การส่งสัญญาณผ่านระบบโมเดม
โดยการเข้ารหัสข้อมูลแบบบล็อคโค๊ดเชิงเส้น
LINEAR BLOCK CODE FOR SIGNAL TRANSMITTING VIA MODEM

โดย

นาย พงษ์ชัย นิลาศ รหัส 33.100231

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต
ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2537

ปีการศึกษา 2537

การส่งสัญญาณผ่านระบบโมเดม
โดยการเข้ารหัสข้อมูลแบบบล็อคโค้ดเชิงเส้น

จัดทำโดย

นาย พงษ์ชัย นิลาศ รหัส 33.100231

อาจารย์ที่ปรึกษา

รศ.ดร. ฟูศักดิ์ ชีวสุวิทย์

ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

การส่งสัญญาณผ่านระบบโมเดม
โดยการเข้ารหัสข้อมูลแบบบล็อคโค้ดเชิงเส้น

จัดทำโดย

นาย พงษ์ชัย นิลาศ รหัส 33.100231

..... อาจารย์ที่ปรึกษา
(รศ.ดร.ฟูศักดิ์ ชีวสุวิทย์)

สารบัญ

หน้า

การส่งสัญญาณผ่านระบบโมเดม
โดยการเข้ารหัสข้อมูลแบบบล็อคโค๊ดเชิงเส้น

จัดทำโดย
นาย พงษ์ชัย นิลาศ 33.100231

อาจารย์ที่ปรึกษา
รศ.ดร. ฟูศักดิ์ ชีวสุวิทย์

ปีการศึกษา 2537

บทคัดย่อ

โครงงานนี้เป็นการเข้ารหัสข้อมูลโดยซอฟท์แวร์ จุดประสงค์หลักในการเข้ารหัสข้อมูลคือทำการป้องกันและแก้ไขความผิดพลาดของข้อมูลอันอาจจะเกิดระหว่างการรับส่งข้อมูลเหล่านั้น (ERROR-CORRECTING CODE) โดยโปรแกรมที่เขียนขึ้นมานี้จะมีความสามารถในการทำการเข้ารหัส และ ถอดรหัส ได้อย่างถูกต้อง มีความเชื่อถือได้ในการทำงานสูงสามารถแก้ไขรหัสข้อมูลที่ผิดพลาด (ERROR BIT) ได้โดยตัวของโปรแกรมเองและสามารถทำการเข้ารหัสและถอดรหัสไฟล์ต่าง ๆ ได้ทุกชนิด ไม่เพียงแต่ไฟล์ตัวอักษรเท่านั้นอัลกอริทึมที่ใช้ในการเข้ารหัส และ ถอดรหัสนี้คือ แบบบล็อคโค๊ดเชิงเส้น (LINEAR BLOCK CODE) ซึ่งได้รับการยอมรับว่ามีความปลอดภัยในการรักษาข้อมูลและมีความสะดวกทั้งในการเข้ารหัส , การถอดรหัสและการแก้ไขบิทที่ผิดพลาด

สำหับโปรแกรมที่เขียนขึ้นมานี้ใช้ภาษา C ในการเขียน โดยใช้คอมไพเลอร์ของ BORLAND C++ version 2.0

LINEAR BLOCK CODE FOR SIGNAL TRANSMITTING VIA MODEM

Colleague

Mr. Phongchai Nilas 33.100231

Advisor

Assoc. Dr. Fusak Cheevasuvit

YEAR 1994

ABSTRACT

Nowadays conceding that an extream problem for tramsmitting and receiving signal between computers is about error bit which occured during the signal transmission and this project is made for this reason. The mainly aim of this project is ciphering signal to prevent and correct the error bit that maybe occured during the time of transmitting. This solfware have an ability to encipher and decipher signal correctly from error bit by itself with high realiability. This program can ciphering and correcting the error signal not only text file but also all others kind of file.

The algorithm that I used for encoding and decoding signal is Linear Block Code method which , widely , accepted in excellent safty for keeping signal and very convenience for encoding , decoding , and correcting the error.

บทนำ

ในปัจจุบันเป็นยุคแห่งข่าวสารและข้อมูล การดำเนินงานต่าง ๆ จำเป็นต้องอาศัยข่าวสารเป็นอย่างมาก ดังนั้นความถูกต้องของข้อมูลหรือข่าวสารจึงเป็นสิ่งที่สำคัญ การนำเอาเครื่องคอมพิวเตอร์มาช่วยในการติดต่อสื่อสารให้เป็นไปอย่างอัตโนมัติ ข้อมูลของข่าวสารต้องเป็นรหัสตัวเลขที่จะป้อนให้กับเครื่องคอมพิวเตอร์ จึงจำเป็นอย่างยิ่งที่จะต้องสร้างความเข้าใจระหว่างผู้ส่งข่าวสารและผู้รับข่าวสารในสามารถทำการส่งและรับข้อมูลต่าง ๆ ได้อย่างถูกต้อง ปัญหาที่เกิดขึ้นในระบบการส่งสัญญาณข้อมูลด้วยรหัสตัวเลขคือการเกิดรหัสที่ผิดพลาดไปเนื่องจากการส่งผ่านไปในช่องสัญญาณที่มีการรบกวน วิศวกรที่เกี่ยวข้องกับการส่งสัญญาณข้อมูลดังกล่าวจึงต้องหาทางควบคุมรหัสที่ผิดพลาดนี้ โครงงานนี้จึงเป็นการสร้างซอฟท์แวร์คอมพิวเตอร์ เพื่อใช้ในการเข้ารหัสข้อมูล ถอดรหัสข้อมูลและสามารถแก้ไขข้อมูลที่มีบิทที่ผิดเนื่องจากการส่งสัญญาณ ให้กลับเป็นข้อมูลที่ถูกต้องตามเดิมได้โดยอัตโนมัติ รูปแบบการเข้ารหัสข้อมูลที่ใช้ในโครงงานนี้คือแบบบล็อคโค๊ดเชิงเส้น ซึ่งมีอัลกอริทึ่มที่สามารถเข้ารหัสและถอดรหัสข้อมูล รวมทั้งสามารถแก้ไขรหัสที่ผิดได้อย่างถูกต้อง รวดเร็ว มีโครงสร้างของข้อมูลที่เข้ารหัสและถอดรหัสไม่ซับซ้อนจนเกิดไป ส่วนภาษาที่ใช้เขียนก็คือภาษา C ซึ่งเป็นภาษาที่ใช้กันแพร่หลาย เพราะใช้ง่ายและทำงานได้คล่องตัว

วัตถุประสงค์ และ ขอบเขตของโครงงาน

โครงงานนี้เป็นการสร้างซอฟท์แวร์โปรแกรมเพื่อใช้ในการเข้ารหัสข้อมูล โดยการเข้ารหัสนี้มีวัตถุประสงค์เพื่อการควบคุมรหัสที่ผิด (ERROR-CORRECTING CODE) อันอาจจะเกิดขึ้นในระหว่างการส่งสัญญาณข้อมูลจากผู้ส่งไปยังผู้รับ โดยเน้นที่ไฟล์ต่าง ๆ ที่อยู่ในดิสค์ เราออกแบบโปรแกรมเพื่อให้ใช้งานได้ง่าย สามารถใช้ได้โดยบุคคลทั่วไปและคอมพิวเตอร์ทั่ว ๆ ไป การเข้ารหัสและถอดรหัสอาศัยวิธีการแบบบล็อคโค๊ดเชิงเส้น ซึ่งทำให้การเข้ารหัสและถอดรหัสมีความปลอดภัยสูง สามารถแก้ไขรหัสที่ผิดได้โดยตัวของโปรแกรมเอง โดยไม่ต้องมีการส่งสัญญาณข้อมูลที่ผิดนั้นจากผู้ส่งซ้ำอีก

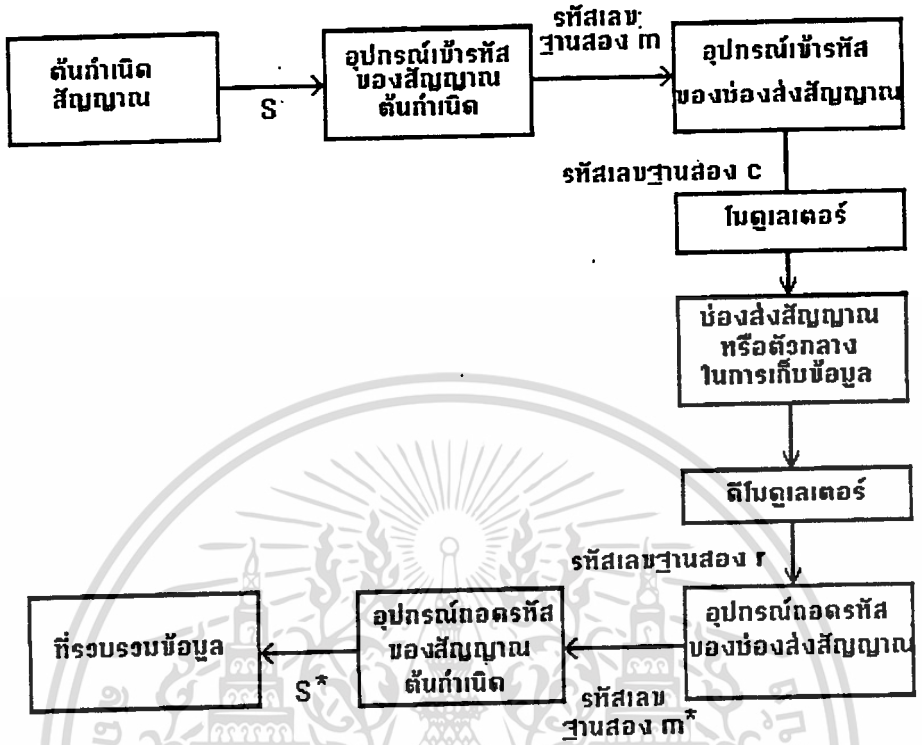
บทที่ 1
การติดต่อสื่อสารและรหัสข้อมูล
(Communication and Coding)

1.1 การติดต่อสื่อสารและรหัสข้อมูล (Communication and Coding)

ในช่วงหลายปีที่ผ่านมาความต้องการที่จะมีระบบการสื่อสารที่มีประสิทธิภาพและความเชื่อถือไว้วางใจได้เพิ่มมากขึ้นอย่างรวดเร็วโดยต้องการประมวลข้อมูลแบบอัตโนมัติ (automatic data processing) สำหรับการติดต่อสื่อสาร ปัญหาใหญ่ที่เกิดขึ้นในระบบส่ง-รับข้อมูลด้วยความเร็วสูงคือ รหัสข้อมูลที่รับได้จะผิดไปจากข้อมูลที่ส่งออกไปกรือเรียกกันว่าเกิด error ขึ้น ด้วยเหตุนี้วิธีการควบคุมรหัสที่ผิดไปของข้อมูลจึงเป็นจุดพื้นฐานที่สำคัญในการออกแบบวิธีการเข้ารหัส (encoding)

ระบบการสื่อสารปัจจุบันได้นำเอาเครื่องคอมพิวเตอร์เข้ามาใช้ในการประมวลข้อมูลอย่างอัตโนมัติ ข้อมูลของการสื่อสารจึงอยู่ในรูปลักษณะของรหัสเลขฐานสอง (binary) ที่มีเลขรหัส "0" กับ "1" แผนผังในการติดต่อสื่อสารด้วยรหัสเลขฐานสองดังแสดงในรูปที่ 1.1

ส่วนแรกของระบบการติดต่อสื่อสารในรูปที่ 1.1 คือต้นกำเนิดสัญญาณซึ่งเป็นข้อมูลที่ต้องการใช้ในการติดต่อสื่อสาร สัญญาณจากต้นกำเนิดอาจจะเป็นสัญญาณแบบต่อเนื่องหรือแบบไม่ต่อเนื่อง (Continuous or discrete) ก็ได้ ปกติแล้วสัญญาณถ้าเป็นแบบต่อเนื่องก็จะถูกแซมปลิ้ง (sampling) ให้เป็นสัญญาณแบบไม่ต่อเนื่องเพื่อจะได้เข้ารหัสสัญญาณ ช่องส่งสัญญาณ (channel) เป็นตัวกลางของการติดต่อสื่อสารซึ่งอาจจะเป็นสายโทรศัพท์ อากาศรอบตัวในกรณีที่มีการติดต่อสื่อสารทางวิทยุ หรือชั้นบรรยากาศ สำหรับการติดต่อสื่อสารผ่านดาวเทียมเป็นต้น ในช่องส่งสัญญาณของการติดต่อสื่อสารจะมีสัญญาณรบกวนอยู่สองประเภทคือสัญญาณรบกวนจากธรรมชาติและสัญญาณรบกวนที่เกิดจากมนุษย์สร้างขึ้นมา (natural and man-made noises) อุปกรณ์เข้ารหัสของสัญญาณต้นกำเนิดจะแปลงสัญญาณต้นกำเนินให้อยู่ในลักษณะของลำดับของสัญญาณ (seguence) ที่ประกอบด้วยตัวเลข "0" กับ "1" สัญญาณที่ถูกแปลงจะมีอยู่ m ชุด ซึ่งเรียกว่าลำดับของข่าวสาร (information sequence) อุปกรณ์เข้ารหัสของช่องส่งสัญญาณจะแปลงรหัสข้อมูล m ให้เป็นลำดับของสัญญาณที่ยาวขึ้นและเรียกว่าโค๊ดเวอด (code word) ซึ่งใช้อักษรย่อว่า C รหัสข้อมูลที่เป็นตัวเลขไม่สามารถใช้ส่งเพื่อการสื่อสารโดยตรงจึงจำเป็นต้องใช้โมดูเลเตอร์ช่วย โมดูเลเตอร์นี้จะเป็นตัวเปลี่ยนรหัสตัวเลขให้อยู่ในลักษณะรูปคลื่น (wave form) บางชนิด (เช่น frequency shift keying หรือ phase shift keying เป็นต้น) ที่จะช่วยลดการรบกวนในช่องส่งสัญญาณ ส่วนดีโมดูเลเตอร์ก็จะแปลงสัญญาณลักษณะรูปคลื่นให้กลับมาเป็นรหัสตัวเลข



รูปที่ 1.1 ผังทั่วไปในการติดต่อสื่อสารข้อมูล

ลำดับสัญญาณรหัสตัวเลขที่ได้คือ ซึ่งเรียกกันว่าลำดับสัญญาณที่รับได้ (recieved sequence) สัญญาณรบกวนในช่องส่งสัญญาณจะทำให้รหัสตัวเลขในลำดับของสัญญาณผิดเพี้ยนไป ตัวอย่างเช่นถ้าลำดับสัญญาณรหัสตัวเลข C = (1100 1100 111011) ถูกส่งออกมาและ r = (1100 0100 101011) เป็นลำดับสัญญาณรหัสตัวเลขที่รับได้ จะพบว่าเกิดรหัสตัวเลขผิด ณ ตำแหน่ง 5 และที่ 11 ดังนั้นอุปกรณ์เข้ารหัสของช่องส่งสัญญาณ จะถูกออกแบบให้สัญญาณเอาท์-พุท (Output) เป็นรหัสของคำที่มีความสามารถในการแก้รหัสผิดที่เกิดขึ้นในระหว่างการส่ง ส่วนอุปกรณ์ถอดรหัสของช่องส่งสัญญาณจึงมีหน้าที่สองอย่างคือ แก้รหัสที่ผิด ในลำดับสัญญาณตัวเลข r ที่รับได้และสร้างลำดับสัญญาณรหัสตัวเลขใหม่เป็น C* ซึ่งเป็นลำดับสัญญาณประมาณของรหัสตัวเลข C อีกหน้าที่หนึ่งก็คือเมื่อเปลี่ยน C* ให้เป็น m* ซึ่งเป็นลำดับสัญญาณรหัสตัวเลขที่ประมาณว่าเป็นสัญญาณที่ใช้ในการติดต่อสื่อสาร

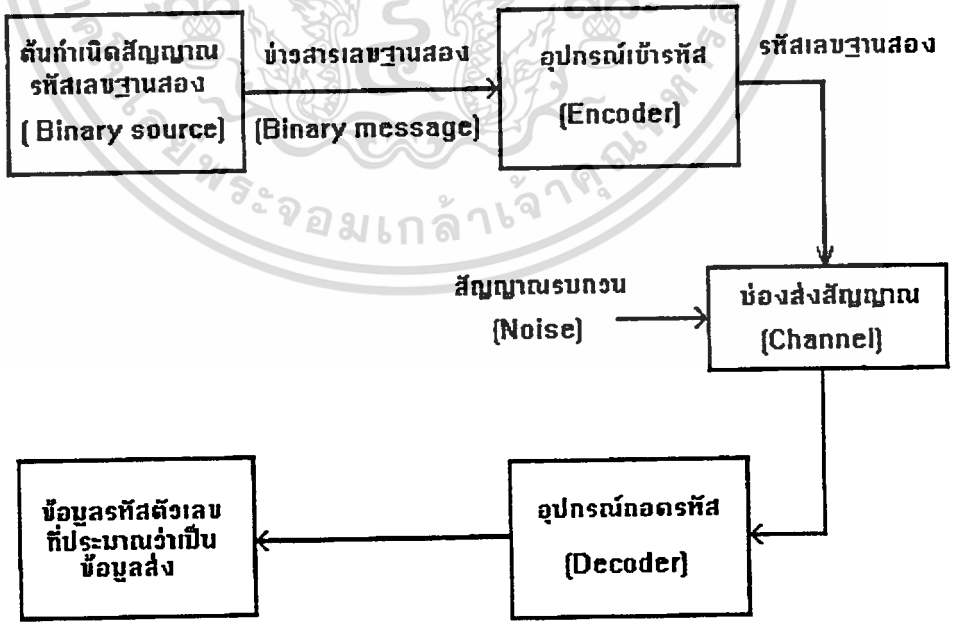
การออกแบบของการเข้ารหัสและการถอดรหัสของช่องส่งสัญญาณมักจะคำนึงถึงความต้องการหลังสองประการด้วยกันคือความเร็วของการส่งสัญญาณผ่านระบบสายส่งที่มีสัญญาณรบกวนและความเชื่อถือ ในการนำสัญญาณ m กลับคืนมาจากสัญญาณเอาท์พุท ของอุปกรณ์ถอดรหัส

1.2 บล็อคโค้ดและการถอดรหัสด้วยวิธีแม็กซิมั่มไลค์รีฮูด

(Block code and Maximum likelinood decoding)

ถ้าหากจะพูดถึงเฉพาะการเข้ารหัสและการถอดรหัสของช่องส่งสัญญาณ (Channel encoding and decoding) ในระบบของรูปที่ 1.1 สามารถเขียนได้เป็นรูปที่ 4.2

สัญญาณจากเอาท์พุทของต้นกำเนินสัญญาณเลขฐานสองจะถูกแบ่งเป็นบล็อค (block) เพื่อนำไปเข้ารหัส แต่ละบล็อคประกอบด้วยรหัสข่าวสาร k บิท (bit) ซึ่งทำให้ได้ข้อมูลที่มีรหัสตัวเลขไม่เหมือนกันจำนวน 2^k ข้อมูล ตัวเข้ารหัสจะแปลงสัญญาณแต่ละบล็อคที่มีรหัสข่าวสารอยู่จำนวน k บิท ให้เป็นรหัสข่าวสารที่ยาวขึ้นเป็น n บิท รหัสข่าวสารที่ถูกเข้ารหัสใหม่นี้ เรียกว่าโค๊ดเวอด (Code word)



รูปที่ 1.2 รูปแบบของ ช่องส่งสัญญาณ (channel modle)

ของ m ซึ่งเขียนย่อด้วยตัวอักษร C เมื่อ m เป็นรหัสข่าวสารในแต่ละบล็อค ชุดของรหัสข้อมูลที่แตกแต่งกันจำนวน 2^k ข้อมูลนี้เรียกบล็อคโค๊ด ส่วนรหัสที่เพิ่มขึ้นจำนวน n - k บิทนี้เรียกว่ารหัสแก้ไข (redundant digits) หรือตัวควบคุมรหัสที่ผิดไป (error control coding) รหัสแก้ไขนี้จะไม่มีข้อมูลที่เป็นข่าวสารผสมอยู่เลยและจะมีหน้าที่ในการแก้รหัสข้อมูลที่ผิด ดังนั้นการออกแบบสร้างรหัสแก้ไขที่สามารถแก้รหัสที่ผิดจึงเป็นบรรทัดฐานของการออกแบบระบบการเข้ารหัส ระบบการเข้ารหัสจะให้อัตราส่วนระหว่างจำนวนรหัสข่าวสารต่อจำนวนรหัสที่ส่งคือ R = k /n เรียกกันว่า โค๊ดเรท (code rate) เป็นตัวบ่งบอกประสิทธิภาพของระบบการเข้ารหัส

ถ้าให้โค๊ดเวอด $c = (c_0 , c_1 , c_2 , ... c_{n-1})$ เป็นรหัสสัญญาณตัวเลขที่ใช้ในการส่งเพื่อติดต่อสื่อสาร และให้ $r = (r_0 , r_1 , r_2 , ... r_{n-1})$ เป็นรหัสสัญญาณตัวเลขที่รับได้จากช่องส่งของสื่อสาร ตัวถอดรหัสจะเป็นผู้ตัดสินใจว่ารหัสแต่ละบิทของ r นั้นควรจะเป็นเลข “0” หรือ “1” จากชุดรหัส r เมื่อนำมาคำนวณหาเงื่อนไขความน่าจะเป็น (Conditional probability) $P(r/c_e)$ สำหรับโค๊ดเวอด 2^k ชด สำหรับโค๊ดเวอด c_i จะถือว่าเป็นรหัสที่ส่งออกมาก็ต่อเมื่อ $P(r/c_e)$ มีค่าสูงสุดซึ่งวิธีการนี้เรียกว่าการถอดรหัสแบบแม็กซิมั่นไลค์ลีฮูด

จากทฤษฎีของ Shannon ที่ว่าด้วยเรื่องของรหัสว่าทุกช่องส่งสัญญาณจะมีขีดจำกัดคือความสามารถของอัตราการส่งสัญญาณมีค่าเท่ากับ c โดยทั่วไปแล้วมักจะให้ RLC ดังนั้นถ้ารู้ความยาวของ n ก็จะได้บล็อคโค๊ดที่ให้ค่าความเป็นไปได้ของความผิดพลาดในการถอดรหัส

$$P(\varepsilon) \leq e^{-nE(R)} \qquad \text{..... (1.1)}$$

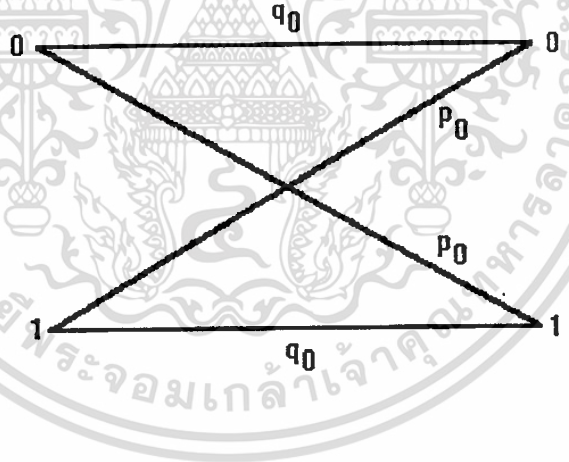
เมื่อ E(R) คือฟังค์ชั่นค่าบวก (positive function) ของ R สำหรับ R < C ดังนั้นการลดค่าความน่าจะเป็นไปได้ของการถอดรหัสผิดทำได้ด้วยการเพิ่มค่า n ซึ่งเป็นการเพิ่มความยาวของโค๊ด และรักษาให้อัตราส่วนของ มีค่าน้อยกว่าค่าความสามารถของอัตราการส่งของช่องส่ง c ทฤษฎีของ Shannon เพียงแต่แสดงให้เห็นว่ามีโค๊ดที่จะทำให้ความน่าจะเป็นไปได้ในการถอรหัสผิดมีค่าต่ำแต่ก็ไม่ได้แสดงให้เห็นว่าโค๊ดที่ว่านั้นจะสร้างได้อย่างไร จึงเป็นปัญหาที่จะต้องศึกษากันต่อไป

จากสมการที่ 1.1 ความยาวของรหัสคือ n มีค่าสูงเพื่อจะให้ได้ผลถูกต้อง ถ้าในการสร้างรหัสนี้จากการเข้ารหัสที่เก็บโค๊ดเวอดไว้ $2^{nR} = (2 k)$ และการถอดรหัสก็จะทำงานได้แบบแม็กซิมั่นไลค์ลีฮูด ดังนั้นทั้งการเข้าและการถอดรหัสจะไม่ซับซ้อน ทางด้านการถอดรหัสจะต้องคำนวณ เงื่อนไขความน่าจะเป็น $P(r/c_e)$ จำนวน 2^{nR} ครั้ง ดังนั้นจึงทำให้เกิดปัญหา 3 อย่าง ตามมาคือ 1) ต้องหารหัสที่ยาวและที่ดี 2) ต้องหาวิธีการเข้ารหัสในเชิงปฏิบัติ และ 3) ต้องหาร

มาคือ 1) ต้องหารหัสที่ยาวและที่ดี 2) ต้องหาวิธีการเข้ารหัสในเชิงปฏิบัติ และ 3) ต้องหาวิธีการถอดรหัสในเชิงปฏิบัติ

ความน่าจะเป็นไปได้ของความผิดพลาดในการถอดรหัสขึ้นอยู่กับคุณสมบัติทางสถิติของความผิดพลาดในสายส่ง ในทางทฤษฎีถ้าความน่าจะเป็นไปได้ของการที่จะได้รหัสที่ส่งเหมือนกับ รหัสที่รับมาคือ g_0 ส่วนความน่าจะเป็นไปได้องการที่จะได้รหัสส่งกับรหัสรับแตกต่างกันคือ $r_0 = 1 - g_0$ ดังแสดงในรูปที่1.3โดยทั่วไปมักจะถือว่าการส่งสัญญาณแต่ละตัวจะแยกอิสระต่อกัน ช่องส่งสัญญาณที่มีคุณสมบัติดังกล่าวนี้เรียกว่า Binary Symmetric Channel (BSC) P_0 เรียกว่าความน่าจะเป็นแบบทรานซิชั่น (transition probability) ดังนั้นเงื่อนไขความน่าจะเป็น $P(r/c_e)$ เขียนได้เป็น

$$P(r / c_e) = \prod_{i=0}^{n-1} P(ri / C_{ei}) \qquad \text{..... (1.2)}$$



รูปที่ 1.3 ระบบ BSC

โดยที่ $P(ri/c_{ei}) = q_0$ สำหรับ $ri = C_{ei}$ และ $P(ri/c_{ei}) = P_0$ ถ้า $ri \neq c_{ei}$ ถ้าให้ d_e เป็นจำนวนตำแหน่งที่โค๊ดเวอด c_e กับลำดับสัญญาณ r ที่รับได้แตกต่างกัน สมการ (1.2) จะกลายเป็น

$$P(r / c_e) = q_0^{n-d_e} \; p_0^{d_e} \qquad \text{..... (1.3)}$$

ถ้า $q_0 > P_0$, $P(r/c_e)$ จะลดลงแบบทางเดียว (decreases monotonically) เมื่อค่า d_e เพิ่มขึ้น ดังนั้นในการหาโค๊ดเวอด c_i ที่มี $P(r/c_i)$ มีค่ามากที่สุดก็คือการหาโค๊ดเวอด c ที่ผิดไปจากลำดับสัญญาณ r ที่รับเข้ามาน้อยที่สุด

1.3 ชนิดของรหัสที่ผิด

รหัสผิดซึ่งเกิดจากระบบการส่งของการสื่อสารแบบรหัสตัวเลขมีสาเหตุมาจากสัญญาณรบกวนในช่องการสื่อสาร (communication channel) โดยทั่วไปสัญญาณรบกวนในช่องของการสื่อสารแบ่งได้ออกเป็นสองประเภท ประเภทแรกเป็นสัญญาณรบกวนแบบเกาส์เซียน (Gaussian noise) ซึ่งเกิดจากโมดูเลเตอร์และดีโมดูเลเตอร์ สัญญาณรบกวนแบบเกาส์เซียนนี้จะรวมไปถึงสัญญาณรบกวนแบบเทอร์มอล (thermal noise) และแบบช๊อค (Shot noise) ที่เกิดจากเครื่องอุปกรณ์การส่งและการรับสัญญาณ Power spectral density ของสัญญาณรบกวนเกาส์เซียนจะมีคุณสมบัติเป็นสัญญาณรบกวนสีขาว (White noise) ซึ่งเรียกกันว่าสัญญาณรบกวนสีขาวแบบเกาส์เซียน ดังนั้นความผิดพลาดที่เกิดจากสัญญาณรบกวนสีขาว แบบเกาส์เซียนมักจะเรียกกันว่าความผิดพลาดแบบแรนดอม (random error)

สัญญาณรบกวนประเภทที่สองเรียกว่าสัญญาณรบกวนแบบอิมพัลส์ (impulse noise) ซึ่งมักจะเกิดในช่องการสื่อสาร ขนาดของสัญญาณรบกวนจะสูง ซึ่งอาจเกิดจากธรรมชาติเช่น ฟ้าผ่า หรือเกิดจากคนทำขึ้น เช่น การเปิดปิดไฟฟ้า สัญญาณรบกวนนี้บางครั้งจะให้ความกว้างของสัญญาณรบกวนเกินกว่า 1 บิทของสัญญาณ

รหัสที่ใช้แก้รหัสผิดที่เกิดจากสัญญาณรบกวนแบบแรนดอม เรียกวันว่า random error correcting codes และรหัสที่ใช้แก้รหัสผิดที่เกิดจากสัญญาณรบกวนแบบอิมพัลส์เรียกกันว่า burst error- correcting codes

1.4 ชนิดของรหัส

รหัสที่ใช้ควบคุมรหัสผิดโดยปกติจะแบ่งเป็นสองพวกคือ พวกบล็อคโค๊ด (block code) และพวกคอนโวลูชั่นแนลโค๊ด (Convolutional code) ในระบบบล็อคโค๊ดนั้นถ้ามีสัญญาณที่เป็นข่าวสารอยู่ k บิท จะถูกตามด้วยกลุ่มของรหัสในการแก้ไข n - k บิททางด้านรับรหัสที่ใช้แก้ไขจะถูกนำมาใช้ในการตรวจสอบรหัสข่าวสารของบล็อคนั้น ๆ ทางด้านรับรหัสที่ใช้แก้ไขจะถูกนำมาใช้ในการตรวจสอบรหัสข่าวสารของบล็อคนั้น ๆ ทางด้านรับรหัสที่ใช้แก้ไขจะถูกนำมาใช้ในการตรวจสอบรหัสข่าวสารของบล็อคนั้น ๆ ในระบบคอนโวลูชั่นแนลโค๊ดรหัสที่ใช้ในการแก้ไขจะถูก

แทรกเข้าระหว่างรหัสของข่าวสาร รหัสแก้ไขไม่เพียงแต่จะใช้ตรวจสอบรหัสในบล็อคของข่าวสารที่อยู่ข้างหน้า แต่จะใช้การตรวจเช็ครหัสของข่าวสารในบล็อคอื่น ๆ ด้วย

1.5 ตัวอย่าง

ตัวอย่างต่อไปนี้เป็นการแสดงการใช้รหัสแก้ไขในการแก้รหัสบิทที่ผิด สมมติว่ามีข่าวสารที่จะส่งอยู่ในรูปเลขฐานสอง 12 บิท ที่จะทำการส่ง จากรหัสของข่าวสาร 12 บิทนี้แยกจัดให้อยู่ในรูปของแมททริกซ์ (matrix) ขนาด 3 x 4 คือ

$$\begin{matrix} X_{11} & X_{12} & X_{13} & X_{14} \\ X_{21} & X_{22} & X_{23} & X_{24} \\ X_{31} & X_{32} & X_{33} & X_{34} \end{matrix}$$

ทำการเพิ่มรหัสที่เรียกว่าพารีตี้บิท (parity bit) ให้กับแต่ละแถว (row) และแต่ละคอลัมน์ (column) จำนวน 1 บิท หลักการเพิ่มรหัสนี้ก็แล้วแต่ว่าใครจะตั้งกฎเกณฑ์ไว้อย่างไร เช่น ถ้าตั้งกฎเกณฑ์ว่าต้องการให้ผลบวกของรหัสในแต่ละแถวและแต่ละคอลัมน์หลังการเพิ่มรหัสแล้วได้ค่าเป็นคู่ (คือ 0 สำหรับเลขฐานสอง) รหัสที่เพิ่มเข้าไปคือ

X_{15}, X_{25}, X_{35}, X_{41}, X_{42}, X_{43}, X_{44} และ X_{45} จะได้เมททริกซ์ใหม่ดังนี้

$$\begin{matrix} X_{11} & X_{12} & X_{13} & X_{14} & X_{15} \\ X_{21} & X_{22} & X_{23} & X_{24} & X_{25} \\ X_{31} & X_{32} & X_{33} & X_{34} & X_{35} \\ \hline X_{41} & X_{42} & X_{43} & X_{44} & X_{45} \end{matrix}$$

ตัวอย่างรหัสของข่าวสารและรหัสที่เพิ่มเข้าไปคือ

$$\begin{matrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{matrix}$$

รหัสทั้งหมดนี้เมื่อถูกส่งออกไปแถวต่อแถว (หรือคอลัมน์ต่อคอลัมน์) และถ้าเกิดรหัสเปลี่ยนไป 1 ตัว กล่าวคือเกิดข้อมูลผิดไป 1 บิท ก็สามารถแก้ได้ด้วยการตรวจสอบพาริตี้ว่าเป็นคู่หรือเปล่า ถ้ามีรหัสผิดไปเพียงตัวเดียวจากการตรวจสอบจะมีเพียงแถวเดียวและคอลัมน์เดียวที่ให้พาริตี้ผิดไป จุดตัดของแถวกับคอลัมน์ ที่พาริตี้ผิดไปคือคือรหัสที่ผิดทำให้สามารถที่จะแก้รหัสนั้นให้ถูกต้องได้ กรณีนี้โค๊ดเวอดจะมีความยาวของรหัส 20 บิท โดยมีรหัสขาวสารอยู่ 12 หลัก (n=2 , k=12) รหัสแก้ไขมีความยาว n-k = 20-12 = 8 บิท จะช่วยในการแก้รหัสที่ผิด สัญญษณ์ทางคณิตศาสตร์ของการเพิ่มรหัสแก้ไขเขียนได้เป็น (n,k) ส่วนการแก้รหัสที่ผิดมากกว่า 1 ตำแหน่งจะต้องใช้คณิตศาสตร์ที่ยุ่งยากเพิ่มขึ้น

1.6 ระยะห่าง (distance)

ระยะห่างระหว่างสองโค๊ดเวอด คือตำแหน่งที่แตกต่างกันในสองโค๊ดเวอดนั้น และระยห่างต่ำสุด (minimum distance) ของรหัสทั้งหมดคือระยะต่ำสุดระหว่างโค๊ดเวอดใด ๆ ตัวอย่างเช่น ระยะห่างแบบแฮมมิ่ง (Hamming distance) ระหว่างสองโค๊ดเวอดก็คือจำนวนตำแหน่งของรหัสที่แตกแต่งกันในสองโค๊ดเวอดนั้น ส่วนระยะห่างต่ำสุดของบล๊อคโค๊ดเชิงเส้น (linear block code) คือ เวทต่ำสุด (minimum weight) ของโค๊ดเวอดที่ไม่เป็นศูนย์ ตัวอย่างของโค๊ดเวอดที่ให้ระยะห่างต่ำสุดของบล๊อคโค๊ดเป็น 3 ดังแสดงข้างล่างนี้

โค๊ดเวอด						น้ำหนัก
0	0	0	0	0	0	0
0	0	1	1	1	1	3
0	1	0	1	0	1	3
0	1	1	0	1	1	4
1	0	0	0	1	1	3

สำหรับการแก้ความผิดพลาด t บิท ระยะห่างต่ำสุดอย่างน้อยคือ 2t + 1 เช่นถ้าเกิดความผิดไป 1 บิทค่าระยะห่างต่ำสุดคือ 2.1 + 1 = 3 ซึ่งจะทำให้แน่ใจได้ว่าถ้าเกิดรหัสผิดไป 1 บิท สัญญาณที่รับได้ยังมีค่าใกล้เคียงกับสัญญาณที่ส่งมา

บทที่ 2

คณิตศาสตร์ที่ใช้ในการแก้รหัสที่ผิด

จุดประสงค์ของบทนี้จะเป็นการแนะนำให้ผู้อ่านได้รู้ถึงพีชคณิตที่ใช้ ซึ่งจะช่วยให้เกิดความเข้าใจได้ดีขึ้นในบทถัดไป ถ้าหากผู้อ่านสนใจอยากจะมีความรู้ทางพีชคณิตให้ลึกซึ้งยิ่งขึ้นก็สามารถหาอ่านได้จากหนังสือประเภท Modern Algebra

2.1 กาลัวส์ฟิลด์ (Galois Field arithmetic)

การบวกและการคูณของจำนวนเลขที่นับได้ (finite number) สามารถทำได้ก็ต่อเมื่อจำนวนเลขเหล่านั้นเป็นกำลัง (power) ของจำนวนเต็มหารไม่ลงตัว (prime number) เมื่อเป็นเช่นนี้ก็สามารถใช้กฎเกณฑ์ธรรมดาของคณิตสาศตร์ในการบวกและการคูณได้ ในกรณีของการใช้รหัสตัวเลขสำหรับการติดต่อสื่อสารซึ่งจะมีสัญญลักษณ์ “0” กับ “1” การบวกและการคูณของรหัสตัวเลขทำได้ตามกฎเกณฑ์ต่อไปนี้

$0+0 = 0$	$0.0 = 0$
$0+0 = 1$	$0.1 = 0$
$1+0 = 1$	$1.0 = 0$
$1+1 = 0$	$1.1 = 1$

ตารางที่ 2.1 การบวกและการคูณของรหัสสองสัญญลักษณ์

การบวกและการคูณดังกล่าวเรียกว่าการบวกและการคูณแบบโมดูโล -2 (Modulo -2) โดยถือว่า 2 เท่ากับ 0 ดังนั้น $1+1 = 0$ และ $1 = -1$ สัญญลักษณ์ 0 กับ 1 พร้อมด้วยการบวกและการคูณแบบโมดูโล-2 นี้ รวมกันเข้าเป็นฟิลด์ (field) ซึ่งเรียกว่าไบนารีฟิลด์ (binary field) เขียนเป็น GF (2)

เราสามารถแสดงให้เห็นว่าพีชคณิตจะสามารถนำมาใช้อย่างไรกับคณิตศาสตร์ที่กล่าวมาข้างต้น ถ้าสมมติว่ามีชุดของสมการเชิงเส้นเป็น

$$x + y = 1$$

$$x + z = 0$$

$$x + y + z = 1$$

การแก้สมการทำได้โดยการลบสมการที่ 1 ออกจากสมการท่ 3 จะให้ผลลัพธ์คือ Z = 0 จากสมการที่ 2 เมื่อ z = 0 จะให้ x = 0 ในที่สุดเมื่อแทนค่า x = 0 ในสมการที่ 1 ก็จะได้ y = 1 เมื่อนำค่าของ x , y และ z แทนลงในสมการทั้ง 3 ก็จะพบว่าค่าที่ได้ถูกต้อง

การแก้สมการเหล่านั้นสามารถทำได้ โดยใช้กฎของ Carmar แต่มีข้อแม้ว่าสมการเหล่านั้นต้องเป็นอิสระเชิงเส้น (linearly independent) และมีค่าดีเทอร์มีแนนท์ (determinant) ของสัมประสิทธิ์ทางด้านซ้ายมือของสมการไม่เป็นศูนย์ซึ่งสามารถตรวจสอบได้ดังนี้

$$\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix} = 1 \cdot \begin{vmatrix} 0 & 1 \\ 1 & 1 \end{vmatrix} - 1 \cdot \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix} + 0 \cdot \begin{vmatrix} 1 & 0 \\ 1 & 1 \end{vmatrix}$$

$$= 1 \cdot 1 - 1 \cdot 0 + 0 \cdot 1 = 1$$

การแก้สมการด้วยกฎของ Carmer

$$x = \frac{\begin{vmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}} = \frac{0}{1} = 0$$

$$y = \frac{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}} = \frac{1}{1} = 1$$

$$z = \frac{\begin{vmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}} = \frac{0}{0} = 0$$

ต่อไปเราจะพิจารณาการคำนวณโพลีโนเมียล (polynomial) ที่มีสัมประสิทธิ์เป็น 1 หรือ 0 สำหรับเลขจำนวนจริงถ้ามี λ เป็นรากของโพลิโนเมียล f(x) นั่นคือ $f(\lambda) = 0$) f(x) จะถูกหารด้วย $x - \lambda$ ลงตัว กฎเกณฑ์นี้ยังคงเป็นจริงสำหรับ f(x) ซึ่งมีสัมประสิทธิ์เป็นเลขฐานสองให้ $f(x) = x^4 + x^3 + x^2 + 1$ ดังนั้น $f(1) = 1^4 + 1^3 + 1^2 + 1 = 1+1+1+1 = 0$ นั่นคือ f(x) หารด้วย x -1 (ซึ่งเท่ากับ x +1) ได้ลงตัว

$$
\begin{array}{r}
x^3 \quad + x \quad + 1 \\
\overline{)\,x^4 + x^3 + x^2 \quad + 1} \\
\underline{x^4 + x^3} \\
x^2 \\
\underline{x^2 + x} \\
x + 1 \\
\underline{x + 1} \\
0
\end{array}
$$

โพลิโนเมียลของดีกรีที่ 1 จะมี x และ x +1 เป็นฟังค์ชั่นของ f(x) โพลิโนเมียลดีกรีที่ 2 ของ f(x) จะมี x^2 , $x^2 + x$, $x^2 + 1$, $x^2 + x + 1$ เนื่องจาก $1^2 + 1 = 0$ ดังนั้น $x^2 + 1$ จะถูกหารด้วย x +1 ลงตัว โดยที่จริงแล้ว $(x+1)^2 = x^2 + x + x + 1 = x^2 + 1$ (เนื่องจาก $x + x = 0$) อย่างไรก็ตาม $x^2 + x + 1$ จะไม่มีรากที่เป็น 0 หรือ 1 ดังนั้นจึงไม่มีแฟคเตอร์ (factor) ใด ๆ มาหารโพลิโนเมียลนี้ได้ลงตัวยกเว้น 1 และตัวของมันเอง โพลิโนเมียล p(x) ขนาดอันดับ (order) เท่ากับ m จะถูกเรียกว่าลดรูปไม่ได้ (irreducible) ภายใต้ GF(2) ก็ต่อเมื่อ p(x) นี้ไม่สามารถแยกแฟกเตอร์อกได้อีกหรือไม่สามารถหารด้วยโพลิโนเมียลใดๆ ที่มีอันดับต่ำกว่า m ได้ลงตัว ตัวอย่างเช่น $x^4 + x + 1$ นี้เป็นโพลิโนเมียลที่ลดรูปไม่ได้ โดยจะพบวา 0 และ 1 ไม่เป็นรากของโพลิโนเมียลนี้ ดังนั้น ทั้ง x และ x +1 ไม่ได้เป็นแฟคเตอร์ของโพลิโนเมียลดังกล่าวถ้าหากนำโพลิโนเมียลดังกล่าวนี้ไปตั้งหารยาวด้ว $x^2 + x + 1$ ก็ยังคงพบว่าไม่สามารถหารได้ลงตัวจึงสรุปได้ว่า $x^4 + x + 1$ นี้ลดรูปไม่ได้

ฟิลด์ (filed) ที่มีสัญญลักษณ์อยู่จำนวน 2^m ตัวเรียกว่า GF (2^m) ซึ่งมีความสำคัญมากในการศึกษาพวกไซคลิคโค๊ด (cyclic code) ในทางปฏิบัติจะถูกนำไปใช้ในการถอดรหัส BCH (Boss-Chaudhuri - Hocquenghem codes) และใช้เป็นสัญญลักษณ์ในรหัสแบบ Read-Solomon คณิตศาสตร์ของสัญญลักษณ์ 2^m สามารถหาได้ตามวิธีการต่อไปนี้ อันดับแรกเราเริ่มจากคณิตศาสตร์ของโพลิโนเมียล p(x) มีอันดับเท่ากับ m ซึ่งมี 2 สัญญลักษณ์ ให้ α เป็นรากของโพลิโนเมียลกล่าวคือ $p(\alpha) = 0$ และกำหนดว่า $2 = 0$ ในฟิลด์ที่มีสองสัญลักษณ์ ถ้าหากเราเลือกโพลิโนเมียล p(x) ที่เหมาะสมจะพบว่ากำลังของ α จะมีค่าถึง $2^m - 2$ ที่แตกต่างกัน โดยที่กำลังของ α ที่ $2^m - 1$ จะกลับมาเท่ากับ 1 ใหม่ ดังนั้น 0 , 1 α , α^2 ..... , α^{2^m-2} จะเป็นเซท (Set) ของ 2^m

field elements แต่ละอีลีเมนท์ (element) สามารถเขียนได้ด้วยผลรวมของพวกอีลีเมนท์ที่เป็น 1 , α ; α^2 , ... α^{m-1} ตัวอย่างสำหรับ $m = 4$, $p(x) = x^4 + x + 1$ ให้ $p(\alpha) = \alpha^4 + \alpha + 1 = 0$ จะได้ $\alpha^4 + \alpha + 1$ α จะมีกำลังที่ให้ค่าแตกต่างกันถึงกำลังที่ $2^4 - 2 = 14$ ดังนั้นการกระจายเทอมกำลังต่าง ๆ ของ α ทำได้ดังตารางที่ 5.2

0

1

α

α^2

α^3

$\alpha^4 = \alpha + 1$

$\alpha^5 = \alpha(\alpha + 1) = \alpha^2 + \alpha$

$\alpha^6 = \alpha(\alpha^2 + \alpha) = \alpha^3 + \alpha^2$

$\alpha^7 = \alpha(\alpha^3 + \alpha^2) = \alpha^4 + \alpha^3 = \alpha^3 + \alpha^2 + 1$

$\alpha^8 = \alpha(\alpha^3 + \alpha^2 + 1) = \alpha^4 + \alpha^3 + \alpha = \alpha^2 + 1$

$\alpha^9 = \alpha(\alpha^2 + 1) = \alpha^3 + \alpha$

$\alpha^{10} = \alpha(\alpha^2 + \alpha) = \alpha^4 + \alpha^2 = \alpha^2 + \alpha + 1$

$\alpha^{11} = \alpha(\alpha^2 + \alpha + 1) = \alpha^3 + \alpha^2 + \alpha$

$\alpha^{12} = \alpha(\alpha^3 + \alpha^2 + \alpha) = \alpha^4 + \alpha^3 + \alpha^2 = \alpha^3 + \alpha^2 + \alpha + 1$

$\alpha^{13} = \alpha(\alpha^3 + \alpha^2 + \alpha + 1) = \alpha^4 + \alpha^3 + \alpha^2 = \alpha^3 + \alpha^2 + + 1$

$\alpha^{14} = \alpha(\alpha^3 + \alpha^2 + 1) = \alpha^4 + \alpha^3 + \alpha = \alpha^3 + 1$

$\alpha^{15} = \alpha(\alpha^3 + 1) = \alpha^4 + \alpha + 1$

$\alpha^{16} = \alpha(1) = \alpha$

<u>ตารางที่ 2.2</u> กาลัวส์ฟิลด์ 2^4 อีลีเมนท์ ($GF(2^4)$ ซึ่ง $P(\alpha) = \alpha^4 + \alpha + 1 = 0$ หรือ $\alpha^4 = \alpha + 1$

อีลีเมนท์ α นี้เรียกว่าไพร($GF(2^m)$โดยทั่ว ๆ ไป อีลีเมนท์ใด ๆ ของ $GF(2^m)$ ที่มีกำลังที่สร้างได้ไม่เป็นศูนย์ ของ $GF(2^m)$ เรียกว่าไพรมีทีฟ ตัวอย่างกำลังของ α^4 ของ $GF(2^4)$ ในตารางที่ 2.2 คือ

$(\alpha^{4})^{0} = 1$, $(\alpha^{4})^{1} = \alpha^{4}$, $(\alpha^{4})^{2}$

$(\alpha^{4})^{3} = \alpha^{12}$, $(\alpha^{4})^{4} = \alpha^{16} = \alpha$, $(\alpha^{4})^{5} = \alpha^{20} = \alpha^{5}$

$(\alpha^{4})^{6} = \alpha^{24} = \alpha^{9}$, $(\alpha^{4})^{7} = \alpha^{28} = \alpha^{13}$, $(\alpha^{4})^{8} = \alpha^{32} = \alpha^{2}$

$(\alpha^{4})^{9} = \alpha^{36} = \alpha^{6}$, $(\alpha^{4})^{10} = \alpha^{40} = \alpha^{10}$, $(\alpha^{4})^{11} = \alpha^{44} = \alpha^{14}$

$(\alpha^{4})^{12} = \alpha^{48} = \alpha^{13}$, $(\alpha^{4})^{13} = \alpha^{52} = \alpha^{7}$, $(\alpha^{4})^{14} = \alpha^{56} = \alpha^{11}$

จะเห็นว่าอีลีเมนท์ทั้ง 15 ไม่เป็นศูนย์ (non-zero element) ของ $GF(2^{4})$ ดังนั้น α^{4} เป็นไพรมีทีฟอีลีเมนท์ของ $GF(2^{4})$ อีลีเมนต์ α^{3} ไม่เป็นไพรมิทีฟอีลีเมนท์ของ $GF(2^{4})$โพลิโนเมียล p(x) ซึ่งมีอันดับเท่ากับ m ที่ให้ตารางอย่างสมบูรณ์ด้วยสัญญลักษณ์ที่แตกต่างกัน 2^{m} มีสัญญลักษณ์ซึ่งประกอบด้วย 0 และ 1 จะถูกเรียกว่าไพรมีทีฟโพลิโนเมียล p(x) เป็นโพลิโนเมียลที่ลดรูปไม่ได้มีอันดับเท่ากับ m จะเป็นไพรมีทีฟอีลีเมนต์ของ $GF(2^{m})$ ซึ่งพิสูจน์ได้ว่าแต่ละจำนวนเต็มบวก m จะมีอย่างน้อยที่สุดหนึ่งไพรมีทีฟโพลิโนเมียลของอันดับที่ m ไพรมีทีฟโพลิโนเมียลต่าง ๆ พอจะสรุปออกมาได้ดังตารางที่ 2.3

อันดับ m	ไพรมีทีฟ โพลิโนเมียล
3	$1 + x + x^{3}$
4	$1 + x + x^{4}$
5	$1 + x^{2} + x^{5}$
6	$1 + x + x^{6}$
7	$1 + x^{3} + x^{7}$
8	$1 + x^{2} + x^{3} + x^{4} + x^{8}$
9	$1 + x^{4} + x^{9}$
10	$1 + x^{3} + x^{4}$
11	$1 + x^{2} + x^{11}$
12	$1 + x + x^{4} + x^{6} + x^{12}$
13	$1 + x + x^{3} + x^{4} + x^{13}$
14	$1 + x + x^{6} + x^{10} + x^{14}$
15	$1 + x + x^{15}$
16	$1 + x + x^{3} + x^{12} + x^{16}$
17	$1 + x^{3} + x^{17}$

18	$1 + x^{7} + x^{18}$
19	$1 + x + x^{2} + x^{5} + x^{19}$
20	$1 + x^{3} + x^{20}$
21	$1 + x^{2} + x^{21}$
22	$1 + x + x^{22}$
23	$1 + x^{5} + x^{23}$
24	$1 + x + x^{2} + x^{7} + x^{24}$

ตารางที่ 2.3 ไพรมีทีฟโพลิโนเมียลอันดับต่าง ๆ

ในการคูณของสัญญลักขณ์เหล่านี้ทำได้โดยการบวกกำลังของสัญญลักขณ์และใช้หลักการของ $\alpha^{15} = 1$ (จาก $\alpha^{2^m - 1} = 1$) ตัวอย่างเช่น $\alpha^{5} . \alpha^{7} = \alpha^{12}$ และ $\alpha^{12} . \alpha^{7} = \alpha^{19} = \alpha^{4}$ การหารก็คล้ายคลึงกันกล่าวคือ $\alpha^{12} / \alpha^{5} = \alpha^{7}$ และ $\alpha^{4} / \alpha^{12} = \alpha^{19} / \alpha^{12} = \alpha^{7}$

ส่วนการบวกของสัญญลักขณ์ใช้คุณสมบัติจากตารางที่กำหนดไว้เช่น

$$\alpha^{5} + \alpha^{7} = (\alpha^{2} + \alpha) + (\alpha^{3} + \alpha + 1) = \alpha^{3} + \alpha^{2} + 1 = \alpha^{13}$$

$$1 + \alpha^{5} + \alpha^{10} = 1 + (\alpha^{2} + \alpha) + (\alpha^{2} + \alpha + 1) = 0$$

พึงระลึกไว้เสมอว่า -1 = 1 ดังนั้นการลบจึงมีค่าเหมือนการบวก

ตัวอย่างของการคำนวณสำหรับสมการเชิงเส้น

$$x + \alpha^{7} y = \alpha^{2}$$
$$\alpha^{12} x + \alpha^{8} y = \alpha^{4} \qquad (5.1)$$

คูณสมการที่ 2 ด้วย α^{3} จะได้

$$\alpha^{15} x + \alpha^{11} y = \alpha^{7}$$

หรือ $x + \alpha^{11} y = \alpha^{7}$ ($\alpha^{15} = 1$ จากตารางที่ 5.2)

ในที่สุดจะได้ว่า

$$x + \alpha^{7} y = \alpha^{2}$$
$$x + \alpha^{11} y = \alpha^{7}$$

บวกสมการทั้งสองเข้าด้วยกัน

$$(\alpha^7+\alpha^7)y = \alpha^2+\alpha^7$$

$$(\alpha^3+\alpha+1+\alpha^3+\alpha^2+\alpha)y = \alpha^2+\alpha^3+\alpha+1$$

$$(\alpha^2+1)y = \alpha^3+\alpha^2+\alpha+1$$

$$\alpha^8 y = \alpha^{12}$$

$$y = \alpha^4$$

แทนค่า $y = \alpha^4$ ลงในสมการแรกของ 2.1

$$x + \alpha^7 \cdot \alpha^4 = \alpha^2$$

$$x = \alpha^2 + \alpha^{11} = \alpha^2 + (\alpha^3 + \alpha^2 + \alpha^2)$$

$$\alpha^3 + \alpha = \alpha^9$$

คำตอบของสมการ 2.1 คือ $X = \alpha^9$ และ $y = \alpha^4$

ถ้าหากใช้กฎของ Carmer แก้สมการ 2.1 จะได้

$$X = \begin{vmatrix} \alpha^2 & \alpha^7 \\ \alpha^4 & \alpha^3 \end{vmatrix} = \alpha^{10} + \alpha^{11}$$

$$\begin{vmatrix} 1 & \alpha^3 \\ \alpha^{12} & \alpha^3 \end{vmatrix} \quad \alpha^3 + \alpha^4$$

$$= \begin{vmatrix} \alpha^3 + 1 & \alpha^{14} \\ \alpha^2 + \alpha & \alpha^5 \end{vmatrix} = \alpha^9$$

$$Y = \begin{vmatrix} \alpha^1 & \alpha^2 \\ \alpha^{12} & \alpha^4 \end{vmatrix} = \alpha^4 + \alpha^4$$

$$\begin{vmatrix} 1 & \alpha^3 \\ \alpha^{12} & \alpha^3 \end{vmatrix} \quad \alpha^3 + \alpha^4$$

$$= \begin{vmatrix} \alpha^3 + \alpha & \alpha^9 \\ \alpha^2 + \alpha & \alpha^5 \end{vmatrix} = \alpha^4$$

อีกตัวอย่าง ถ้าสมมติว่าต้องการแก้สมการต่อไปนี้

$f(x) = x^2 + \alpha^7 x + \alpha = 0$

วิธีการที่ว ๆ ไปเอามาใช้ไม่ได้เพราะจากสูตรในการหารากต้องการด้วย 2 ซึ่งในพีชคณิตของ GF(2) จะได้ว่า 2 = 0 ถ้า f(X) = 0 มีคำตอบอยู่ใน $GF(2^2)$ คำตอบสามารถคำนวณได้จากการแทน x ด้วยสัญญาลักษณ์ต่าง ๆ จากตารางที่ 2.2 ก็จะพบว่า $f(\alpha^6) = 0$ และ $f(\alpha^{10}) = 0$ เพราะว่า

$$f(\alpha^6) = (\alpha^6)^2 + \alpha^7.\alpha^6 + \alpha = \alpha^{12} + \alpha^{13} + \alpha$$
$$= (\alpha^3 + \alpha^2 + \alpha + 1) + (\alpha^3 + \alpha^2 + 1) + \alpha = 0$$
$$f(\alpha^{10}) = (\alpha^{10})^2 + \alpha^7.\alpha^{10} + \alpha = \alpha^{20} + \alpha^{17} + \alpha$$
$$\alpha^5 + \alpha^2 + v = (\alpha^2 + \alpha) + \alpha^2 + \alpha = 0$$

และจะพบว่าไม่มีอีลีเมนท์อื่นที่จะให้ f(X) มีค่าเป็นศูนย์ ดังนั้น $f(x) = (x + \alpha^6)(x + \alpha^{10})$ กล่าวคือ α^6 และ α^{10} เป็นรากของ f(x)

การคำนวณที่กล่าวมานี้จะต้องนำไปใช้ในการถอดรหัส BCH ซึ่งสามารถเขียนเป็นโปรแกรมอย่างง่าย ๆ เก็บไว้ในเครื่องคอมพิวเตอร์

ให้ f(x) เป็นโพลิโนเมียลใด ๆ อันดับที่ k

$$f(x) = f_k x^k + f_{k-1} x^{k-1} + ...f_1 x + f_0$$

โดย f_i อาจจะเป็น 0 หรือ 1 เมื่อพิจารณา

$$f2(x) = (f_k x^k + f_{k-1} x^{k-1} + ...+f_1 x + f_0)^2$$
$$= (f_k x^k)^2 + 2(f_k x^k)(f_{k-1} x^{k-1} + ...+f_1 x + f_0)$$
$$+(f_{k-1} x^{k-1} + ...+f_1 x + f_0)^2$$

สมการข้างบนได้จาก $(b+c)^2 = b^2 + 2bc + c^2$ แต่ตามกฎของโมดูโล-2 จะได้ว่า 1+1 =2=0 และ $1.1 = 1^2 = 1$

$$f^2(x) = f_k x^{2k} + (f_{k-1} x^{k-1} + ...+f_1 x + f_0)^2$$

.ในที่สุดเมื่อทำการกระจายไปเรื่อย ๆ จะได้

$$f^2(x) = f_k x^{2k} + f_{k-1} x^{2(k-1)} + ...+f_1 x^2 + f_0$$
$$= x^8 + x^2 + 1 = f(x^2)$$

ตัวอย่างเช่น

$$f^2(x) = (x^4 + x + 1)^2 = x^8 + x^2 + 1 + 2x^5 + 2x^4 + 2x$$

$$= x^8 + x^2 + 1 = f(x^2)$$

จะได้ว่าเมื่อ e เป็นเลขจำนวนเต็มใด ๆ

$$[f(x)]^{2^e} = f(x^{2^e})$$

ถ้าให้ β เป็นอีลีเมนท์ใด ๆ ของกาลัวส์ฟิลด์ $GF(2^m)$ m(x) เป็นโพลิโนเมียลอันดับน้อยที่สุด(smallest polynomail) มีสัมประสิทธิ์เป็นเลขฐานสองโดยที่ $m(\beta) = 0$ จะเป็นโพลิโนเมียลต่ำสุด (minimum polynomail) ของ β นี้ลดรูปไม่ได้ (irreducible) ถ้าสมมรุว่า m(x) ยังลดรูปได้ กล่าวคือ $m(x) = m_1(x)\, m_2(x)$ โดย $m_1(x)$ และ $m_2(x)$ไม่ซ้ำกัน แต่ $m(\beta) = m(\beta)\, m_1(\beta) = 0$ นั่นคือ $m_2(\beta)$ จะต้องเป็นศูนย์ซึ่งจะเป็นการขัดกับสมมติฐานที่ว่า m(x) เป็นโพลิโนเมียลอันน้อยสุดที่ให้ $m(\beta) = 0$ ดังนั้น m(x) จะต้องลดรูปไม่ได้สามารถพิสูจน์ได้ว่าทุกอีลีเมนท์ของ GF(2m) จะเป็นโพลิโนเมียลต่ำสุดโดยจะมีอันดับเท่ากับ m หรือต่ำกว่าเพราะ $m(\beta) = 0$ และ

$[m(x)]^{2^e} = m(x^{2^e})$ ดังนั้น

$$[m(\beta)]^{2^e} = m(\beta^{2^e}) = 0$$

นั่นคือ β^{2^e} เป็นรากของ m(x)โดยที่ = 1,2,3,..... แต่เนื่องจาก m(x)มีอันดับจำกัด ดังนั้นรากจึงมีจำนวนจำกัด ถ้าให้ e เป็นอันดับของ m(x) ซึ่งอาจแสดงให้เห็นได้ว่า

$\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{e-1}}$ เป็นรากที่ไม่ซ้ำกันของ m(X) โดยอีลีเมนต์ทั้งหลายจะซ้ำ ๆกันหลังจาก $\beta^{2^{e-1}}$ ไปแล้ว

ในการหาโพลิโนเมียลต่ำสุดของอีลีเมนท์ β ที่กำหนดให้ใน $GF(2^m)$ซึ่งสา" \s 16□ ข้าใจได้ด้วยตัวอย่าง พิจารณาถึง $GF(2^4)$ ของตารางที่ 2.2 ให้ $\beta = \alpha^3$ ซึ่งได้

$$\beta = \alpha^3, \beta^2 = \alpha^6, \beta^{2^2} = \alpha^{12}, \beta^{2^3} = \alpha^{24} = \alpha^9$$

$$\beta^{2^4} = \alpha^{48} = \alpha^3, \quad \beta^{2^5} = \alpha^{96} = \alpha^6, \ldots\ldots$$

สังเกตว่าจะมีการซ้ำ ๆ กันเริ่มต้นจาก β^{2^4} ดังนั้นโพลิโนเมียลต่ำสุดของ α^3 มีรากเป็น

จะได้ $m(x) = (x+\alpha^{3})(x+\alpha^{6})(x+\alpha^{9})(x+\alpha^{12})$

จากการคูณกระจายทางขวามือของสมการข้างบนและใช้อีลีเมนท์ในตารางที่ 2.2 ช่วยก็จะได้

$$m(x) = x^{4} + (\alpha^{2} + \alpha^{6} + \alpha^{9} + \alpha^{12})x^{3} + (\alpha^{9} + \alpha^{12} + \alpha^{15} + \alpha^{18} + \alpha^{21})X^{2} + (\alpha^{18} + \alpha^{21} + \alpha^{24} + \alpha^{27})x + \alpha^{30}$$

$$= x^{4} + x^{3} + x^{2} + x + 1$$

2.2 เวคเตอร์สแปซ (Vector space)

จากหัวข้อที่ 2.1 สัญญลักษณ์ 0 กับ 1 พร้อมด้วยการบวกและการคูณแบบโมดูโล -2 ที่กำหนดไว้ในตารางที่ 5.1 ซึ่งเรียกกันว่าไบนารีฟิลด์ GF (2) ต่อไปเมื่อพิจารณาถึงลำดับของสัญญลักษณ์เลขฐานสอง

$$\underline{V} = (V_1, V_2, V_1, \ldots\ldots V_n) \quad \ldots\ldots (5.2)$$

เมื่อ V_i เป็นรหัสบิทที่ i จะมีค่าเป็น 0 หรือ 1 เท่านั้น ดังนั้นลำดับของรหัส $\underline{V}$ สามารถเขียนได้ถึง Z^{n} ลำดับของรหัสที่แตกต่างกัน ในการบวกของลำดับสัญญาณสองชุดโดยที่แต่ละชุดจะมี n อีลีเมนท์ (หรือเรียกกวันว่า n - tuples) ทำได้ดังนี้

$$\underline{V} = (V_1, V_2, V_1, \ldots\ldots V_n) \quad \ldots\ldots (5.2)$$

$$\underline{U} = (U_1, U_2, U_1, \ldots\ldots U_n) \quad \ldots\ldots (5.2)$$

$$\underline{V} + \underline{U} = (V_1 + U_1, V_2 + U_2, V_3 + U_3, \ldots\ldots, V_n + U_n) \quad \ldots\ldots (5.2)$$

การบวกระหว่าง V กับ U เป็นไปตามกฎโมดูโล -2 ดังนั้นลำดับสัญญาณ $\underline{V}+\underline{U}$ ก็ยังคงเป็นลำดับของรหัสที่มีอีลีเมนท์เป็นเลขฐานสองอยู่ n อีลีเมนท์ คุณสมบัติการบวกอีกอย่างคือ

$$\underline{V} + \underline{U} = \underline{U} + \underline{V}$$

การคูณเลขฐานสองของ n - ทูเปิลส์ (n-tuples) ด้วยสคาลาร์ (Scalar) ใน GF (2) คือ

$$\sigma(V_1, V_2, V_3, \ldots\ldots, V_n) = (\sigma V_1, \sigma V_2, \sigma V_3, \ldots\ldots, \sigma V_n) \qquad (5.5)$$

เซทของเลขฐานสองของ n - ทูเปิ้ลส์นี้เรียกว่าเวคเตอร์สแปซของ GF(2) ซึ่งเขียนด้วย $\underline{V}_n$ เวคเตอร์สแปซจะมีบทบาทสำคัญในการถอดรหัส n - ทูเปิลส์ของ $\underline{V}_n$ โดยทั่วไปจะเรียกว่าเวคเตอร์

ตัวอย่างที่ 2.1 เมื่อ $n = 4$ เวคเตอร์สแปซ $\underline{V}_4$ จะประกอบด้วย

$(0\ 0\ 0\ 0), (0\ 0\ 0\ 1)$

$(0\ 0\ 1\ 0), (0\ 0\ 1\ 1)$

$(0\ 1\ 0\ 0), (0\ 1\ 0\ 1)$

$(0\ 1\ 1\ 0), (0\ 1\ 1\ 1)$

$(1\ 0\ 0\ 0), (1\ 0\ 0\ 1)$

$(1\ 0\ 1\ 0), (1\ 0\ 1\ 1)$

$(1\ 1\ 0\ 0), (1\ 1\ 0\ 1)$

$(1\ 1\ 1\ 0), (1\ 1\ 1\ 1)$

ในการบวกเลขเวคเตอร์ทำได้โดย

$$(0\ 1\ 0\ 1) + (1\ 1\ 1\ 0) = (0+1, 1+1, 0+1, 1+0)$$
$$= (1\ 0\ 1\ 1)$$

การคูณของสองเวคเตอร์ด้วยสคาลาร์ (Scalar)

$$1.(1\ 0\ 1\ 1) = (1.1, 1.0, 1.1, 1, 1)$$
$$= (1\ 0\ 1\ 1)$$

$$0.(1\ 0\ 1\ 1) = (0.1, 0.0, 0.1, 0.1)$$
$$= (0\ 0\ 0\ 0)$$

ถ้า S เป็นซับเซท (subset) ของ $\underline{V}_n$ ซึ่งเรียกว่าซับสแปซ (subspzce) ต่อเมื่อ

(i) มีเวคเตอร์ศูนย์ (Zero vector) อยู่ใน S

(ii) ผลรวมของสองเวคเตอร์ใน S ยังคงอยู่ใน S

ตัวอย่างที่ 2.2 พิจารณาถึงเซทของเวคเตอร์

$v_0 = (0\ 0\ 0\ 0)$

$v_1 = (0\ 1\ 0\ 1)$

$v_2 = (1\ 0\ 1\ 0)$

$v_3 = (1\ 1\ 1\ 1)$

ซึ่งเป็นซับเซทของเวคเตอร์สแปซ $\underline{V}_4$ จะประกอบด้วยซีโร่เวคเตอร์และสามารถตรวจสอบได้ง่าย ๆ สำหรับสองเวคเตอร์ใด ๆ ใน S ว่าผลบวกที่ได้จะยังคงเป็นเวคเตอร์ใน S ดังนั้น

$S = \{V_0, V_1, V_2, V_3\}$ จะรวมกันเป็นซับสแปซของ $\underline{V}_4$

ให้ $V_1, V_2, \ldots\ldots, V_K$ เป็น K เวคเตอร์ของ $\underline{V}_n$ และ การรวมอย่างเชิงเส้นของเวคเตอร์เหล่านี้จะให้

$$\underline{u} = c_1 V_1 + c_2 V_2 + \ldots\ldots + c_k V_k$$

เมื่อ c_i ได้จาก GF(2) ซึ่งเรียกว่าเป็นสัมประสิทธ์ของ V_i

ตัวอย่างที่ 2.3 พิจารณาถึงเวคเตอร์สแปซ V_4 ที่ให้ในตัวอย่างที่ 2.1 ให้

$V_1 = (0\ 1\ 0\ 0)$

$V_2 = (1\ 0\ 1\ 0)$

$V_3 = (1\ 0\ 1\ 1)$

$V_4 = (1\ 1\ 0\ 0)$

และ $c_1 = 1, c_2 = 0, c_3 = 1$ และ $c_4 = 1$ ดังนั้น

$$\underline{u} = c_1 V_1 + C_2 V_2 + C_3 V_3 + C_4 V_4$$

$$= 1.(0100) + 0.(1010) + 1.(1011) + .1\ (1100)$$

$$= (0100) + (1011) + (1100)$$

$$= (0\ 0\ 1\ 1)$$

เซทของเวคเตอร์เพ็นลิเนียร์ลีดีเพ็นเด็นท์ (linearly dependent) ถ้าและเพียงแต่ถ้า (if and only if) สคาลาร์ $c_1, c_2, \ldots c_n$ จาก GF(2) ทุกตัวไม่เท่ากับศูนย์ ดังนั้น

$$c_1 v_1 + c_2 v + \ldots + c_n v_n = 0$$

ตัวอย่าง 2.4 พิจารณาถึงชุดของเวคเตอร์ต่อไปนี้

$V_1 = (1\ 1\ 0\ 0)$

$V_2 = (1\ 0\ 1\ 0)$

$V_3 = (1\ 0\ 1\ 1)$

$V_4 = (1\ 1\ 0\ 1)$

ทำให้ $c_1 = c_2 = c_3 = c_4 = 1$ ดังนั้น

$$u = c_1 v_1 + c_2 v_2 + c_3 v_3 + c_4 v_4$$

$$= 1.(1100) + 1.(1010) + 1.(1011) + 1.(1101)$$

$$= (1100) + (1010) + (1011) + (1101)$$

$$= (0\ 0\ 0\ 0)$$

ซึ่ง V_1, V_2, V_3 และ V_4 เป็นลิเนียร์ลีดีเพ็นเด็นท์

เซทของเวคเตอร์ที่ไม่เป็นลิเนียร์ลีดีเพ็นเด็นท์ก็จะเรียกว่าลิเนียร์ลีอินดีเพ็นเด็นท์ ถ้าเซทของเวคเตอร์เรียกว่าสแปน (span) ของเวคเตอร์สแปซ ถ้าทุกเวคเตอร์ในเวคเตอร์สแปซ $\underline{V}_n$ เกิดจากการรวมอย่างเชิงเส้นของเซทของเวคเตอร์

ตัวอย่างที่ 2.5 เซทของเวคเตอร์

$v_1 = (1\ 0\ 0\ 0)$

$v_2 = (0\ 1\ 0\ 0)$

$v_3 = (0\ 0\ 1\ 0)$

$v_4 = (0\ 0\ 0\ 1)$

เป็นลิเนียร์ลีอินดีเพ็นเด็นท์อย่างชัดเจน เวคเตอร์ใด ๆ ใน $\underline{v}_4$ เกิดจากการรวมอย่างเชิงเส้นของเซทของเวคเตอร์เหล่านี้ ดังนั้น v_1, v_2, v_3 และ V_4 เป็นสแปนของเวคเตอร์สแปซ $\underline{V}_4$

ในเวคเตอร์สแปซหรือซับสแปซใด ๆ จะมีอย่างน้อยเซทหนึ่งของลิเนียร์ลีดีเพ็นเด้นท์เวคเตอร์ที่ขยายสแปซ เซทของเวคเตอร์นี้เรียกว่าแบซิส (basis) ของเวคเตอร์สแปซ $\underline{V}_n$ ของเลขฐานสองชนิด n-ทูเปิ้ลส์เป็นเวคเตอร์ขนาด n-มิติ (n-dimensional) โดยเวคเตอร์สแปซนี้ได้จากการกระจายของเบซิสที่มี n เวคเตอร์อิสระอย่างเชิงเส้น

ถ้าหาก $k<n$ และ $v_1, v_2 \ldots v_k$ เป็น k เวคเตอร์อิสระเชิงเส้น ดังนั้นการรวมอย่างเชิงเส้นของ $v_1, v_2, \ldots, v_k$ จะให้

$$u = c_1v_1 + c_2v_2 + \ldots + c_kv_k$$

ซึ่ง u เป็นซับสแปซ k มิติของ s ใน $\underline{V}_n$ โดย ci มีค่าเป็น 0 หรือ 1 จะมีการรวมอย่างเชิงเส้นของ $v_1, v_2, \ldots, v_k$ ที่ไม่เหมือนกันอยู่ 2^k ชุด ดังนั้น s จะมี 2^k เวคเตอร์ที่ต่างกันและเป็นซับสแปซ k มิติของ $\underline{v}_n$

5.3 แมทริกซ์ (Matrices)

ถ้า k x n เป็นอะเร (array) ที่มี k แถวและ n คอลัมน์

$$G = \begin{matrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \\ g_{k1} & g_{k2} & \cdots & g_{kn} \end{matrix}$$

โดยอีลิเมนท์ g_{ij} จะเป็น "0" หรือ "1" เท่านั้น อะเร G นี้เรียกว่า k x n แมททริกซ์ของ GF(2) แต่ละแถวจะมีรหัสเลขฐานสองอยู่ n บิท และแต่ละคอลัมน์ จะมีรหัสเลขฐานสอง k บิท ถ้าทั้ง k แถว (เมื่อ $k < n$) ต่างก็มี n - ทูเปิ้ลส์ที่เป็นอิสระเชิงเส้นใน $\underline{V}_n$ ดังนั้นการรวมอย่างเชิงเส้นของแถวต่าง ๆ ของ G จะให้ซับสแปช K-มิติ ของ $\underline{V}_n$ ซึ่งเราเรียกว่าโรว์สแปช (row space) ของ G

ตัวอย่างที่ 2.6

$$G = \begin{matrix} 1\ 0\ 0\ 1\ 0\ 1 \\ 0\ 1\ 0\ 0\ 1\ 1 \\ 0\ 0\ 1\ 1\ 1\ 0 \end{matrix}$$

G เป็นแมททริกซ์ขนาด 3x6 จะมี 3 แถว และ 6 คอลัมน์ แต่ละแถวเป็น 6-ทูเปิ้ลส์แต่ละคอลัมน์เป็น 3-ทูเปิ้ลส์ โดยการรวมอย่างเชิงเส้นของแถวเหล่านี้เป็น

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

เซทนี้เป็นซับเซทของเวคเตอร์สแปช $\underline{V}_6$ จะประกอบด้วย $2^6 = 64$ ชุดของ 6-ทูเปิ้ลส์ ซึ่งตรงกับคำจำกัดความของซับสแปชโดยซับสแปชนี้มี 2^3 อีลีเมนท์เป็นซับสแปช 3-มิติของ $\underline{V}_6$

ถ้าพิจารณาถึงอินเนอร์โพรดัคท์ (inner product) ของสองเวคเตอร์ใด ๆ

$\underline{v} = (v_1 , v_2,....v_n)$

$\underline{u} = (u_1 , u_2,....u_n)$

การทำอินเนอร์ดพรดัคท์จะได้

$$\underline{v}.\underline{u} = v_1.u_1 + v_2.u_2 + ...+v_n.u_n \qquad (2.8)$$

โดยการบวกและการคูณใช้หลักการของโมดูโล-2 ถ้า $\underline{v}.\underline{u} = 0$ v และ u จะตั้งฉากกัน (orthogonal)

สำหรับแมททริกซ์ G ขนาด k x n ที่มี k แถวที่เป็นอิสระอย่างเชิงเส้น ถ้ามีแมททริกซ์ H ขนาด (n-k)xn

$$H = \begin{matrix} h_1 \\ h_2 \\ \\ h_{n-k} \end{matrix} \begin{matrix} h_{11} & h_{12} & \cdots & h_{1n} \\ h_{21} & h_{22} & \cdots & h_{2n} \\ & & & \\ h_{n-k,1} & h_{n-k,2} & \cdots & h_{n-k,n} \end{matrix} \qquad 2.9$$

แมททริกซ์ H มี $h_j = (h_{j1}, ..., h_{j2})$ ดังนั้น n-k แถวที่เป็นอิสระอย่างเชิงเส้นและเวคเตอร์ v ในโรว์สแปซของ T จะตั้งฉากกับทุกแถวของ H นั้นคืออินเนอร์โพรดัคท์

$$\underline{v}.h_j = \text{สำหรับ } 1 < j < n-k$$

ถ้า g_i เป็นเวคเตอร์ในโรว์สแปซของ G ซึ่งจะให้อินเนอร์โพรดัคท์

$$g_i \cdot h_j = 0$$

สำหรับ $1 \le i \le k$ และ $1 \le i \le n-k$ ให้ $\underline{u}$ เป็นเวคเตอร์ในโรว์สแปซของ H ดังนั้น U เป็นการรวมอย่างเชิงเส้นของ H

$$\underline{u} = d_1 h_1 + d_2 h_2 + \cdots + d_{n-k} h_{n-k}$$

เมื่อ d_1 เท่ากับ 0 หรือ 1 สำหรับ $1 \le i \le n-k$ ดังนั้นอินเนอร์โพรดัคท์ของ $\underline{v}$ และ $\underline{u}$ คือ

$$\underline{v}.\underline{u} = \underline{v}.(d_1 h_1 + \cdots = d_{n-k} h_{n-k})$$

$$= d_1 (\underline{v}.h_1) + \cdots + d_{n-k} (\underline{v}.h_{n-k}) \qquad 2.10$$

แต่ $\underline{v}.h_j = 0$ จะได้ $\underline{v}.\underline{u} = 0$ นั่นคือเวคเตอร์ใด ๆ ในโรว์สแปซของ G กับเวคเตอร์ $\underline{u}$ ในโรว์สแปซของ H ต่างก็ตั้งฉากซึ่งกันและกัน ดังนั้นโรว์สแปซของ G เรียกว่า นูลล์สแปซ (null space) ของ H หรือกลับกกัน H เป็นนูลล์สแปซของ G

บทที่ 3

บล็อคโค๊ดเชิงเส้น (Linear block code)

ในบทนี้จะได้กล่าวถึงหลักการพื้นฐานของบล็อคโค๊ดโดยเฉพาะบล็อคโค๊ดเชิงเส้นในหัวข้อ 3.2 และ 3.3 ได้อธิบายถึงแมททริกซ์ของบล็อคโค๊ดเชิงเส้นและสมการพารีตี้สำหรับรหัสตามระบบ (systematic code) ในหัวข้อ 3.4 จะได้พูดถึงระยะห่างต่ำสุด (minimum distance) ของบล็อคโค๊ดพร้อมทั้งแสดงให้เห็นถึงความสามารถในการแก้รหัสที่ผิดไปอย่างแรนดอม (random-error correcting) และการตรวจจับรหัสที่ผิดนั้น (random-error detecting) ของบล็อคโค๊ดจากการใช้ระยะห่างต่ำสุด ในหัวข้อ 3.5 จะได้พูดถึงการถอดรหัสของบล็อคโค๊ดพร้อมแสดงตารางการถอดรหัสของบล็อคโค๊ดเชิงเส้น

3.1 นิยาม

ถ้าสมมติว่าข่าวสารที่ส่งออกมาจากต้นกำเนิดอยู่ในลักษณะของรหัสเลขฐานสองคือเป็นชุดลำดับของรหัสเลขฐานสอง (sequence of binary digits) กระบวนการเข้ารหัสจะประกอบด้วยขั้นตอนพื้นฐาน 2 ขั้นตอนคือ 1) แบ่งชุดลำดับของสัญญาณออกเป็นบล็อคข่าวสาร (message block) ต่าง ๆ แต่ละบล็อคจะประกอบด้วยรหัสของข่าวสารอยู่ k บิท 2) ทำการเข้ารหัสโดยการแปลงชุดลำดับของรหัสแต่ละบล็อคให้เป็นชุดลำดับรหัสที่ยาวขึ้นกว่าเดิมเป็น n บิท (n>k) ซึ่งบล็อคของสัญญาณใหม่นี้เรียกว่าโค๊ดเวอด (code word) เนื่องจากแต่ละบล็อคของข่าวสารจะประกอบด้วยรหัส k บิท ดังนั้นจึงมีรหัสของข่าวสารที่แตกต่างกัน 2^k ชุด หรือพูดง่าย ๆ ว่ามี 2^k โค๊ดเวอดที่แตกต่างกัน ชุดของ 2^k โค๊ดเวอดนี้เรียกว่าบล็อคโค๊ดบางครั้งอาจจะเรียกโค๊ดเวอดว่า โค๊ดเวคเตอร์ (code vector) เพราะเป็น n-ทูเปิ้ลส์จากเวคเตอร์สแปซ v_{-n} ของทุก ๆ n-ทูเปิ้ลส์

สำหรับบล็อคโค๊ดที่ให้คำจำกัดความข้างบนนั้นจะพิจารณาถึงโครงสร้างของ 2^k โค๊ดเวค- เตอร์ของแต่ละชุดจาก k-มิติของซับสแปซในทุก ๆ n-ทูเปิ้ลส์ จากโครงสร้างของรหัสนี้จะ
สามารถช่วยลดความยุ่งยากในการเข้ารหัส

นิยามที่ 3.1 ชุดของ 2^k n-ทูเปิ้ลส์ เรียกว่าโค๊ดเชิงเส้น ถ้าและเพียงแต่ถ้าเป็นซับสแปซของเวคเตอร์สแปซ v_{-n} ของทุก ๆ n-ทูเปิ้ลส์

ตัวอย่างที่ 3.1 ถ้าหากว่าตัวเข้ารหัสแบ่งลำดับของข่าวสารออกเป็นบล็อค ๆ แต่ละบล็อคจะประ- กอบด้วยรหัสข่าวสาร 3 บิท หลังจากนั้นตัวเข้ารหัสจะแปลงบล็อคข่าวสารให้เป็นโค๊ดเวอดหรือ

เมื่อ $\underline{v}_i = (v_{i1}, v_{i2}, v_{i3}, \ldots v_{in})$ สำหรับ $i = 1,2,\ldots,k$ ให้ $\underline{m} = (m_1, m_2, \ldots, m_k)$ เป็นบล็อคของข่าวสาร โค๊ดเวอดจะได้จาก

$$\underline{u} = \underline{m}\, G$$

$$= (m_1, m_2, \ldots, m_k) \begin{bmatrix} \underline{V}_1 \\ \underline{V}_2 \\ M \\ \underline{V}_k \end{bmatrix} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.3)$$

$$= m_1 \underline{V}_1 + m_2 \underline{V}_2 + \ldots\ldots + m_k \underline{V}_k \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.1)$$

ดังนั้นโค๊ดเวอดที่สอดคล้องกับชุดรหัสข่าวสาร $(m_1, m_2, \ldots, m_k)$ เกิดจากการรวมแบบเชิงเส้นของแถวใน G กลุ่มแถวต่าง ๆ ของแมททริกซ์ G จะเป็นตัวผลิตรหัสเชิงเส้นและเราเรียกแมททริกซ์ G ว่าเจนเนอเรเตอร์แมททริกซ์ของรหัส รหัสเชิงเส้นที่กล่าวนี้เรียกว่ารหัส (n,k) โดยในแต่ละบล็อคจะมีข่าวสารอยู่ k บิทที่ถูกเข้ารหัสเป็นโค๊ดเวอดที่มีรหัสอยู่ n บิทและถูกส่งผ่านออกไปในช่องส่งที่มีสัญญาณรบกวน อัตราส่วน $R = k/n$ เรียกว่าความเร็วรหัส (coderate) เนื่องจากรหัสเชิงเส้นได้จากเจนเนอเรเตอร์แมททริกซ์ G ไว้แทนที่จะเก็บโค๊ดเวคเตอร์จำนวน 2^k โค๊ดเวคเตอร์

ตัวอย่างที่ 3.2 รหัสที่แสดงไว้ในตัวอย่างที่ 3.1 คือ (6,3) ที่มีเจนเนอเรเตอร์แมททริกซ์ G

$$G = \begin{bmatrix} \underline{V}_1 \\ \underline{V}_2 \\ \underline{V}_3 \end{bmatrix} = \begin{bmatrix} 1\,0\,0\,1\,1\,0 \\ 0\,1\,0\,0\,1\,1 \\ 0\,0\,1\,1\,0\,1 \end{bmatrix}$$

โค๊ดเวอดที่สอดคล้องกับรหัสข่าวสาร $\underline{m} = (1\ 0\ 1)$ คือ

$$\underline{U} = (1\ 0\ 1) \begin{bmatrix} \underline{V}_1 \\ \underline{V}_2 \\ \underline{V}_3 \end{bmatrix}$$

$$= 1.V_1 + 0.V_2 + 1.V_3$$

รหัสข่าวสาร	การเข้ารหัส	โค๊ดเวอด
0 0 0		0 0 0 0 0 0
0 0 1		0 0 1 1 0 1
0 1 0		0 1 0 0 1 1
0 1 1		0 1 1 1 1 0
1 0 0		1 0 0 1 1 0
1 0 1		1 0 1 0 1 1
1 1 0		1 1 0 1 0 1
1 1 1		1 1 1 0 0 0

แต่เนื่องจาก k=3 จะมีจำนวนข้อความอยู่ $2^3 = 8$ ข้อความของข่าวสารที่แตกต่างกัน แต่ละข้อความของข่าวสารจะถูกแปลงเป็นโค๊ดเวอดขนาด 6 บิท ฉะนั้นการส่งข่าวสารก็คือการส่งโค๊ดเวอดเหล่านี้ออกไปนั้นเอง จะเห็นว่าชุดของโค๊ดเวอดนี้จะฟอร์มเป็นซับสแปซ 3-มิติของเวคเตอร์สแปซที่เป็น 6-ทูเปิ้ลส์จึงเป็นรหัสเชิงเส้น

3.2 เจนเนอเรเตอร์แมททริกซ์ (Generator matrix)

สำหรับซับสแปซ s ของ v_{-n} และแต่ละ n-ทูเปิ้ลส์ของ s เป็นการรวมแบบเชิงเส้นของ $v_1, v_2, \ldots, v_k$ กล่าวคือ

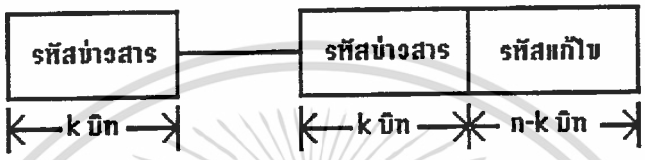
$$u = m_1 v_1 + m_2 v_2 + \ldots\ldots + m_k v_k \qquad \ldots\ldots\ldots\ldots\ldots\ldots (3.1)$$

เมื่อ $m_i = 0$ หรือ 1 สำหรับ $i = 1,2,\ldots,k$ ซับสแปซนี้มีขนาด k-มิติของ v_{-n} ซึ่งประกอบด้วย 2^k ของ n-ทูเปิ้ลส์ จากข้อเท็จจริงที่กล่าวมาซึ่งสามารถอธิบายถึงโค๊ดเชิงเส้นของ 2^k โค๊ดเวอดโดยเซทของ k โค๊ดเวคเตอร์เป็นอิสระเชิงเส้น ถ้าจัด k โค๊ดเวอดซึ่งเป็นอิสระต่อกันได้แมททริกซ์ k x n

$$G = \begin{bmatrix} \underline{v}_1 \\ \underline{v}_2 \\ M \\ \underline{v}_k \end{bmatrix} = \begin{bmatrix} v_{11}\ v_{12}\ v_{13} \cdots\cdots v_{1n} \\ v_{21}\ v_{22}\ v_{23} \cdots\cdots v_{2n} \\ M \\ v_{k1}\ v_{k2}\ v_{k3} \cdots\cdots v_{kn} \end{bmatrix} \qquad \ldots\ldots\ldots\ldots (3.2)$$

$$= 1.(1\ 0\ 0\ 1\ 1\ 0) + 0.(0\ 1\ 0\ 0\ 1\ 1)$$
$$+ 1.(0\ 0\ 1\ 1\ 0)$$
$$= (1\ 0\ 1\ 0\ 1\ 1)$$

ในโค๊ดเวอดขนาด n บิท โดยมีรหัสของสารอยู่ k บิท ก็หมายความว่าได้มีการเพิ่มรหัสเข้าไปในแต่ละบล็อคของข่าวสารด้วยขนาด n-k บิท รหัสที่เพิ่มขึ้นนี้เรียกว่ารหัสแก้ไข (redundant digits) โดยจะอยู่ต่อท้ายรหัสข่าวสาร k บิทดังแสดงในรูปที่ 6.1



รูปที่ 3.1

รหัสที่ได้นี้เรียกว่ารหัสระบบ (systematic code) โดยที่รหัสแก้ไขนี้จะเป็นตัวแก้รหัสที่ผิดที่เกิดขึ้นในระหว่างการส่งผ่านช่องส่งที่มีสัญญาณรบกวน หรือพูดอีกนัยหนึ่งว่ารหัสแก้ไขมีความสามารถที่จะช่วยป้องกันข่าวสาร ปัญหาจึงอยู่ที่ว่าจะจัดตั้งรูปแบบรหัสแก้ไขได้อย่างไร สำหรับรหัสเชิงเส้น (n,k) สามารถอธิบายได้ด้วยเจนเนอเรเตอร์แมทริกซ์ขนาด k x n ดังนี้

$$G = \begin{bmatrix} 1\ 0\ 0\ 0.....0 & p_{11}\ p_{12} \cdots p_{1,n-k} \\ 0\ 1\ 0\ 0.....0 & p_{21}\ p_{22} \cdots p_{2,n-k} \\ 0\ 0\ 1\ 0.....0 & p_{31}\ p_{32} \cdots p_{3,n-k} \\ M & M \\ 0\ 0\ 0\ 0.....1 & p_{k1}\ p_{k2} \cdots p_{k,n-k} \end{bmatrix} \quad \text{.....(3.4)}$$

โดย $p_{ij} = 0$ หรือ 1 ให้ i_k เป็นไอเด็นติตี้แมททริกซ์ (identity matrix) ขนาด k x k และให้ p เป็นแมททริกซ์ขนาด k x (n-k) ที่มีอีลีเมนท์เป็น p_{ij} ดังนั้นเจนเนอเรเตอร์แมททริกซ์ของรหัสระบบเขียนใหม่ได้เป็น

$$G = \left[i_k \ p_{n-k} \right]$$

พิจารณาถึงบล็อคของข่าวสาร $m = (m_1, m_2, ..., m_k)$ เมื่อใช้เจนเนอเรเตอร์แมท- ทริกซ์ของสมการ (3.4) ซึ่งจะได้โค๊ดเวอดเป็น

$$u = (u_1, u_2, u_3, ..., u_n)$$

$$= (m_1, m_2, ..., m_k)$$

$$= (m_1, m_2, ..., m_k) \begin{bmatrix} 1\ 0\ 0\ 0 \ldots 0 & p_{11}\ p_{12} \cdots p_{1k} \\ 0\ 1\ 0\ 0 \ldots 0 & p_{21}\ p_{22} \cdots p_{2k} \\ M & M \\ 0\ 0\ 0\ 0 \ldots 0 & p_{k1}\ p_{k2} \cdots p_{k,n-k} \end{bmatrix} \quad (3.5)$$

จากการคูณของแมททริกซ์จะได้

$$u_i = m_i \quad \text{สำหรับ } i = 1,2,....,k \quad .....(3.6a)$$

และ

$$u_{k-j} = p_{1j}\, m_1 + p_{2j}\, m_2 + ... + p_{kj}\, m_k \quad .....(3.6b)$$

สำหรับ j = 1,2, ...n-k จากสมการที่ (3.6a) และ (3.6b) จะพบว่ารหัส k บิทแรกของโค๊ดเวอดคือรหัสของข่าวสารที่ต้องการจะส่ง ส่วน (n-k) บิทหลังเป็นฟังก์ชั่นเชิงเส้นของรหัสข่าวสาร ซึ่งเรียกว่ารหัสแก้ไข n-k บิทของ u หรือรหัสพารีตี้เช็ค (parity check degits) ของโค๊ดเวอด สมการที่ (3.6b) จึงเรียกว่าสมการพารีตี้ของรหัส

ตัวอย่างที่ 3.3 จากเจนเนอเรเตอร์แมททริกซ์ที่ให้ไว้ในตัวอย่างที่ 3.2 โค๊ดเวอดที่สอดคล้องกับบล็อคของรหัสข่าวสาร (m1 m2 m3) คือ

$$u = (u_1, u_2, u_3, u_4, u_5, u_6)$$

$$= (m_1\ m_2\ m_3) \begin{matrix} 1\ 0\ 0\ 1\ 1\ 0 \\ 0\ 1\ 0\ 0\ 1\ 1 \\ 0\ 0\ 1\ 1\ 0\ 1 \end{matrix}$$

$$= (m_1, m_2, m_3, m_1 + m_3, m_1 + m_2, m_2 + m_3)$$

ดังนั้น $u_1 = m_1, u_2 = m_1, u_3 = m_3$ และ

$u_4 = m_1 + m_3$

$u_5 = m_1 + m_2$

$u_6 = m_2 + m_3$

สำหรับรหัสเชิงเส้นในรูปแบบรหัสระบบนั้น ความยุ่งยากในการเข้ารหัสสามารถลดขนาดหน่วยความจำได้โดยเพียงแต่เก็บรหัสข้อมูล p_{ij} ของแมททริกซ์ p ขนาด k x (n-k) แทนที่จะเก็บเจนเนอเรเตอร์แมททริกซ์ G ขนาด k x n ไว้

ตัวอย่างที่ 6.4 ถ้ามีเจนเนอเรเตอร์แมททริกซ์ G ของบล็อคโค๊ด ให้หาเวอดทั้งหมด

$$G = \begin{matrix} 1\ 0\ 0\ 0\ 1\ 1 \\ 0\ 1\ 0\ 1\ 0\ 1 \\ 0\ 0\ 1\ 1\ 1\ 0 \end{matrix}$$

เนื่องจากแต่ละบล็อคของรหัสข่าวสารมีขนาด k = 3 จะให้รหัสข่าวสาร 8 ชุดที่แตกต่างกัน ดังนี้ (0 0 0),(0 0 1),(0 1 0),(0 1 1),(1 0 0),(1 0 1),(1 1 0)และ(1 1 1) การคำนวณหาโค๊ดเวอดของ (1 1 1) ทำได้โดย

$$C = mG$$

$$= (1\ 1\ 1) \begin{matrix} 1\ 0\ 0\ 0\ 1\ 1 \\ 0\ 1\ 0\ 1\ 0\ 1 \\ 0\ 0\ 1\ 1\ 1\ 0 \end{matrix}$$

$$= (1\ 1\ 1\ 0\ 0\ 0)$$

ดังนั้นโค๊ดเวอดทั้งหมดคือ

รหัสข่าวสาร	โค๊ดเวคเตอร์
0 0 0	0 0 0 0 0 0
0 0 1	0 0 1 1 1 0

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

3.3 แมททริกซ์ในการตรวจสอบพารีตี้ (Parity check metrix)

ดังที่ได้กล่าวไว้แล้วในบทที่ 2 ว่าแมททริกซ์ G ขนาด k x n จะมีแมททริกซ์ H ขนาด (k-n)x n ซึ่งโรว์สแปซของ G จะตั้งฉากอยู่กับ H อินเนอร์โพรดัคท์ของเวคเตอร์ในโรว์สแปซของ G กับ แถวของ H จะเป็นศูนย์

$$H = \begin{matrix} h_1 \\ h_2 \\ M \\ h_{n-k} \end{matrix} = \begin{matrix} h_{11} & h_{12} & .... & h_{1n} \\ h_{21} & h_{22} & .... & h_{2n} \\ M & & & \\ h_{n-k,1} & h_{n-k,2} & .... & h_{n-k,n} \end{matrix} \qquad \text{.....(3.7)}$$

และให้ $\underline{u} = (u_1, u_2, ..., u_n)$ เป็นเวคเตอร์ในโรว์แปซของ G จะได้

$$\underline{u}H^T = (0\ 0\ .....\ 0) \qquad \text{.....(3.8)}$$

หรือ

$$\underline{u}hi = u1\ hi1 + u2\ hi2 + ... + un\ hin = 0 \qquad \text{.....(3.9)}$$

สำหรับ i = 1,2.....,n-k จึงสรุปได้ว่า $\underline{u}$ จะเป็นโค๊ดเวอดที่ได้จาก G ถ้าและเพียงแต่ถ้า $\underline{u}H^T$ =0 แมททริกซ์ H นี้เรียกว่าแมททริกซ์ในการตรวจสอบพารีตี้ หรือเรียกย่อว่าพารีตี้แมททริกซ์ถ้าเจนเนอเรเตอร์แมททริกซ์ของรหัสระบบได้มาจากสมการที่ (3.4) พารีตี้แมททริกซ์ของรหัสคือ

$$H = \begin{bmatrix} p_{11} & p_{21} & \cdots\cdots & p_{k1} & 1 & 0 & 0 & \cdots\cdots & 0 \\ p_{12} & p_{22} & \cdots\cdots & p_{k2} & 0 & 1 & 0 & \cdots\cdots & 0 \\ M & & & & & M & & & \\ p_{1,n-k} & p_{2,n-k} & \cdots & p_{k,n-k} & 0 & 0 & 0 & \cdots\cdots & 1 \end{bmatrix} \quad \text{.....(6.10)}$$

$$H = [p^T \; I_{n-k}]$$

P^T เป็นทรานสโพซ (transpose) ของแมททริกซ์ P สมการของพารีตี้ (3.6b)ได้จากแมททริกซ์ H นั้นคือ $\underline{u} = (u^1, u^2, \ldots, u^n)$ เป็นโค๊ดเวอดของรหัสข้อมูล $\underline{m} = (m^1, m^2, \ldots, m^k)$เมื่อ $u^i = m^i$ สำหรับ i = 1,2, ...,k แต่

$$\underline{u}H^T = 0$$

จะได้ว่า

$$u_{k-j} = p_{1j}\, u_1 + p_{2j}\, u_2 + \ldots\ldots + p_{kj}\, u_k$$
$$= p_{1j}\, m_1 + p_{2j}\, m_2 + \ldots\ldots + p_{kj}\, m_k \quad \text{.....(3.11)}$$

เมื่อ j = 1,2 ,....,n-k ซึ่งสมการข้างบนเป็นสมการเดียวกันกับสมการที่ (3.6b) ในการออกแบบรหัสเชิงเส้นนั้นแมททริกซ์ p จะถูกเลือกเพื่อให้มีคุณสมบัติในการแก้บิทที่ผิด

จากเจนเนอเรเตอร์แมททริกซ์ G ของรหัสเชิงเส้น (n,k) และมีพารีตี้แมททริกซ์ H ให้ $\underline{u}$ เป็นโค๊ดเวคเตอร์ที่ใช้ส่งผ่านช่องส่งที่มีสัญญาณรบกวน ทางด้านรับจะได้รับสัญญาณเวคเตอร์ $\underline{r}$ ซึ่งเป็นเวคเตอร์รวมระหว่างรหัสกำเนิด $\underline{u}$ และเวคเตอร์ของรหัสบิทที่ผิด $\underline{e}$ นั้นคือ

$$\underline{r} = \underline{u} + \underline{e} \quad \text{.....(3.12)}$$

ทางด้านรับจะไม่รู้ว่า $\underline{u}$ และ $\underline{e}$ คืออะไร จุดประสงค์ของตัวถอดรหัสคือจะต้องนำเอาเวคเตอร์ $\underline{u}$ คืนมาจากเวคเตอร์ $\underline{r}$ โดยทางด้านรับจะหารหัสบิทที่ผิดได้จากการคำนวณหาซีโดรม (syndrom) $\underline{s}$ ซึ่งเป็นเวคเตอร์ขนาด (n-k)

$$\underline{s} = \underline{r}\, H^T \quad \text{.....(3.13)}$$

จากสมการที่ (3.8) ถ้าหากว่าเวคเตอร์ $\underline{r}$ ที่รับเข้ามาเป็นโค๊ดเวคเตอร์แล้ว เวคเตอร์ซีนโดรมจะเป็นศูนย์ จากสมการ (3.13) เขียนใหม่ได้เป็น

$$\underline{s} = (\underline{u} + \underline{e})\, H$$

$$\underline{s} = \underline{u}H^T + \underline{e}H^T$$

$$\underline{s} = \underline{e}H^T$$

เนื่องจาก $\underline{u}H^T = 0$ นั้นคือซินโดรมที่เกิดจากเวคเตอร์ $\underline{r}$ ที่รับเข้ามาจะเป็นศูนย์ถ้า $\underline{r}$ เป็นโค๊ดเวคเตอร์ที่ส่งมา ถ้าหากเกิดความผิดพลาดในการส่งดังนั้นซินโดรม $\underline{s}$ ของเวคเตอร์ที่รับเข้ามาจะไม่เป็นศูนย์ยิ่งไปกว่านั้น $\underline{s}$ จะมีความสัมพันธ์กับ $\underline{e}$ ตัวถอดรหัสจะใช้ $\underline{s}$ ในการตรวจจับและแก้บิทที่ผิดไปซึ่งรายละเอียดจะได้กล่าวต่อไป แต่ตอนนี้จะพิจารณาและทำความเข้าใจกับสัญญาลักษณ์ และวิธีการที่กล่าวมา

ตัวอย่างที่ 3.5 พิจารณาถึงบล็อคโค๊ด (7,4) ที่ได้จากเจนเนอเรเตอร์แมททริกซ์ G

$$G = \begin{matrix} & \overbrace{\quad}^{I_4} & & \overbrace{\quad}^{P} \\ \end{matrix}$$

$$G = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

แต่จาก

$$H = [P^T \; I_{n-k}]$$

นั้นคือ

$$H = \left[\begin{array}{ccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

ถ้ามีรหัสข่าวสาร $\underline{m} = (1\ 0\ 1\ 1)$ จะได้โค๊ดเวคเตอร์ $\underline{u}$ เป็น

$$\underline{u} = \underline{m}G$$
$$= (1\ 0\ 1\ 1\ 0\ 0\ 0\ 1)$$

สำหรับโค๊ดเวคเตอร์นี้จะให้ซินโดรม $\underline{s}$ เป็น

$$\underline{s} = \underline{u}H^T$$
$$= (0\ 0\ 0)$$

ถ้าหากว่าบิทที่ 3 ของ $\underline{u}$ เกิดผิดไปเนื่องจากสัญญาณรบกวนในช่องส่ง ดังนั้นทางด้านรับจะได้เวคเตอร์ r เป็น

$$\begin{aligned} r &= \underline{u} + \underline{e} \\ &= (1\ 0\ 1\ 1\ 0\ 0\ 1) + (0\ 0\ 1\ 0\ 0\ 0\ 0) \\ &= (1\ 0\ 0\ 1\ 0\ 0\ 1) \end{aligned}$$

และซีนโดรมที่ได้คือ

$$s = r H^T = (1\ 0\ 1)$$

3.4 ความสามารถในการตรวจแก้บิทที่ผิดของรหัสเชิงเส้น

ในหัวข้อนี้จะได้กล่าวถึงคำศัพท์พื้นฐานที่ใช้ในการแก้บิทที่ผิดของรหัสเชิงเส้น

นิยามที่ 3.2 เวท (weight) ของแฮมมิ่งสำหรับเวคเตอร์ $\underline{v}$ n-ทูเปิ้ลส์คือ $w(\underline{v})$ ซึ่งหมายถึงจำนวนบิทของรหัสของ $\underline{v}$ ที่ไม่เป็นศูนย์ ตัวอย่างเช่นถ้า $\underline{v} = (1\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 1)$ จะได้ $w(\underline{v}) = 5$

นิยามที่ 3.3 ให้ $\underline{u}$ และ $\underline{v}$ เป็นเวคเตอร์ n-ทูเปิ้ลส์ ค่าระยะห่างแบบแฮมมิ่งระหว่าง $\underline{u}$ และ $\underline{v}$ เขียนได้เป็น $d(\underline{u},\underline{v})$ คือจำนวนบิทรหัส "1" ที่แตกต่างระหว่างเวคเตอร์ทั้งสองเช่น

$$\underline{u} = (1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 1)$$

$$\underline{v} = (1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1)$$

จะได้ $d(\underline{u}, \underline{v}) = 5$

จากคำจำกัดความของการบวกแบบโมดูโล-2 จะเขียนใหม่ได้ว่า

$$d(\underline{u}, \underline{v}) = w(\underline{u} + \underline{v}) \qquad \text{.....(3.14)}$$

นั้นก็หมายความว่าระยะห่างระหว่างเวคเตอร์ทั้งสองคือเวทของผลรวมของเวคเตอร์ทั้งสองเมื่อพิจารณาจากตัวอย่างข้างบนจะได้

$$\underline{u} + \underline{v} = (0\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 0)$$

และ $w(\underline{u} + \underline{v}) = 5$ หรือว่า $d(\underline{u} + \underline{v}) = 5$

สำหรับรหัสเชิงเส้นในการหาระยะห่างของแต่ละคู่ของโค๊ดเวอด ระยะห่างที่ต่ำที่สุดเขียนย่อเป็น dmin ถ้า $\underline{u}$ และ $\underline{v}$ เป็นโค๊ดเวคเตอร์ 2 ชุดของรหัสเชิงเส้นดังนั้น $\underline{u} + \underline{v}$ ก็ยังคงเป็นโค๊ดเวคเตอร์เพราะเซทของทุกโค๊ดเวคเตอร์เป็นซับสแปซของทุก n-ทูเปิ้ลส์ ดังนั้น จากคำนิยามที่ว่าระยะห่างระหว่างโค๊ดเวคเตอร์ทั้งสองคือเวทของโค๊ดเวคเตอร์ที่ 3 ก็จะได้ระยะห่างต่ำสุดของ

ของรหัสเชิงเส้นเท่ากับเวทต่ำสุดของโค๊ดเวคเตอร์ที่ไม้เป็นศูนย์ ค่าระยะ ห่าง ต่ำสุดและเวทต่ำสุดจะเป็นตัวกำหนดความสามารถในการแก้รหัสบิทที่ผิดของรหัสเชิงเส้น

ในระบบการสื่อสารบางชนิดรหัสที่ผิดเกิดขึ้นอย่างอิสระต่อกัน ความผิดพลาดที่เกิดขึ้นอย่างนี้เรียกกันว่าความผิดพลาดแบบแรนดอม รหัสที่ผิดอีกอย่างที่เกิดขึ้นไม่จำกัดช่วงเวลา รหัสผิดที่เกิดขึ้นจะรวมตัวเข้าด้วยกันเป็นเบอสท์ (burst) รหัสที่ออกแบบไว้แก้ความผิดพลาดแบบนี้เรียกว่ารหัสอดแก้รหัสที่ผิดแบบเบอสท์ ในบทนี้จะพูดเฉพาะเรื่องการแก้รหัสที่ผิดแบบเรนดอม

เมื่อพิจารณาจากรหัสที่ใช้สำหรับช่องส่งแบบ BSC (binary systematic channel) โดยให้ $\underline{v} = (v_1, v_2, \ldots, v_n)$ เป็นโค๊ดเวคเตอร์ที่ใช้ส่งผ่านช่องส่งและให้ $\underline{r} = (r_1, r_2, \ldots, r_n)$ เป็นเวคเตอร์ที่รับได้จากช่องส่ง แต่เนื่องจากภายในช่องส่งมีสัญญาณรบกวนอยู่ ดังนั้นเวคเตอร์ $\underline{r}$ ที่รับได้จะเป็นเวคเตอร์อะไรก็ได้ใน 2_n เวคเตอร์ของ n-ทูเปิ้ลส์ ความแตกต่างระหว่าง $\underline{r}$ และ $\underline{v}$ คือ $\underline{e}$

$$\begin{aligned}\underline{e} &= (e_1, e_2, \ldots, e_n) \\ &= -\underline{r} + \underline{v} \\ &= -(r_1, r_2, \ldots, r_n) + (v_1, v_2, \ldots, v_n) \\ &= (r_1 + v_1, r_2 + v_2, \ldots, r_n + v_n)\end{aligned}$$

ซึ่ง $\underline{e}$ เป็นรูปแบบของรหัสที่ผิด (error pattern หรือ error vector) อันเกิดจากการถูกรบกวน ในช่องส่งสัญญาณ เมื่อ $e_i = r_i + v_i = 1$ ก็หมายความว่าโค๊ดเวคเตอร์ที่ส่งมาถูกรบกวนที่ตำแหน่งบิทที่ i แต่เนื่องจากในหนึ่งโค๊ดเวคเตอร์มีรหัสอยู่ n บิทจึงทำให้เกิดความผิดพลาด 2^n แบบที่แตกต่างกันใน BSC รูปแบบของรหัสที่ผิดที่มีบิทที่ผิดจำนวนมาก

ที่ทางด้านรับตัวถอดรหัสมีหน้าที่ในการตรวจหาโค๊ดเวคเตอร์ที่ส่งจากโค๊ดเวคเตอร์ $\underline{r}$ ที่รับได้ สำหรับการถอดรหัสโดยใช้วิธีแม็กซิมั่มไลค์รีฮูดกับ BSC นั้นตัวถอดรหัสจะตรวจสอบว่า $\underline{v}$ เป็นเวคเตอร์ที่ใช้ส่งซึ่งจะมีค่าเข้าใกล้เวคเตอร์ $\underline{r}$ โดยอาศัยการดูจากระยะห่างของแฮมมิ่ง (นั่นคือ $d(\underline{v},\underline{r})$ จะมีค่าต่ำสุด) ตัวถอดรหัสสามารถแก้รหัสที่ผิดอย่างแรนดอมจำนวน t บิทในโค๊ดเวคเตอร์ที่รับเข้ามาโดยที่ $2t + 2 \geq d_{min} \geq 2t + 1$ ตัวถอดรหัสสามารถแก้ทุกรูปแบบของรหัสบิทที่ผิดไป t บิทจากเวคเตอร์ r ที่รับได้ ซึ่งแสดงให้เห็นได้ดังต่อไปนี้ ให้ $\underline{v}$ เป็นโค๊ดเวคเตอร์ที่ต้องการส่งและ $\underline{u}$ เป็นโค๊ดเวคเตอร์ใด ๆ ระยะห่างของแฮมมิ่งระหว่าง $\underline{v},\underline{u}$ และ $\underline{r}$ จะต้องเป็นไปตามสมการข้างล่างนี้

$$d(\underline{v}, \underline{r}) + d(\underline{u}, \underline{r}) \geq d(\underline{u}, \underline{v}) \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3.15)$$

ถ้าเกิดรหัสผิดไป t บิท ($t \leq t$) ดังนั้นระยะห่างของแฮมมิ่งระหว่างโค๊ดเวคเตอร์ที่ส่ง $\underline{v}$ กับโค๊ดเวคเตอร์ที่รับ $\underline{r}$ คือ $d(\underline{v}, \underline{r}) = t$ แต่ $d(\underline{v}, \underline{u}) \geq d_{min} \geq 2t + 1$ สมการที่ (3.15) จะให้

$$d(\underline{u}, \underline{r}) \geq 2t + 1 - t'$$

$$d(\underline{u}, \underline{r}) \geq t + 1 \qquad \text{.....(3.16)}$$

$$d(\underline{u}, \underline{r}) \geq t'$$

จากสมการที่ (3.16) แสดงให้เห็นว่าถ้ารูปแบบของรหัสบิทที่ผิดมีรหัสผิดไป t บิทหรือน้อยกว่า เวคเตอร์ $\underline{r}$ ที่รับได้จะเข้าใกล้โค๊ดเวคเตอร์ $\underline{v}$ กว่าโค๊ดเวคเตอร์ $\underline{u}$ ดังนั้นตัวถอดรหัสจะสามารถถอดรหัสได้ถูกต้องความคลาดเคลื่อนก็จะถูกแก้ไข ตัวถอดรหัสไม่สามารถจะแก้ทุกรูปแบบของรูหัสที่ผิดไป λ บิท เมื่อ $\lambda \geq t + 1$ โดยปกติแล้วการแก้รหัสผิดของโค๊ดเชิงเส้นจะทำได้เมื่อ

$$t = (d_{min} - 1) / 2$$

โดย $(d_{min} - 1) / 2$ เป็นค่าจำนวนเต็มที่ไม่คิดค่าทศนิยม และสามารถตรวจสอบรหัสผิดที่เกิดขึ้นถึง $(d_{min} - 1)$ บิทในแต่ละโค๊ดเวอด

ตัวอย่างที่ 3.6 พิจารณาจากรหัสในตัวอย่างที่ 3.1 จะมีระยะห่างต่ำสุด = 3 ดังนั้นความสามารถในการแก้รหัสผิดไม่เกิน 1 บิท ทุกรูปแบบของรหัสที่ผิดไป 1 บิทสามารถที่จะแก้ไขกับรหัสนั้นได้ แต่จะไม่สามารถแก้ไขรหัสที่ผิดไป 2 บิททางด้านรับได้รับโค๊ดเวคเตอร์เป็น (0 0 1 0 0 1) โค๊ดที่รับได้จะผิดไปจากโค๊ดเวคเตอร์ (0 0 1 1 0 1) เพียงตำแหน่งเดียว ก็หมายความว่าโค๊ดที่ใกล้เคียงกับโค๊ดที่รับได้คือ (0 0 1 1 0 1) ตัวถอดรหัสจะแก้โค๊ดที่รับได้เป็น (0 0 1 1 0 1) ซึ่งไม่ใช้โค๊ดที่ส่งมา

เมื่อพิจารณาถึงความสามารถในการตรวจจับรหัสที่ผิดของรหัสที่มีระยะห่างต่ำสุดเท่ากับ d_{min} ตัวถอดรหัสจะตรวจจับเซทของรูปแบบของรหัสที่ผิดที่มีจำนวนบิทที่ผิด $\leq (d_{min} - 1)$ บิทในแต่ละโค๊ดเวคเตอร์ หรือพูดอีกนัยหนึ่งว่าถ้า λเป็นจำนวนบิทที่ผิด ในแต่ละโค๊ดเวคเตอร์ และ $\lambda \geq d_{min}$ แล้วจะทำการตรวจจับรหัสที่ผิดไม่ได้ เช่นสมมติว่ามีโค๊ดเวคเตอร์อยู่คู่หนึ่งคือ $\underline{u}$ และ $\underline{v}$ โดยจะให้ระยะห่างของแฮมมิ่ง $d(\underline{u},\underline{v}) = \lambda$ เมื่อพิจารณาถึงรูปแบบรหัสที่ผิด $\underline{e} = \underline{u} + \underline{v}$ โดยรูปแบบของรหัสที่ผิดจะเปลี่ยนแปลงให้ $\underline{u}$ เป็น $\underline{v}$ และ $\underline{v}$ กลายเป็น $\underline{u}$ ถ้า $\underline{u}$ เป็นโค๊ดเวคเตอร์ที่ส่งออกมาและ $\underline{e}$ เป็นรูปแบบของรหัสที่ผิด และ $\underline{v}$ เป็นเวคเตอร์ที่รับเข้ามา แต่ $\underline{v}$ ก็เป็นโค๊ดเวคเตอร์ จึงทำให้ตัวถอดรหัสไม่พบรหัสที่ผิดและยอมรับ $\underline{v}$ เป็นเวคเตอร์ที่ส่งมา นั้นคือไม่สามารถตรวจจับรูปแบบของ $\underline{e}$

3.5 การแก้รหัสที่ผิดไปบิทเดียวของโค๊ดแบบแฮมมิ่ง

จากการเกิดรหัสที่ผิไป t บิท $\leq [(d_{min} - 1) / 2]$ ถ้าหาก $d_{min} = 3$ การแก้รหัสที่ผิดของบล็อคโค๊ดเชิงเส้นทำได้ต่อเมื่อมีรหัสผิดไปเพียงบิทเดียวในโค๊ดเวคเตอร์นั้นรหัสที่ผิดไปเพียงบิทเดียวนีสามารถบอกได้ว่าเกิดที่ตำแหน่งไหนของโค๊ดเวคเตอร์โดยใช้การคำนวณหาซินโดรมของ

$$H^T = \left[\begin{array}{c} P \\ \hline I_{n-k} \end{array}\right]$$

$$= \begin{matrix} 1100 \\ 0110 \\ 0011 \\ 1001 \\ 1010 \\ 0101 \\ 1110 \\ 0111 \\ 1000 \\ 0100 \\ 0010 \\ 0001 \end{matrix}$$

จะได้เจนเนอเรเตอร์แมททริกซ์ G เป็น

$$G = [I_k \;\; P] = \begin{matrix} 10000000 & 1100 \\ 01000000 & 0110 \\ 00100000 & 0011 \\ 00010000 & 1001 \\ 00001000 & 1010 \\ 00000100 & 0101 \\ 00000010 & 1110 \\ 00000001 & 0111 \end{matrix}$$

ทางด้านรับจะได้ซินโดรม $s = r H^T$ เมื่อ r คือโค๊ดเวคเตอร์ที่รับได้และจะยอมรับว่าเป็นโค๊ดเวคเตอร์ที่ส่งมา ถ้า $s = 0$ หมายความว่า $r = u_i$ และ $\min d(r,u_j) = d(r,u_i)$ เมื่อ $u_1, u_2, \ldots, u_{2k}$ เป็นโค๊ดเวคเตอร์ของรหัส ดังนั้นการเกิดรหัสผิเพียงบิทเดียวในโค๊ดเวคเตอร์สามารถแก้ได้ แต่ถ้าเป็นการผิดไป 2 บิทจะไม่สามารถแก้ได้

ประสิทธิภาพของบล็อคโค๊ดที่มีระยะห่างต่ำสุดเป็น 3 จะดียิ่งขึ้นถ้าขนาดของบล็อคข่าวสารเพิ่ม เช่น ถ้า k = 64 จะให้ประสิทธิภาพเท่ากับ 0.90 และถ้า k = 256 จะได้ประสิทธิภาพเป็น 0.97

การทำงานของตัวถอดรหัสคือจะหาโค๊ดเวคเตอร์ $\underline{u}_i$ ที่ใกล้เคียงกับ $\underline{r}$ ที่สุดซึ่งทำได้โดยการเก็บ 2^k โค๊ดเวคเตอร์ไว้ในหน่วยความจำของตัวถอดรหัส จากนั้นก็ทำการเปรียบเทียบ $\underline{r}$ กับโค๊ดเวคเตอร์ทั้งหมด เนื่องจากแต่ละโค๊ดเวคเตอร์มีขนาด n บิทต้องใช้หน่วยความจำถึง $n.2^k$ บิท ถ้าค่า n สูงมาก ๆ จะทำให้สิ้นเปลืองหน่วยความจำอีกทั้งยังต้องใช้เวลาในการเปรียบเทียบหารหัส

3.6 ตารางถอดรหัสที่ใช้อะเรมาตรฐาน (standard array)

สมมติว่ามี (n,k) ของโค๊ดเชิงเส้นโดยมี $\underline{u}_1, \underline{u}_2, \underline{u}_{2k}$ เป็นโค๊ดเวคเตอร์ทั้งหลายของ $\underline{u}$ ตัวถอดรหัสมีหน้าที่ในการหาว่าโค๊ดเวคเตอร์ $\underline{r}$ ที่รับได้เป็นอะไรในจำนวนโค๊ดเวคเตอร์ 2^k ชุด หน้าที่อีกอย่างของตัวถอดรหัสคือจะแบ่ง 2^n ของ n-ทูเปิ้ลส์เป็น 2^k disjorint set T_1 , T_2 ,....., T_{2k} โดยแต่ละซับเซทจะมีเพียงโค๊ดเวคเตอร์ $\underline{u}_i$ ชุด เดียวถ้า $\underline{r} \in T_i$ ตัวถอดรหัสจะให้ว่า $\underline{u}_i$ ถูกส่งมา ถ้าหากว่า $\underline{e}$ เป็นเวคเตอร์ที่ผิดไป การตรวจแก้จะถูกต้องเมื่อ $(\underline{u}_i + \underline{e}) \in T_i$ และจะไม่ถูกต้องเมื่อ $(\underline{u}_i + \underline{e}) \notin T_i$

วิธีการหนึ่งของการทำการแยกของเซทของ 2^n n-ทูเปิ้ลส์ ดังแสดงตามตารางข้างล่างโดยโค๊ดเวคเตอร์ต่าง ๆ คือ $\underline{u}_1, \underline{u}_2, ..., \underline{u}_{2k}$ ถูกเขียนอยู่บนแถวแรกที่รวมอยู่กับโค๊ดเวคเตอร์ที่เป็นศูนย์หมดที่ตำแหน่งซ้ายมือสุด $\underline{u}_1$ กับ $\underline{e}_1$ ต่างก็เป็นเวคเตอร์ศูนย์ อีลีเมนท์แรกของแถวที่สองคือ $\underline{e}_2$ ซึ่งเป็นเวคเตอร์หนึ่งใน $(2^n - 2^k)$ n-ทูเปิ้ลส์ และ $\underline{e}_2$ จะบวกกับทุกโค๊ดเวคเตอร์จะได้แถวที่ 2 ส่วน $\underline{e}_3$ ก็จะบวกกับ $\underline{u}_i$ $(i = 1,2,...,2^k)$ ได้เป็นแถวที่ 3 ซึ่งเป็นอย่างนี้ไปเรื่อย ๆ จนถึง $\underline{e}_{2^{n-k}}$ ก็จะได้อะเรมาตรฐานสำหรับโค๊ดของ 2^k คอลัมน์ แต่ละคอลัมน์จะมี 2^{n-k} n-ทั้เปิ้ลส์ โดยที่แถวบนสุดของทุก ๆ n-ทู่เปิ้ลส์คือโค๊ดเวคเตอร์ ดังนั้นคอลัมน์ที่ j คือ แยก T_j ที่ถูกใช้สำหรับการถอดรหัส แถวของอะเรมาตรฐานเรียกว่าโคเซท (coset) และอีลีเมนท์แรกของแต่ละแถวเรียกวาโคเซทลีดเดอร์ (coset leader)

ตารางของอะเรมาตรฐานสำหรับบล็อคโค๊ดเชิงเส้น (n,k) คือ

$\underline{u}_1$	$\underline{u}_2$	$\underline{u}_3$	..... $\underline{u}_{2^k}$
$\underline{e}_2$	$\underline{u}_2 + \underline{e}_2$	$\underline{u}_3 + \underline{e}_2$	$\underline{u}_{2^k} + \underline{e}_2$
$\underline{e}_3$	$\underline{u}_2 + \underline{e}_3$	$\underline{u}_3 + \underline{e}_2$	$\underline{u}_{2^k} + \underline{e}_3$
M	M	M	M
$\underline{e}_{2^{n-k}}$	$\underline{u}_2 + \underline{e}_{2^{n-k}}$	$\underline{u}_3 + \underline{e}_{2^{n-k}}$	$\underline{u}_{2^k} \underline{e}_{2^{n-k}}$

คุณสมบัติของอะเรมาตรฐาน

1.แต่ละอีลีเมนท์ในอะเรมาตรฐนจะแตกต่างกัน จึงเป็นเหตุให้คอลัมน์ของอะเรมาตรฐาน T_j เป็น disjoint

2.ถ้ารูปแบบของรหัสที่ผิดเกิดจากช่องส่งสัญาณตรงกับโคเซทลีดเดอร์ ดังนั้นเวคเตอร์ที่รับมาจะถูกถอดรหัสอย่างถูกต้อง ในทางตรงกันข้ามถ้ารูปแบบของรหัสที่ผิดไม่เป็นโคเซทลีดเดอร์ การถอดรหัสจะทำได้ไม่ถูกต้อง ดังนั้นโคเซทลีดเดอร์จึงเรียกอีกชื่อ correctable error pattern

ในการลดความน่าจะเป็นไปได้ของการถอดรหัสที่ไม่ถูกต้องนั้น โคเซทลีดเดอร์ 2^{n-k} จะถูกเลือกเพื่อให้รูปแบบของรหัสที่ผิดคล้ายกับความผิดพลาดที่เกิดในช่องส่งสัญญาณมากที่สุด ถ้า e_i และ e_j เป็นรูปแบบของรหัสที่ผิดที่มีเวท w_i และ w_j ดังนั้นรหัสที่ผิดแบบแรนดอม e_i น่าจะเกิดขึ้นมากกว่า e_j ถ้า $w_j > w_i$ ในการสร้างอะเรมาตรฐานควรจะเลือกโคเซทลีดเดอร์ที่มีเวทต่ำสุกจากเวคเตอร์ทั้งหลายที่มีอยู่

ตัวอย่างที่ 3.8 ให้สร้างอะเรมาตรฐานของ 6,3) บล็อคโค้ดเชิงเส้นที่มีเจนเนอเรเตอร์แมททริกซ์

$$G = \begin{matrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{matrix}$$

วิธีทำ โค้ดเวคเตอร์ที่เป็นไปได้คือ

รหัสข่าวสาร	โค้ดเวคเตอร์
0 0 0	0 0 0 0 0 0
1 0 0	1 0 0 1 1 0
0 1 0	0 1 0 0 1 1
0 0 1	0 0 1 1 0 1
1 1 0	1 1 0 1 0 1
0 1 1	0 1 1 1 1 0
1 0 1	1 0 1 0 1 1
1 1 1	1 1 1 0 0 0

ตารางอะเรมาตรฐานเป็น

โคเซท ลีดเดอร์							
000000	001110	010101	100011	011011	101101	110110	111000
000001	001111	010100	100010	011010	101100	110100	111010
000010	001100	010111	100001	011001	101111	110100	111010
000100	001010	010001	100111	011111	101001	110010	111100
001000	000110	011101	101011	010011	100101	111110	110000
010000	011110	000101	110011	001011	111101	100110	101000
100000	101110	110101	000011	111011	001101	010110	011000
001001	000111	011100	101010	010010	100100	111111	110001

ถ้าอะเรมาตรฐานใช้สำหรับการถอดรหัสดังแสดงในตารางข้างบน ดังนั้นสัญญาณของตัวถอดรหัสจะเป็นโค๊ดเวคเตอร์ที่ i ถ้าเวคเตอร์ที่รับเข้ามาตรงกับเวคเตอร์ใดเวคเตอร์หนึ่งในคอลัมน์ที่ i ตัวอย่างเช่นถ้าเวคเตอร์ที่รับเข้ามาคือ (0 1 0 1 0 0) จะถูกถอดรหัสเป็น (1 1 0 1 0 1) และเวคเตอร์ที่รับได้เป็น (1 1 1 1 0 0) จะถูกถอดรหัสเป็น (1 1 1 0 0 0)

ทฤษฎีบททุก 2^k n-ทูเปิ้ลส์ของโคเซทมีซินโดรมอย่างเดียวกันและซินโดรมสำหรับโคเซทที่ต่างกันจะต่างกัน

พิสูจน์ พิจารณาจากโคเซทลีดเดอร์ที่ j คือ $\underline{e}_j$ ดังนั้น n-ทูเปิ้ลส์ในโคเซทคือ $\underline{e}j + \underline{v}j$ สำหรับ i ใด ๆ ซินโดรมของ n-ทูเปิ้ลส์นี้คือ

$$[\underline{e}_j + \underline{v}_i] H^T = \underline{e}_j H^T$$

เนื่องจาก $\underline{v}_i H^T = 0$ ดังนั้นซินโดรมของ n-ทูเปิ้ลส์ใด ๆ ในโคเซทจะเท่ากัน

พิสูจน์ข้อที่ว่าซินโดรมสำหรับค่าโคเซทที่ต่างกันจะต่างกัน เช่น ซินโดรมที่แถวที่ i และที่ j ($i < j$) ของโคเซทที่ต่างกันเมื่อเท่ากันจะได้ว่า

$$\underline{e}_i HT = \underline{e}_j H^T$$

หรือ

$$(\underline{e}_i + \underline{e}_j) H^T = 0$$

โดยที่ H^T ไม่เท่ากับศูนย์แล้วจะให้ $(\underline{e}_i + \underline{e}_j) = 0$ แต่การบวกของ $\underline{e}i$ กับ $\underline{e}j$ เป็นแบบโมดูโล-2 ดังนั้นจะได้ว่า $\underline{e}i = \underline{e}j$ จึงทำให้ได้ผลบวกโมดูโล -2ของทั้งสองเวคเตอร์เป็นศูนย์ ซึ่งจะขัดกับกฎในการสร้างอะเรมาตรฐาน จึงสรุปได้ว่าถ้า $i* *j$ แล้ว $\underline{e}_i H^T \underline{e}_j H^T$

ซีนโดรมของ n-ทูเปิ้ลส์คือ (n-k) ทูเปิ้ลส์มี 2^{n-k} ของ (n-k) ทูเปิ้ลส์ที่แตกต่างกันเนื่องจากมี 2^{n-k} โคเซท ดังนั้นจากทฤษฎีที่กล่าวมาข้างต้นจะเกิดความสอดคล้องหนึ่งต่อหนึ่งระหว่างโคเซทหนึ่งกับ (n-k) ทูเปิ้ลส์ของซีนโดรมตัวหนึ่งหรือจะมีความสอดคล้องหนึ่งต่อหนึ่งระหว่างหนึ่งโคเซทลีดเดอร์กับหนึ่งซีนโดรม เราจึงสามารถสร้างตารางของการถอดรหัสที่ง่ายกว่าอะเรมาตรฐาน ตารางนี้จะประกอบด้วยด้วย 2^{n-k} โคเซทลีดเดอร์ และซีนโดรมที่เกี่ยวข้องการทำการถอดรหัสของแวคเตอร์ที่รับเข้ามาจะแบ่งเป็น 3 ขั้นตอนคือ

1. คำนวณหาซีนโดรม $\underline{s} = \underline{r} H^T$
2. หาโคเซทลีดเดอร์ $\underline{e}_i$ ที่ให้ซีนโดรมเท่ากับ $\underline{r}H^T$ และประมาณว่า $\underline{e}_i$ เป็นรูปแบบของรหัสที่ผิดอันเนื่องจากสัญญาณรบกวนในช่องส่ง
3. โค๊ดเวคเตอร์ $\underline{v}_i$ ได้จาก $\underline{v}_i = \underline{r} + \underline{e}_i$ ซึ่งถือว่าเป็นโค๊ดเวคเตอร์ที่ส่งมา

ตัวอย่างที่ 3.9 จากโค๊ด (6,3) ที่มีโค๊ดเวคเตอร์ดังต่อไปนี้

รหัสข่าวสาร	โค๊ดเวคเตอร์
0 0 0	0 0 0 0 0 0
0 0 1	0 0 1 1 1 0
0 1 0	0 1 0 1 0 1
0 1 1	0 1 1 0 1 1
1 0 0	1 0 0 0 1 1
1 0 1	1 0 1 1 0 1
1 1 0	1 1 0 1 1 0
1 1 1	1 1 1 0 0 0

โค๊ดเวคเตอร์เหล่านี้ได้จากเจนเนอเรเตอร์แมททริกซ์ G

$$G = \begin{matrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{matrix}$$

จะได้พารีตี้แมททริกซ์คือ

$$H = \begin{matrix} 0\ 1\ 1 & 1\ 0\ 0 \\ 1\ 0\ 1 & 0\ 1\ 0 \\ 1\ 1\ 0 & 0\ 0\ 1 \end{matrix}$$

ความสัมพันธ์ของโคเซทลีดเดอร์กับซีนโดรมคือ

ซีนโดรม	โคเซทลีดเดอร์
0 0 0	0 0 0 0 0 0
0 0 1	0 0 0 0 0 1
0 1 0	0 0 0 0 1 0
0 1 1	0 0 0 1 0 0
1 0 0	0 0 1 0 0 0
1 0 1	0 1 0 0 0 0
1 1 0	1 0 0 0 0 0
1 1 1	0 0 1 0 0 1

ถ้าสมมติว่าโค๊ดเวคเตอร์ $\underline{v}_8$ = (1 1 1 0 0 0) ถูกส่งมา และ $\underline{r}$ รับได้เป็น (1 1 1 0 0 1) ในการถอดรหัส $\underline{r}$ ได้ซีนโดรมเป็น

$$\underline{r}H^T = (1\ 1\ 1\ 0\ 0\ 1) \begin{matrix} 0\ 1\ 1 \\ 1\ 0\ 1 \\ 1\ 1\ 0 \\ 1\ 0\ 0 \\ 0\ 1\ 0 \\ 0\ 0\ 1 \end{matrix} = (0\ 0\ 1)$$

พบว่า (0 0 1) เป็นซีนโดรมของโคเซทลีดเดอร์ (0 0 0 0 0 1) ซึ่งถือว่าเป็นรูปแบบของรหัสที่ผิดที่เกิดขึ้นในช่องส่งที่มีสัญญาณรบกวน ดังนั้นสัญญาณที่ส่งมาจึงเป็น (1 1 1 0 0 1) + (0 0 0 0 0 1) = (1 1 1 0 0 0) = $\underline{v}_i$ ถ้าหากว่า $\underline{v}_i$ = (1 1 1 0 0 0) ถูกส่งมาและ $\underline{r}$ = (1 1 1 0 1 1)

รับเข้ามาจะได้ซีนโดรมเป็น (0 1 1) ซึ่งตรงกับโคเซทลีดเดอร์ของ (1 0 0 0 0 0) และถือเป็นรูปแบบของรหัสที่ผิด ทำให้คิดไปว่ารหัสที่ส่งมาเป็น (1 1 1 0 1 1) + (1 0 0 0 0 0) = (0 1 1 0 1 1) การถอดรหัสจึงทำได้ไม่ถูกต้องเพราะรูปแบบของรหัสที่ผิดที่แท้จริงคือ (1 1 1 0 1 1)+ (1 1 1 0 0 0) = (0 0 0 0 1 1) ซึ่งไม่เป็นโคเซทลีดเดอร์ในอะเรมาตรฐาน เพราะเกิดรหัสที่ผิดไป 2 บิทในหนึ่งโค๊ดเวคเตอร์ การแก้บิทที่ผิดไปเพียงบิทเดียวนี้เรียกว่า single-error-correcting code

บทที่ 4
การโปรแกรมและข้อสรุป

จากบทที่แล้วมา ได้แสดงวิธีการเข้ารหัสแบบบล็อคโค๊ดเชิงเส้น ในบทนี้จะกล่าวถึงวิธีการโปรแกรมการเข้ารหัสและการแก้ไขรหัสที่ผิด โดยโปรแกรมที่ใช้จะใช้เป็นภาษา C โดยใช้ BORLAND C++ , version 2.0

สำหรับโปรแกรมส่วนที่ทำการเข้ารหัสและถอดรหัสที่สำคัญ มีดังนี้

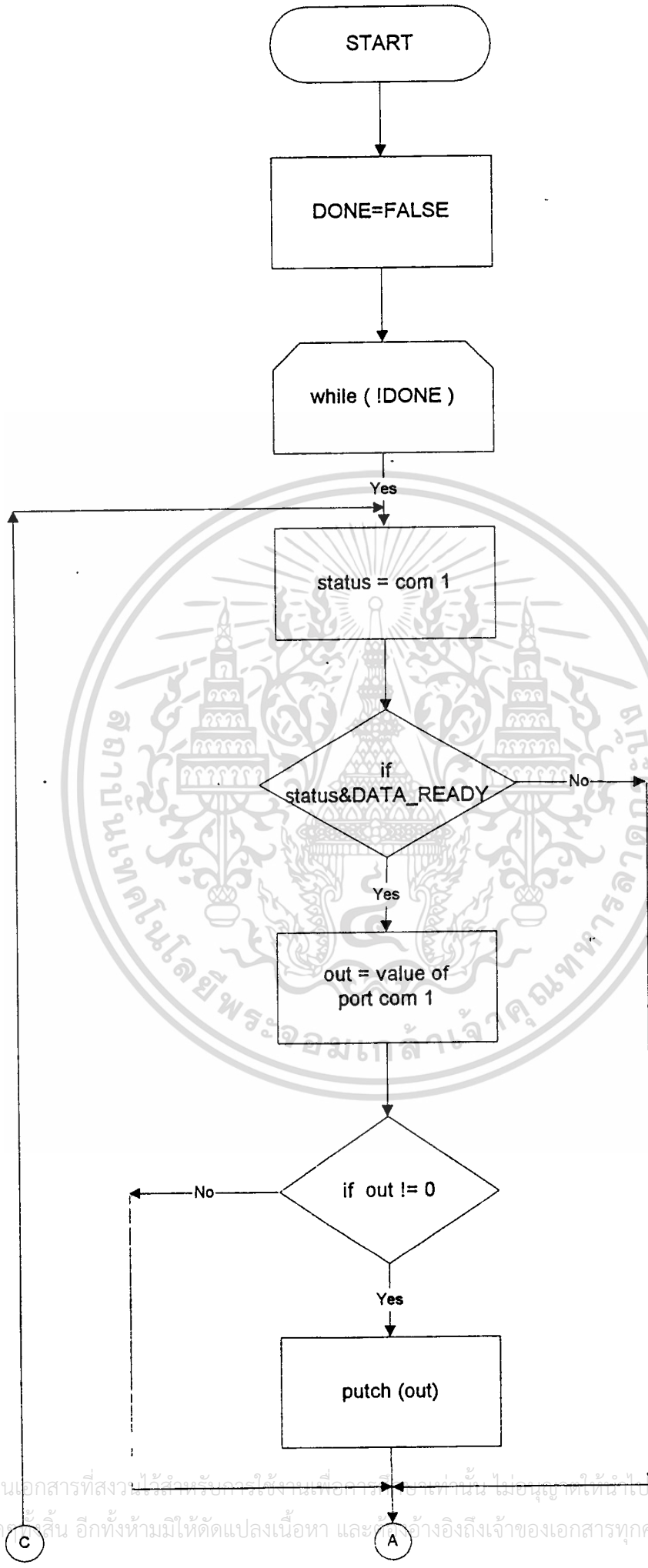
ส่วนแรกของโปรแกรมนี้เป็นส่วนของการประกาศโปรแกรมย่อยที่สำคัญ และกำหนดค่าตัวแปรต่าง ๆ ที่ใช้ในโปรแกมมีดังนี้

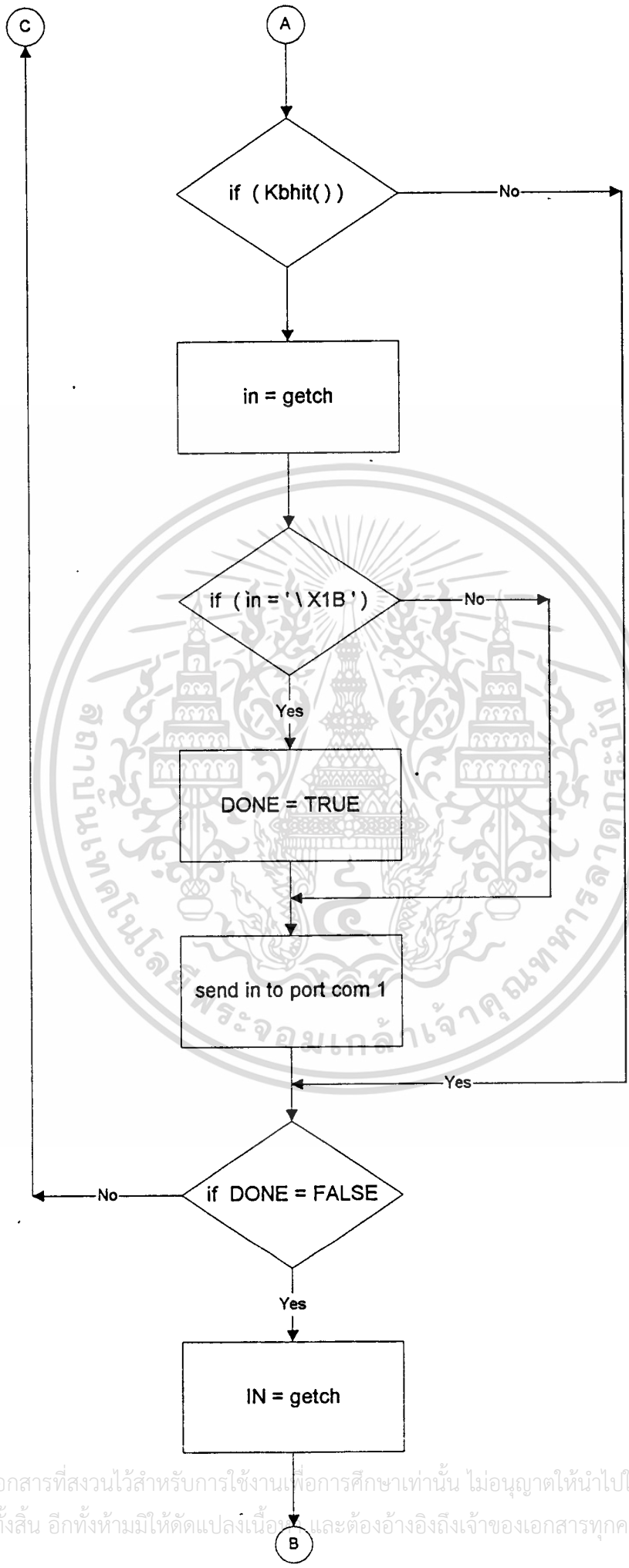
```
#define PORT 0
#define XSIZE 128
#define YSIZE 128
#define RETURN 0x1C0D
#define ESCAPE 0x011B
#define COM1      0
#define DATA_READY 0x100
#define TRUE      1
#define FALSE     0
#define SETTINGS ( 0x80 | 0x02 | 0x00 | 0x00)
typedef struct heading {char *choice;};
typedef struct menu_struct {
    int frame[4];
    int row[NO_CHOICE];
    int col;
    struct heading item[NO_CHOICE];
    int last_choice;
};
        int modem_char();
int modem_send(char *);
int modem_recv();
```

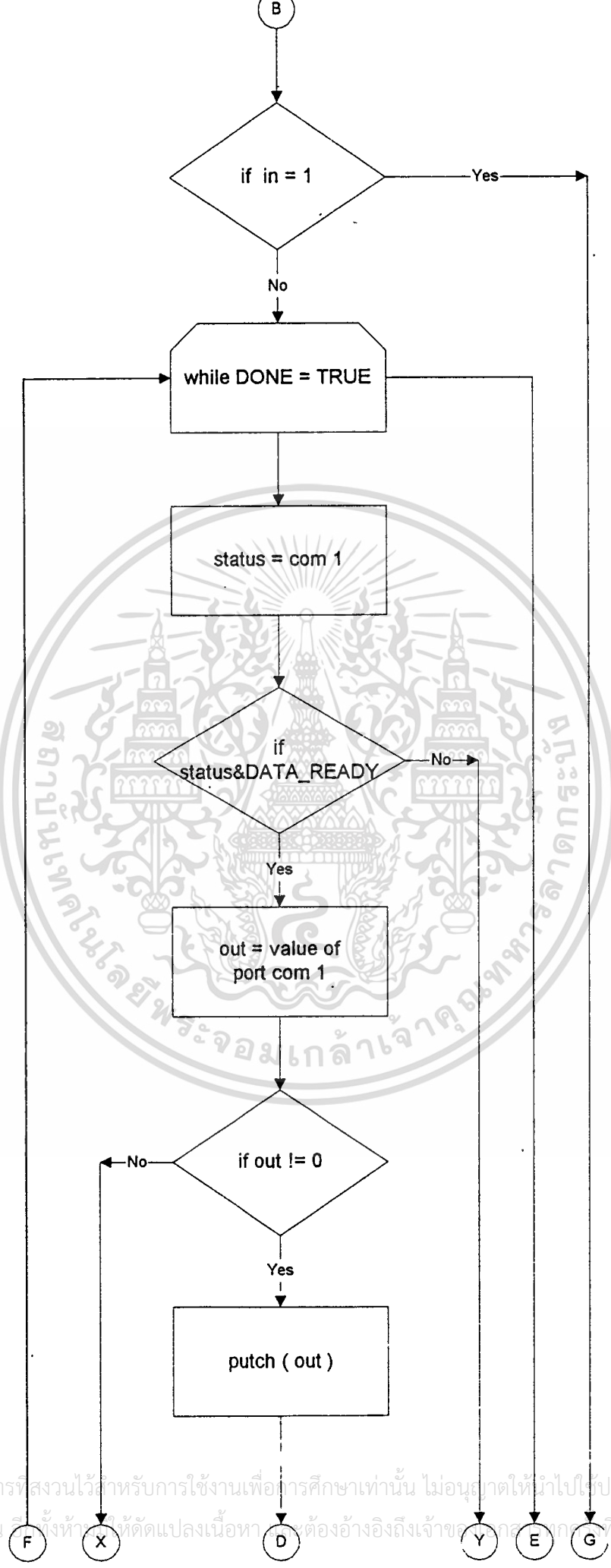
ส่วนที่สองของโปรแกรมเป็นส่วนของการติดต่อสื่อสารระหว่างเครื่องคอมพิวเตอร์และระบบโมเดม โดยส่วนนี้มีความสำคัญในการทำงานที่เกี่ยวข้องกับโทรศัพท์ทั้งหมด ทั้งที่เป็นการ DIAL , HANG UP , ANSWER CALL เป็นต้น

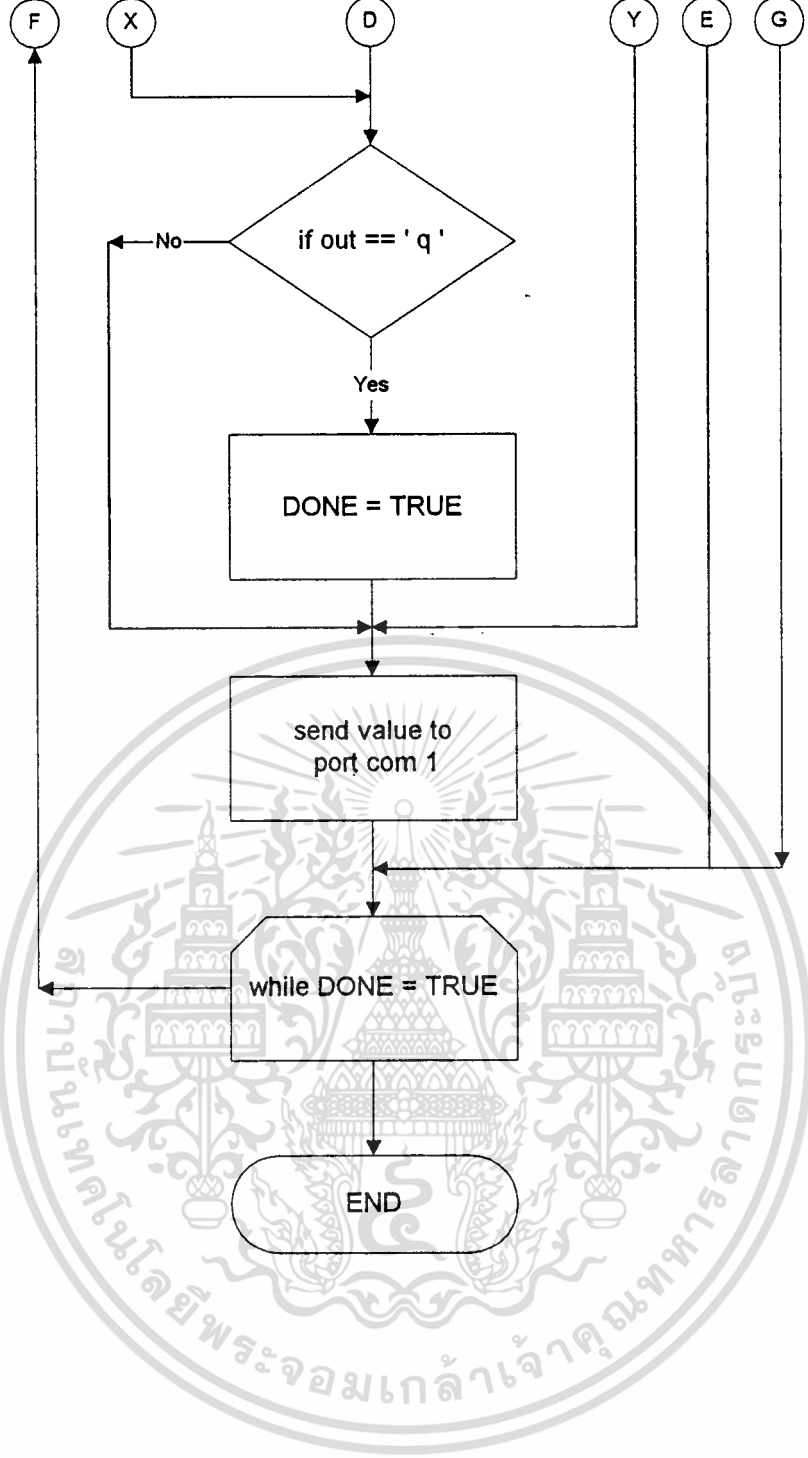
โปรแกรมมีดังนี้

```
int modem_char(void)
{
   int in, out, status, DONE = FALSE;
   clrscr();
   bioscom(0, SETTINGS, COM1);
    cprintf("... BIOSCOM [ESC] to exit ...\n");
   while (!DONE)
 { status = bioscom(3, 0, COM1);
    if (status & DATA_READY)
if ((out = bioscom(2, 0, COM1)) != 0)
   putch(out);
if (kbhit())
{if ((in = getch()) == '\x1B')
    DONE = TRUE;
   bioscom(1, in, COM1);
}                    }
in = getch();
if(in==1)exit(1);
while (DONE)
 {status = bioscom(3, 0, COM1);
    if (status & DATA_READY)
if ((out = bioscom(2, 0, COM1)) != 0)
   putch(out);
   if(out == 'q')DONE=FALSE;
   bioscom(1, 0, COM1);
     }
   return 0;
}
```









ส่วนของโปรแกรมส่วนที่สามเป็นส่วนของการเข้ารหัสข้อมูล โดยจะมีการเปิดไฟล์เพื่ออ่านเอารหัสตัวเลขของไฟล์มาทำการเข้ารหัสแบบบล็อคโค๊ดเชิงเส้น แล้วส่งออกไปยังช่องส่งสัญญาณของคอมพิวเตอร์

โปรแกรมการเข้ารหัส มีดังนี้

```
int modem_send(fname)
char *fname;
{
   FILE *fp;
   unsigned char ch,code;
   int x,y,port;
   int y1,z,ran,m[12],counta;
   unsigned int a,x1;
   unsigned char sendword,codeword;
   union {
         char c[2];
         unsigned int count;
         } cnt;
   int in, out, status, DONE = FALSE;
   x=0;
   y=0;
   ran=0;
   port = 0;
   clrscr();
   bioscom(0, SETTINGS, COM1);
   cprintf("... BIOSCOM [ESC] to exit ...\n");
   while (!DONE)
   {
     status = bioscom(3, 0, COM1);
     if (status & DATA_READY)
         if ((out = bioscom(2, 0, COM1)) != 0)
            putch(out);
         if (kbhit())
```

```
        {
          if ((in = getch()) == '\x1B')
            DONE = TRUE;
          bioscom(1, in, COM1);
        }
      }
  if(!(fp=fopen(fname,"rb")))
    {
        printf("can not open file...\n");
          exit(1);
    }
  in=getch();
  bioscom(0, SETTINGS, COM1);
  status = bioscom(3, 0, COM1);
  bioscom(1,'.',COM1);
  bioscom(1,name,COM1);
  s_f_name1(fname);
  wait2(PORT);
  cnt.count = 16384;
  bioscom(1,cnt.c[0],COM1);
  wait2(PORT);
  bioscom(1,cnt.c[1],COM1);
  opengraph_320_200();
 do{
      code=getc(fp);
      if(ferror(fp))
         {
          closegraph();
          printf("error reading input file");
          break;
          }
         {
```

```
sendword = 0;
codeword = 0;
for(z=12;z>0;z--)
{
counta =0;
if(z==12)a = (code)&(128);  /* 100000001100 */
if(z==11)a = (code)&(64);   /* 010000000110 */
if(z==10)a = (code)&(32);   /* 001000000011 */
if(z==9)a = (code)&(16);    /* 000100001001 */
if(z==8)a = (code)&(8);     /* 000010001010 */
if(z==7)a = (code)&(4);     /* 000001000101 */
if(z==6)a = (code)&(2);     /* 000000101110 */
if(z==5)a = (code)&(1);     /* 000000010111 */
if(z==4)a = (code)&(154);
if(z==3)a = (code)&(199);
if(z==2)a = (code)&(107);
if(z==1)a = (code)&(53);
for(y1=0;y1<=7;y1++)
{
    x1 = a&0x1;
    if(x1==1){ counta++;}
    a = a>>1;
}
if(counta%2){ m[z]=1;}
else m[z]=0;
}
for(z=12;z>0;z--)
{
if(z>=5){
    if(m[z]==1){ sendword = sendword<<1;
        sendword = sendword|1;}
    else sendword = sendword<<1;
```

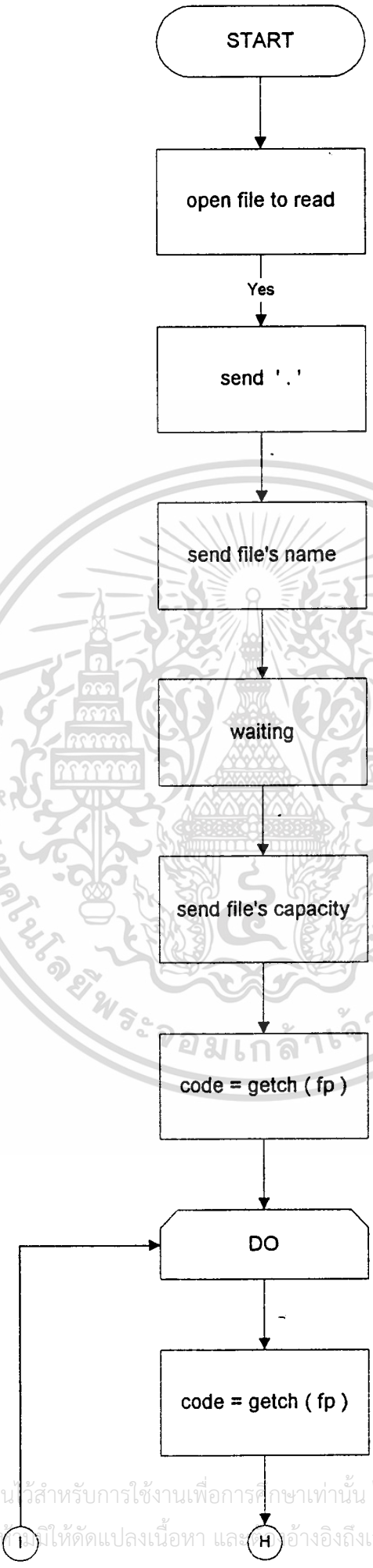
```
        }
      if(z<5){
            if(m[z]==1){ codeword = codeword<<1;
                         codeword = codeword|1;}
          else codeword = codeword<<1;
        }
    }
```

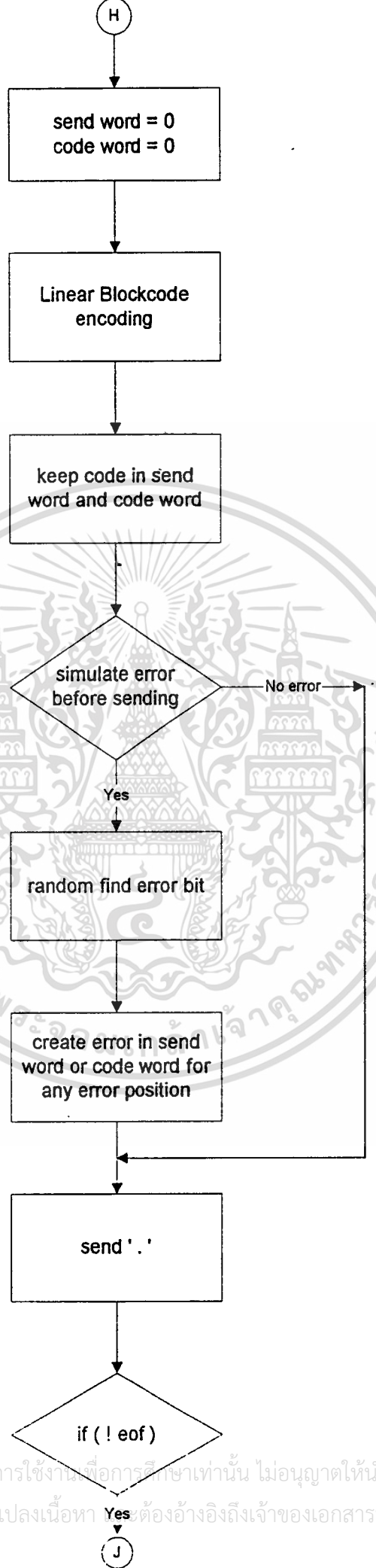
นอกจากนี้ในโปรแกรมจะมีการสร้างสัญญาณรบกวนขึ้นเพื่อทำให้เกิดรหัสที่ผิดอย่างซุ่มให้กับข้อมูล (random error) ก่อนที่จะมีการส่ง เป็นการ simulate noise เพื่อทดสอบการทำงานของโปรแกรมในการแก้ไขรหัสที่ผิด

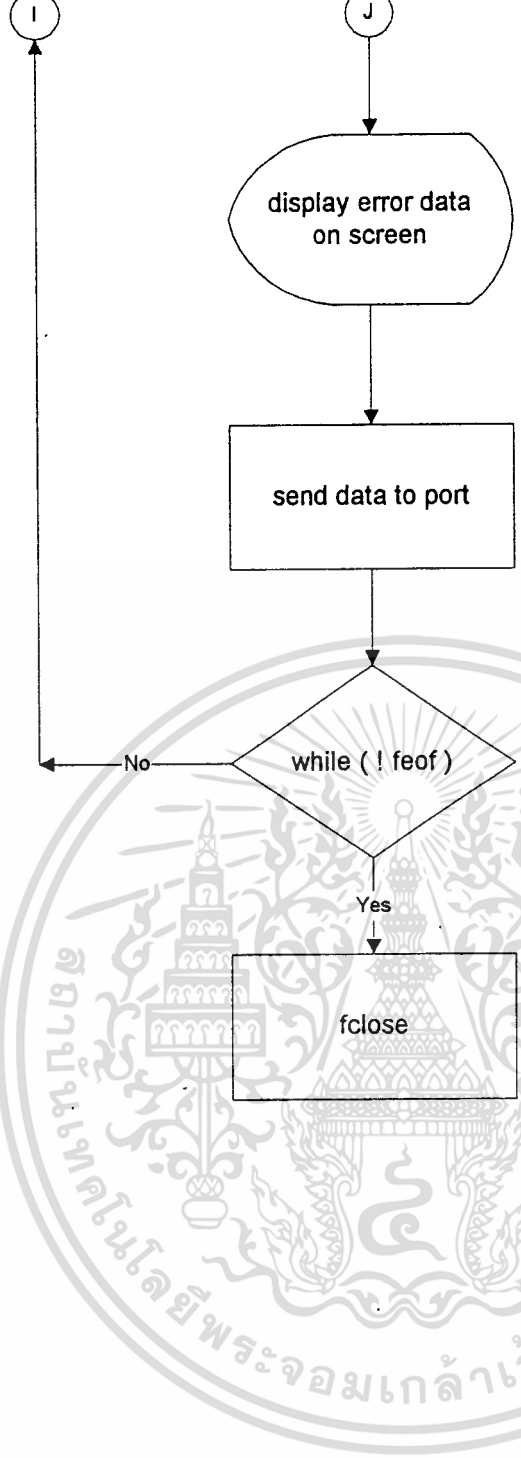
```
    {
     ran=random(12);
     if(ran==11)  codeword = codeword^8;
     if(ran==10)  codeword = codeword^4;
     if(ran==9)  codeword = codeword^2;
     if(ran==8)  codeword = codeword^1;
     if(ran==7)  sendword = sendword^128;
     if(ran==6)  sendword = sendword^64;
     if(ran==5)  sendword = sendword^32;
     if(ran==4)  sendword = sendword^16;
     if(ran==3)  sendword = sendword^8;
     if(ran==2)  sendword = sendword^4;
     if(ran==1)  sendword = sendword^2;
     if(ran==0)  sendword = sendword^1;
  }
     ch=sendword;
}
bioscom(1,'.',COM1);
if(!feof(fp))
   {delay(22);
    wair2(port)
    putpixel(x+64,y+36,(ch>>2)+64);
```

```
            bioscom(1,'~',COM1);
            delay(10);


            bioscom(1,sendword,COM1);
            delay(10);
            bioscom(1,codeword,COM1);
            x++;
          }
            if(x>=XSIZE)
               {
                  y++;
                  x=0;
            if(ferror(fp))
               {
                  closegraph();
                  printf("error writing output file");
                  clearerr(fp);
                  exit(1);
               }
            bioscom(1,'.',COM1);
            }
       }while(!feof(fp));
       wait2(port);
       fclose(fp);
       getch();
       cls();
       closegraph();
    return;
}
```

Encoding Programe







โปรแกรมส่วนที่สี่เป็นส่วนของการถอดรหัสข้อมูล โดยโปรแกรมจะรับสัญญาณข้อมูลจากช่องสัญญาณมาทำการถอดรหัสและตรวจสอบความถูกต้องของข้อมูล ถ้าหากพบว่าสัญญาณข้อมูลที่รับมาเกิดความผิดพลาด (syndrom > 0) โปรแกรมจะทำการแก้ไขรหัสที่ผิดให้กลับเป็นรหัสที่ถูกต้องได้ภายในตัวของโปรแกรมเอง

โปรแกรมการถอดรหัส มีดังนี้

```
int modem_recv(void)
{
   FILE *fp;
   unsigned char ch,ch1;/*,ich[1];*/
   char fname[14];
   int x,y;
   int a1,y2,i,k,syndrom;
   unsigned int g1,g2,counta,n[5],m[15];
   unsigned long int word1,word2,g3,q;
   unsigned char c1,z2;
   char h;
/*   unsigned long count;*/
   union {
         char c[2];
         unsigned int count;
         }cnt;
   int in, out, status, DONE = FALSE;
   clrscr();
   bioscom(0, SETTINGS, COM1);
    cprintf("... BIOSCOM [ESC] to exit ...\n");
   while (!DONE)
  {
      status = bioscom(3, 0, COM1);
      if (status & DATA_READY)
          if ((out = bioscom(2, 0, COM1)) != 0)
             putch(out);
          if (kbhit())
```

```
                {
                  if ((in = getch()) == '\x1B')
                    DONE = TRUE;
                  bioscom(1, in, COM1);
                }
              }
            in = getch();
  if(in==1)exit(1);
  wait3(PORT,0);
  bioscom(0, SETTINGS, COM1);
  status = bioscom(3, 0, COM1);
  g_f_name1(fname);
  printf("receiving file %s\n",fname);
  remove(fname);
  bioscom(1,'.',COM1);
  cnt.c[0] = bioscom(2,0,COM1);
  bioscom(1,'.',COM1);
  cnt.c[1] = bioscom(2,0,COM1);
  bioscom(1,'.',COM1);
  if(!(fp=fopen("ama.128","rb")))
    {
          printf("cannot open output file\n");
          exit(1);
    }
  closegraph();
  opengraph_320_200();
 while(!feof(fp)){ ch1=getc(fp);
            putpixel(x+64,y+36,(ch1>>2)+64);
           }
  for(y=0;y<YSIZE;y++)
      for(x=0;x<XSIZE;x++)
```

```
{
        cnt.count--;
        if(cnt.count == 0)
            break;
     /*   ch = rport(PORT);*/
    //     ch1=getc(fp);
    {
        word1 = 0;
        word2 = 0;
        syndrom =0;
        counta = 0;
        g1=0;g2=0;g3=0;q=0;
        c1=bioscom(2,0,COM1);
        if(c1=='~')
        { delay(5);
          h='s';
          }else h='r';
        if(h=='s')
        {
        for(k=1;k<=2;k++)
   {   delay(10);
       c1 = bioscom(2,0,COM1);
       {
        if(k==1){ g1 = c1;
                  word1 = (g1)<<4;
                  word2 = (g1)<<4;
                 }
if(k==2){
                   g2 = c1;
                   word1 = word1|g2;
                   word2 = word2|g2;
                   printf("word1=%u\n",word1);
```

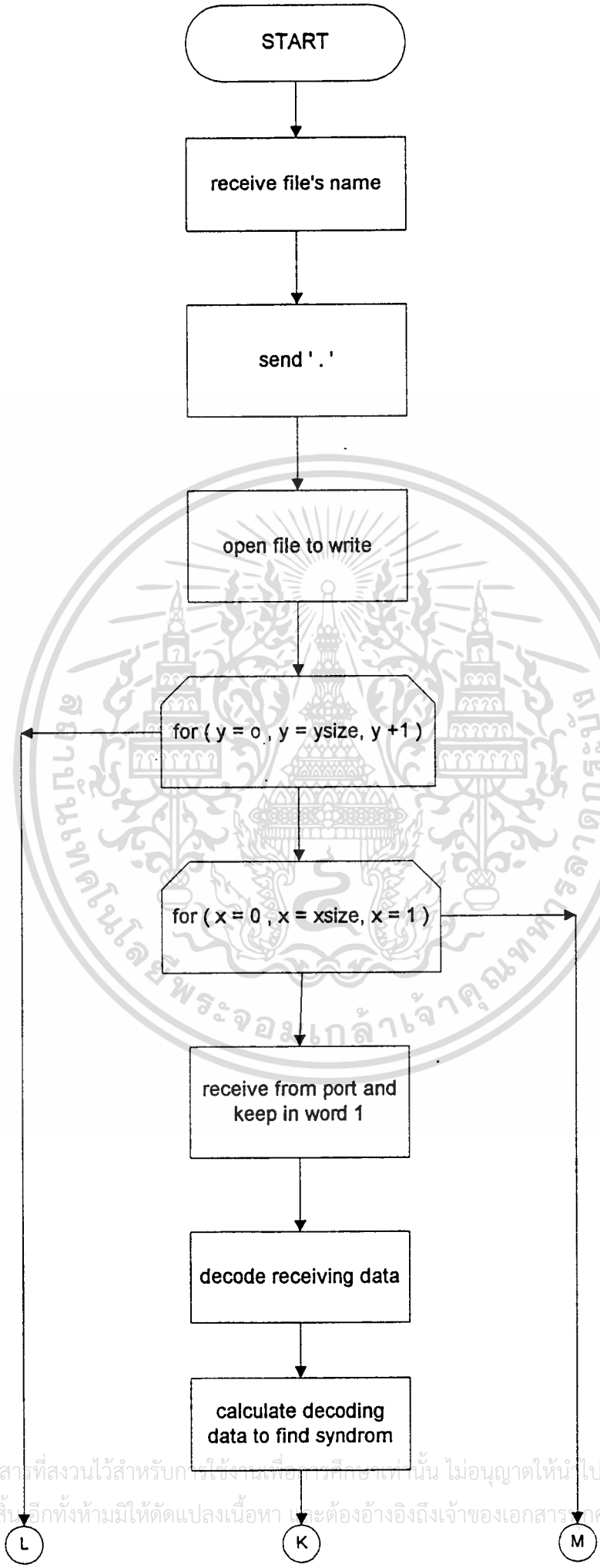
```
                    printf("word2=%u\n",word2);
                   }


        }
    }
   for(i=12;i>=1;i--)

   for(a1=4;a1>=1;a1--)
   { g3=0;counta=0;
   if(a1==4)g3 = (word1)&(2472);
   if(a1==3)g3 = (word1)&(3188);
   if(a1==2)g3 = (word1)&(1714);
   if(a1==1)g3 = (word1)&(849);

   for(y2=1;y2<=12;y2++)
   {
       q = g3&0x1;
       if(q==1){ counta++;}
       g3 = g3>>1;
   }
   if(counta%2){ n[a1]=1; }
   else n[a1]=0;
   }
   for(i=4;i>=1;i--)
      {
        if(n[i]==1){ syndrom = syndrom<<1;
                    syndrom = syndrom|1;}
else syndrom = syndrom<<1;
       }

   if(syndrom>0)
   {
```

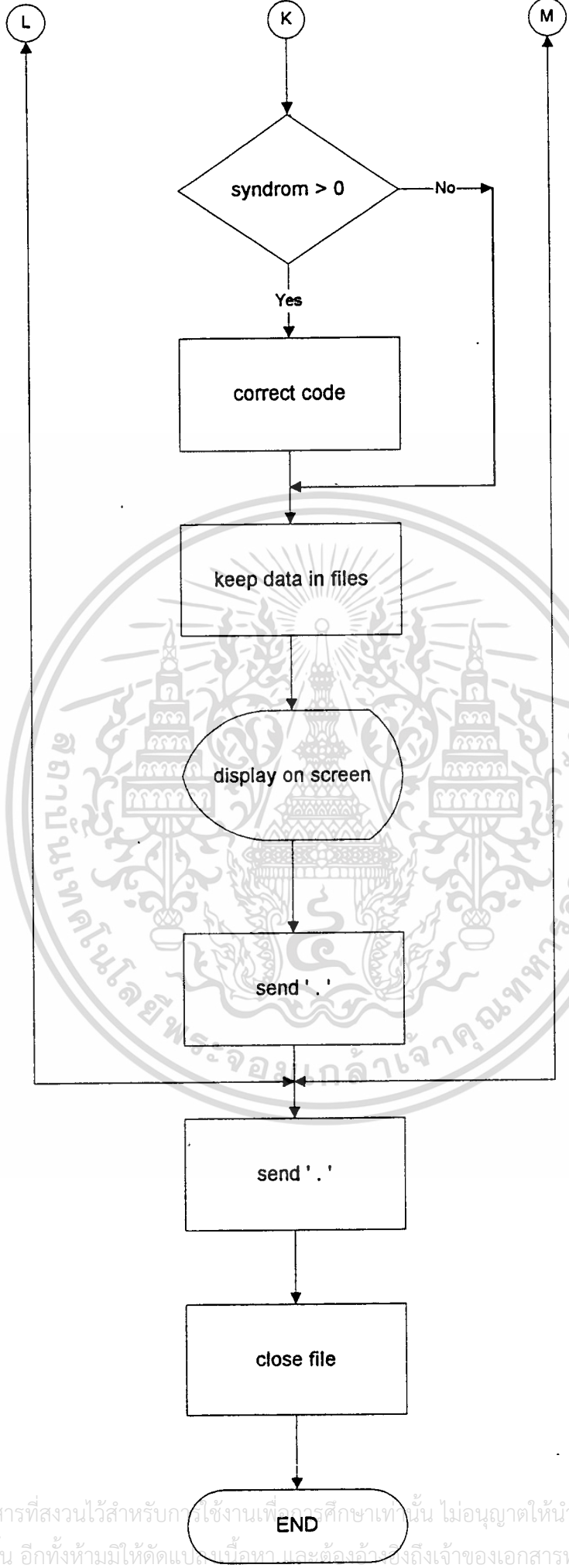
```
            if(syndrom==1)    word1 = word1^1;
            if(syndrom==2)    word1 = word1^2;
            if(syndrom==3)    word1 = word1^512;
            if(syndrom==4)    word1 = word1^4;
            if(syndrom==5)    word1 = word1^64;
            if(syndrom==6)    word1 = word1^1024;
            if(syndrom==7)    word1 = word1^16;
            if(syndrom==8)    word1 = word1^8;
            if(syndrom==9)    word1 = word1^256;
            if(syndrom==10)   word1 = word1^128;
            if(syndrom==12)   word1 = word1^2048;
            if(syndrom==14)   word1 = word1^32;
           }
       z2 = word1>>4;

      /*for(i=1;i<=12;i++)
       { x2 = word1&0x1;
          if(x2==1)m[i]=1;
          else m[i]=0;
          word1=word1>>1;
          }
       for(i=12;i>=1;i--)
       printf(".....%d",m[i]);
       printf("\n");
       if(z == 'q')exit(1);*/
     }
  }
 putpixel(x+64,y+36,(ch1>>2)+64);
                if(ferror(fp))
                  { closegraph();
                     printf("error writing output file");
                     clearerr(fp);
```

```
                exit(1);
            }
        getch();
        bioscom(1,'.',COM1);
        /*  sport(PORT,'.');*/
      }
  bioscom(1,'.',COM1);
  getch();
  fclose(fp);
  closegraph();

 return 0;
}
```

จากโปรแกรมข้างบนนี้เป็นส่วนของการเข้ารหัสข้อมูล และการถอดรหัสข้อมูล รวมทั้งการแก้ไข้รหัสที่ผิดโดยการตรวจสอบค่าของ syndrom ส่วนโปรแกรมที่สมบรูณ์ทั้งหมดจะแสดงไว้ในภาคผนวก

Decoding Programe





ข้อสรุป

จากการศึกษาและทดลองเขียนโปรแกรมการส่งสัญญาณผ่านระบบโมเดมโดยการเข้ารหัสข้อมูลแบบบล็อคโค๊ดเชิงเส้น ได้เริ่มศึกษาและทำการเขียนโปรแกรมรับ-ส่งตัวอักษร โปรแกรมรับ-ส่งไฟล์ระหว่างเครื่องคอมพิวเตอร์ หลังจากนั้นได้เขียนโปรแกรมการติดต่อและสั่งงานโมเดมด้วยคอมพิวเตอร์ เช่น dial , hang up , answer call เป็นต้น และได้พัฒนาโปรแกรมการเข้ารหัส-ถอดรหัสตัวอักษร โดยโปรแกรมเข้ารหัสจะรับตัวอักษรจากแป้นพิมพ์มาทำการเข้ารหัส แบบบล็อกโค๊ดเชิงเส้นดังกล่าวข้างต้นแล้วส่งไปยังผู้รับ ส่วนโปรแกรมถอดรหัสจะรับรหัสข้อมูลจากช่องสัญญาณ มาตรวจสอบและสามารถแก้ไขรหัสตัวอักษรที่มีรหัสผิดได้โดยอัตโนมัติภายในตัวของโปรแกรมเอง โดยมีการแสดงผลออกทางหน้าจอด้านส่งเป็นรหัสตัวอักษรที่รับเข้ามา และรหัสตัวอักษรที่มี error bit ที่จะถูกส่งออกไป ส่วนหน้าจอทางด้านเครื่องรับจะแสดงผลข้อมูลที่รับเข้ามาทางช่องสัญญาณ และแสดงผลข้อมูลที่ผ่านการแก้ไขรหัสที่ผิดให้ถูกต้องแล้ว นอกจากนั้นยังได้เขียนโปรแกรมการเข้ารหัสและถอดรหัสไฟล์ข้อมูลเพื่อทำการส่งผ่านระบบโมเดม โดยอาศัยหลักการเดียวกันคือ ฝ่ายส่งจะเปิดไฟล์เพื่ออ่านค่ารหัสของข้อมูลมาทำการเข้ารหัสแบบบล็อกโค๊ดเชิงเส้น แล้วส่งไปยังฝ่ายรับโดยมีการ simulate สัญญาณรบกวน เข้ากับรหัสข้อมูลก่อนการส่ง ส่วนฝ่ายรับจะรับสัญญาณข้อมูล มาถอดรหัสและตรวจสอบความถูกต้อง ถ้าหากข้อมูลที่รับมามี error bit โปรแกรมถอดรหัสจะทำการแก้ไขรหัสที่ผิดนั้นให้เป็นรหัสที่ถูกต้องได้โดยอัตโนมัติ

จากโปรแกรมที่ได้เขียนขึ้นมานี้ สามารถทำการทดลองได้ผลถูกต้องตามวัตถุประสงค์ที่ต้องการ

ข้อเสนอแนะ

โปรแกรมนี้สามารถนำไปพัฒนาโดยการเพิ่มขนาดเจนเนอเรเตอร์เมตริกที่ใช้เข้ารหัส (Generator Metrix) ให้มีรหัสแก้ไขมากขึ้น จะทำให้สามารถตรวจสอบและแก้ไขจำนวนบิทที่ผิดของรหัสข้อมูลได้มากขึ้น แต่มีข้อจำกัดประการหนึ่งคือ เมื่อเพิ่มจำนวนของรหัสแก้ไขจะทำให้ Codeword ยาวขึ้นมีผลทำให้การส่งข้อมูลต้องส่งจำนวนบิทเพิ่มขึ้น การติดต่อจะช้าลงด้วย

กิตติกรรมประกาศ

โครงงานในภาคเรียนนี้สำเร็จลุล่วงไปได้ด้วยดี โดยได้รับคำแนะนำ และการชี้แนะแนวทางจากท่านอาจารย์ที่ปรึกษา คือ อาจารย์ ดร. ฟูศักดิ์ ชีวสุวิทย์ จึงกราบขอบพระคุณอาจารย์ไว้ ณ โอกาสนี้ อีกทั้งยังได้รับความรู้ และคำแนะนำจากอาจารย์ท่านอื่น ๆ ทำให้โครงงานชิ้นนี้มีความสมบูรณ์เพิ่มขึ้น ต้องขอขอบคุณ อ. สักรียา ชิตวงศ์ และ อ. อาโมทย์ สมบูรณ์แก้ว เป็นอย่างยิ่ง

รวมทั้งขอขอบคุณเพื่อนๆ ทุกคนที่ให้กำลังใจระหว่างการทำงาน

หนังสืออ้างอิง

1. ฟูศักดิ์ ชีวสุวิทย์ , " การแก้รหัสที่ผิด " , คณะวิศวกรรมศาสตร์ , สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

2. SHU LIN , " **An Introduction o Error - Correcting Code** " , Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1970

3. J. WAKERIY , " **Error Detecting Codes, Self- Checking Circuits and Applications** " , North-Holland , New York ,1974.

4. V.PLESS , " **Introduction to the theory of Error-correcting Codes** " ,John Wiley & Sons, Inc. , 1978

ภาคผนวก

```
#include<dos.h>
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
#include<bios.h>
#include<stdlib.h>
#include<time.h>
#include<alloc.h>
#define PORT 0
#define XSIZE 128
#define YSIZE 128
{

unsigned long filesize();
void sport(int,unsigned char);
unsigned char rport1(int);
void sfile(char *);
void rfile(void);
s_f_name(char *);
s_f_name1(char *);
void g_f_name(char *);
void g_f_name1(char *);
check_stat(int);
void port_ini(int,int);
void wait(int);
void main_send();
void main_disp(void);
void opengraph(void);
void opengraph_320_200(void);
void opengraph_640_480(void);
void display();
void main_char();
```

```
void send_graph_error();
void modem_graph_error();

unsigned char far *pict,*img;

#define NO_CHOICE 5
#define NO_MENU 5

#define UP_ARROW 0x4800
#define DOWN_ARROW 0x5000
#define LEFT_ARROW 0x4B00
#define RIGHT_ARROW 0x4D00
#define RETURN 0x1C0D
#define ESCAPE 0x011B
#define COM1        0
#define DATA_READY 0x100
#define TRUE        1
#define FALSE       0

#define SETTINGS ( 0x80 | 0x02 | 0x00 | 0x00)


typedef struct heading {char *choice;};
typedef struct menu_struct {
    int frame[4];
    int row[NO_CHOICE];
    int col;
    struct heading item[NO_CHOICE];
    int last_choice;
};

void Initialize_Graphics_Mode(void);
```

```
void Close_Graphics_Mode(void);
void Draw_Rectangle(int lt,int tp,int rt,int bt);
void Draw_Fill_Rectangle(int lt,int tp,int rt,int bt);
void Erase_Fill_Rectangle(int lt,int tp,int rt,int bt);
void Set_Fill_Pattern(int pattern,int color);
void Set_Color(int color);
void Out_Text_XY(int x,int y,char text[]);
int Get_Max_X(void);
int Get_Max_Y(void);

int Title(void);
void Menu_Assignment(void);
void Display_Main_Menu(void);
void Display_Menu(int menu_no);
void Select_Menu(int menu_no,int choice_no);
void Unselect_Menu(int menu_no,int choice_no);
int Read_Key(int key);
void Inverse(int menu_no,int choice_no);
void Normal(int menu_no,int choice_no);
void cls(void);
void encode(int,unsigned char);
void decode(int);
void wait1(int,int);
void wait2(int);
void wait3(int,int);
void sfile_error();
void rfile_error(void);
int modem_char();
int modem_send(char *);
int modem_recv();
void Do_Choice(void);
void Do_Choice1(void);
```

```
void cursor(int x,int y);

struct menu_struct menus[NO_MENU];

int start = 1,
    one_choice_width = 20;
int end = 0;

int modem_send(fname)

char *fname;
{
    FILE *fp;
    unsigned char ch,code;
    int x,y,port;
    int y1,z,ran,m[12],counta;
    unsigned int a,x1;
    unsigned char sendword,codeword;
    union {
            char c[2];
            unsigned int count;
          } cnt;
   int in, out, status, DONE = FALSE;
   x=0;
   y=0;
   ran=0;
   port = 0;
   clrscr();
   bioscom(0, SETTINGS, COM1);
```

```
cprintf("... BIOSCOM [ESC] to exit ...\n");
while (!DONE)
{
   status = bioscom(3, 0, COM1);
   if (status & DATA_READY)
       if ((out = bioscom(2, 0, COM1)) != 0)
          putch(out);
       if (kbhit())
       {
          if ((in = getch()) == '\x1B')
             DONE = TRUE;
          bioscom(1, in, COM1);
       }
   }
if(!(fp=fopen(fname,"rb")))
   {
      printf("cannot open input file\n");
      exit(1);
   }
in=getch();
bioscom(0, SETTINGS, COM1);
status = bioscom(3, 0, COM1);
bioscom(1,'.',COM1);
bioscom(1,name,COM1);
s_f_name1(fname);
wait2(PORT);
cnt.count = 16384;
bioscom(1,cnt.c[0],COM1);
wait2(PORT);
bioscom(1,cnt.c[1],COM1);
opengraph_320_200();
```

```
do
 {
  code=getc(fp);
  if(ferror(fp))
     {
      closegraph();
      printf("error reading input file");
      break;
     }
    {
        sendword = 0;
        codeword = 0;
        for(z=12;z>0;z--)
        {
        counta =0;
        if(z==12)a = (code)&(128);  /* 100000001100 */
        if(z==11)a = (code)&(64);   /* 010000000110 */
        if(z==10)a = (code)&(32);   /* 001000000011 */
        if(z==9)a = (code)&(16);    /* 000100001001 */
        if(z==8)a = (code)&(8);     /* 000010001010 */
        if(z==7)a = (code)&(4);     /* 000001000101 */
        if(z==6)a = (code)&(2);     /* 000000101110 */
        if(z==5)a = (code)&(1);     /* 000000010111 */
        if(z==4)a = (code)&(154);
        if(z==3)a = (code)&(199);
        if(z==2)a = (code)&(107);
        if(z==1)a = (code)&(53);
        for(y1=0;y1<=7;y1++)
        {
            x1 = a&0x1;
            if(x1==1){ counta++;}
            a = a>>1;
```

```
}
if(counta%2){ m[z]=1;}
else m[z]=0;

}

for(z=12;z>0;z--)
{
if(z>=5){
        if(m[z]==1){ sendword = sendword<<1;
                sendword = sendword|1;}
         else sendword = sendword<<1;
      }
if(z<5){
      if(m[z]==1){ codeword = codeword<<1;
                codeword = codeword|1;}
         else codeword = codeword<<1;
      }
}
{
ran=random(12);

if(ran==11)  codeword = codeword^8;
if(ran==10)  codeword = codeword^4;
if(ran==9)  codeword = codeword^2;
if(ran==8)  codeword = codeword^1;
if(ran==7)  sendword = sendword^128;
if(ran==6)  sendword = sendword^64;
if(ran==5)  sendword = sendword^32;
if(ran==4)  sendword = sendword^16;
if(ran==3)  sendword = sendword^8;
if(ran==2)  sendword = sendword^4;
```

```
            if(ran==1)  sendword = sendword^2;
            if(ran==0)  sendword = sendword^1;

          }
          ch=sendword;
    }

    bioscom(1,'.',COM1);
    if(!feof(fp))
       {delay(22);
        wair2(port)
        putpixel(x+64,y+36,(ch>>2)+64);
        bioscom(1,'~',COM1);
        delay(10);
        bioscom(1,sendword,COM1);
        delay(10);
        bioscom(1,codeword,COM1);
        x++;
      }
        if(x>=XSIZE)
           { y++;
             x=0;

        if(ferror(fp))
            { closegraph();
              printf("error writing output file");
              clearerr(fp);
              exit(1);
            }
        bioscom(1,'.',COM1);
       }
  }while(!feof(fp));
```

```
        wait2(port);
        fclose(fp);
        getch();
        cls();
        closegraph();
    return ;
}


int modem_recv(void)
{
    FILE *fp;
    unsigned char ch,ch1;/*,ich[1];*/
    char fname[14];
    int x,y;
    int a1,y2,i,k,syndrom;
    unsigned int g1,g2,counta,n[5],m[15];
    unsigned long int word1,word2,g3,q;
    unsigned char c1,z2;
    char h;
/*    unsigned long count;*/
    union {
            char c[2];
            unsigned int count;
            }cnt;

   int in, out, status, DONE = FALSE;
   clrscr();
   bioscom(0, SETTINGS, COM1);
    cprintf("... BIOSCOM [ESC] to exit ...\n");
   while (!DONE)
```

```
{
   status = bioscom(3, 0, COM1);
   if (status & DATA_READY)
      if ((out = bioscom(2, 0, COM1)) != 0)
         putch(out);
      if (kbhit())
      {
         if ((in = getch()) == '\x1B')
           DONE = TRUE;
         bioscom(1, in, COM1);
      }
   }
in = getch();
if(in==1)exit(1);
wait3(PORT,0);
bioscom(0, SETTINGS, COM1);
status = bioscom(3, 0, COM1);
g_f_name1(fname);
printf("receiving file %s\n",fname);
remove(fname);
bioscom(1,'.',COM1);
cnt.c[0] = bioscom(2,0,COM1);
bioscom(1,'.',COM1);
cnt.c[1] = bioscom(2,0,COM1);
bioscom(1,'.',COM1);
if(!(fp=fopen("ama.128","rb")))
 {
      printf("cannot open output file\n");
      exit(1);
 }
closegraph();
opengraph_320_200();
```

```
while(!feof(fp)){ ch1=getc(fp);
        putpixel(x+64,y+36,(ch1>>2)+64);
      }
for(y=0;y<YSIZE;y++)
   for(x=0;x<XSIZE;x++)
      {
        cnt.count--;
        if(cnt.count == 0)
            break;
       /*   ch = rport(PORT);*/
      //     ch1=getc(fp);
      {
          word1 = 0;
          word2 = 0;
          syndrom =0;
          counta = 0;
          g1=0;g2=0;g3=0;q=0;
          c1=bioscom(2,0,COM1);
          if(c1=='~')
          { delay(5);
            h='s';
            }else h='r';
          if(h=='s')
          {
          for(k=1;k<=2;k++)
   {  delay(10);
       c1 = bioscom(2,0,COM1);
       {
        if(k==1){ g1 = c1;
                  word1 = (g1)<<4;
                  word2 = (g1)<<4;
                 }
```

```
        if(k==2){
                    g2 = c1;
                    word1 = word1|g2;
                    word2 = word2|g2;
                    printf("word1=%u\n",word1);
                    printf("word2=%u\n",word2);
                   }

      }
 }
for(i=12;i>=1;i--)

for(a1=4;a1>=1;a1--)
{ g3=0;counta=0;
if(a1==4)g3 = (word1)&(2472);
if(a1==3)g3 = (word1)&(3188);
if(a1==2)g3 = (word1)&(1714);
if(a1==1)g3 = (word1)&(849);

for(y2=1;y2<=12;y2++)
{
    q = g3&0x1;
    if(q==1){ counta++;}
    g3 = g3>>1;
}
if(counta%2){ n[a1]=1; }
else n[a1]=0;
}
for(i=4;i>=1;i--)
   {
     if(n[i]==1){ syndrom = syndrom<<1;
                  syndrom = syndrom|1;}
```

```
            else syndrom = syndrom<<1;
        }

    if(syndrom>0)
    {
        if(syndrom==1)    word1 = word1^1;
        if(syndrom==2)    word1 = word1^2;
        if(syndrom==3)    word1 = word1^512;
        if(syndrom==4)    word1 = word1^4;
        if(syndrom==5)    word1 = word1^64;
        if(syndrom==6)    word1 = word1^1024;
        if(syndrom==7)    word1 = word1^16;
        if(syndrom==8)    word1 = word1^8;
        if(syndrom==9)    word1 = word1^256;
        if(syndrom==10)   word1 = word1^128;
        if(syndrom==12)   word1 = word1^2048;
        if(syndrom==14)   word1 = word1^32;
       }
    z2 = word1>>4;

   /*for(i=1;i<=12;i++)
    { x2 = word1&0x1;
      if(x2==1)m[i]=1;
      else m[i]=0;
      word1=word1>>1;
      }
    for(i=12;i>=1;i--)
    printf(".....%d",m[i]);
    printf("\n");
    if(z == 'q')exit(1);*/
  }
}
```

```
/*              *ich = ch; */
/*              fwrite((char *)ich,1,1,fp); */
/*              img[count] = ch;
                count++; */
                putpixel(x+64,y+36,(ch1>>2)+64);
                if(ferror(fp))
                  { closegraph();
                    printf("error writing output file");
                    clearerr(fp);
                    exit(1);
                  }
                getch();
                bioscom(1,'.',COM1);
                /*  sport(PORT,'.');*/
            }
    bioscom(1,'.',COM1);
    getch();
    fclose(fp);
    closegraph();

  return 0;
}


void wait3(port,num)
     int port,num;
     { int status;
       char k;
       bioscom(0,SETTINGS,COM1);
       status=bioscom(3,0,COM1);
       if(num==1){
                    do{ getch();
```

```
                    bioscom(1,'?',COM1);//sport(port,'?');
                    delay(50);
                    k=bioscom(2,0,COM1);//rport1(port);
                    }while(k!='*');

                }
        if(num==0){
                do{ getch();
                    delay(4);
                    k=bioscom(2,0,COM1);//rport1(port);
                    if(k=='?'){
                            bioscom(1,'*',COM1);//sport(port,'*');
                            }
                    } while(k!='?');
                }
    return;
    }


void modem_graph_error(argc,argv)
int argc;
char *argv[];
{   unsigned char c;
    char n,m;
    /*int port;*/
    closegraph();
    clrscr();
    printf("send or recieve....(s/r) ....");
    n = getch();
    printf("%c\n",n);
    printf("with error or not....(e/n)....");
```

```
        m = getch();
        printf("%c\n",m);


      MemAlloc();
      printf("FINE TRANFER PROGRAM IN OPERATION.\n");
      printf("TO ABORT , PRESS ANY KEY.\n \n");
      port_ini(PORT,231);
      if((n=='s')||(n=='S'))
      { wait3(PORT,1);
        argv[2]="ama.128";
        modem_send(argv[2]);
      }

       if((n=='r')||(n=='R'))
       { wait3(PORT,0);
           modem_recv();
           clrscr();

      }
      clrscr();
      closegraph();
      return;
}


int modem_char(void)
{
   int in, out, status, DONE = FALSE;
   clrscr();
   bioscom(0, SETTINGS, COM1);
```

```
    cprintf("... BIOSCOM [ESC] to exit ...\n");
    while (!DONE)
    {
      status = bioscom(3, 0, COM1);
      if (status & DATA_READY)
          if ((out = bioscom(2, 0, COM1)) != 0)
             putch(out);
          if (kbhit())
          {
             if ((in = getch()) == '\x1B')
                DONE = TRUE;
             bioscom(1, in, COM1);
          }
      }
  in = getch();
  if(in==1)exit(1);
  while (DONE)
  {
     status = bioscom(3, 0, COM1);
     if (status & DATA_READY)
         if ((out = bioscom(2, 0, COM1)) != 0)
            putch(out);
            if(out == 'q')DONE=FALSE;
            bioscom(1, 0, COM1);
      }

  return 0;
}


MemAlloc()
{
```

```
    pict = (unsigned char far *) farmalloc(XSIZE);
    if((img = (unsigned char far *) farmalloc(XSIZE*YSIZE)) == NULL)
    {
        printf("\nMemory allocation Error !");
        exit(1);
    }
    return;
}

unsigned char rport1(port)
    int port;
    {
        union REGS r;
        r.x.dx = port;
        r.h.ah = 2;
        int86(0x14,&r,&r);
        if(!(r.h.ah)&128){
                        printf("read error detected in serial port");
                        exit(1);
                        }
        return(r.h.al);
    }


void encode(port,code)
        int port;
        unsigned char code;
       {
            int y,z,ran,m[12],count;
            unsigned int a,x;
            unsigned long int sendword,codeword;
            sendword = 0;
```

```
codeword = 0;
for(z=12;z>0;z--)
{
count =0;
if(z==12)a = (code)&(128);  /* 100000001100 */
if(z==11)a = (code)&(64);   /* 010000000110 */
if(z==10)a = (code)&(32);   /* 001000000011 */
if(z==9)a = (code)&(16);    /* 000100001001 */
if(z==8)a = (code)&(8);     /* 000010001010 */
if(z==7)a = (code)&(4);     /* 000001000101 */
if(z==6)a = (code)&(2);     /* 000000101110 */
if(z==5)a = (code)&(1);     /* 000000010111 */
if(z==4)a = (code)&(154);
if(z==3)a = (code)&(199);
if(z==2)a = (code)&(107);
if(z==1)a = (code)&(53);
for(y=0;y<=7;y++)
{
    x = a&0x1;
    if(x==1){ count ++;}
    a = a>>1;
}
if(count%2){ m[z]=1;}
else m[z]=0;

}

for(z=12;z>0;z--)
{
if(z>=5){
            if(m[z]==1){ sendword = sendword<<1;
                         sendword = sendword|1;}
```

```
                else sendword = sendword<<1;
            }
        if(z<5){
            if(m[z]==1){ codeword = codeword<<1;
                        codeword = codeword|1;}
              else codeword = codeword<<1;
            }
    /*  printf("%d..",m[z]);*/
      }
    /*  printf("\nsendword=%u...codeword=%u\n",sendword,codeword);*/
    /* if((code>'a')&&(code<'g'))*/
     {
      ran = random(12);
      printf("\nrandom=%d\n",ran);
      if(ran==11) sendword = sendword^128;
      if(ran==10) sendword = sendword^64;
      if(ran==9)  sendword = sendword^32;
      if(ran==8)  sendword = sendword^16;
      if(ran==7)  sendword = sendword^8;
      if(ran==6)  sendword = sendword^4;
      if(ran==5)  sendword = sendword^2;
      if(ran==4)  sendword = sendword^1;
      if(ran==3)  codeword = codeword^8;
      if(ran==2)  codeword = codeword^4;
      if(ran==1)  codeword = codeword^2;
      if(ran==0)  codeword = codeword^1;
     }
      sport(port,'~');
      delay(200);
      sport(port,sendword);
      printf("sendwod = %u\n",sendword);
      delay(200);
```

```
                sport(port,codeword);
                printf("codeword = %u\n\n",codeword);
        }

/* recieve */

void decode(port1)
  int port1;
{    int a1,y,i,k,syndrom;
     unsigned int g1,g2,count,n[5],m[15];
     unsigned long int word1,word2,g3,q,z,x;
     unsigned char c1;
     char h;
     word1 = 0;
     word2 = 0;
     syndrom =0;
     count = 0;
     g1=0;g2=0;g3=0;q=0;
     c1=rport1(port1);
     if(c1=='~')
     { delay(100);
        h='s';
        }else h='r';
    if(h=='s')
    {
    for(k=1;k<=2;k++)
    {   delay(200);
         c1 = rport1(port1);
         {
           printf("read....%c...\n",c1);
           if(k==1){ g1 = c1;
                     word1 = (g1)<<4;
```

```
                word2 = (g1)<<4;
                }
        if(k==2){
                g2 = c1;
               /*  printf("g1=%u....g2=%u....word=%u\n",g1,g2,word1);*/
                word1 = word1|g2;
                word2 = word2|g2;
               /*  printf("word1=%u\n",word1);
                printf("word2=%u\n",word2);*/
                }

      }
  }
 for(i=1;i<=12;i++)
 { x = word2&0x1;
    if(x==1)m[i]=1;
    else m[i]=0;
    word2=word2>>1;
    }
 for(i=12;i>=1;i--)
/* printf(".....%d",m[i]);
 printf("\n");*/

 for(a1=4;a1>=1;a1--)
 { g3=0;count=0;
 if(a1==4)g3 = (word1)&(2472);     /*1100*/ /*[H]T*/
 if(a1==3)g3 = (word1)&(3188);     /*0110*/
 if(a1==2)g3 = (word1)&(1714);     /*0011*/
 if(a1==1)g3 = (word1)&(849);      /*1001*/
                                   /*1010*/
                                   /*0101*/
                                   /*1110*/
```

```
                                        /*0111*/
                                        /*1000*/
                                        /*0100*/
                                        /*0010*/
                                        /*0001*/
  for(y=1;y<=12;y++)
  {
      q = g3&0x1;
      if(q==1){ count ++;}
      g3 = g3>>1;
  }
  if(count%2){ n[a1]=1; }
  else n[a1]=0;
/*  printf(".....%d",n[a1]);
  printf("count%d=%d...",a1,count);*/
  }
 /* printf("\n");*/
  for(i=4;i>=1;i--)
     {
      if(n[i]==1){ syndrom = syndrom<<1;
                   syndrom = syndrom|1;}
      else syndrom = syndrom<<1;
     /*  printf("..%d",n[i]);*/
     }
/*  printf("\nsyndrom=%d\n",syndrom);*/

  if(syndrom>0)
  {
      if(syndrom==1)    word1 = word1^1;
      if(syndrom==2)    word1 = word1^2;
      if(syndrom==3)    word1 = word1^512;
      if(syndrom==4)    word1 = word1^4;
```

```
            if(syndrom==5)    word1 = word1^64;
            if(syndrom==6)    word1 = word1^1024;
            if(syndrom==7)    word1 = word1^16;
            if(syndrom==8)    word1 = word1^8;
            if(syndrom==9)    word1 = word1^256;
            if(syndrom==10)   word1 = word1^128;
            if(syndrom==12)   word1 = word1^2048;
            if(syndrom==14)   word1 = word1^32;
          }
     /* printf("word1=%u\n",word1);*/
      z = word1>>4;
      printf("real word = %c\n",z);

      for(i=1;i<=12;i++)
      { x = word1&0x1;
          if(x==1)m[i]=1;
          else m[i]=0;
          word1=word1>>1;
          }
     /*  for(i=12;i>=1;i--)
       printf(".....%d",m[i]);
       printf("\n");*/
       if(z == 'q')exit(1);
    }
}


void wait1(port,num)
     int port,num;
     {
       unsigned char k;
       if(num==1){
            •          do{ sport(port, );
```

```
                    delay(500);
                    k=rport1(port);
                   }while(k!='*');


                }
        if(num==0){
                do{ delay(400);
                    k=rport1(port);
                    if(k=='?'){
                                sport(port,'*');
                              }
                  } while(k!='?');
                }
    return;
    }

void main_char()
{    unsigned char c;
     char n;
     int port;
     closegraph();
     clrscr();
     port = 0;
     port_ini(0,27);
     printf("send or recieve ....");
     n = getch();
     if((n=='s')||(n=='S'))
     { wait1(port,1);
       clrscr();
     while(1)
     {
       c = getch();
```

```
            printf("real word = %c\n",c);
            /*  sport(1,c);*/
            encode(0,c);
            if(c =='q' ) break;
        }}
        if((n=='r')||(n=='R'))
        { wait1(port,0);
            clrscr();
        while(1)
        {
            decode(0);
            if(kbhit())break;
        }}
        clrscr();
        return;
    }

void sfile_error(fname)
char *fname;
{
    FILE *fp;
    unsigned char ch,code;
    int x,y,port;
    int y1,z,ran,m[12],counta;
    unsigned int a,x1;
    unsigned char sendword,codeword;
    union {
            char c[2];
            unsigned int count;
          } cnt;
    if(!(fp=fopen(fname,"rb"))) {
            printf("cannot open input file\n");
```

```
            exit(1);
        }
        x=0;
        y=0;
        ran=0;
        port = 0;
        s_f_name(fname);
        wait(PORT);
/*      cnt.count = filesize(fp); */
        cnt.count = 16384;
        sport(PORT,cnt.c[0]);
        wait(PORT);
        sport(PORT,cnt.c[1]);
        opengraph_320_200();
        do
          {
           code=getc(fp);
           if(ferror(fp))
              {
               closegraph();
               printf("error reading input file");
               break;
              }

             {
                  sendword = 0;
                  codeword = 0;
                  for(z=12;z>0;z--)
                  {
                  counta =0;
                  if(z==12)a = (code)&(128);  /* 100000001100 */
                  if(z==11)a = (code)&(64);   /* 010000000110 */
```

```
if(z==10)a = (code)&(32);   /* 001000000011 */
if(z==9)a = (code)&(16);    /* 000100001001 */
if(z==8)a = (code)&(8);     /* 000010001010 */
if(z==7)a = (code)&(4);     /* 000001000101 */
if(z==6)a = (code)&(2);     /* 000000101110 */
if(z==5)a = (code)&(1);     /* 000000010111 */
if(z==4)a = (code)&(154);
if(z==3)a = (code)&(199);
if(z==2)a = (code)&(107);
if(z==1)a = (code)&(53);
for(y1=0;y1<=7;y1++)
{
    x1 = a&0x1;
    if(x1==1){ counta++;}
    a = a>>1;
}
if(counta%2){ m[z]=1;}
else m[z]=0;

}

for(z=12;z>0;z--)
{
if(z>=5){
            if(m[z]==1){ sendword = sendword<<1;
                         sendword = sendword|1;}
             else sendword = sendword<<1;
          }
if(z<5){
          if(m[z]==1){ codeword = codeword<<1;
                       codeword = codeword|1;}
            else codeword = codeword<<1;
```

```
                }
        /* printf("%d..",m[z]);*/
         }
       /*  printf("\nsendword=%u...codeword=%u\n",sendword,codeword);*/
       /* if((code>'a')&&(code<'g'))*/
        {
         ran=random(12);

         if(ran==11)  codeword = codeword^8;
         if(ran==10)  codeword = codeword^4;
         if(ran==9)  codeword = codeword^2;
         if(ran==8)  codeword = codeword^1;
         if(ran==7)  sendword = sendword^128;
         if(ran==6)  sendword = sendword^64;
         if(ran==5)  sendword = sendword^32;
         if(ran==4)  sendword = sendword^16;
         if(ran==3)  sendword = sendword^8;
         if(ran==2)  sendword = sendword^4;
         if(ran==1)  sendword = sendword^2;
         if(ran==0)  sendword = sendword^1;

       }
       ch=sendword;
}

if(!feof(fp))
   {

    wait(PORT);

   /* putpixel(x+64,y+36,(ch>>2)+64);*/
    x++;
```

```
            /*  sport(PORT,ch);*/
              sport(port,'~');
              delay(10);
              sport(port,sendword);
            /*  printf("k=%u....\n",sendword);*/
              delay(10);
              sport(port,codeword);
            /*  printf("codeword = %u\n\n",codeword);*/
              if(x>=XSIZE)
                 { y++;
                   x=0;
                 }
              if(ferror(fp))
                   { closegraph();
                     printf("error writing output file");
                     clearerr(fp);
                     exit(1);
                   }
            }
        }while(!feof(fp));
        wait(PORT);
        fclose(fp);
        getch();
        cls();
        closegraph();
        return;
}

void rfile_error()
{
    FILE *fp;
    unsigned char ch;/*,ich[1];*/
```

```
    char fname[14];
    int x,y;
    int a1,y2,i,k,syndrom;
    unsigned int g1,g2,counta,n[5],m[15];
    unsigned long int word1,word2,g3,q,x2;
    unsigned char c1,z2;
    char h;
/*  unsigned long count;*/
    union {
          char c[2];
          unsigned int count;
          }cnt;

    g_f_name(fname);
    printf("receiving file %s\n",fname);
    remove(fname);
    sport(PORT,'.');
    cnt.c[0] = rport(PORT);
    sport(PORT,'.');
    cnt.c[1] = rport(PORT);
    sport(PORT,'.');

    if(!(fp=fopen(fname,"wb"))){
          printf("cannot open output file\n");
          exit(1);
    }
    opengraph_320_200();
/*  count = 0L;*/
   for(y=0;y<YSIZE;y++)
     for(x=0;x<XSIZE;x++)
         {
           cnt.count--;
```

```
        if(cnt.count == 0)
            break;
       /*  ch = rport(PORT);*/
      {
        word1 = 0;
        word2 = 0;
        syndrom =0;
        counta = 0;
        g1=0;g2=0;g3=0;q=0;
        c1=rport1(PORT);
        if(c1=='~')
        { delay(5);
          h='s';
          }else h='r';
        if(h=='s')
        {
        for(k=1;k<=2;k++)
{   delay(10);
    c1 = rport1(PORT);
    {
     /* printf("read....%c...\n",c1);*/
      if(k==1){ g1 = c1;
                word1 = (g1)<<4;
              /*   word2 = (g1)<<4;*/
                }
      if(k==2){
                g2 = c1;
             /*    printf("g1=%u....g2=%u....word=%u\n",g1,g2,word1);*/
                word1 = word1|g2;
             /*    word2 = word2|g2;
                printf("word1=%u\n",word1);
                printf("word2=%u\n",word2);*/
```

```
    }
/*  for(i=1;i<=12;i++)
  { x2 = word2&0x1;
      if(x2==1)m[i]=1;
      else m[i]=0;
      word2=word2>>1;
      } */
  for(i=12;i>=1;i--)
/*  printf(".....%d",m[i]);
  printf("\n");*/

  for(a1=4;a1>=1;a1--)
  { g3=0;counta=0;
  if(a1==4)g3 = (word1)&(2472);     /*1100*/ /*[H]T*/
  if(a1==3)g3 = (word1)&(3188);     /*0110*/
  if(a1==2)g3 = (word1)&(1714);     /*0011*/
  if(a1==1)g3 = (word1)&(849);      /*1001*/
                                    /*1010*/
                                    /*0101*/
                                    /*1110*/
                                    /*0111*/
                                    /*1000*/
                                    /*0100*/
                                    /*0010*/
                                    /*0001*/

   for(y2=1;y2<=12;y2++)
   {
       q = g3&0x1;
       if(q==1){ counta++;}
```

```
        g3 = g3>>1;
    }
    if(counta%2){ n[a1]=1; }
    else n[a1]=0;
/*  printf(".....%d",n[a1]);
    printf("count%d=%d...",a1,count);*/
    }
/*  printf("\n");*/
    for(i=4;i>=1;i--)
        {
         if(n[i]==1){ syndrom = syndrom<<1;
                      syndrom = syndrom|1;}
         else syndrom = syndrom<<1;
        /*  printf("..%d",n[i]);*/
         }
 /*  printf("\nsyndrom=%d\n",syndrom);*/

    if(syndrom>0)
    {
         if(syndrom==1)    word1 = word1^1;
         if(syndrom==2)    word1 = word1^2;
         if(syndrom==3)    word1 = word1^512;
         if(syndrom==4)    word1 = word1^4;
         if(syndrom==5)    word1 = word1^64;
         if(syndrom==6)    word1 = word1^1024;
         if(syndrom==7)    word1 = word1^16;
         if(syndrom==8)    word1 = word1^8;
         if(syndrom==9)    word1 = word1^256;
         if(syndrom==10)   word1 = word1^128;
         if(syndrom==12)   word1 = word1^2048;
         if(syndrom==14)   word1 = word1^32;
         }
```

```
    /* printf("word1=%u\n",word1);*/
     z2 = word1>>4;
   /*  ch = z2&128;*/
   /*  printf("z=%c\n",z);*/

   /*  for(i=1;i<=12;i++)
     { x2 = word1&0x1;
        if(x2==1)m[i]=1;
        else m[i]=0;
        word1=word1>>1;
        }
     for(i=12;i>=1;i--)
     printf(".....%d",m[i]);
     printf("\n");
     if(z == 'q')exit(1);*/
   }
}
/*          *ich = ch; */
/*          fwrite((char *)ich,1,1,fp); */
/*          img[count] = ch;
            count++;*/
            putpixel(x+64,y+36,(z2>>2)+64);
            if(ferror(fp))
              { closegraph();
                printf("error writing output file");
                clearerr(fp);
                exit(1);
               }
              sport(PORT,'.');
          }
    sport(PORT,'.');
    getch();
```

```
    fclose(fp);
    closegraph();
}

void send_graph_error(argc,argv)
int argc;
char *argv[];
{    unsigned char c;
     char n,m;
     /*int port;*/
     closegraph();
     clrscr();
     printf("send or recieve....(s/r) ....");
     n = getch();
     printf("%c\n",n);
     printf("with error or not....(e/n)....");
     m = getch();
     printf("%c\n",m);

       /*------------*/

    MemAlloc();
    printf("FINE TRANFER PROGRAM IN OPERATION.\n");
    printf("TO ABORT , PRESS ANY KEY.\n \n");
    port_ini(PORT,231);
/*    if(tolower(*argv[1]) == 's') sfile(argv[2]);
    else rfile(); */
    if((n=='s')||(n=='S'))
    { wait1(PORT,1);
      argv[2]="ama.128";
      if((m=='n')||(m=='N'))sfile(argv[2]);
```

```
      else sfile_error(argv[2]);
/*  main_disp(); */
/*  display(img,64,36); */
    }

        /*------------*/
      if((n=='r')||(n=='R'))
      { wait1(PORT,0);
        clrscr();
        if((m=='n')||(m=='N'))rfile();
        else rfile_error();

    }
    clrscr();
    closegraph();
    return;
}

/*---------*/

void Initialize_Graphics_Mode(void)
{
    int gdriver = DETECT, gmode, errorcode;
    initgraph(&gdriver, &gmode, "c:\\cpp\\bgi");
    errorcode = graphresult();

    if (errorcode != grOk)
    {
        printf("Graphics error : %s\n", grapherrormsg(errorcode));
        printf("Press any key to halt : ");
        getch();
        exit(1);
```

```
    }
}


void Close_Graphics_Mode(void)
{
    closegraph();
}


void Draw_Rectangle(int lt,int tp,int rt,int bt)
{
    rectangle(lt,tp,rt,bt);
}


void Draw_Fill_Rectangle(int lt,int tp,int rt,int bt)
{
    Set_Fill_Pattern(1,15);
    bar(lt,tp,rt,bt);
}


void Erase_Fill_Rectangle(int lt,int tp,int rt,int bt)
{
    Set_Fill_Pattern(1,0);
    bar(lt,tp,rt,bt);
}


void Set_Fill_Pattern(int pattern,int color)
{
```

```
    setfillstyle(pattern,color);
}


void Set_Color(int color)
{
    setcolor(color);
}


void Out_Text_XY(int x,int y,char text[])
{
    outtextxy(x,y,text);
}


int Get_Max_X(void)
{
    return(getmaxx());
}


int Get_Max_Y(void)
{
    return(getmaxy());
}

void cls()   /* Clear the screen. */
{
  union REGS r;

  r.h.ah = 6;  /* screen scroll code */
```

```
  r.h.al = 0;  /* clear screen code */
  r.h.ch = 0;  /* start row */
  r.h.cl = 0;  /* start column */
  r.h.dh = 29; /* end row */
  r.h.dl = 79; /* end column */
  r.h.bh = 0;  /* blank line is black */
  int86(0x10, &r, &r);
  return;
}


int Title(void)
{
   int i;
   opengraph_640_480();
   setfillstyle(1,1);                    /* Background */
   bar(0,0,639,479);
   rectangle(10,10,629,469);
   setcolor(12);   /* IN LINE red */
   setfillstyle(1,7);
   bar(11,11,628,468);
   rectangle(15,15,624,464);
   rectangle(17,17,622,462);

   setcolor(5);
 /*  settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);*/
   settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
   outtextxy(25,25,"King Mongkut Institute of Technology");
   outtextxy(150,65,"L A D K R A B A N G");
   setcolor(9);
   outtextxy(28,25,"King Mongkut Institute of Technology");
   outtextxy(153,65,"L A D K R A B A N G");
```

```
    setfillstyle(1,15);
    bar(40,160,599,165);
    bar(40,160,45,220);
    bar(160,390,479,400);   /* Producer */
    bar(469,240,479,400);
    setfillstyle(1,8);
    bar(40,215,599,220);
    bar(594,160,599,220);
    bar(160,240,479,250);   /* Producer */
    bar(160,240,170,400);

    setcolor(6);
    outtextxy(57,180,"CODING CONTROL & SIGNAL TRANSFER");
/*  rectangle(158,238,482,403); *//* producer */
    setcolor(11);
    outtextxy(61,180,"CODING CONTROL & SIGNAL TRANSFER");
    rectangle(158,238,482,403); /* producer */
    line(40,220,45,215);
    line(594,165,599,160);
    setcolor(15);
    for(i=0;i<5;i++)     /* Conner . */
      {
       line(40,215+i,45-i,215+i);
       line(594,160+i,599-i,160+i);
      }
    for(i=0;i<10;i++)       /* Conner Producer */
      {
       line(160+i,400-i,170,400-i);
       line(469+i,250-i,479,250-i);
      }
    setfillstyle(9,8);
    bar(40,115,599,140);
```

```
   settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
   setcolor(2);
   outtextxy(92,120,"Instrumentation   Department");
   setcolor(10);
   outtextxy(95,120,"Instrumentation   Department");
   setfillstyle(1,4);   /* producer */
   bar(169,249,469,390);

   settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
/* settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);*/
   outtextxy(240,305,"MR. PHONGCHAI NILAS");
   outtextxy(250,328,"ID. 33.100231");
   setcolor(11);
   rectangle(39,159,600,221);  /* Program name */
   rectangle(38,113,601,142);  /* Dept. Ins. */
   outtextxy(250,265,"..Presented By..");
/*   outtextxy(225,420,"COPYRIGHT (C) April 1994");*/
   if(getch());   /* switch-on */
    {
      setfillstyle(9,7);
      bar(39,159,600,221);
      setfillstyle(1,8);
      bar(39,159,600,165);
      bar(39,159,45,221);
      setfillstyle(1,15);
      bar(39,215,600,221);
      bar(594,160,600,221);
           setcolor(8);
      for(i=0;i<7;i++)
          {
           line(39,215+i,45-i,215+i);
           line(594,159+i,600-i,159+i);
```

```
        }
          setcolor(4);
   /* settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);*/
     settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
     outtextxy(65,178,"Switch-On to Start Program Now");
     settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
     delay(1000);
    }
   cls();
   cleardevice();
   return(0);
}


void Menu_Assignment(void)
{
    int one_part;
    setfillstyle(1,9);
    one_part = (Get_Max_X()/NO_MENU);

    /* assign menu no. 0 menu About */
    menus[0].frame[0] = start;
    menus[0].frame[1] = one_choice_width;
    menus[0].frame[2] = start+one_part;
    menus[0].frame[3] = one_choice_width+one_choice_width*4;


    menus[0].row[0] = 6;
    menus[0].row[1] = 26;
    menus[0].row[2] = 46;
    menus[0].row[3] = 66;
    menus[0].row[4] = 86;
```

```
menus[0].col = start;
start += one_part;

menus[0].item[0].choice = " About          ";
menus[0].item[1].choice = "  Information  ";
menus[0].item[2].choice = "  Programmers  ";
menus[0].item[3].choice = "  Send_File    ";
menus[0].item[4].choice = "  Display      ";

menus[0].last_choice = 4;


/* assign menu no. 1 menu File */
menus[1].frame[0] = start;
menus[1].frame[1] = one_choice_width;
menus[1].frame[2] = start+one_part;
menus[1].frame[3] = one_choice_width+one_choice_width*4;

menus[1].row[0] = 6;
menus[1].row[1] = 26;
menus[1].row[2] = 46;
menus[1].row[3] = 66;
menus[1].row[4] = 86;

menus[1].col = start;
start += one_part;

menus[1].item[0].choice = " File           ";
menus[1].item[1].choice = "  Modem_Char   ";
menus[1].item[2].choice = "  Modem_Send   ";
menus[1].item[3].choice = "  Modem_Recv   ";
menus[1].item[4].choice = "  Quit         ";
```

```
menus[1].last_choice = 4;


/* assign menu no. 2 menu Run */
menus[2].frame[0] = start;
menus[2].frame[1] = one_choice_width;
menus[2].frame[2] = start+one_part;
menus[2].frame[3] = one_choice_width+one_choice_width*2;

menus[2].row[0] = 6;
menus[2].row[1] = 26;
menus[2].row[2] = 46;
menus[2].row[3] = 66;
menus[2].row[4] = 86;

menus[2].col = start;
start += one_part;

menus[2].item[0].choice = " Run          ";
menus[2].item[1].choice = "  Auto        ";
menus[2].item[2].choice = "  Manual      ";
menus[2].item[3].choice = "";
menus[2].item[4].choice = "";

menus[2].last_choice = 2;


/* assign menu no. 3 menu Simulation */
menus[3].frame[0] = start;
menus[3].frame[1] = one_choice_width;
menus[3].frame[2] = start+one_part;
```

```
menus[3].frame[3] = one_choice_width+one_choice_width*2;

menus[3].row[0] = 6;
menus[3].row[1] = 26;
menus[3].row[2] = 46;
menus[3].row[3] = 66;
menus[3].row[4] = 86;

menus[3].col = start;
start += one_part;

menus[3].item[0].choice = " Simulation    ";
menus[3].item[1].choice = "  Send_char    ";
menus[3].item[2].choice = "  Send_graph_er";
menus[3].item[3].choice = "";
menus[3].item[4].choice = "";

menus[3].last_choice = 2;

/* assign menu no. 4 menu Help */
menus[4].frame[0] = start;
menus[4].frame[1] = one_choice_width;
menus[4].frame[2] = start+one_part;
menus[4].frame[3] = one_choice_width+one_choice_width*2;

menus[4].row[0] = 6;
menus[4].row[1] = 26;
menus[4].row[2] = 46;
menus[4].row[3] = 66;
menus[4].row[4] = 86;
```

```
    menus[4].col = start;
    start += one_part;

    menus[4].item[0].choice = "     Help       ";
    menus[4].item[1].choice = "  Information  ";
    menus[4].item[2].choice = "  Programmer   ";
    menus[4].item[3].choice = "";
    menus[4].item[4].choice = "";

    menus[4].last_choice = 2;

}


void Display_Main_Menu(void)
{
    int i;
    setfillstyle(1,7);
    bar(0,0,Get_Max_X(),19);
    setcolor(10); /* up menu line color */
    Draw_Rectangle(0,0,Get_Max_X(),19);

    setcolor(9);  /* menu color */
    for(i=0; i<NO_MENU; i++){
        Out_Text_XY(menus[i].col,menus[i].row[0],
            menus[i].item[0].choice);
    }
}


void Display_Menu(int menu_no)
{
```

```
    int i;
    Draw_Rectangle(menus[menu_no].frame[0],menus[menu_no].frame[1],
                    menus[menu_no].frame[2],menus[menu_no].frame[3]);
    for(i=1 ;i<NO_CHOICE; i++)
          Out_Text_XY(menus[menu_no].col,menus[menu_no].row[i],
                    menus[menu_no].item[i].choice);

}

void Select_Menu(int menu_no,int choice_no)
{
    int new_menu_no,
    new_choice_no = 0;
    int i;
    char chr;
   /* int px=0,py=0;*/

    Display_Menu(menu_no);
    Inverse(menu_no,choice_no);

    for(;end==0;)
    { i = Read_Key(0);
      switch (i)
      {   case LEFT_ARROW  : if(menu_no == 0)
                                new_menu_no = NO_MENU-1;
                                else new_menu_no = menu_no-1;
                            Unselect_Menu(menu_no,choice_no);
                            Select_Menu(new_menu_no,1);
                            break;

          case RIGHT_ARROW : if(menu_no == NO_MENU-1)
```

```
                    new_menu_no = 0;
                else new_menu_no = menu_no+1;
                Unselect_Menu(menu_no,choice_no);
                Select_Menu(new_menu_no,1);
                break;

case UP_ARROW    : if(choice_no == 1)
                    new_choice_no
                    = menus[menu_no].last_choice;
                else new_choice_no = choice_no-1;
                Normal(menu_no,choice_no);
                Inverse(menu_no,new_choice_no);
                choice_no = new_choice_no;
                break;

case DOWN_ARROW  : if(choice_no
                    == menus[menu_no].last_choice)
                      new_choice_no = 1;
                else new_choice_no = choice_no+1;
                Normal(menu_no,choice_no);
                Inverse(menu_no,new_choice_no);
                choice_no = new_choice_no;
                break;

case RETURN : switch (menu_no)
            {  case 0 : switch (choice_no)
                      { case 1 : Title();
                                 Display_Main_Menu();
                                 Select_Menu(menu_no,choice_no);
                                 break;
                        case 2 : Do_Choice1();
                                 break;
```

```
                case 3 : closegraph();
                         main_send();
                         Title();
                         Display_Main_Menu();
                         Select_Menu(menu_no,choice_no);
                         break;
                case 4 : closegraph();
                         rfile();/*Do_Choice();*/
                         Title();
                         Display_Main_Menu();
                         Select_Menu(menu_no,choice_no);
                         break;
               };
               break;

        case 1 : switch (choice_no)
               { case 1 : closegraph();
                          modem_char();
                          Title();
                          Display_Main_Menu();
                          Select_Menu(menu_no,choice_no);
                          break;
                 case 2 : closegraph();
                          modem_graph_error();
                          /*modem_send();*/
                          Title();
                          Display_Main_Menu();
                          Select_Menu(menu_no,choice_no);
                          break;
                case 3 : closegraph();
                          modem_recv();
                          Title();
```

```
                    Display_Main_Menu();
                    Select_Menu(menu_no,choice_no);
                    break;
            case 4 : end = 1;
                    break;
            };
            break;
    case 2 : switch (choice_no)
            { case 1 : Do_Choice();
                    break;
            case 2 : Do_Choice();
                    break;
            case 3 : Do_Choice();
                    break;
            case 4 : Do_Choice();
                    break;
            };
            break;
    case 3 : switch (choice_no)
            { case 1 : main_char();
                    Title();
                    Display_Main_Menu();
                    Select_Menu(menu_no,choice_no);
                    break;
            case 2 : send_graph_error();
                    Title();
                    Display_Main_Menu();
                    Select_Menu(menu_no,choice_no);
                    break;
            case 3 : Do_Choice();
                    break;
            };
```

```
                        break;
                case 4 : switch (choice_no)
                        { case 1 : cls();
                                   Title();
                                   Display_Main_Menu();
                                   Select_Menu(menu_no,choice_no);
                                   break;
                          case 2 : Do_Choice1();
                                   break;
                        };
                        break;
                }
                break;
        case ESCAPE : end = 1;
                break;
        } /* switch */
    }     /* for */
}

void Inverse(int menu_no,int choice_no)
{
    Draw_Fill_Rectangle(menus[menu_no].col+2,
                        menus[menu_no].row[choice_no],
                        menus[menu_no].col+124,
                        menus[menu_no].row[choice_no]+10);
        Set_Color(2);/* MENU COLOR */
    Out_Text_XY(menus[menu_no].col,menus[menu_no].row[choice_no],
                menus[menu_no].item[choice_no].choice);
        Set_Color(9);

}
```

```
void Normal(int menu_no,int choice_no)
{
    Erase_Fill_Rectangle(menus[menu_no].col+2,
                         menus[menu_no].row[choice_no],
                         menus[menu_no].col+124,
                         menus[menu_no].row[choice_no]+10);

    Out_Text_XY(menus[menu_no].col,menus[menu_no].row[choice_no],
                menus[menu_no].item[choice_no].choice);
}

void Unselect_Menu(int menu_no,int choice_no)
{
    Normal(menu_no,choice_no);
        Set_Color(0);
    Display_Menu(menu_no);
        Set_Color(9);  /* menu out line block */
}

int Read_Key(int key)
{
    return bioskey(key);
}

void Do_Choice(void)
{
    int x,y;

    x = Get_Max_X()/2;
    y = Get_Max_Y()/2;

    Out_Text_XY(x,y,"Do Choice");
```

```
    delay(400);
        Set_Color(0);
    Out_Text_XY(x,y,"Do Choice");
        Set_Color(9);
}

void Do_Choice1(void)
{
    int x,y;

    x = Get_Max_X()/3;
    y = Get_Max_Y()/2;

    Out_Text_XY(x,y,"MR. PHONGCHAI NILAS");
    delay(1000);
        Set_Color(0);
        Out_Text_XY(x,y,"MR. PHONGCHAI NILAS");
        Set_Color(9);
}

void cursor(int x,int y)
{
  int i,j;
  char n;
  unsigned char hor[9],ver[9];

  i = x;
  j = y;
  for (n=-4; n<5; n++)
  {  hor[n+4] = getpixel(i+n,j);
     putpixel(i+n,j,15-hor[n+4] );
     ver[n+4] = getpixel(i,j+n);
```

```
        putpixel(i,j+n,15-ver[n+4] );
    }
    return;
  }



int huge detect256_320_200(void)
{
  return 0;
  /* 2->640*480  3->800*600  4->1024*768 */
}

int huge detect256_640_480(void)
{
  return 2;
  /* 2->640*480  3->800*600  4->1024*768 */
}

/*
void opengraph(void)
{
  int gd,gm,i;
  installuserdriver("svga256",detect256);
  gd = DETECT;
  initgraph(&gd,&gm,"");
  for(i=0;i<64;i++)
    setrgbpalette(i,i,i,i);
}
*/
void opengraph_320_200(void)
{
```

```
  int gd,gm,i;
  installuserdriver("svga256",detect256_320_200);
  gd = DETECT;
  initgraph(&gd,&gm,"");
  for(i=64;i<128;i++)
    setrgbpalette(i,i-64,i-64,i-64);
}

void opengraph_640_480(void)
{
  int gd,gm,i;
  installuserdriver("svga256",detect256_640_480);
  gd = DETECT;
  initgraph(&gd,&gm,"");
  for(i=64;i<128;i++)
    setrgbpalette(i,i-64,i-64,i-64);
}

void  display(ImgDataFile,xposi,yposi)
unsigned char *ImgDataFile;
int xposi,yposi;
{
 register int x,y;
 unsigned long count;
 count = 0L;
 opengraph_320_200();
 for(y=0;y<YSIZE;y++)
   for(x=0;x<XSIZE;x++)
     {
     putpixel(x+xposi,y+yposi,ImgDataFile[count]>>2);
      count++;
     }
```

```
getch();
closegraph();
}

void main_disp()
{
int x,y,x1,y1;
char fname[20];
unsigned long count;
FILE *FP1;

clrscr();
FP1 = fopen("ama.128","rb");
if(FP1 == NULL)
  {
   printf("Open file fail ! ");
   getch();
  }
count = 0L;
for(y=0;y<YSIZE;y++)
  {
   fread(pict,1,XSIZE,FP1);
   for(x=0;x<XSIZE;x++)
     {
        img[count] = pict[x];
        count++;
     }
  }
x1=64;y1=36;
display(img,x1,y1);
fclose(FP1);
getch();
```

```
 closegraph();
}

void sfile(fname)
char *fname;
{
    FILE *fp;
    unsigned char ch;
    int x,y;
    union {
            char c[2];
            unsigned int count;
          } cnt;
    if(!(fp=fopen(fname,"rb"))) {
          printf("cannot open input file\n");
          exit(1);
    }
    x=0;
    y=0;
    cls();
    closegraph();
    s_f_name(fname);
    wait(PORT);
/*    cnt.count = filesize(fp); */
    cnt.count = 16384;
    sport(PORT,cnt.c[0]);
    wait(PORT);
    sport(PORT,cnt.c[1]);
    opengraph_320_200();
    do
      {
       ch=getc(fp);
```

```
        if(ferror(fp))
            {
             closegraph();
             printf("error reading input file");
             break;
            }
        if(!feof(fp))
            {
             wait(PORT);
             putpixel(x+64,y+36,(ch>>2)+64);
             x++;
             sport(PORT,ch);
             if(x>=XSIZE)
                { y++;
                  x=0;
                }
             if(ferror(fp))
                { closegraph();
                  printf("error writing output file");
                  clearerr(fp);
                  exit(1);
                }
            }
        }while(!feof(fp));
        wait(PORT);
        getch();
        fclose(fp);
        closegraph();
        return;
}
```

```
void rfile()
{
    FILE *fp;
    unsigned char ch;/*,ich[1];*/
    char fname[14];
    int x,y;
/*  unsigned long count;*/
    union {
          char c[2];
          unsigned int count;
          }cnt;

    g_f_name(fname);
    printf("receiving file %s\n",fname);
    remove(fname);
    sport(PORT,'.');
    cnt.c[0] = rport(PORT);
    sport(PORT,'.');
    cnt.c[1] = rport(PORT);
    sport(PORT,'.');

    if(!(fp=fopen(fname,"wb"))){
          printf("cannot open output file\n");
          exit(1);
    }
    opengraph_320_200();
/*  count = 0L;*/
    for(y=0;y<YSIZE;y++)
      for(x=0;x<XSIZE;x++)
          {
             cnt.count--;
             if(cnt.count == 0)
```

```
                break;
            ch = rport(PORT);

/*          *ich = ch; */
/*          fwrite((char *)ich,1,1,fp); */
/*          img[count] = ch;
            count++;*/
            putpixel(x+64,y+36,(ch>>2)+64);
            if(ferror(fp))
              { closegraph();
                printf("error writing output file");
                clearerr(fp);
                exit(1);
              }
             sport(PORT,'.');
        }
    sport(PORT,'.');
    getch();
    fclose(fp);
    closegraph();
    return;
}

unsigned long filesize(fp)

    FILE *fp;
    {
    unsigned long int i;
    i=0;
    do{
        getc(fp);
        i++;
```

```
    }while(!feof(fp));
    rewind(fp);
    return i-1;
}


s_f_name(f)
char *f;
{
    printf("tranmitter witing....\n");
    do{
        sport(PORT,'?');
    }while(!kbhit() && (!(check_stat(PORT)&256)));
    if(kbhit()){
        getch();
        exit(1);
    }

    wait(PORT);
    printf("SENDING....%s\n\n",f);
    while(*f){
            sport(PORT,*f++);
            wait(PORT);
    }

    sport(PORT,'\0');
}


s_f_name1(f)
char *f;
{   int status;
```

```
    bioscom(0, SETTINGS, COM1);
    status = bioscom(3, 0, COM1);
    printf("tranmitter witing....\n");
    do{
        /* sport(PORT,'?');*/
        bioscom(1,'?',PORT);
    }while(!kbhit() && (!(check_stat(PORT)&256)));
    if(kbhit()){
        getch();
        exit(1);
    }

    wait2(PORT);
    printf("SENDING....%s\n\n",f);
    while(*f){
            /*sport(PORT,*f++);*/
            bioscom(1,*f++,PORT);
            wait2(PORT);
    }

    bioscom(1,'\0',PORT);
   /* sport(PORT,'\0');*/
}


void g_f_name(f)
char *f;
{
    printf("receiver waiting....\n");
    while(rport(PORT)!='?'){
      printf("."); }
      sport(PORT,'.');
```

```
    while((*f)=(rport(PORT))){
                                if(*f!='?'){
                                                f++;
                                                sport(PORT,'.');
                                }
    }
}


void g_f_name1(f)
char *f;
{   int out,status;
    bioscom(0, SETTINGS, COM1);
    status = bioscom(3, 0, COM1);
    printf("receiver waiting....\n");
    do{
       out=bioscom(2,0,COM1);
       printf("%c",out);
       /*sport(PORT,'.');*/
       bioscom(1,'.',COM1);

       }while(out!='?');
//    if(out=='?')bioscom(1,'*',COM1);
    while((*f)=(bioscom(2,0,COM1))){
                                if(*f!='?'){
                                                f++;
                                                bioscom(2,0,COM1);
                                               /* sport(PORT,'.');*/
                                }
    }
}
```

```
void wait(port)
int port;
{
    if(rport(port)!='.'){
                                printf("commumnication error\n");
                                exit(1);
    }
}


void wait2(port)
int port;
{   int out,status;
    bioscom(0, SETTINGS, COM1);
    status = bioscom(3, 0, COM1);

    if((out=bioscom(2,0,port))!='.'){
                                printf("commumnication error\n");
                        //      exit(1);
    }               //
}


void sport(port,c)
int port;
char c;
{
    union REGS r;
    r.x.dx = port;
```

```
        r.h.al = c;
        r.h.ah = 1;
        int86(0x14,&r,&r);
        if((r.h.ah)&128){
                        printf("send error detected inserial port ");
                        exit(1);
        }
    }


rport(port)
int port;
{
    union REGS r;
    while(!(check_stat(PORT)&256))
            if(kbhit()){
                        getch();
                        exit(1);
            }
    r.x.dx = port;
    r.h.ah = 2;
    int86(0x14,&r,&r);
    if(r.h.ah & 128){
                    printf("read reeor detected inserial port");
    }
/*      printf("r.h.al=%c..",r.h.al); */
   return r.h.al;

}
```

```
check_stat(port)
int port;
{
    union REGS r;
    r.x.dx = port;
    r.h.ah = 3;
    int86(0x14,&r,&r);
    return r.x.ax;
}


void port_ini(port,code)
int port;
unsigned char code;
{
    union REGS r;
    r.x.dx = port;
    r.h.ah = 0;
    r.h.al = code;
    int86(0x14,&r,&r);
}

void main_send(argc,argv)
int argc;
char *argv[];
{
    closegraph();
    clrscr();
 /*  setfillstyle(1,0);
    bar(0,0,6330,428); */                    /* Background */
    MemAlloc();
/*  if(argc<2) {
```

```
        printf(" USAGE : TRANS S FILENAME OR TRANS R\n");
        exit(1);
    }*/
    printf("FINE TRANFER PROGRAM IN OPERATION.\n");
    printf("TO ABORT , PRESS ANY KEY.\n \n");
    port_ini(PORT,231);
/*    if(tolower(*argv[1]) == 's') sfile(argv[2]);
    else rfile(); */
    argv[2]="ama.128";
    sfile(argv[2]);
/*  main_disp(); */
/*  display(img,64,36); */
    clrscr();
    return;
    closegraph();
}

main()
{


    Title();
    do
    { closegraph();
      opengraph_640_480();
      Menu_Assignment();
      Display_Main_Menu();
      Select_Menu(1,1);
    }while(end==0);
    Close_Graphics_Mode();
    return(0);
}
```

ผลการทดลอง

ตอน 1 การส่ง text file ระหว่างคอมพิวเตอร์โดยผ่านระบบโมเดม

เช่น ข้อมูลจริงส่งคำว่า Instrumentation Ladkrabang

ข้อมูลที่ได้รับพร้อมกับ error เป็น *omq$-(,:pwcsau slieoceweu

ข้อมูลที่ได้รับพร้อมกับแก้ error แล้วเป็น Instrumentation Ladkrabang

ตอน 2 การส่งไฟล์ข้อมูล ระหว่างคอมพิวเตอร์โดยผ่านระบบโมเดม

รูปที่ 1 ไฟล์ข้อมูลจริงที่ส่ง

รูปที่ 2 ไฟล์ข้อมูลที่ได้รับพร้อมกับ error bit

รูปที่ 3 ไฟล์ข้อมูลที่ได้รับพร้อมกับแก้ไข error bit แล้ว