อุปกรณ์เข้ารหัสลับแบบผสม

HYBRID CRYPTOSYSTEM DEVICE

โดย

นายปกรณ์ โตนดแก้ว

นายสุทธิ์เขต เถกิงศรี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

HYBRID CRYPTOSYSTEM DEVICE

BY

MR. PAKON TANODEKEAW

MR. SUTKHET THAKERNGSRI

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIRMENT FOR THE DEGREE OF

BACHELOR IN DEPARTMENT OF INFORMATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2004

หัวข้อปริญญานิพนธ์	อุปกรณ์เข้ารหัสลับแบบผสม	
ชื่อนักศึกษา	นายปกรณ์ โตนดแก้ว	รหัสนักศึกษา 45015854
	นายสุทธิ์เขต เถกิงศรี	รหัสนักศึกษา 45015873
อาจารย์ที่ปรึกษา	อาจารย์กฤดากร กล่อมการ	
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตรบัณฑิต	
	สาขาวิศวกรรมสารสนเทศ	
ภาควิชา	วิศวกรรมสารสนเทศ	
ปีการศึกษา	2547	

ปริญญานิพนธ์ฉบับนี้ได้รับการอนุมัติเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

กฤดากร

.....

(อาจารย์กฤดากร กล่อมการ)

อาจารย์ที่ปรึกษา

หัวข้อปริญญานิพนธ์	อุปกรณ์เข้ารหัสลับแบบผสม	
ชื่อนักศึกษา	นายปกรณ์ โตนดแก้ว	รหัสนักศึกษา 45015854
	นายสุทธิ์เขต เถกิงศรี	รหัสนักศึกษา 45015873
อาจารย์ที่ปรึกษา	อาจารย์กฤดากร กล่อมการ	
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตรบัณฑิต	
	สาขาวิศวกรรมสารสนเทศ	
ภาควิชา	วิศวกรรมสารสนเทศ	
ปีการศึกษา	2547	

บทคัดย่อ

โครงงานนี้แสดงเกี่ยวกับ ระบบการเข้ารหัสที่ซับซ้อนโดยเรียกอุปกรณ์นี้ว่า อุปกรณ์เข้ารหัสแบบผสม โดยเป็นการรวมกันระหว่างสัญญาณเดออสติก และ Pseudo Random Sequence ฟังก์ชันนี้ถูกนำมาใช้สำหรับเป็นคีย์สตีมในการเข้ารหัสโดยการเพิ่มความปลอดภัยด้วยอัลกอริทึมของการเข้ารหัสแบบเดออสติก ซึ่งได้ออกแบบบนคอมพิวเตอร์ โดยทำให้ซับซ้อนขึ้นโดยการใช้ฮาร์ดแวร์แบบง่ายๆ ในส่วนของการติดต่อสื่อสารระหว่างคอมพิวเตอร์ และวงจร Pseudo Random Sequence โดยใช้พอร์ตยูเอสบี โดยใช้เทคนิคการรวมกันระหว่างสัญญาณเดออสติกและ Pseudo Random Sequence ทำให้มีประสิทธิภาพและความซับซ้อนซึ่งจะได้ศึกษาและออกแบบในรายละเอียด

Thesis Title	Hybrid Cryptosystem Device	
Student	Mr. Pakon Tanodekeaw	ID. 45015854
	Mr. Sutkhet Thakerngsri	ID. 45015873
Advisor	Mr. Kitdakorn KlomKarn	
Graduate Level	Bachelor Degree of Information Engineering	
Department	Information Engineering	
Academic Year	2003	

Abstract

This project present about a strong cryptosystem called hybrid encryption device. The combination between chaotic signal and Pseudo Random Sequence: PRS function is introduced for use as a key stream of encryption. The function of PRS is uses to enhance the security of chaotic encryption algorithm which designs on a computer, are implemented by using a simple hardware. In order to communicating between computer and PRS circuit, USB port is employed. The combination technique between chaotic signal and PRS that is effective and complexity are study and design in detail.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ ได้เสร็จสมบูรณ์ไปได้ด้วยดี ตามวัตถุประสงค์ที่ตั้งเอาไว้ทางผู้จัดทำขอขอบพระคุณอาจารย์กฤดากร กล่อมการ ที่คอยให้คำแนะนำและคำปรึกษาดีๆ ในการทำโครงงานนี้ ตลอดจนอาจารย์ท่านอื่นๆ ในภาควิชาวิศวกรรมสารสนเทศ ที่ได้อบรมสั่งสอนตลอด 3 ปี ที่ผ่านมา และเพื่อนๆ 3F/3 ที่ให้การช่วยเหลือในเรื่องต่างๆ ตลอดมาและยังคงความเป็นเพื่อนที่ดีต่อกันเสมอมา และขอขอบพระคุณพ่อและแม่ที่คอยให้การสนับสนุนและให้กำลังใจในทุกๆ เรื่องในการทำโครงงานนี้

สุดท้าย โครงงานนี้คงไม่สามารถเกิดขึ้นได้โดยปราศจากความคิดสร้างสรรค์จากทฤษฎีของทุกๆ ท่านที่มีส่วนร่วมในโครงงานนี้

ปกรณ์ โตนดแก้ว
สุทธิ์เขต เถกิงศรี
ผู้จัดทำ

สารบัญ

สารบัญ(ต่อ)

สารบัญ(ต่อ)

สารบัญรูปภาพ

สารบัญรูปภาพ(ต่อ)

สารบัญรูปภาพ (ต่อ)

หน้า

สารบัญตาราง

บทที่ 1
บทนำ

1.1 แนวคิดและที่มาของปัญหา

ข้อมูลข่าวสารในชีวิตประจำวันของคนเรานั้นมีความสำคัญมากยิ่งขึ้น ไม่ว่าจะเป็นในด้านธุรกิจหรือในด้านทหาร เป็นต้น ถ้าเราจำเป็นต้องส่งข้อมูลไปยังผู้รับก็จะต้องทำการเข้ารหัสข้อมูลเหล่านั้นเอาไว้เพื่อป้องกันมิให้ข้อมูลของเราไว้ถูกลักลอบนำไปใช้งาน

แต่ในปัจจุบันด้วยเทคโนโลยีที่ก้าวหน้าทำให้มีการพยายามที่จะถอดรหัสเพื่อที่จะถูกลักลอบนำเอาข้อมูลไปใช้งานหรือทำให้ข้อมูลต่าง ๆ เกิดความเสียหายได้ ดังนั้นการเข้ารหัสลับจึงอาจจะไม่มีความปลอดภัยมากเท่าที่ควร จึงได้มีแนวคิดที่จะทำการเข้ารหัสโดยอาศัยทั้งซอฟท์แวร์และฮาร์ดแวร์เข้าด้วยกัน จึงทำให้ระบบมีความซับซ้อนยากต่อการถอดรหัสเพราะถ้าไม่มีฮาร์ดแวร์ก็จะไม่สามารถถอดรหัสได้

1.2 จุดประสงค์

1.2.1 เพื่อทำการเข้ารหัสลับแบบผสมโดยใช้ทั้งซอฟท์แวร์และฮาร์ดแวร์ทำการเข้ารหัสลับร่วมกัน

1.2.2 เพื่อทำให้การเข้ารหัสลับมีความซับซ้อนมากขึ้นยากต่อการถอดรหัส

1.2.3 เพื่อป้องกันการลักลอบถอดรหัสโดยปราศจากฮาร์ดแวร์

1.2.4 เพื่อป้องกันการทำซ้ำของซอฟท์แวร์ลิขสิทธิ์

1.3 ขอบเขตของโครงงาน

1.3.1 สร้างอุปกรณ์เข้ารหัสลับแบบผสมโดยใช้ทั้งซอฟท์แวร์และฮาร์ดแวร์ทำการเข้ารหัสลับร่วมกัน

1.3.2 อุปกรณ์เข้ารหัสลับแบบผสมสามารถใช้อัลกอริทึมในการเข้ารหัสลับแบบต่างๆ ได้

1.3.3 ทำการเข้ารหัสและถอดรหัสลับโดยใช้อุปกรณ์เข้ารหัสลับแบบผสมได้

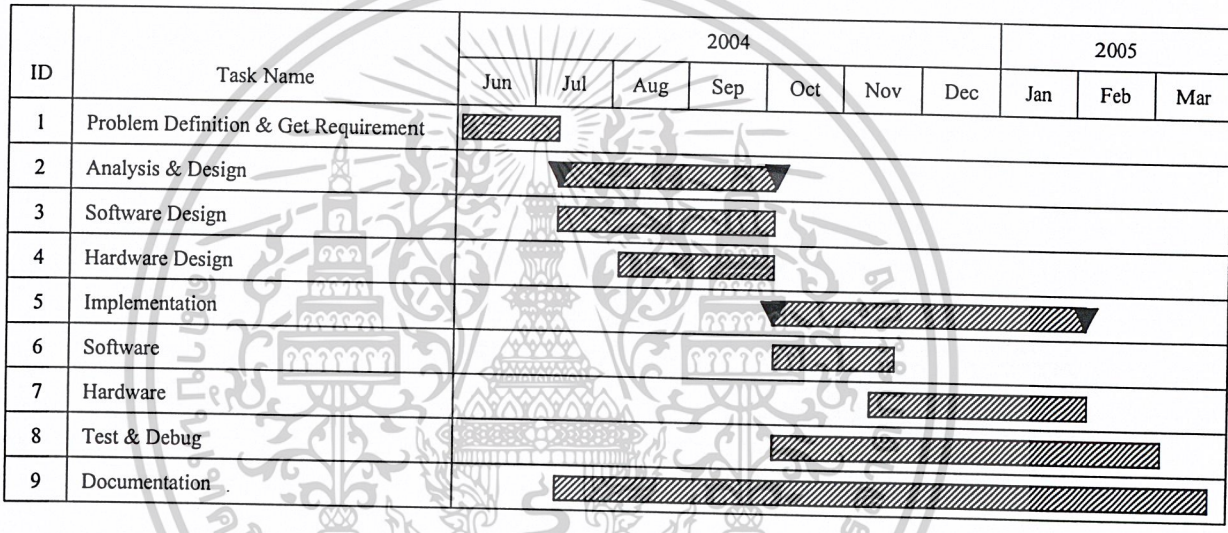
1.4 ผลที่คาดว่าจะได้รับ

1.4.1 จะได้เทคนิคการเข้ารหัสและถอดรหัสลับแบบใหม่โดยต้องอาศัยทั้งซอฟท์แวร์และฮาร์ดแวร์ทำงานร่วมกันจึงทำให้ระบบมีความซับซ้อนมากยิ่งขึ้น

1.4.2 เพิ่มความปลอดภัยให้แก่ข้อมูลข่าวสารทำให้ผู้อื่นไม่สามารถลักลอบถอดรหัสได้

1.4.3 นำความรู้เกี่ยวกับการเข้ารหัสที่ได้ไปประยุกต์ใช้กับงานอื่นๆ ได้ในอนาคต

1.5 ขั้นตอนการดำเนินงาน



บทที่ 2
ทฤษฎีที่เกี่ยวข้องกับโครงงาน

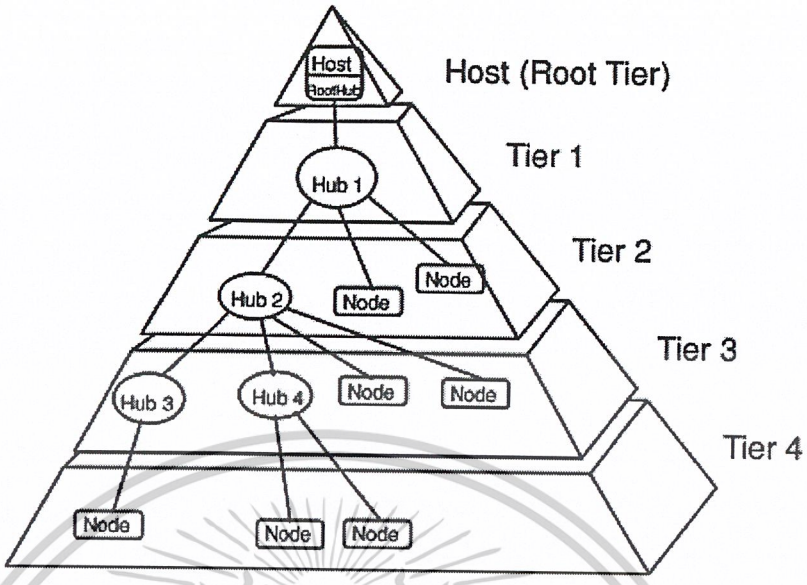
โครงงานอุปกรณ์เข้ารหัสลับแบบผสม นี้ประกอบด้วย การเข้ารหัสบนซอฟท์แวร์และฮาร์ดแวร์ที่เชื่อมต่อกับพอร์ตยูเอสบี ดังนั้นในบทนี้จะขอกล่าวถึงทฤษฎีของระบบยูเอสบีและอัลกอริทึมการเข้ารหัสในรูปแบบต่างๆ

2.1 ระบบบัสยูเอสบี

ระบบบัสยูเอสบีเป็นระบบบัสที่ถูกออกแบบมาให้มีความง่ายในการเชื่อมต่อ และมีประสิทธิภาพในการสื่อสารข้อมูลกับอุปกรณ์รอบข้างหลายๆ ชนิด การเชื่อมต่ออุปกรณ์ยูเอสบีเข้ากับระบบยังสามารถทำได้ในขณะที่เครื่องคอมพิวเตอร์ยังคงทำงานอยู่ได้ ซึ่งการใช้งานในลักษณะนี้คือรูปแบบของระบบปลั๊กแอนด์เพลย์ (Plug and Play) และในส่วนของการเพิ่มจำนวนพอร์ตการสื่อสารข้อมูลก็ทำได้ ด้วยการใช้ยูเอสบีฮับ (USB Hub) มาต่อพ่วงเข้ากับระบบ

2.1.1 โครงสร้างของระบบบัสยูเอสบี

โครงสร้างการจัดการเชื่อมต่อบนระบบบัสยูเอสบีจะเป็นแบบไทร์สตาร์ (Tiered Star) หรือสตาร์แบบลำดับชั้น (รูปที่ 2.1) โครงสร้างนี้ประกอบไปด้วยโครงสร้างแบบสตาร์หลายๆชุดเชื่อมต่อกันอยู่ โดยมีฮับเป็นจุดศูนย์กลางของโครงสร้างย่อย ๆ เหล่านั้น ที่ส่วนบนสุดของสตาร์จะเป็นโฮสคอนโทรลเลอร์ ซึ่งในระบบบัสยูเอสบีจะมีโฮสเพียงตัวเดียวเท่านั้นที่ควบคุมการติดต่อสื่อสารทั้งหมดของระบบและมีฮับเป็นตัวเพิ่มจำนวนของพอร์ตที่ใช้เชื่อมต่อกับอุปกรณ์รอบข้าง จุดปลายแต่ละจุดของโครงสร้างสตาร์จะเป็นอุปกรณ์รอบข้างที่ต่อเข้ากับพอร์ตใดพอร์ตหนึ่งบนฮับหรืออาจนำฮับมาเชื่อมต่อแทนอุปกรณ์เพื่อเพิ่มจำนวนพอร์ต ซึ่งความสามารถในการเชื่อมต่อฮับเข้าด้วยกันนี้จึงทำให้เกิดโครงสร้างย่อยได้หลายชุดและมีลักษณะเป็นลำดับชั้น สตาร์แบบลำดับชั้นเป็นเพียงการอธิบายการเชื่อมต่อทางกายภาพเท่านั้น แต่ในความเป็นจริงนั้นระบบบัสยูเอสบีจะมีเส้นทางข้อมูลสำหรับการสื่อสารเพียง 1 เส้นทางและโฮสกับอุปกรณ์ทุกตัวที่ต่ออยู่ในระบบจะส่งถ่ายข้อมูลโดยอาศัยเส้นทางดังกล่าวเท่านั้น



รูปที่ 2.1 โครงสร้างการเชื่อมต่อบนระบบบัสยูเอสบี

2.1.2 ความเร็วการสื่อสารข้อมูลของอุปกรณ์ยูเอสบี

ตารางที่ 2.1 แสดงความเร็วในการสื่อสารข้อมูลของอุปกรณ์ยูเอสบี

ความเร็ว	สำหรับอุปกรณ์
Low speed	เมาส์ , คีย์บอร์ด
Full speed	อุปกรณ์เกี่ยวกับเสียง, สัญญาณภาพที่มีการบีบอัด
High speed	มัลติมีเดียภาพและเสียง, ฮาร์ดดิสก์

2.1.3 โพรโตคอลของระบบบัสยูเอสบี

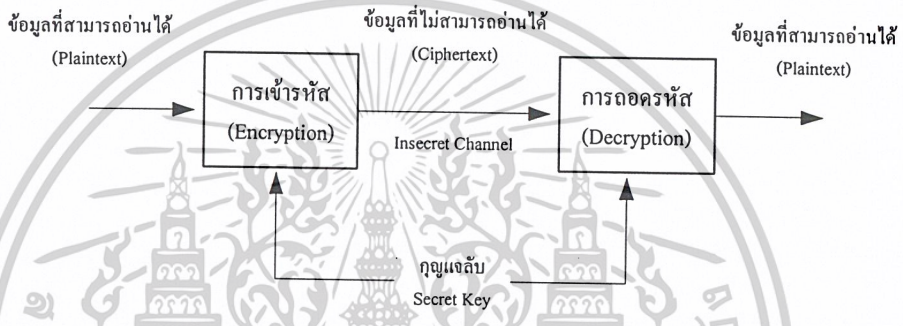
การเชื่อมต่อบนระบบบัสยูเอสบีจะต่างจากการเชื่อมต่อพอร์ตแบบอนุกรมหรือ RS-232 ซึ่งไม่มีการกำหนดรูปแบบของการส่งข้อมูลอย่างเคร่งครัด โดยการสื่อสารข้อมูลบนระบบบัสยูเอสบีจะสร้างขึ้นจากโพรโตคอลที่ค่อนข้างและมีข้อกำหนดที่เคร่งครัด แต่ในความเป็นจริงแล้วการทำงานในส่วนต่างๆ ของโพรโตคอลนี้จะถูกจัดการอยู่ภายในไอซีคอนโทรลเลอร์ยูเอสบีอย่างสมบูรณ์ โพรโตคอลยูเอสบีประกอบขึ้นจากทรานแซคชัน (Transaction) หลายชุดซึ่งแต่ละชุดจะประกอบไปด้วย โทเค็นแพ็คเก็ต (Token Packet), ดาต้าแพ็คเก็ต (Data packet), สเตตัสแพ็คเก็ต (Status packet)

2.2 รูปแบบของกุญแจในการเข้ารหัสลับ

รูปแบบของกุญแจในการเข้ารหัสลับโดยทั่วๆ ไปจะมีอยู่ 2 รูปแบบ คือ Symmetric key และ Asymmetric key Encryption ซึ่งมีรายละเอียดดังนี้

2.2.1 การเข้ารหัสแบบกุญแจสมมาตร (Symmetric-key Cryptography หรือ Secret key Encryption)

เป็นการเข้ารหัสและถอดรหัสโดยใช้กุญแจตัวเดียวกัน ซึ่งจะเรียกกุญแจตัวนี้ว่า กุญแจลับ (Secret key) ซึ่งแสดงได้ดังรูปที่ 2.2



รูปที่ 2.2 การเข้ารหัสแบบกุญแจลับ (Secret key)

ข้อดีของการเข้ารหัสแบบสมมาตร

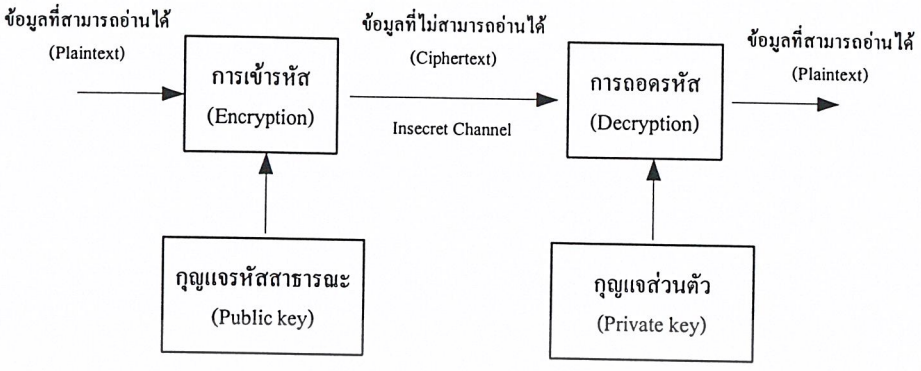
- การเข้ารหัสและถอดรหัสจะใช้เวลาน้อย (ขึ้นอยู่กับอัลกอริทึมที่เลือกใช้)
- ขนาดของข้อมูลหลังจากทำการเข้ารหัสแล้วเปลี่ยนแปลงไปไม่มาก

ข้อด้อยของการเข้ารหัสแบบสมมาตร

- การจัดการกุญแจลับค่อนข้างยุ่งยากเพราะใช้กุญแจตัวเดียวกันในการเข้ารหัสและถอดรหัส

2.2.2 การเข้ารหัสแบบกุญแจอสมมาตร (Asymmetric key cryptography or Public key)

ในการเข้ารหัสและถอดรหัสจะใช้กุญแจคนละตัวกัน โดยในการเข้ารหัสลับจะใช้กุญแจสาธารณะ (Public key) ซึ่งสามารถแจกจ่ายให้กับทุกคนเพื่อใช้ในการเข้ารหัส แต่จะมีเพียงคนเดียวที่มีกุญแจส่วนตัว (Private key) ที่สามารถทำการถอดรหัสลับได้ และจะต้องเก็บกุญแจส่วนตัวไว้เป็นความลับ ซึ่งแสดงได้ดังรูปที่ 2.3



รูปที่ 2.3 การเข้ารหัสแบบกุญแจสาธารณะ (Public key)

ข้อดีของการเข้ารหัสแบบกุญแจอสมมาตร

- การจัดการกุญแจทำได้ง่าย เพราะใช้กุญแจในการเข้ารหัสและถอดรหัสต่างกัน
- สามารถทำการพิสูจน์ตัวตนได้โดยใช้ร่วมกับลายมือชื่อดิจิตอล

ข้อด้อยของการเข้ารหัสแบบกุญแจอสมมาตร

- ใช้เวลาในการเข้ารหัสและถอดรหัสค่อนข้างนาน เพราะมีการคำนวณที่ซับซ้อน
- ขนาดของข้อมูลหลังจากทำการเข้ารหัสแล้วมีการเปลี่ยนแปลงไปมาก

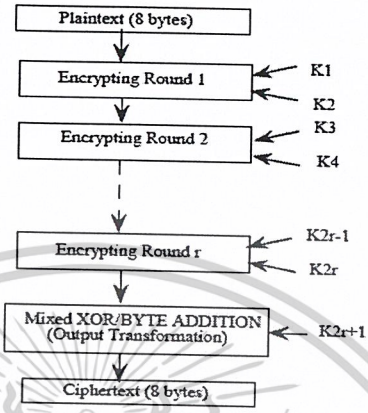
2.3 อัลกอริทึมการเข้ารหัสลับ SAFER K-64 [4]

ในการเข้ารหัสลับจะใช้ SAFER K-64 เนื่องจากใช้เวลาในการเข้ารหัสน้อยและอัลกอริทึมไม่ซับซ้อนซึ่งง่ายต่อการศึกษาและการนำมาใช้ โดย SAFER K-64 เป็นการเข้ารหัสแบบกุญแจสมมาตร และมีรูปแบบ Byte-Oriented โดยที่จำนวนตัวเลขด้านหลัง คือ ขนาดของค่ากุญแจ มีขนาดของบล็อกเป็น 8 ไบต์ (64 บิต) SAFER K-64 เป็นรูปแบบของการเข้ารหัสที่แสดงการเปลี่ยนแปลงของแต่ละรอบ จากนั้นจะได้ข้อมูลที่เข้ารหัสลับแล้วออกมา

2.3.1 การเข้ารหัสลับของ SAFER K-64

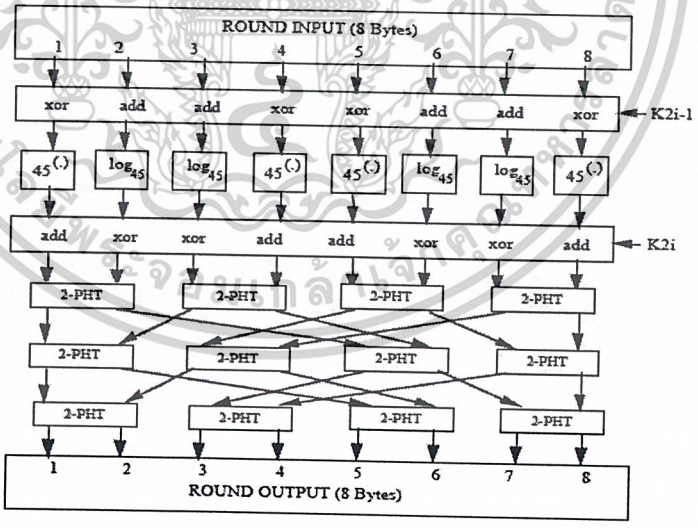
โครงสร้างการเข้ารหัสของ SAFER K-64 ถูกแสดงในรูปที่ 2.4 อัลกอริทึมการเข้ารหัสประกอบด้วย Plaintext คือ ข้อมูลที่จะนำไปเข้ารหัส โดยเอาท์พุทของการเข้ารหัส คือ ข้อมูลที่ถูก

เข้ารหัส (Ciphertext) โดยกำหนดรอบการเข้ารหัสเท่ากับ 6 รอบ ซึ่งแต่ละรอบจะใช้กุญแจ 2 ตัว ขนาด 8 ไบต์



รูปที่ 2.4 โครงสร้างการเข้ารหัสของ SAFER K-64

โดยภายในรอบของการเข้ารหัสของ SAFER K-64 ถูกแสดงในรูปที่ 2.5



รูปที่ 2.5 โครงสร้างภายในของการเข้ารหัสของ SAFER K-64

จากรูปที่ 2.5 โครงสร้างของการเข้ารหัสจะประกอบด้วยกุญแจย่อย K2i ในขั้นตอนการเข้ารหัสจะประกอบด้วย การบวก byte-by-byte (modulo-256 addition) ของไบต์ 1, 4, 5 และ 8 ด้วย

กุญแจย่อย K2i สำหรับการ bit-by-bit XOR (modulo-2sum) ของไบต์ 2, 3, 6 และ 7 ด้วยกุญแจย่อย K2i จากนั้นจึงผ่านฟังก์ชัน 2-PHT โดยแสดงฟังก์ชันการทำงานดังนี้

XOR เป็นฟังก์ชัน bit-by-bit XOR (modulo-2sum) ของแต่ละบิต

ADD เป็นฟังก์ชัน byte-by-byte (modulo-256 addition) ของแต่ละบิต

EXP เป็นฟังก์ชัน exptab(x) = 45^x modulo 257 โดยกำหนดให้ exptab(128) = 0

LOG เป็นฟังก์ชัน logtab(x) = $\log_{45}$ โดยกำหนดให้ logtab(0) = 128

SUB เป็นฟังก์ชัน Mixed Byte-Subtraction/XOR (modulo-256 subtraction) อินเวิร์ดค่าฟังก์ชัน ADD

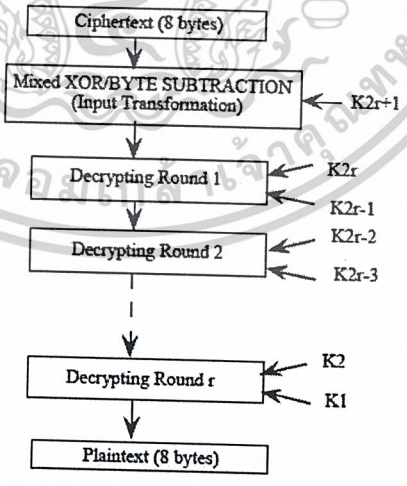
ฟังก์ชัน 2-PHT จะมีรูปแบบสมการดังนี้

$$b1 = 2a1 + a2$$

$$b2 = a1 + a2 \qquad (1)$$

2.3.2 การถอดรหัสลับของ SAFER K-64

โครงสร้างการถอดรหัสของ SAFER K-64 ถูกแสดงในรูปที่ 2.6 ซึ่งอัลกอริทึมในการถอดรหัสจะประกอบด้วย บล็อกข้อมูลที่ถูกเข้ารหัส (Block Cipher) ขนาด 8 ไบต์



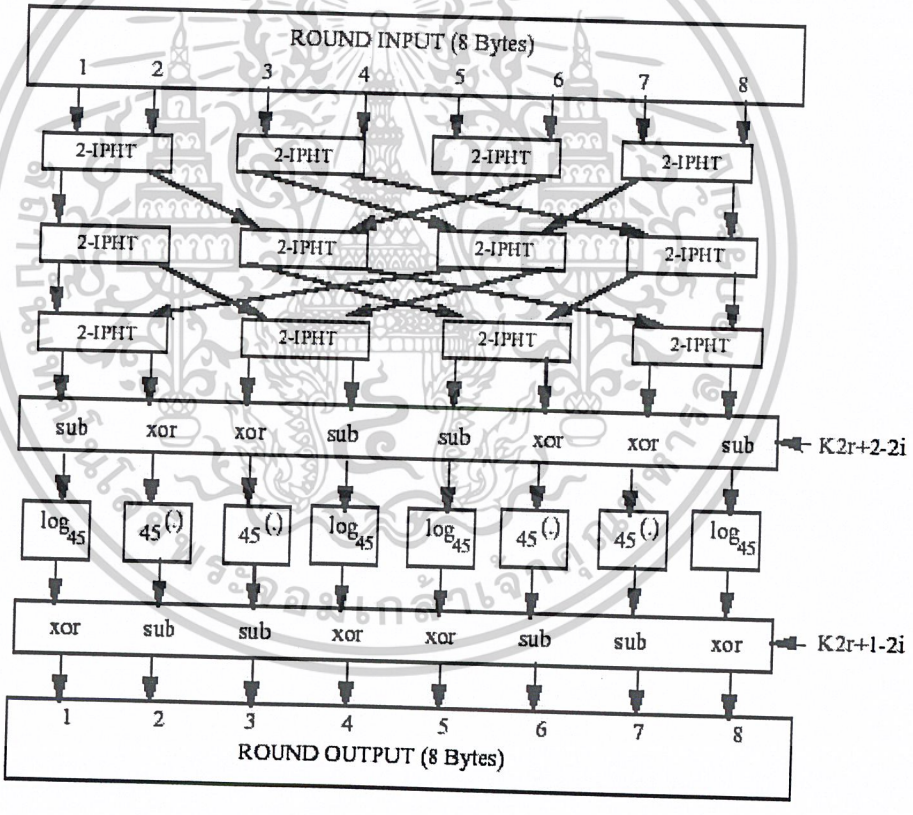
รูปที่ 2.6 โครงสร้างการถอดรหัสของ SAFER K-64

จากรูปที่ 2.6 การถอดรหัสจะประกอบด้วยฟังก์ชัน Mixed XOR/Byte-Subtraction โดยใช้กุญแจย่อย K2r+1 จากนั้นจะผ่านรอบการถอดรหัส ซึ่งจะประกอบด้วยกุญแจย่อย

ฟังก์ชัน 2-IPHT จะมีรูปแบบสมการดังนี้

$$a1 = b1 - b2$$
$$a2 = -b1 + 2b2 \qquad (2)$$

จากสมการที่ (2) Inverse PHT (2-IPHT) เป็นการคำนวณกลับของ 2-PHT ซึ่งโครงสร้างของรอบการถอดรหัส SAFER K-64 ถูกแสดงในรูปที่ 2.7



รูปที่ 2.7 โครงสร้างรอบการถอดรหัสของ SAFER K-64

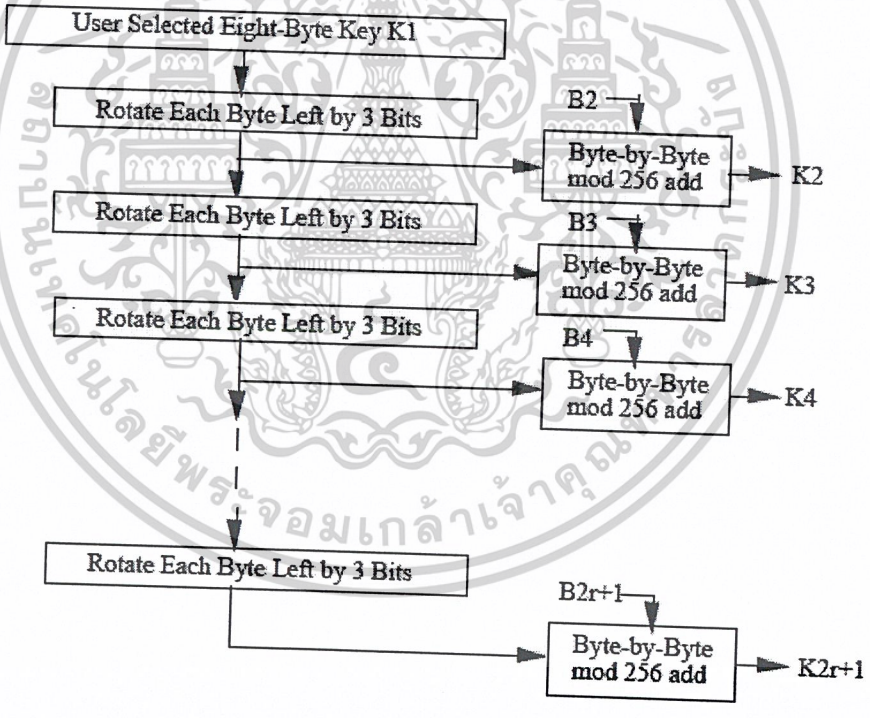
จากรูปที่ 2.7 ภายในรอบการถอดรหัสจะผ่านฟังก์ชัน 2-IPHT, Mixed Byte-Subtraction/XOR และ inverse linear layer โดยภายในฟังก์ชัน Mixed Byte-Subtraction/XOR ซึ่งประกอบด้วย byte-by-byte เป็นการคำนวณโดยวิธีการลบ (modulo-256 subtraction) ของไบต์ 1, 4,

5 และ 8 ด้วยกุญแจย่อย K2r+2-2i และ bit-by-bit XOR (modulo-2 sum) ของไบต์ 2, 3, 6 และ 7 ด้วยกุญแจย่อย K2r+2-2i

โดยขั้นตอนสุดท้ายภายในรอบของการถอดรหัส คือ Mixed XOR/Byte-Subtraction โดยใช้กุญแจย่อย K2r+1-2i ซึ่งประกอบด้วย bit-by-bit XOR (modulo-2 sum) ของไบต์ 1, 4, 5 และ 8 โดยใช้กุญแจย่อย K2r+1-2i และเช่นเดียวกับ byte-by-byte เป็นการคำนวณโดยวิธีการลบ (modulo-256 subtraction) ของไบต์ 2, 3, 6 และ7 โดยใช้กุญแจย่อย K2r+1-2i

2.3.3 ตารางกุญแจสำหรับ SAFER K-64

ตารางกุญแจสำหรับ SAFER K-64 มีฟังก์ชันสำหรับสร้างกุญแจย่อย K2, K3,..., K2r+1 โดยผู้ใช้จะกำหนดกุญแจ K1 เพื่อใช้ในการสร้างกุญแจย่อย ดังแสดงในรูปที่ 2.8



รูปที่ 2.8 แสดงตารางกุญแจสำหรับ SAFER K-64

จากรูปที่ 2.8 แสดงวิธีที่กุญแจ K1 ถูกใช้สร้างกุญแจย่อยเพิ่ม 64 บิต K2, K3,..., K2r+1 ซึ่งถูกเรียกใช้ภายในรอบของการเข้ารหัสและถอดรหัสของอัลกอริทึม SAFER K-64 ในกระบวนการสร้างกุญแจย่อย แสดงเป็นไบต์ที่ถูกหมุนไปทางซ้าย 3 บิต ในระหว่างรอบการสร้างกุญแจย่อย

จุดประสงค์ในการออกแบบตารางกุญแจ คือ จะทำให้แยกออกจากกันเป็นอิสระจากความซับซ้อนด้วยการหมุนและการคำนวณเพื่อเพิ่มกุญแจย่อยภายในตารางกุญแจ SAFER K-64

การกำหนดกุญแจย่อยจะมีรูปแบบสมการดังนี้

$$b[i, j] = 45^{**}[45^{**}(9i+j) \bmod 257] \bmod 257 \quad (3)$$

ข้อดีของ SAFER K-64

- การเข้ารหัสและถอดรหัสมีความรวดเร็วและอัลกอริทึมไม่ค่อยซับซ้อน
- มีลักษณะยืดหยุ่นตามลักษณะการใช้งาน

ข้อจำกัดของ SAFER K-64

- ไม่มีการพิสูจน์ที่สมบูรณ์แบบด้านความปลอดภัย
- การเข้ารหัสและการถอดรหัสมีความแตกต่างกัน

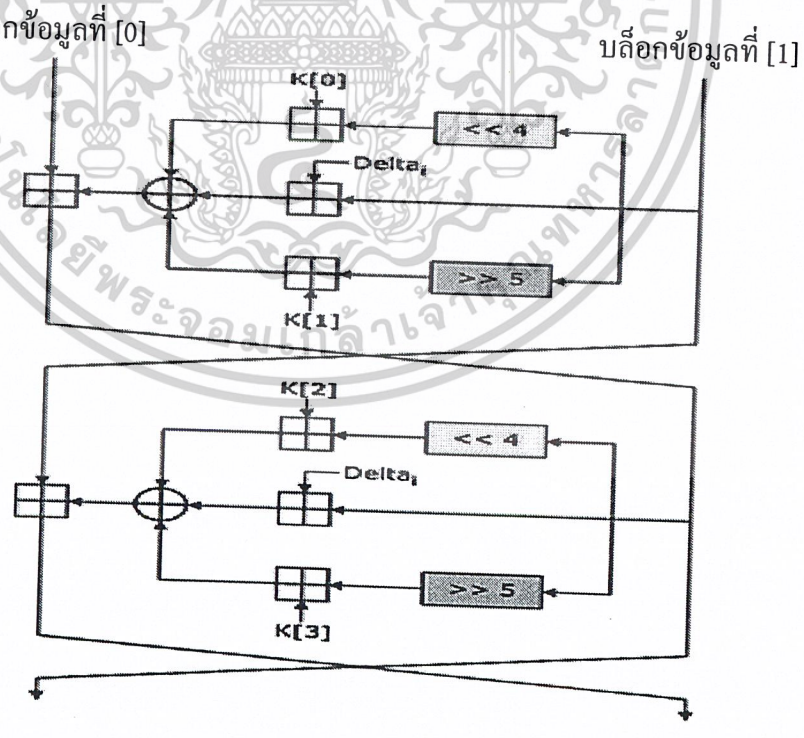
2.4 อัลกอริทึมการเข้ารหัสลับแบบ Tiny Encryption Algorithm: TEA [5]

อัลกอริทึมในการเข้ารหัสลับถัดมาจะเป็นแบบ Tiny Encryption Algorithm ซึ่งเป็นอัลกอริทึมง่ายๆ ซึ่งสามารถถูกดัดแปลงไปได้หลายภาษา และมีขนาดเล็กกะทัดรัดพอที่จะถูกโปรแกรมลงในไมโครคอนโทรลเลอร์ได้

โปรแกรมเข้ารหัสลับ Tiny Encryption Algorithm โดยใช้กุญแจ k [0] - k [3] และบล็อกข้อมูล v [0] - v [1]

```
void code(long* v, long* k) {
        unsigned long y=v[0],z=v[1], sum=0, /* set up */
        delta=0x9e3779b9, /* a key schedule constant */
        n=32 ;
        while (n-->0) { /* basic cycle start */
                sum += delta ;
                y += ((z<<4)+k[0]) ^ (z+sum) ^ ((z>>5)+k[1]) ;
                z += ((y<<4)+k[2]) ^ (y+sum) ^ ((y>>5)+k[3]) ;
        } /* end cycle */
        v[0]=y ; v[1]=z ; }
```

โดยอัลกอริทึม Tiny Encryption Algorithm มีโครงสร้างแบบ Feistel ซึ่งเป็นรูปแบบเฉพาะ ดังนั้นโครงสร้างแต่ละรอบของการเข้ารหัสจะประกอบด้วย กุญแจ, บล็อกข้อมูลและค่าคงที่ ซึ่งแสดงได้ดังรูปที่ 2.9



รูปที่ 2.9 แสดงการเข้ารหัสลับของ Tiny Encryption Algorithm

จากรูปที่ 2.9 โดยบล็อกข้อมูลจะถูกแบ่งออกเป็น 64 บิต โดยในส่วนของบล็อกข้อมูลที่ [1] จะถูกแยกการเข้ารหัสดังนี้ ส่วนที่ 1 ทำการเลื่อนไปทางซ้าย 4 บิตแล้วนำไปบวกกับบล็อกกุญแจที่ [0] ส่วนที่ 2 บวกกับค่าคงที่ ส่วนที่ 3 ทำการเลื่อนไปทางขวา 5 บิตแล้วนำไปบวกกับบล็อกกุญแจที่ [1] แล้วนำมาทำการ XOR กันแล้วนำไปบวกกับบล็อกข้อมูลที่ [0] ส่วนของบล็อกข้อมูลที่ [0] จะถูกแยกการเข้ารหัสดังนี้ ส่วนที่ 1 ทำการเลื่อนไปทางซ้าย 4 บิตแล้วนำไปบวกกับบล็อกกุญแจที่ [2] ส่วนที่ 2 บวกกับค่าคงที่ ส่วนที่ 3 ทำการเลื่อนไปทางขวา 5 บิตแล้วนำไปบวกกับบล็อกกุญแจที่ [3] แล้วนำมาทำการ XOR กันแล้วนำไปบวกกับบล็อกข้อมูลที่ [1]

สมการที่ (4) ใช้ในการหาค่าคงที่

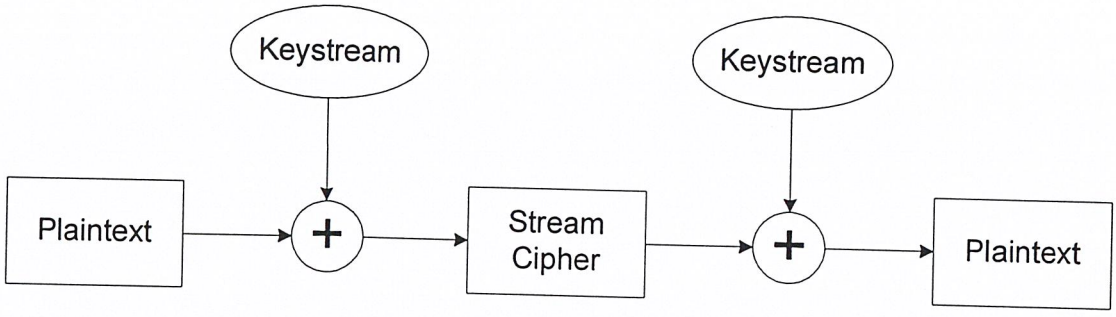
$$Delta = (\sqrt{5} - 1)2^{31} \tag{4}$$

โปรแกรมถอดรหัสลับ Tiny Encryption Algorithm โดยใช้กุญแจ k [0] - k [3] และบล็อกข้อมูล v [0] - v [1]

```
void decode(long* v,long* k) {
        unsigned long n=32, sum, y=v[0], z=v[1],
        delta=0x9e3779b9 ;
        sum=delta<<5 ;
        /* start cycle */
        while (n-->0) {
                z-= ((y<<4)+k[2]) ^ (y+sum) ^ ((y>>5)+k[3]) ; Z = shift y 4 bit
                y-= ((z<<4)+k[0]) ^ (z+sum) ^ ((z>>5)+k[1]) ;
                sum-=delta ; }
        /* end cycle */
        v[0]=y ; v[1]=z ; }
```

2.5 Blum-Blum-Shub Generator [6]

การเขียนโปรแกรมเข้ารหัสลับในไมโครคอนโทรลเลอร์ จะใช้ในการสร้างสัญญาณ Pseudo random sequence ซึ่งจะเลือกใช้แบบ Blum-Blum-Shub Generator เนื่องจากสามารถสร้างสัญญาณ Pseudo random sequence ได้ดี โดยในการเข้ารหัสแบบ Blum-Blum-Shub Generator จะเป็นการสร้างคีย์สตรีมที่ใช้ในการเข้ารหัสโดยที่จะนำคีย์สตรีมที่ได้ไปบวกเข้ากับตัวอักษรที่จะนำไปเข้ารหัสจากนั้นจะได้ Stream Cipher ออกมา โดยมีการทำงานดังรูปที่ 2.10



รูปที่ 2.10 แสดงการเข้ารหัสแบบ Stream Cipher โดยใช้ Blum-Blum-Shub Generator

หลักการทำงานของ Blum-Blum-Shub Generator จะเริ่มจากการเลือกค่า Prime ขึ้นมา 2 ค่า p, q โดยทั้งคู่ต้องสอดคล้องกับ 3 mod 4 และคำนวณหาค่า N = p*q และกำหนดค่าเริ่มต้น S_0 เพื่อที่จะคำนวณลำดับ a_p, s_2, s_3... โดยจะใช้สมการที่ (5)

$$S_{i+1} = S^2_i \bmod N \tag{5}$$

จากลำดับของตัวเลขในช่วง 0, 1, 2,..., n-1 เราจะสร้างลำดับของ pseudo-random บิตโดยสมการที่ (6)

$$b_i = S_i \bmod 2 \tag{6}$$

ค่า Prime p จะเท่ากับ 3 modulo 4 สำหรับ z ยกกำลัง modulo p จะได้จากสมการที่ (7)

$$y = z^{(p+1)/4} \bmod p \tag{7}$$

จากหลักการทำงานของ Blum-Blum-Shub Generator ข้างต้นเราจะนำมาใช้ในการเขียนโปรแกรมเข้ารหัสลับในไมโครคอนโทรลเลอร์ เนื่องจากมีความเหมาะสมในการเขียนโปรแกรมและสร้างสัญญาณ Pseudo random sequence ได้ดี

2.6 อัลกอริทึมการเข้ารหัสลับแบบ RSA (Rivest-Shamir-Adelman Encryption)

อัลกอริทึมการเข้ารหัสลับแบบ RSA เป็นการเข้ารหัสแบบกุญแจสาธารณะ โดยจะใช้กุญแจสาธารณะ (Public key) ในการเข้ารหัสและถอดรหัสด้วย กุญแจส่วนตัว (Private key) โดยหลักการในการเข้ารหัสลับแบบนี้คือ ความยากในการแยกส่วนประกอบที่เป็นค่า Prime และสามารถนำไปประยุกต์ใช้งานอื่นๆ ได้ โดยมีหลักการทำงานดังนี้

2.6.1 การสร้างกุญแจ (Key Generator)

- เลือกจำนวนเฉพาะ (Primes) p และ q
- คำนวณค่า N (modulo) จาก p*q
- คำนวณหา $\Phi(N) = (p-1)(q-1)$
- เลือกค่า e (กุญแจสาธารณะ) ที่เป็น relative prime โดยที่ $e < \Phi(N)$ และ GCD $(e, \Phi(N)) = 1$
- คำนวณหาค่า d (กุญแจส่วนตัว) จากสมการที่ (8)

$$d = e^{-1} \bmod \Phi(N) \tag{8}$$

โดยในการเข้ารหัสลับด้วย RSA จะต้องเก็บ p และ q ไว้เป็นความลับ

2.6.2 การเข้ารหัสลับแบบ RSA

นำข้อมูลที่ต้องการเข้ารหัสยกกำลังด้วยกุญแจสาธารณะ (e) และ modulo ด้วยค่า N จากสมการที่ (9)

$$C = M^e \bmod N \tag{9}$$

โดยที่ M คือ ข้อมูลที่ใช้ในการเข้ารหัส (Plaintext)

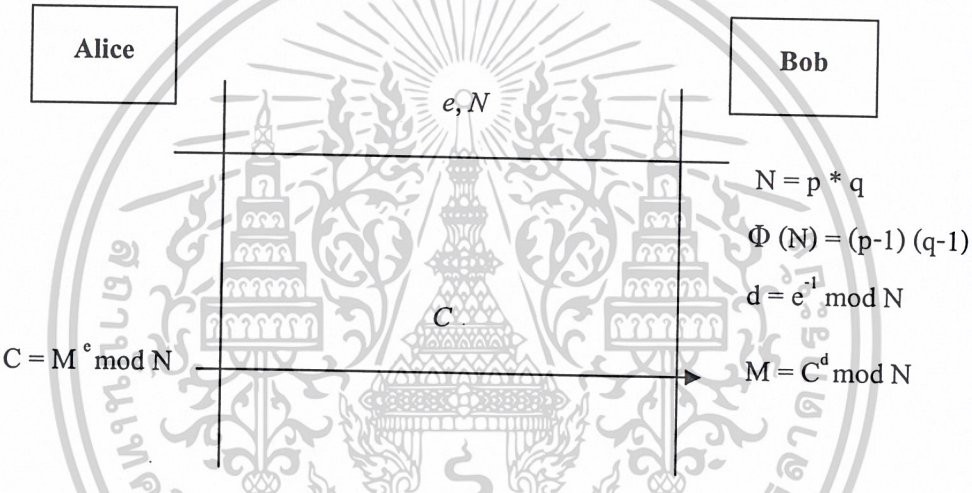
C คือ ข้อมูลที่เข้ารหัสแล้ว (Ciphertext)

2.6.3 การถอดรหัสลับแบบ RSA

กระบวนการถอดรหัสจะทำเช่นเดียวกับการเข้ารหัสโดยการนำข้อมูลที่เข้ารหัสแล้วยกกำลังด้วยกุญแจส่วนตัว (d) และ modulo ด้วยค่า N ดังสมการที่ (10)

$$M = C^d \bmod N \tag{10}$$

เนื่องจากกระบวนการในการเข้ารหัสข้อมูลและการถอดรหัสข้อมูลใช้วิธีการเดียวกันแต่ใช้กุญแจที่ต่างกัน เราจึงสามารถที่จะเปลี่ยนใช้ e เป็นกุญแจส่วนตัวและใช้ d เป็นกุญแจสาธารณะได้



รูปที่ 2.11 แสดงโพรโตคอลการเข้ารหัสลับของ RSA

จากรูป (2.11) แสดงการติดต่อระหว่าง Alice กับ Bob โดยใช้โพรโตคอลการเข้ารหัสลับของ RSA โดยที่ Alice ต้องการส่งข้อมูลไปยัง Bob โดยจะมีขั้นตอนการติดต่อสื่อสารดังนี้ Bob จะทำการประกาศกุญแจสาธารณะ (e) และ N จากนั้น Alice จะนำกุญแจสาธารณะ (e) และ N มาใช้ในการเข้ารหัสลับและส่งไปให้กับ Bob จากนั้น Bob จะนำข้อมูลที่เข้ารหัสลับมาทำการถอดรหัสโดยใช้กุญแจส่วนตัว (d)

ข้อดีของ RSA

- มีความปลอดภัยสูง
- การจัดการกุญแจทำได้ง่าย เพราะใช้กุญแจในการเข้ารหัสและถอดรหัสต่างกัน
- สามารถทำการพิสูจน์ตัวตนได้โดยใช้ร่วมกับลายมือชื่อดิจิตอล

ข้อเสียของ RSA

- ใช้เวลาในการเข้ารหัสและถอดรหัสค่อนข้างนานเพราะมีการคำนวณที่ซับซ้อน

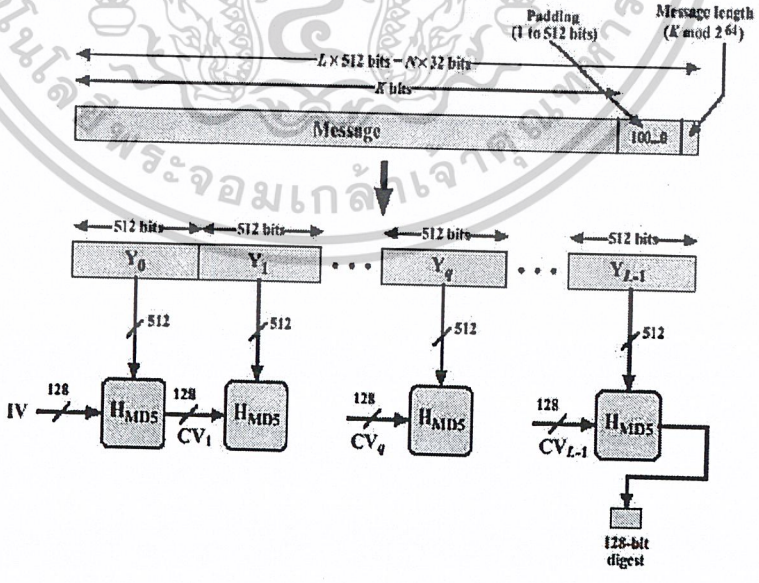
2.7 อัลกอริทึมการเข้ารหัสแบบ MD5 [7]

อัลกอริทึมการเข้ารหัส MD5 เป็นรูปแบบฟังก์ชันแฮช (hash function) โดยกระบวนการในการเข้ารหัสจะเป็นแบบทางเดียว (one way function) ซึ่งอินพุทจะถูกแบ่งออกเป็นบล็อกขนาด 512 บิต และเมื่อผ่านขั้นตอนทั้งหมดแล้วจะได้ข้อมูลขนาด 128 บิต ออกมาเรียกว่า เมสเซสไดเจสต์ (Messages digest)

โดยขั้นตอนการทำงานหากอินพุทเกิดการเปลี่ยนแปลงแม้แต่ตัวอักษรเดียว (หรือแค่บิตเดียว) ก็จะทำให้เมสเซสไดเจสต์เปลี่ยนแปลงไปอย่างมาก ซึ่งการที่จะหาข้อมูล 2 ชุดที่แตกต่างกันและได้ค่าเมสเซสไดเจสต์ที่เหมือนกันนั้นทำได้ยากมาก

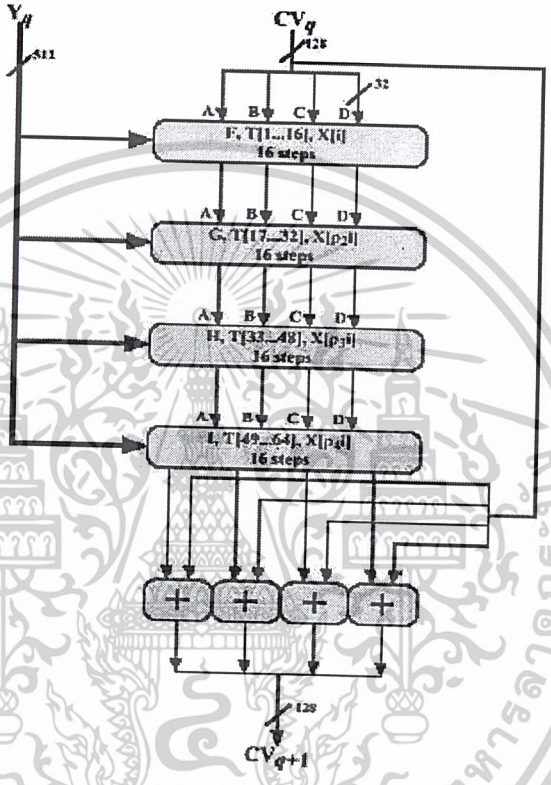
2.7.1 ขั้นตอนการเข้ารหัสของอัลกอริทึม MD5

นำข้อมูลที่ต้องการเข้ารหัสแบ่งออกเป็นบล็อกขนาด 512 บิต จากนั้นกำหนดเมสเซสไดเจสต์ขนาด 128 บิตแล้วนำข้อมูลผ่านขั้นตอนการทำงานของ MD5 เมื่อเสร็จสิ้นขั้นตอนจะได้เอาท์พุทเป็นเมสเซสไดเจสต์ขนาด 128 บิตซึ่งแสดงได้ดังรูป 2.12

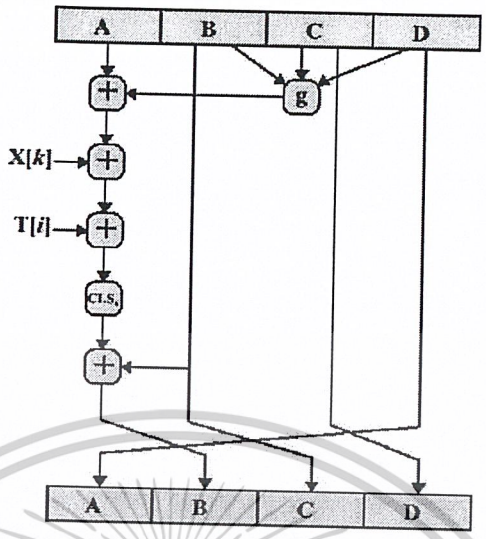


รูปที่ 2.12 แสดงขั้นตอนการเข้ารหัสของอัลกอริทึม MD5

ขั้นตอนการทำงานจะเก็บค่าเมสเซสไดเจสต์ ขนาด 128 บิต ซึ่งถูกใช้เป็นค่ากลางและค่าสุดท้ายซึ่งแบ่งออกเป็นขนาด 32 บิต 4 ค่า (*A, B, C, D*) ในรูปแบบเลขฐาน 16 จากนั้นจะทำการบีบอัดข้อมูลทั้งหมด 4 รอบ โดยแต่ละรอบจะใช้ค่า Primitive Function ที่แตกต่างกันซึ่งอ้างอิงจากตำแหน่ง *F, G, H, I* หลังจากนั้นจะได้เอาท์พุทแมสเซสไดเจสต์ขนาด 128 บิต ซึ่งแสดงดังรูปที่ 2.13



รูปที่ 2.13 กระบวนการฟังก์ชันแฮชของ MD5



รูปที่ 2.14 ขั้นตอนการคำนวณค่าในฟังก์ชัน F, G, H, I ของ MD5

รูปแบบฟังก์ชัน (*A, B, C, D*) ที่ใช้ในขั้นตอนการบีบอัด ซึ่งสามารถแสดงได้ดังสมการที่ (11)

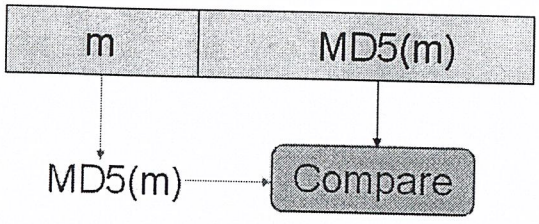
$$A = B + ((A + F(B, C, D) + X[k] + T[i]) <<< s)$$
$$D = A + ((D + G(A, B, C) + X[k] + T[i]) <<< s)$$
$$C = D + ((C + H(D, A, B) + X[k] + T[i]) <<< s)$$
$$B = C + ((B + I(C, D, A) + X[k] + T[i]) <<< s) \quad (11)$$

จากรูปที่ 2.14 โดยที่ *F* เป็นค่า primitive functions *F, G, H, I* สามารถคำนวณได้จากสมการที่ (12)

$$F(X, Y, Z) = (X \text{ and } Y) \text{ or } ((\text{not } X) \text{ and } Z)$$
$$F(X, Y, Z) = (X \text{ and } Z) \text{ or } (Y \text{ and } (\text{not } Z))$$
$$F(X, Y, Z) = X \text{ xor } Y \text{ xor } Z$$
$$F(X, Y, Z) = Y \text{ xor } (X \text{ or } (\text{not } Z)) \quad (12)$$

ข้อดีของ MD5

- ข้อมูลที่ใช้ในการเข้ารหัสมีขนาดไม่จำกัด
- สามารถนำไปใช้ในการทำลายมือชื่อดิจิตอลได้

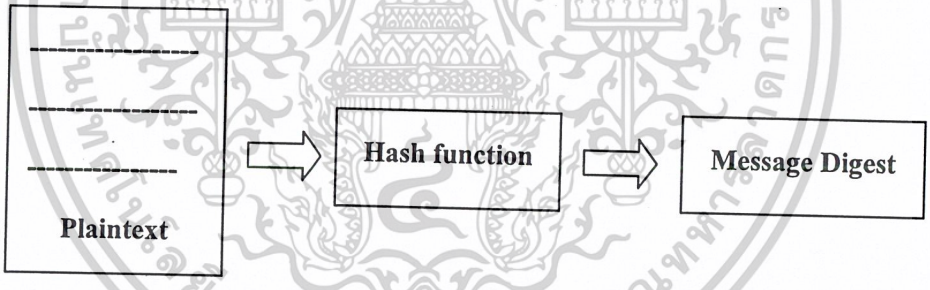


รูปที่ 2.15 แสดงขั้นตอนการเปรียบเทียบเมสเซสไดเจสต์

2.8 การพิสูจน์ตัวตนด้วยลายมือชื่อดิจิตอล (Digital Signature)

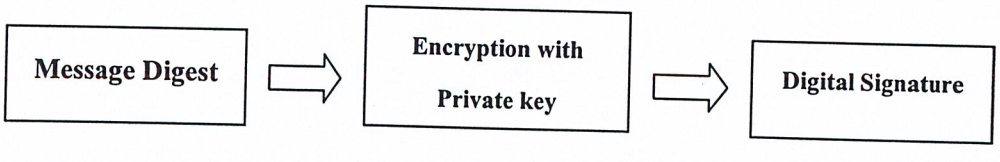
เป็นการประยุกต์การเข้ารหัสด้วยกุญแจสาธารณะและฟังก์ชันแฮช มาใช้งานในการพิสูจน์ตัวตนด้วยการทำลายมือชื่อดิจิตอล ซึ่งสามารถแบ่งออกเป็นขั้นตอนได้ดังนี้

1. นำข้อมูลที่ต้องการเข้ารหัสมาผ่านฟังก์ชันแฮช จากนั้นจะได้เมสเซสไดเจสต์ออกมา สามารถแสดงได้ดังรูปที่ 2.16



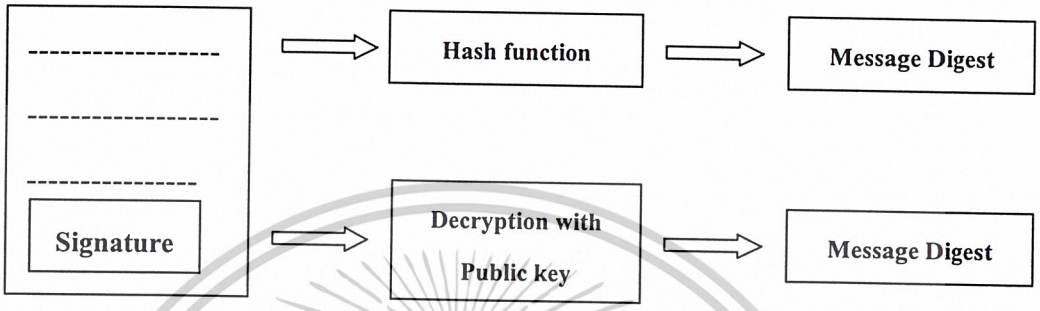
รูปที่ 2.16 แสดงขั้นตอนการนำข้อมูลไปเข้ารหัสด้วยฟังก์ชันแฮช

2. นำเมสเซสไดเจสต์มาทำเข้ารหัสด้วยกุญแจส่วนตัว เพื่อเป็นการทำลายมือชื่อดิจิตอลซึ่งแสดงดังรูปที่ 2.17



รูปที่ 2.17 การเข้ารหัสเมสเซสไดเจสต์ด้วยกุญแจส่วนตัว

3. การตรวจสอบลายมือชื่อดิจิตอลทำได้โดยการนำข้อมูลมาผ่านฟังก์ชันแฮชเพื่อหาเมสเซสไดเจสต์ จากนั้นจะนำลายมือชื่อดิจิตอลมาถอดรหัสด้วยกุญแจสาธารณะ ซึ่งสามารถแสดงได้ดังรูปที่ 2.18



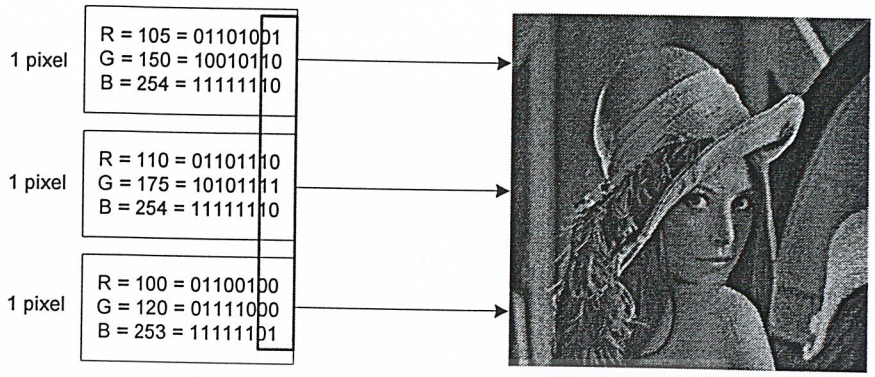
รูปที่ 2.18 ขั้นตอนการเปรียบเทียบความถูกต้อง

ลายมือชื่อดิจิตอลนิยมนำไปใช้ในระบบรักษาความปลอดภัยในการชำระเงินผ่านระบบอินเตอร์เน็ต ซึ่งในปัจจุบันนี้การทำธุรกรรมการเงินอิเล็กทรอนิกส์ได้รับความนิยมเป็นอย่างมาก

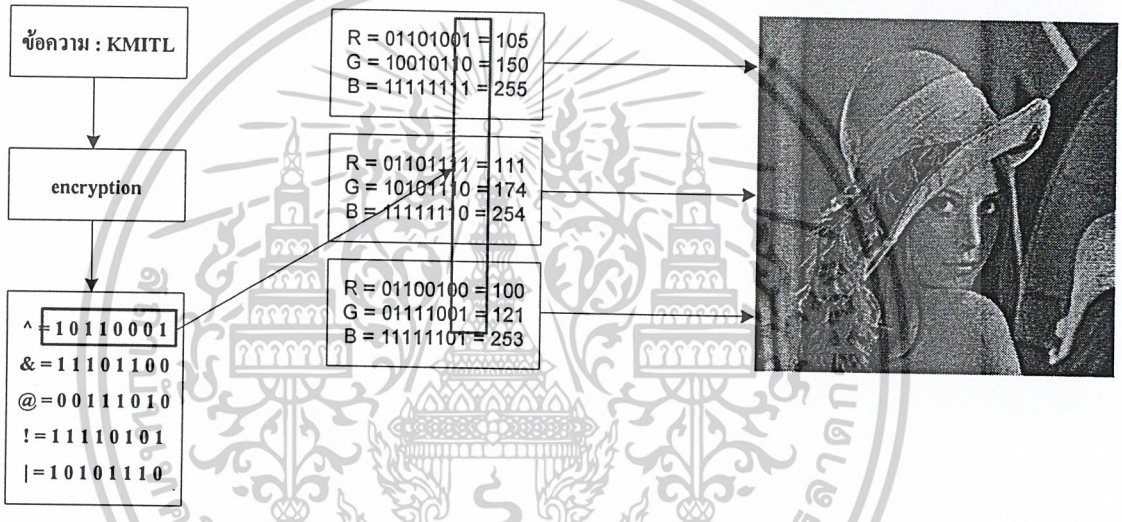
2.9 การซ่อนข้อมูลในภาพบิตแมป (Steganography)

การซ่อนข้อมูลเอาไว้ในภาพบิตแมป ซึ่งภาพนั้นยังคงมองเห็นเป็นภาพเดิมหรือมีการเปลี่ยนแปลงน้อยมาก ซึ่งจะใช้หลักการที่เรียกว่า Steganography โดยจะนำภาพสี 24 บิต ที่มีค่าสี RGB จากนั้นก็เลือก LSB (Least Significant Bit) ของแต่ละไบต์มาใช้ เพราะการเปลี่ยนแปลงที่บิต 1 จะส่งผลกระทบน้อยมากกับภาพ โดยที่ต้องแปลงจากเลขฐานสิบเป็นสองก่อน จากนั้นจะนำข้อมูลเลขฐานสองมาเรียงลงไปในบิต 1 จากซ้ายไปขวา จากบนลงล่าง และในการซ่อนข้อความ 1 ตัวอักษรจะใช้ 8 ไบต์ เช่น ภาพขนาด 32*32 จะสามารถซ่อนตัวอักษรได้ 48 ตัว ในบิต 0 ซึ่งสามารถคำนวณได้จาก $[3*(32*32)]*0.125^2 = 48$ ตัว หลังจากนั้นจะนำเลขฐานสองมาแปลงเป็นฐานสิบเพื่อแปลงกลับเป็นภาพอีกครั้ง

เพื่อความปลอดภัยในการใช้งาน จะนำข้อมูลมาทำการเข้ารหัสลับก่อนแล้วค่อยนำไปซ่อนเอาไว้ในภาพอีกที ซึ่งจะทำให้การถอดรหัสทำได้ยากขึ้น และได้แสดงดังรูปที่ 2.19



(ก)



(ข)

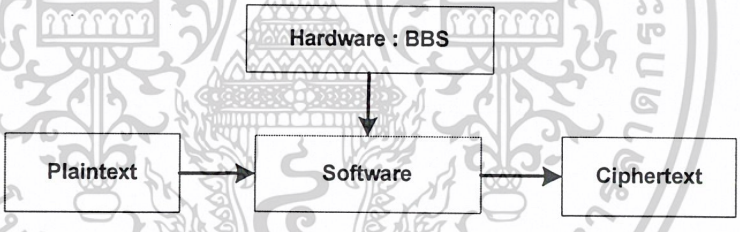
รูปที่ 2.19 แสดงขั้นตอนการซ่อนข้อมูลในภาพบิตแมป

บทที่ 3
การออกแบบและโครงสร้าง

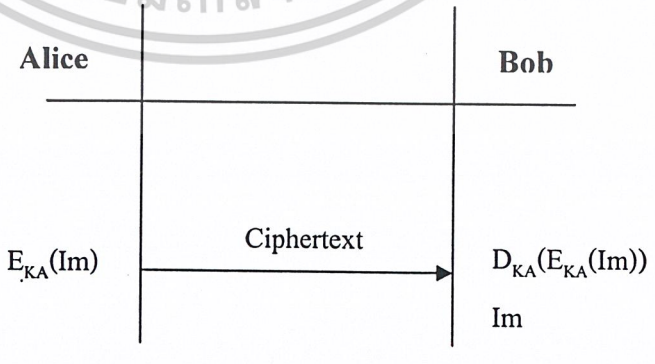
3.1 ภาพโดยรวมของระบบ

ระบบนี้เป็นการพัฒนาการเข้ารหัสลับให้มีความแข็งแกร่งมากยิ่งขึ้น เมื่อเรามีข้อมูลที่ต้องการจะเข้ารหัสลับเราจะนำข้อมูลนั้นไปผ่านกระบวนการเข้ารหัสลับในคอมพิวเตอร์แล้วจะได้ข้อมูลที่เข้ารหัสแล้วออกมาจากนั้นจะนำข้อมูลดังกล่าวไปผ่านกระบวนการเข้ารหัสลับในไมโครคอนโทรลเลอร์อีกครั้ง โดยที่คอมพิวเตอร์จะติดต่อกับไมโครคอนโทรลเลอร์ผ่านทางพอร์ตยูเอสบี ดังนั้นกระบวนการเข้ารหัสลับจะต้องอาศัยทั้งซอฟท์แวร์กับฮาร์ดแวร์ทำงานร่วมกัน ส่วนการถอดรหัสลับนั้นก็จะมีวิธีทำงานเช่นเดียวกัน โดยแต่ละโหมดมีการทำงานดังนี้

โหมดที่ 1 การเข้ารหัสลับร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Blum-Blum-Shub Generator

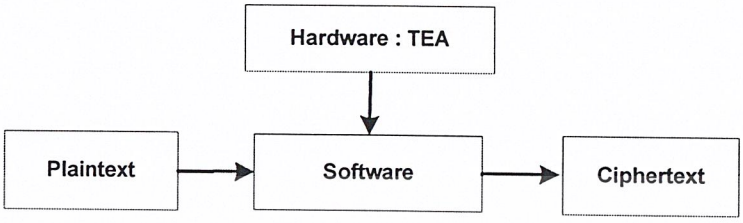


รูปที่ 3.1 การเข้ารหัสลับโดยใช้ Blum-Blum-Shub Generator

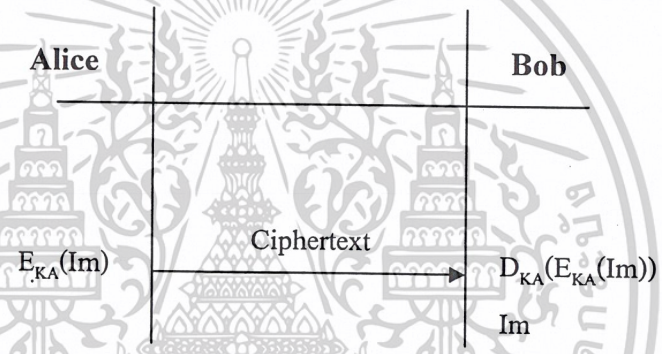


รูปที่ 3.2 แสดงโพรโตคอลการเข้ารหัสลับโดยใช้ Blum-Blum-Shub Generator

โหมดที่ 2 การเข้ารหัสลับร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm

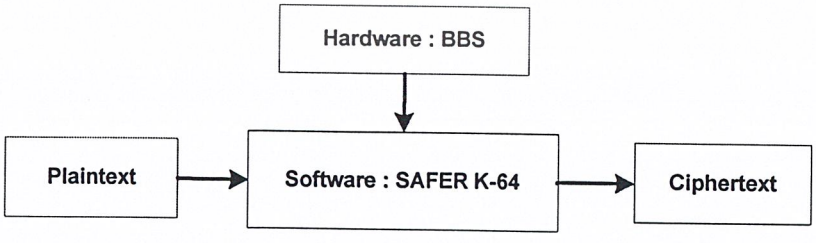


รูปที่ 3.3 การเข้ารหัสลับโดยใช้ Tiny Encryption Algorithm

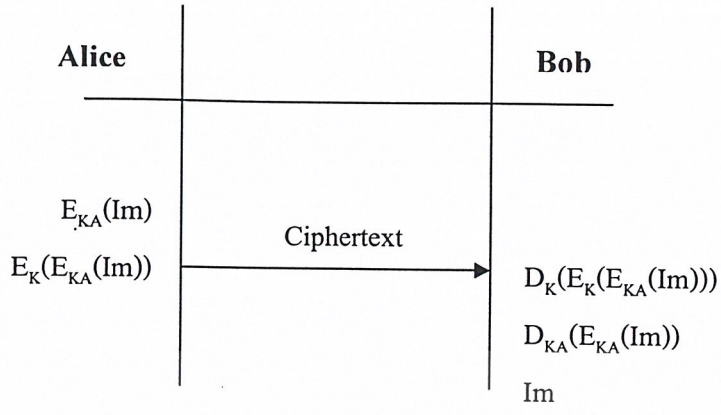


รูปที่ 3.4 แสดงโพรโตคอลการเข้ารหัสลับโดยใช้ Tiny Encryption Algorithm

โหมดที่ 3 การเข้ารหัสลับร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ SAFER K-64 ร่วมกับ Blum -Blum-Shub Generator

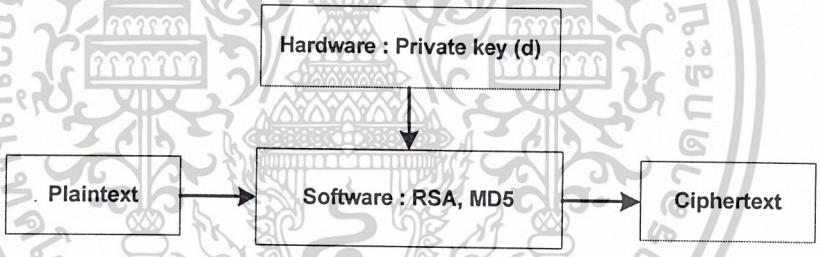


รูปที่ 3.5 การเข้ารหัสลับโดยใช้ SAFER K-64 ร่วมกับ Blum -Blum-Shub Generator

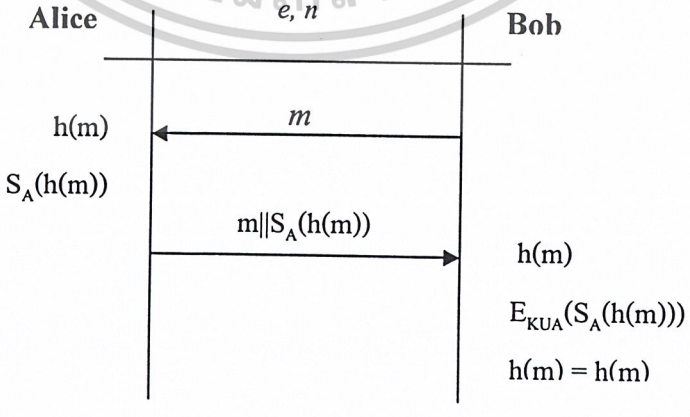


รูปที่ 3.6 แสดงโพรโตคอลการเข้ารหัสลับโดยใช้ SAFER K-64 ร่วมกับ Blum -Blum-Shub Generator

โหมดที่ 4 การทำลายมือชื่อดิจิตอล

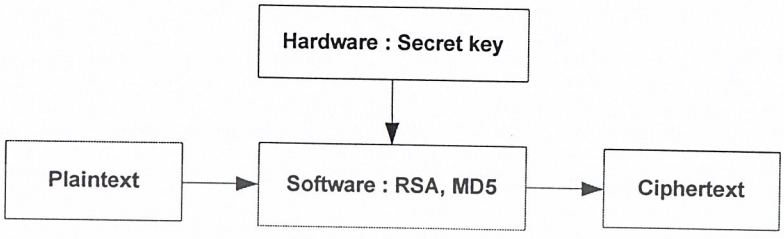


รูปที่ 3.7 การทำลายมือชื่อดิจิตอล

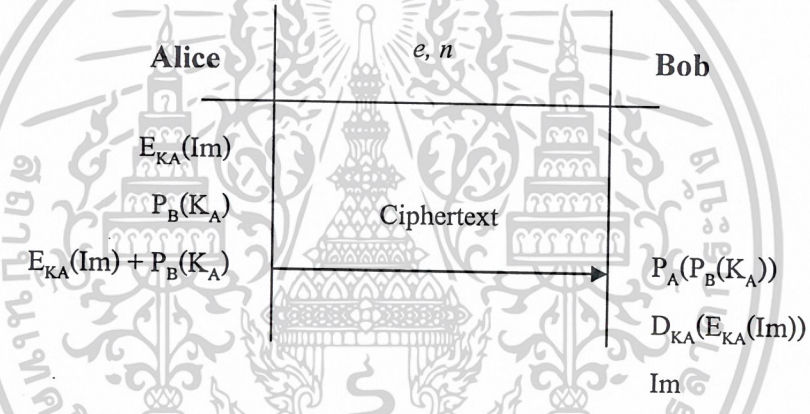


รูปที่ 3.8 แสดงโพรโตคอลการทำลายมือชื่อดิจิตอล

โหมดที่ 5 การซ่อนกุญแจลับในภาพบิตแมป

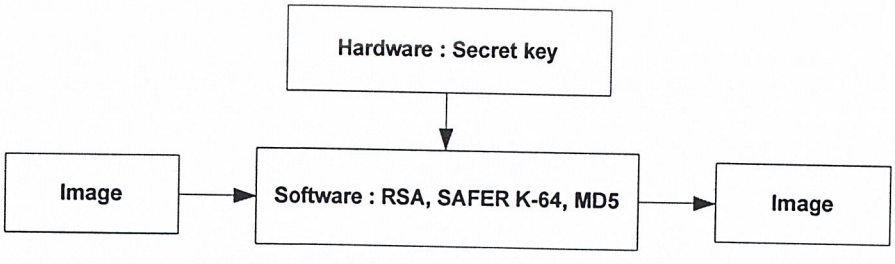


รูปที่ 3.9 การซ่อนกุญแจลับในภาพบิตแมป

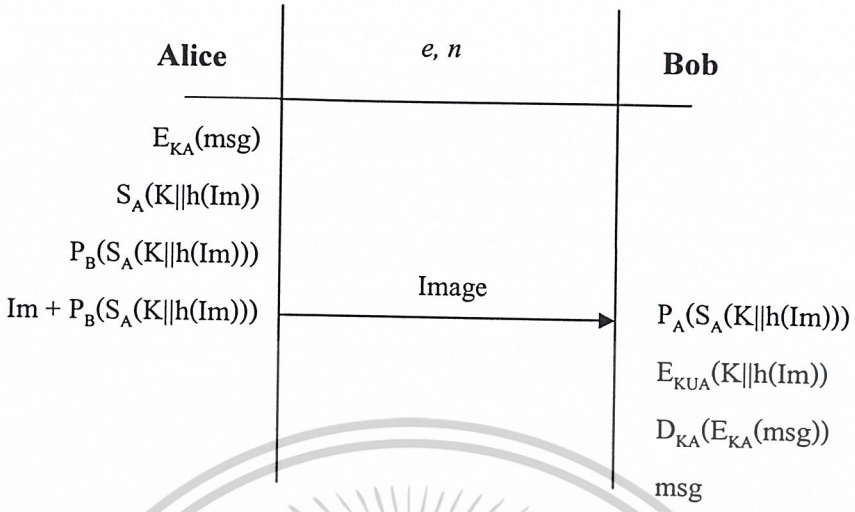


รูปที่ 3.10 แสดงโพรโตคอลการซ่อนกุญแจลับในภาพบิตแมป

โหมดที่ 6 การซ่อนข้อความในภาพบิตแมป



รูปที่ 3.11 การซ่อนข้อความในภาพบิตแมป

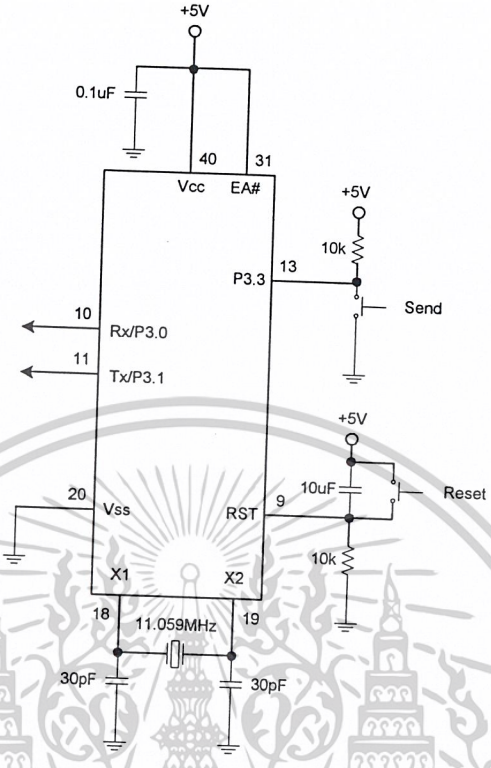


รูปที่ 3.12 แสดงโพรโตคอลการการซ่อนข้อความในภาพบิตแมป

3.2 โครงสร้างทางฮาร์ดแวร์

3.2.1 การเชื่อมต่อวงจรไมโครคอลโทรลเลอร์ MCS-51

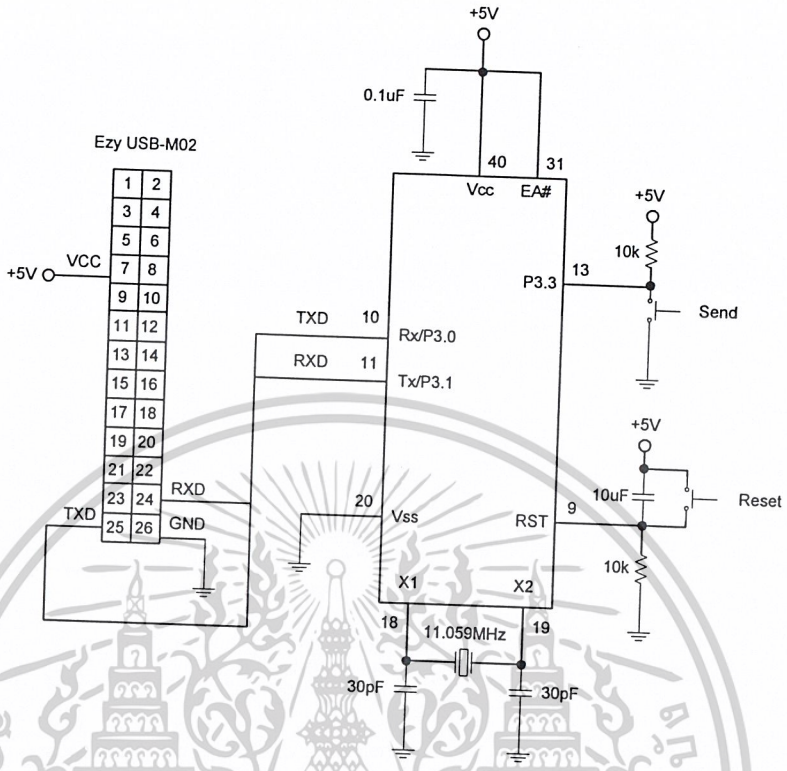
การเชื่อมต่อวงจรที่ใช้ในโครงงานนี้จะใช้ไมโครคอนโทรลเลอร์ในตระกูล MCS-51 เบอร์ AT89C51 ซึ่งเป็นการต่อวงจรไมโครคอนโทรลเลอร์พื้นฐานที่สามารถจะติดต่อสื่อสารผ่านทางพอร์ตอนุกรมได้ โดยที่จะมีขา 10 และ 11 ที่จะต่อกับโมดูล Ezy USB-M02 เพื่อติดต่อกับคอมพิวเตอร์ ดังแสดงในรูปที่ 3.13



รูปที่ 3.13 แสดงการเชื่อมต่อวงจรไมโครคอนโทรลเลอร์ MCS-51 (AT89C51)

3.2.2 การเชื่อมต่อระหว่างวงจรไมโครคอนโทรลเลอร์ MCS-51 กับโมดูล Ezy USB-M02

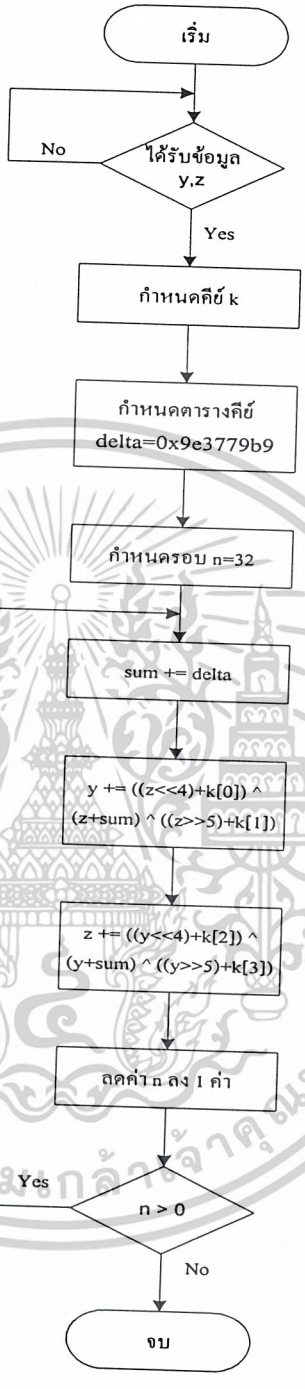
การเชื่อมต่อระหว่างวงจรไมโครคอนโทรลเลอร์ MCS-51 กับโมดูล Ezy USB-M02 ซึ่งเป็นโมดูลสำเร็จรูป แสดงดังรูปที่ 3.14 การทำงานทั้งหมดของวงจรจะถูกควบคุมด้วยไมโครคอนโทรลเลอร์ MCS-51 ซึ่งไมโครคอนโทรลเลอร์จะรับส่งข้อมูลออกทางพอร์ตอนุกรมที่ติดอยู่กับขา TXD และ RXD ของโมดูลแล้วโมดูลจะแปลงจากพอร์ตอนุกรมเป็นพอร์ตยูเอสบี จากนั้นจะส่งข้อมูลให้กับคอมพิวเตอร์ต่อไป



รูปที่ 3.14 แสดงการเชื่อมต่อระหว่างวงจรไมโครคอนโทรลเลอร์ MCS-51 กับโมดูล Ezy USB-M02

3.2.3 การเขียนโปรแกรมบนไมโครคอนโทรลเลอร์ MCS-51

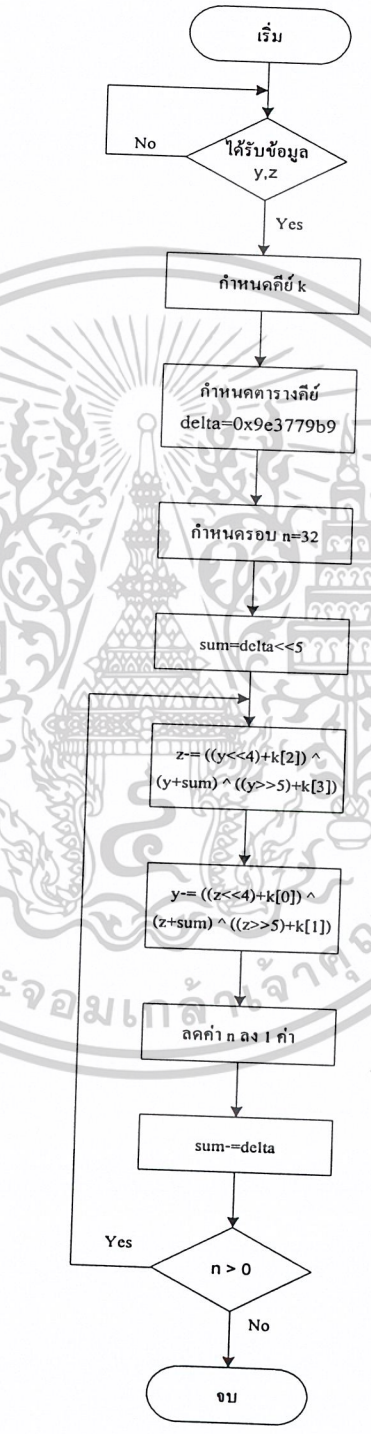
การเขียนโปรแกรมเข้ารหัสลับบนไมโครคอนโทรลเลอร์ MCS-51 จะมีการเขียนโปรแกรมเข้ารหัสลับโดยใช้ Tiny Encryption Algorithm และการสร้างสัญญาณ Pseudo random sequence แบบ Blum-Blum-Shub Generator แสดงโฟลว์ชาร์ตดังรูปต่อไปนี้



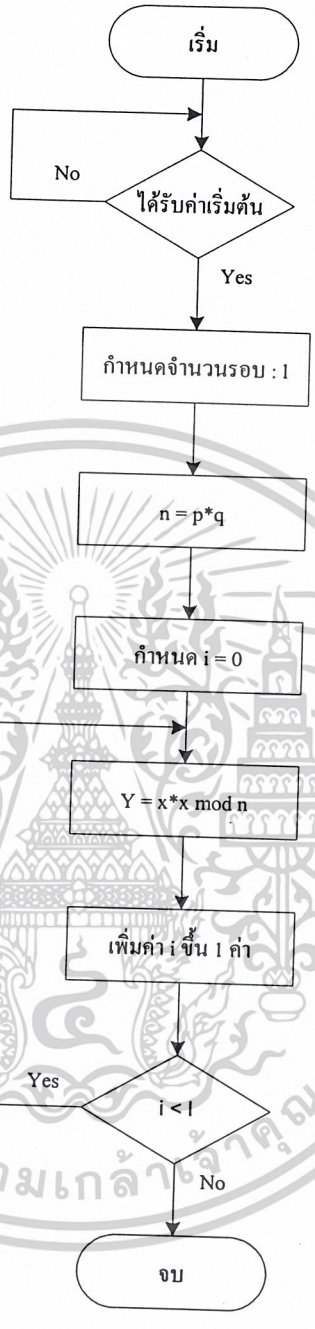
รูปที่ 3.15 โฟลว์ชาร์ตแสดงการเข้ารหัสของ Tiny Encryption Algorithm

จากรูปที่ 3.15 จะแสดงการทำงานของโปรแกรม Tiny Encryption Algorithm จะเริ่มจากรับข้อมูลเข้ามาเก็บในตัวแปร *y*, *z* และกำหนดคีย์ในตัวแปร *k* จากนั้นจะกำหนดตารางคีย์ค่าคงในตัวแปร *delta* และกำหนดรอบการทำงาน *n* ของโปรแกรม จากนั้นจะนำข้อมูล *y*, *z* ไปผ่าน

กระบวนการ XOR และ ADD ตามลำดับจนครบจำนวนรอบ โดยรอบการถอดรหัสจะทำย้อนกลับรอบการเข้ารหัส



รูปที่ 3.16 โฟลว์ชาร์ตแสดงการถอดรหัสของ Tiny Encryption Algorithm



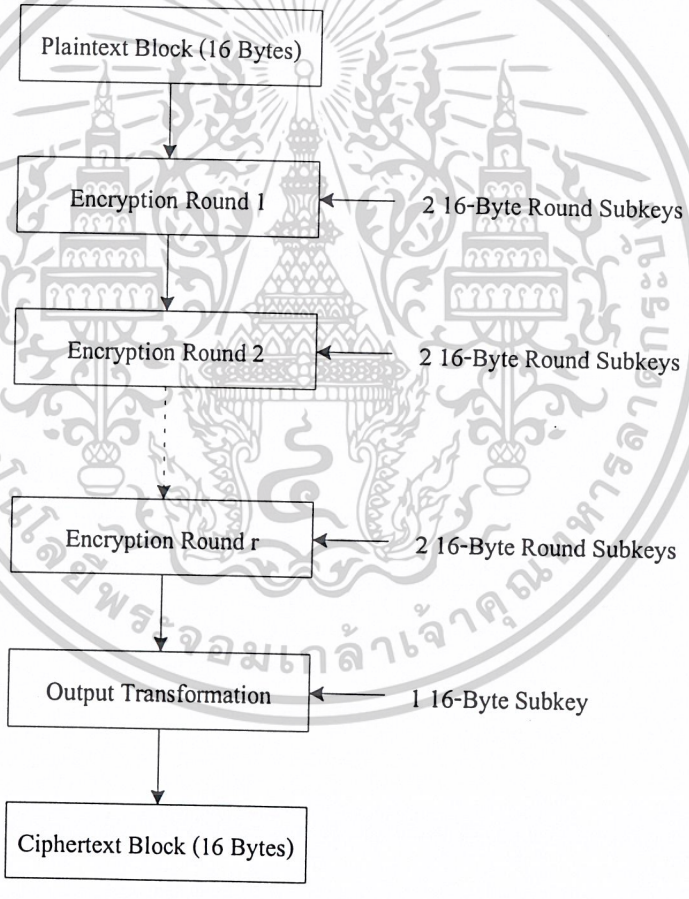
รูปที่ 3.17 โฟลว์ชาร์ตแสดงการทำงานของ Blum-Blum-Shub Generator

จากรูปที่ 3.17 แสดงการทำงานของโปรแกรม Blum-Blum-Shub Generator จะเริ่มจากรอรับค่าเริ่มต้นมาเก็บในตัวแปร x จากนั้นจะกำหนดความยาวให้ตัวแปร l นำค่า p คูณกับ q แล้วนำผลลัพธ์ที่ได้ไปเก็บในตัวแปร n จากนั้นกำหนดค่า i เริ่มต้นเท่ากับศูนย์ แล้วนำค่า x ไปหาค่าตามสมการต่อไปจนครบจำนวนรอบแล้วจึงออกจากโปรแกรม

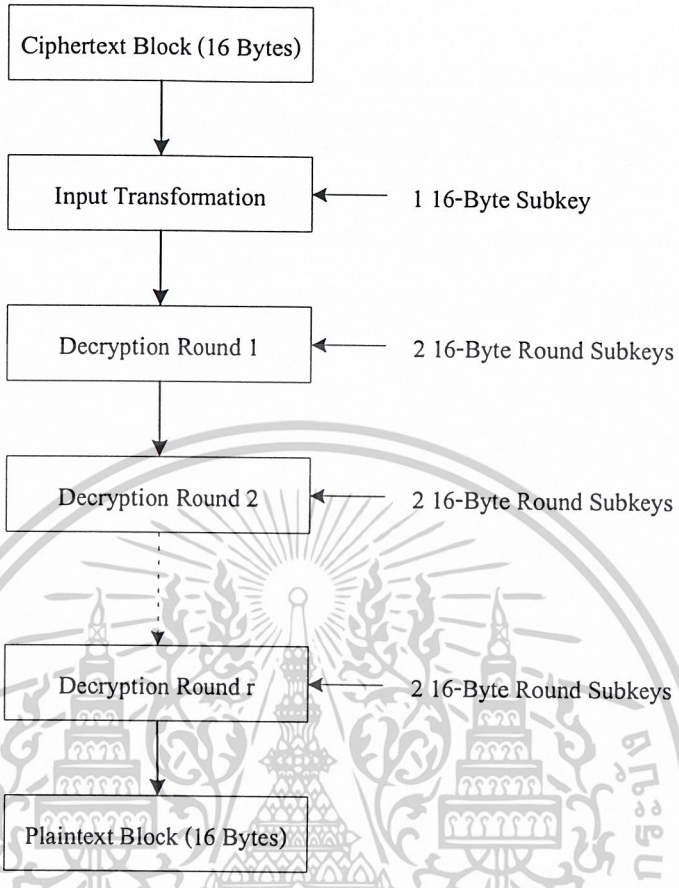
3.3 การออกแบบซอฟท์แวร์

การเขียนโปรแกรมในส่วนของซอฟท์แวร์นั้นจะมีการเข้ารหัสลับแบบกุญแจลับและกุญแจสาธารณะ โดยที่แบบกุญแจลับจะใช้อัลกอริทึม SAFER K-64 และแบบกุญแจสาธารณะจะใช้อัลกอริทึม RSA ในการการเข้ารหัสและถอดรหัส นอกจากนี้ยังมีการใช้การเข้ารหัสทางเดียว (hash function) ในการทำลายมือชื่อดิจิตอล (Digital Signature) ร่วมกับ RSA โดยที่แต่ละแบบมีการทำงานดังต่อไปนี้

3.3.1 การเข้ารหัสและถอดรหัสโดยใช้อัลกอริทึม SAFER K-64



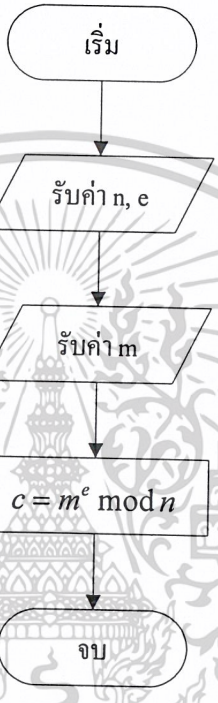
รูปที่ 3.18 บล็อกไดอะแกรมการเข้ารหัสของ SAFER K-64



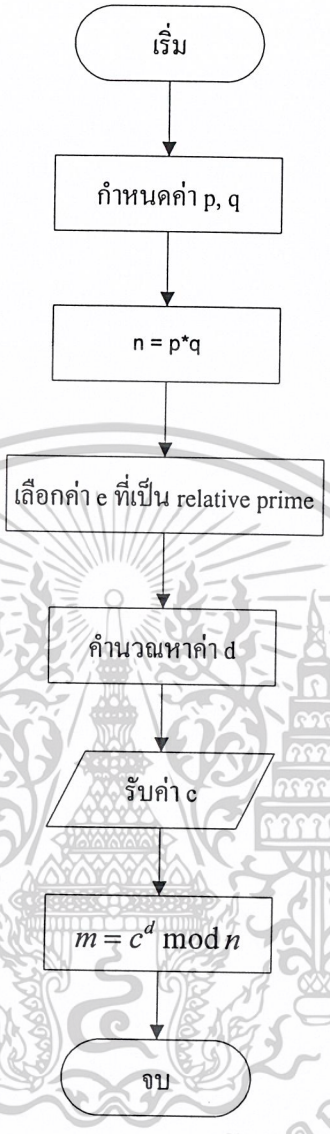
รูปที่ 3.19 บล็อกไดอะแกรมการถอดรหัสของ SAFER K-64

3.3.2 การเข้ารหัสและถอดรหัสโดยใช้อัลกอริทึม RSA

การทำงานของโปรแกรมการเข้ารหัส RSA จากรูปที่ 3.20 เริ่มจากรับค่ารับค่า N, e เข้ามา และเข้ารหัสข้อมูลด้วย $C = m^e \ mod \ N$ ส่วนการถอดรหัสเริ่มจากคำนวณ $n = p*q$ และคำนวณค่าหา d จากนั้นคำนวณค่า m จากสมการ $m = C^d \ mod \ N$ ซึ่งแสดงในรูปที่ 3.21



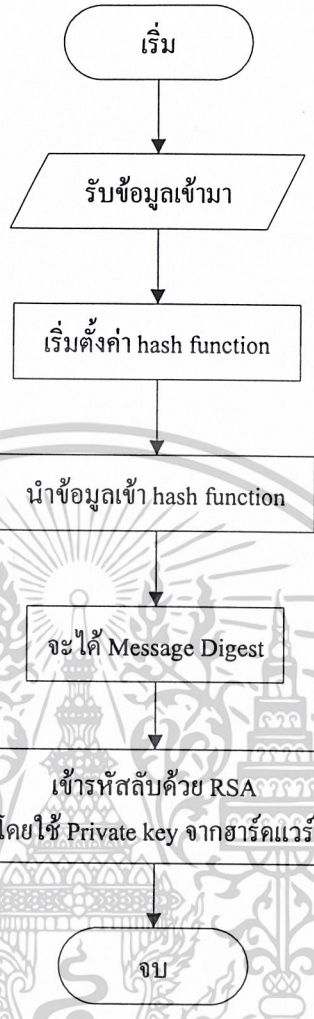
รูปที่ 3.20 โฟลว์ชาร์ตแสดงการเข้ารหัสลับของ RSA



รูปที่ 3.21 โฟลว์ชาร์ตแสดงการถอดรหัสลับของ RSA

3.3.3 การทำลายมือชื่อดิจิตอล (Digital Signature)

โฟลว์ชาร์ตแสดงการทำลายมือชื่อดิจิตอล เริ่มจากรับค่าข้อมูลเข้ามา จากนั้นจะนำข้อมูลไปผ่านฟังก์ชันแฮช และจะได้เมสเซสไดเจสต์ออกมานำค่าเมสเซสไดเจสต์ที่ได้มาทำการเข้ารหัสลับด้วย RSA ด้วยกุญแจส่วนตัว (Private key) ซึ่งแสดงในรูปที่ 3.22



รูปที่ 3.22 โฟลว์ชาร์ตแสดงการทำลายมือชื่อดิจิตอล (Digital Signature)

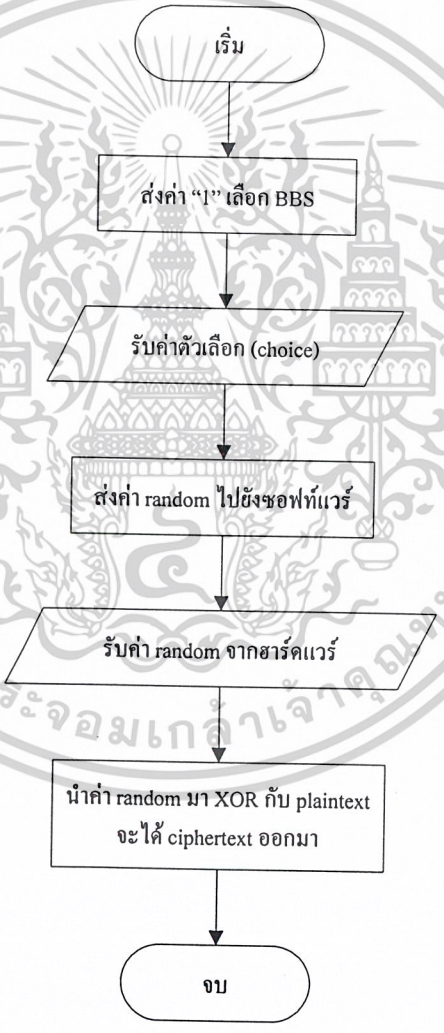
3.4 โครงสร้างทางซอฟท์แวร์กับฮาร์ดแวร์

เมื่อต้องการทำให้การเข้ารหัสลับมีความแข็งแกร่งมากขึ้นต้องทำการเข้ารหัสร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์จึงจะได้ข้อมูลที่เข้ารหัสแล้วออกมา ดังนั้นการเข้ารหัสต้องอาศัยทั้งสองอย่างเข้าด้วยกันจึงจะสมบูรณ์และถ้าในการถอดรหัสขาดอุปกรณ์ในทางฮาร์ดแวร์ไปก็จะไม่สามารถถอดรหัสได้เช่นเดียวกัน

3.4.1 การเข้ารหัสลับร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Blum-Blum-Shub Generator

การเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Blum-Blum-Shub Generator เริ่มจากซอฟท์แวร์จะส่งค่า 1 เพื่อเลือกใช้อัลกอริทึม Blum-Blum-Shub Generator จากนั้นฮาร์ดแวร์ส่งค่า Blum-Blum-Shub ไปยังซอร์ฟแวร์ จากนั้นจะนำค่า Blum-Blum-Shub จากฮาร์ดแวร์ มา XOR กับ Plaintext จะได้ cipher text ออกมา ซึ่งแสดงในรูปที่ 3.23

ในส่วนของการถอดรหัสนั้นจะใช้วิธีการเช่นเดียวกับการเข้ารหัส

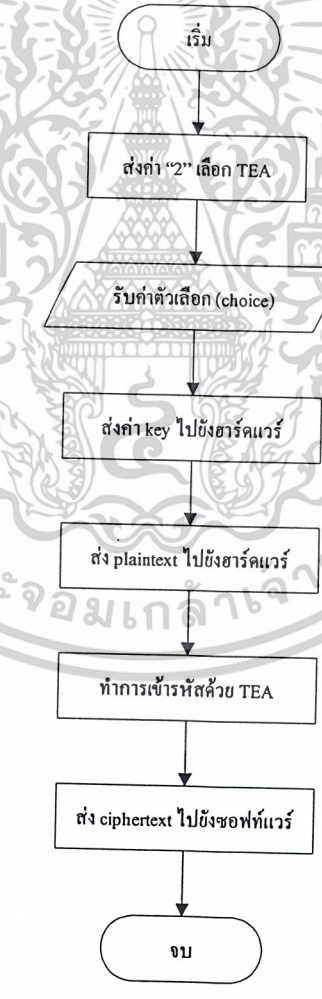


รูปที่ 3.23 โฟลว์ชาร์ตการเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์
โดยใช้ Blum-Blum-Shub Generator

3.4.2 การเข้ารหัสลับร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm

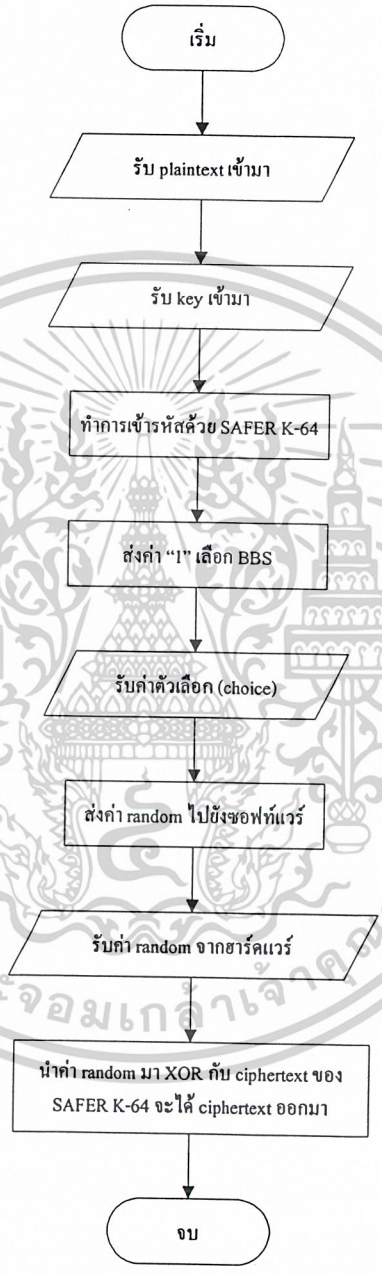
การเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm เริ่มจากซอฟท์แวร์ส่งค่า 2 เพื่อเลือกใช้อัลกอริทึมการเข้ารหัสแบบ Tiny Encryption Algorithm จากนั้นซอฟท์แวร์ส่งกุญแจไปยังฮาร์ดแวร์ส่ง Plaintext ไปยังฮาร์ดแวร์ หลังจากนั้นฮาร์ดแวร์จะทำการเข้ารหัสลับด้วย Tiny Encryption Algorithm และส่ง Ciphertext ไปยังซอฟท์แวร์ ซึ่งแสดงในรูปที่ 3.24

ในส่วนของการถอดรหัสนั้นจะใช้วิธีการเช่นเดียวกับการเข้ารหัสแต่จะต่างกันตรงที่จะใช้ฟังก์ชั่นในการถอดรหัสแทน



รูปที่ 3.24 โฟลว์ชาร์ตการเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm

3.4.3 การเข้ารหัสลับร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator

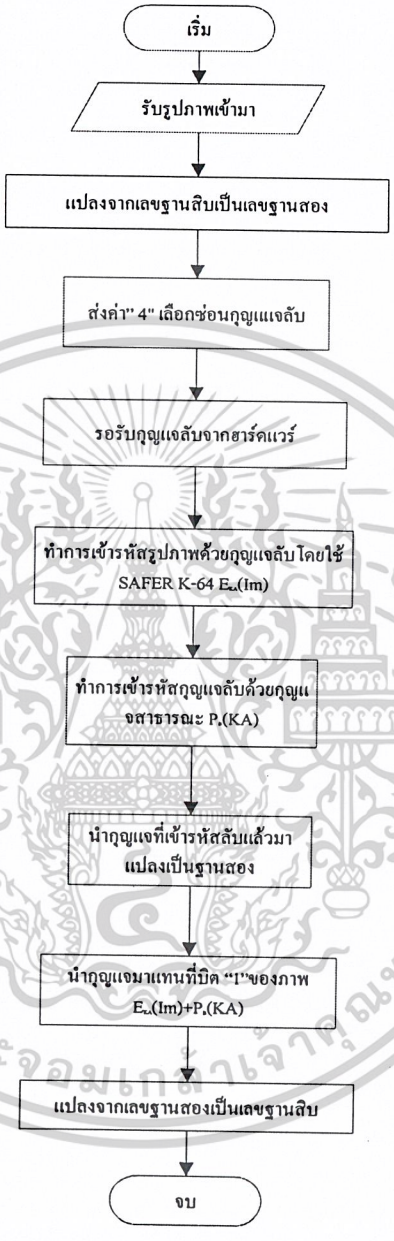


รูปที่ 3.25 โฟลว์ชาร์ตการเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์ โดย SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator

จากรูปที่ 3.25 แสดงการเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดย SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator เริ่มจากซอฟท์แวร์รับค่า plaintext และกุญแจเข้ามา จากนั้นทำการเข้ารหัสด้วย SAFER K-64 แล้วส่งค่า 1 เพื่อเลือก Blum-Blum-Shub Generator หลังจากนั้นฮาร์ดแวร์จะส่ง Blum-Blum-Shub ไปยังซอฟท์แวร์ แล้วนำ Blum-Blum-Shub มา XOR กับ Ciphertext ของ SAFER K-64

ในส่วนของการถอดรหัสนั้นจะใช้วิธีการเช่นเดียวกับการเข้ารหัสแต่จะต่างกันตรงที่จะทำย้อนกลับกันโดยที่จะนำ Blum-Blum-Shub Generator มา XOR กับ Ciphertext ก่อนจากนั้นจึงนำมาถอดรหัสด้วย SAFER K-64

3.4.4 การซ่อนกุญแจลับในภาพบิตแมป

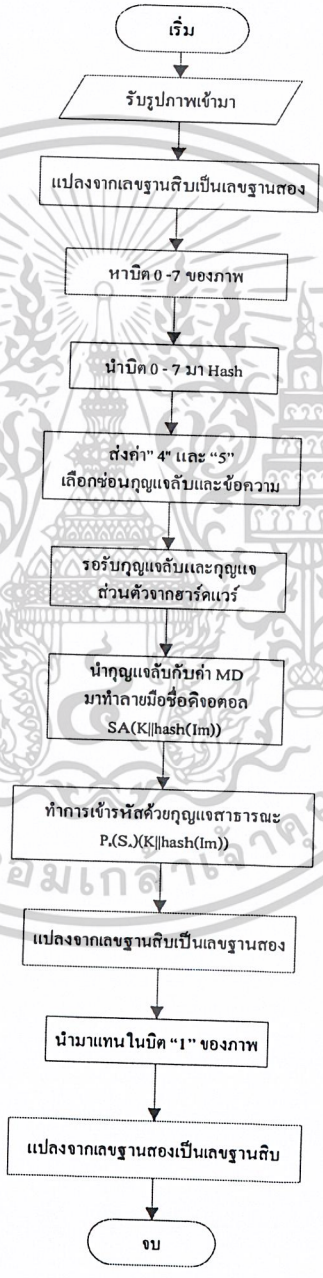


รูปที่ 3.26 โฟลว์ชาร์ตการซ่อนกุญแจลับไว้ในภาพบิตแมป

จากรูปที่ 3.26 เริ่มจากรับภาพเข้ามาจากนั้นจะทำการจากเลขฐานสิบเป็นเลขฐานสอง จากนั้นซอฟท์แวร์จะส่งค่า 4 ไปยังฮาร์ดแวร์เพื่อเลือกการซ่อนกุญแจลับ หลังจากนั้นฮาร์ดแวร์จะส่งกุญแจลับกลับไปยังซอฟท์แวร์ แล้วนำมาเข้ารหัสลับโดยใช้ SAFER K-64 และทำการเข้ารหัสกุญแจลับด้วยกุญแจสาธารณะ นำกุญแจที่เข้ารหัสลับแล้วมาแปลงเป็นเลขฐานสองแล้วนำมาแทน

บิต 1 ของรูปภาพและทำการแปลงจากเลขฐานสองเป็นเลขฐานสิบเพื่อแปลงกลับเป็นภาพเหมือนเดิม

3.4.5 การซ่อนข้อความในภาพบิตแมป



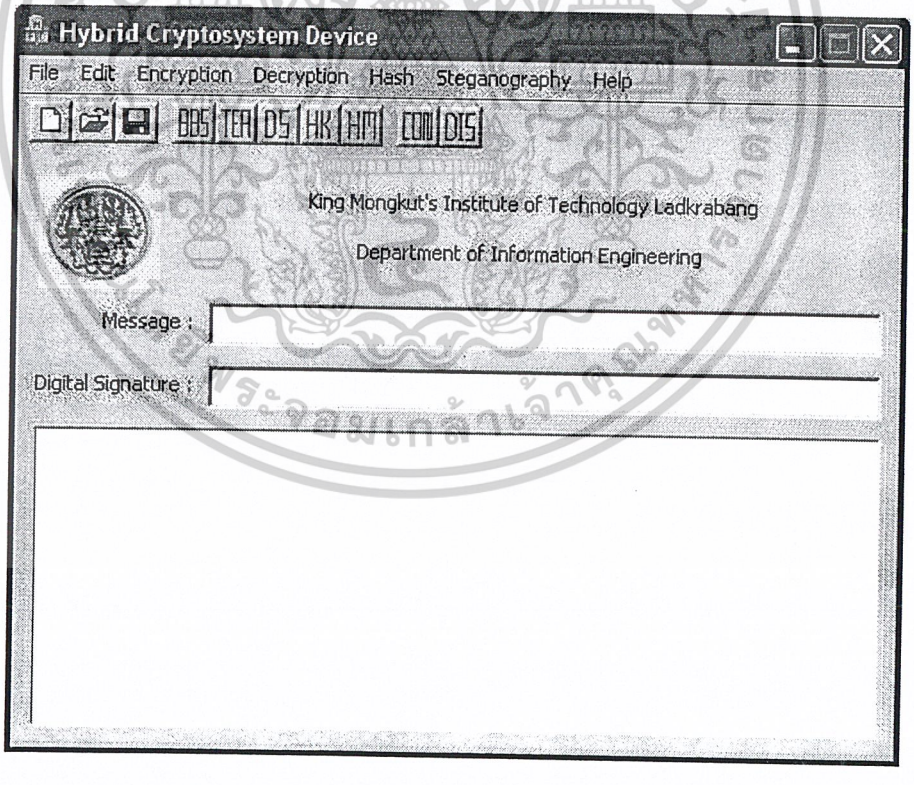
รูปที่ 3.27 โฟลว์ชาร์ตการซ่อนข้อความในภาพบิตแมป

จากรูปที่ 3.27 เริ่มจากรับภาพเข้ามา จากนั้นทำการแปลงเลขฐานสิบเป็นเลขฐานสอง และหาบิต 0-7 ของภาพแล้วนำผ่านฟังก์ชันแฮช จากจะส่งค่า 4 และ 5 ไปยังฮาร์ดแวร์เพื่อเลือกการซ่อนกุญแจลับและข้อความ และนำกุญแจลับมาเข้ารหัสข้อความด้วย SAFER K-64 จากนั้นนำกุญแจลับกับเมสเซสไดเจสต์มาทำลายมือชื่อดิจิตอล และทำการเข้ารหัสด้วยกุญแจสาธารณะ หลังจากนั้นแปลงเป็นเลขฐานสองแล้วนำมาแทนในบิต 1 ของภาพ

3.5 ส่วนติดต่อกับผู้ใช้ (User Interface)

ส่วนติดต่อกับผู้ใช้จะประกอบด้วย

- เมนูบาร์ซึ่งจะใช้ในการเลือกอัลกอริทึมในการเข้ารหัสและถอดรหัสรวมทั้งเปิดไฟล์และเซฟไฟล์
- ปุ่มกดเอาไว้ติดต่อกับฮาร์ดแวร์และเลือกอัลกอริทึมที่ใช้ติดต่อกับฮาร์ดแวร์
- ลิสต์บล็อกเป็นส่วนที่ใช้ในการแสดงผล



รูปที่ 3.28 ส่วนติดต่อกับผู้ใช้ (User Interface)

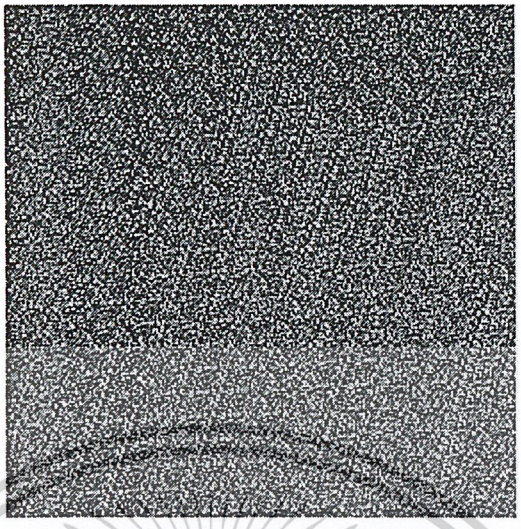
บทที่ 4
ผลการทดลอง

4.1 การทดลองเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Blum-Blum-Shub Generator

การทดลองในส่วนนี้จะเป็นการทดลองเข้ารหัสร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Blum-Blum-Shub Generator ซึ่งกำหนดค่าเริ่มต้นเท่ากับ 20 โดยจะใช้ไฟล์บิตแมปขนาด 256*256 ในการเข้ารหัส ซึ่งจะแสดงดังรูปที่ 4.1

รูปที่ 4.1 แสดงรูปภาพที่ต้องการเข้ารหัส

จากนั้นจะทำการเข้ารหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Blum-Blum-Shub Generator ซึ่งจะได้ผลการทดลองดังรูปที่ 4.2



รูปที่ 4.2 แสดงรูปภาพที่เข้ารหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์
โดยใช้ Blum-Blum-Shub Generator

จากนั้นจะทำการถอดรหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Blum-Blum-Shub Generator ซึ่งจะได้ผลการทดลองดังรูปที่ 4.3

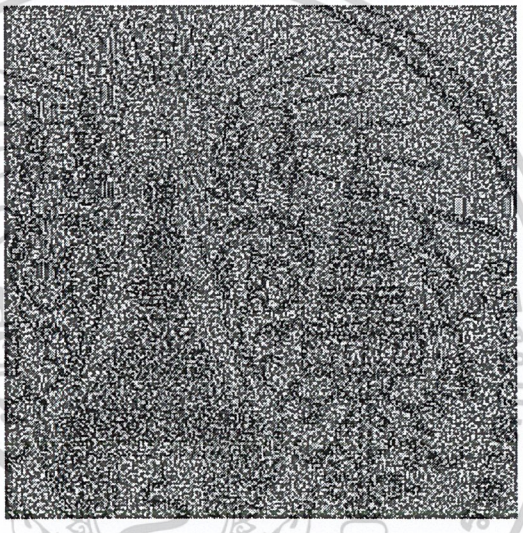
รูปที่ 4.3 แสดงรูปภาพที่ถอดรหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์
โดยใช้ Blum-Blum-Shub Generator

ซึ่งจากการทดลองถ้าไม่ได้ใช้ฮาร์ดแวร์ด้วยในการถอดรหัสจะทำให้ไม่สามารถถอดรหัสได้และจะแสดงหน้าต่างให้ผู้ใช้ติดต่อกับฮาร์ดแวร์

4.2 การทดลองเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm

การทดลองในส่วนนี้จะเป็นการทดลองเข้ารหัสร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm ซึ่งกำหนดค่าคีย์เท่ากับ 1234 โดยจะใช้ไฟล์บิตแมปขนาด 256*256 ในการเข้ารหัส ซึ่งจะแสดงดังรูปที่ 4.1

จากนั้นจะทำการเข้ารหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm ซึ่งจะได้ผลการทดลองดังรูปที่ 4.4



รูปที่ 4.4 แสดงรูปภาพที่เข้ารหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์
โดยใช้ Tiny Encryption Algorithm

จากนั้นจะทำการถอดรหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ Tiny Encryption Algorithm ซึ่งจะได้ผลการทดลองดังรูปที่ 4.5

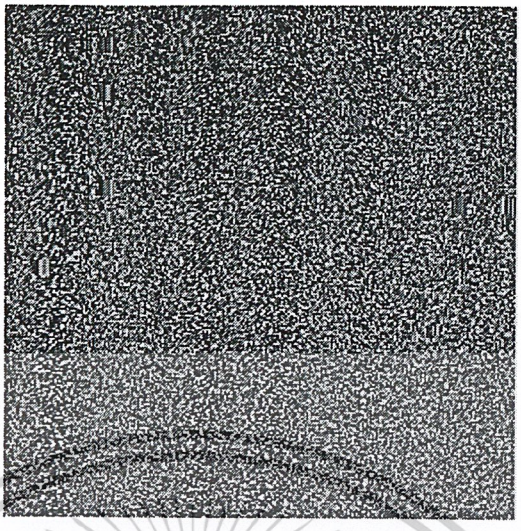
รูปที่ 4.5 แสดงรูปภาพที่ถอดรหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์
โดยใช้ Tiny Encryption Algorithm

ซึ่งจากการทดลองถ้าไม่ได้ใช้ฮาร์ดแวร์ด้วยในการถอดรหัสหรือใส่กุญแจผิดก็จะทำให้ไม่สามารถถอดรหัสได้และจะแสดงหน้าต่างแสดงความผิดพลาดขึ้น

4.3 การทดลองเข้ารหัสลับระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator

การทดลองในส่วนนี้จะเป็นการทดลองเข้ารหัสร่วมกันระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator ซึ่งกำหนดกุญแจเท่ากับ 12345678 และค่าเริ่มต้นเท่ากับ 20 โดยจะใช้ไฟล์บิตแมปขนาด 256*256 ในการเข้ารหัส ซึ่งจะแสดงดังรูปที่ 4.1

จากนั้นจะทำการเข้ารหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator ซึ่งจะได้ผลการทดลองดังรูปที่ 4.6



รูปที่ 4.6 แสดงรูปภาพที่เข้ารหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์
โดยใช้ SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator

จากนั้นจะทำการถอดรหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์โดยใช้ SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator ซึ่งจะต้องทำย้อนกลับกับการเข้ารหัสโดยที่จะต้องถอดรหัสด้วย Blum-Blum-Shub Generator ก่อนจากนั้นจึงถอดรหัสด้วย SAFER K-64 ซึ่งจะได้ผลการทดลองดังรูปที่ 4.7

ซึ่งจากการทดลองถ้าไม่ได้ทำย้อนกลับกับการเข้ารหัสจะทำให้ไม่สามารถถอดรหัสได้

รูปที่ 4.7 แสดงรูปภาพที่ถอดรหัสระหว่างซอฟท์แวร์กับฮาร์ดแวร์
โดยใช้ SAFER K-64 ร่วมกับ Blum-Blum-Shub Generator

4.4 การทดลองการทำลายมือชื่อดิจิตอล (Digital Signature)

การทดลองในส่วนนี้จะเป็นการทำลายมือชื่อดิจิตอลโดยใช้อัลกอริทึม MD5 ร่วมกับ RSA โดยที่จะกำหนดค่า p = 7, q = 17 และกุญแจส่วนตัว d = 77 ซึ่งได้จากฮาร์ดแวร์ โดยจะใช้ไฟล์บิตแมปขนาด 256*256 ในการเข้ารหัส ซึ่งจะแสดงดังรูปที่ 4.1

ทำการเข้ารหัสโดยใช้อัลกอริทึม MD5 จากนั้นจะได้ค่าเมสเซสไดเจสต์ (Message Digest) ออกมา

Message Digest: 9f586f78111dc396b103398eb84201bc

จากนั้นทำการเข้ารหัสโดยใช้อัลกอริทึม RSA ด้วยการใช้กุญแจส่วนตัวเข้ารหัสเมสเซสไดเจสต์ซึ่งหมายความว่าได้ลงลายมือชื่อดิจิตอลแล้ว จากนั้นจะได้ลายมือชื่อดิจิตอลออกมา

Digital Signature: 4e336469503337694646463516664e50

ในส่วนของการตรวจสอบว่าข้อมูลที่ได้รับเป็นข้อมูลของผู้ส่งจริงโดยการนำข้อมูลมาเข้ารหัสด้วยอัลกอริทึม MD5 เพื่อหาค่าเมสเซสไดเจสต์ (Message Digest) จะได้

Message Digest: 9f586f78111dc396b103398eb84201bc

จากนั้นนำลายมือชื่อดิจิตอลมาถอดรหัสด้วยกุญแจสาธารณะ e = 5 จะได้

Message Digest: 9f586f78111dc396b103398eb84201bc

แสดงว่าเป็นข้อมูลของผู้ส่งจริงและข้อมูลมีความถูกต้อง ซึ่งในการทดลองถ้าข้อมูลมีการเปลี่ยนแปลงไปจะทำให้ค่าเมสเซสไดเจสต์เปลี่ยนไปแสดงว่าไม่ใช้ข้อมูลที่ถูกต้อง

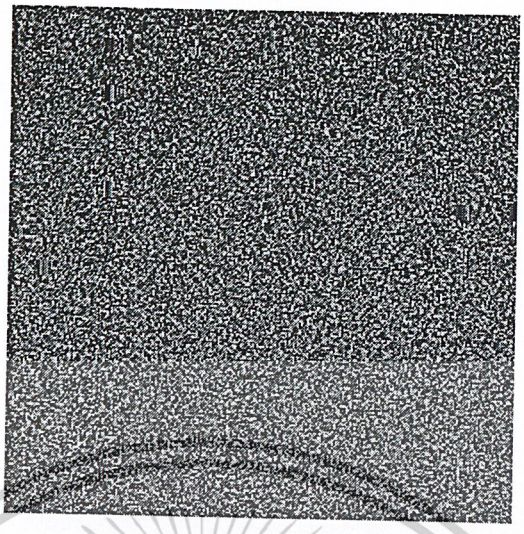
4.5 การทดลองซ่อนข้อมูล (Steganography)

การทดลองในส่วนนี้จะเป็นการทดลองซ่อนข้อมูลในภาพบิตแมป ซึ่งจะแบ่งออกเป็น 2 ส่วน ดังนี้

4.5.1 การทดลองซ่อนกุญแจลับในภาพบิตแมป

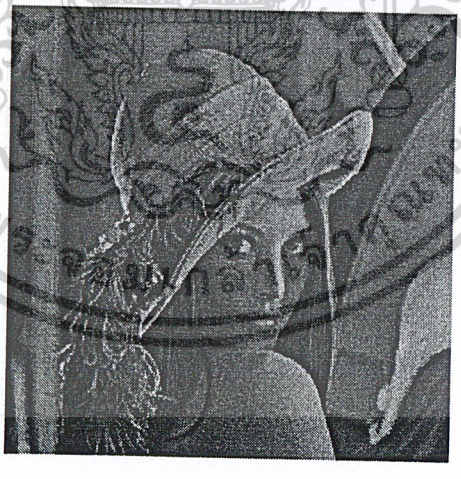
การทดลองซ่อนกุญแจลับในภาพบิตแมปที่ถูกเข้ารหัสลับด้วย SAFER K-64 ซึ่งกุญแจลับที่ใช้ในการเข้ารหัสจะได้จากฮาร์ดแวร์เท่ากับ 12345678 และกำหนดค่า n = 119 และกุญแจสาธารณะ e = 5 ซึ่งจะใช้ในการเข้ารหัสกุญแจลับ โดยจะใช้ไฟล์บิตแมปขนาด 256*256 ในการเข้ารหัสลับภาพซึ่งจะแสดงดังรูปที่ 4.1

จากนั้นจะทำการซ่อนกุญแจลับในภาพที่ถูกเข้ารหัสลับ ซึ่งจะได้ผลการทดลองดังรูปที่ 4.8



รูปที่ 4.8 แสดงรูปภาพที่เข้ารหัสด้วย SAFER K-64
และซ่อนกุญแจลับในภาพบิตแมป

จากนั้นจะนำภาพมาทำการถอดรหัสด้วยกุญแจส่วนตัว d = 77 ซึ่งจะได้กุญแจลับออกมาแล้วนำกุญแจลับมาถอดรหัสด้วย SAFER K-64 อีกครั้งหนึ่ง ซึ่งจะได้ผลการทดลองดังรูปที่ 4.9



รูปที่ 4.9 แสดงรูปภาพที่ถอดรหัสด้วย SAFER K-64
จากกุญแจลับที่ซ่อนในภาพบิตแมป

ซึ่งจากการทดลองถ้าใช้กุญแจในการถอดรหัสไม่ถูกต้องจะไม่สามารถถอดรหัสกุญแจลับออกมาได้จึงทำให้ไม่สามารถถอดรหัสรูปภาพด้วย SAFER K-64 ได้เช่นกัน

4.5.2 การทดลองซ่อนข้อความไว้ในภาพบิตแมป

การทดลองซ่อนข้อความไว้ในภาพบิตแมป ซึ่งในการทดลองจะกำหนดข้อความว่า "KMITL" จากนั้นจะนำข้อความมาเข้ารหัสลับด้วย SAFER K-64 โดยใช้กุญแจลับ 12345678 ซึ่งได้จากฮาร์ดแวร์และในส่วนการทำลายมือชื่อดิจิตอลจะกำหนด p = 7, q = 17 และกุญแจส่วนตัว d = 77 ซึ่งได้จากฮาร์ดแวร์ และในส่วนการเข้ารหัสข้อความจะใช้กุญแจสาธารณะ e = 7 และ n = 77 ในการเข้ารหัส โดยจะใช้ไฟล์บิตแมปขนาด 256*256 ในการเข้ารหัสลับภาพซึ่งจะแสดงดังรูปที่ 4.1

จากนั้นจะทำการซ่อนข้อความไว้ในภาพ ซึ่งจะได้ผลการทดลองดังรูปที่ 4.10

รูปที่ 4.10 แสดงรูปภาพที่ถูกซ่อนข้อความแล้ว

จากรูปที่ 4.10 จะเห็นได้ว่ารูปภาพที่ถูกซ่อนข้อความแล้วจะเกิดการเปลี่ยนแปลงน้อยมาก ทำให้ดูผิวเผินเป็นรูปภาพธรรมดา

จากนั้นจะนำภาพมาทำการถอดรหัสโดยใช้กุญแจส่วนตัว d = 43, p = 7 และ q = 11 และในการตรวจสอบลายมือชื่อดิจิตอลจะใช้กุญแจสาธารณะ e = 5 และ n = 119 จากนั้นจะได้กุญแจลับออกมาหลังจากนั้นนำกุญแจลับที่ได้มาถอดรหัสข้อความด้วย SAFER K-64 สุดท้ายจะได้ข้อความ "KMITL" ออกมา

บทที่ 5
สรุปผลการทดลอง

5.1 สรุปผลการทดลอง

จากผลการทดลองโครงงาน Hybrid Cryptosystem Device เป็นอุปกรณ์เข้ารหัสลับแบบผสมระหว่างซอฟท์แวร์และฮาร์ดแวร์ โดยในการทดลองจะนำข้อมูลที่ต้องเข้ารหัสลับมาทำการเข้ารหัสลับโดยใช้อัลกอริทึมแบบต่างๆ ในการทำงาน โดยที่แต่ละแบบสามารถทำการเข้ารหัสและถอดรหัสลับได้ ซึ่งจะทำให้ระบบมีความแข็งแกร่งยากต่อการลักลอบถอดรหัสโดยปราศจากฮาร์ดแวร์ซึ่งจะเป็นประโยชน์ในการนำไปใช้ในทางทหารหรืองานที่ต้องการความปลอดภัยของข้อมูลสูง

การทดลองในส่วนของลายมือชื่อดิจิตอลสามารถที่จะตรวจสอบว่าใครเป็นคนทำรายการและรายการนั้นได้ทำอย่างถูกต้องปราศจากการปลอมแปลง ซึ่งสามารถนำไปใช้ในการทำธุระกรรมทางด้านการเงินและอื่นๆ ได้

ในส่วนสุดท้ายจะเป็นการทดลองซ่อนข้อมูล (Steganography) โดยในการทดลองซ่อนกุญแจลับในภาพบิตแมป และสามารถนำกุญแจลับมาทำการถอดรหัสลับได้ ซึ่งสามารถไปใช้ในการแลกเปลี่ยนกุญแจได้ และการทดลองในส่วนสุดท้ายจะเป็นการซ่อนข้อความในภาพบิตแมป ซึ่งจำนวนของข้อความจะขึ้นอยู่กับขนาดของภาพ ถ้าภาพมีขนาดใหญ่ก็จะสามารถซ่อนข้อความได้มาก และสามารถนำไปประยุกต์ใช้ในการซ่อนไฟล์ประเภทอื่นๆ ได้

5.2 ปัญหาที่เกิดขึ้น

ในการเข้ารหัสลับแบบผสมระหว่างซอฟท์แวร์และฮาร์ดแวร์นั้น ถ้าข้อมูลมีขนาดใหญ่มากจะใช้เวลาในการเข้ารหัสและถอดรหัสลับนานมากและอาจจะไม่สามารถทำการถอดรหัสลับได้ เนื่องจากถ้าใช้อัลกอริทึม Blum-Blum-Shub Generator ในการสร้างสัญญาณ Pseudo random sequence ถ้าข้อมูลมีขนาดใหญ่มากฮาร์ดแวร์จะต้องสร้างสัญญาณ Pseudo random sequence เท่ากับขนาดของข้อมูลเพื่อใช้ในการเข้ารหัสแบบ Stream cipher ซึ่งจะทำให้ต้องใช้เวลามากในการเข้ารหัสลับ

ในส่วนของอัลกอริทึม Tiny Encryption Algorithm ในการเข้ารหัสลับนั้นทั้งซอฟท์แวร์และฮาร์ดแวร์จะต้องรอส่งค่ากลับโดยที่ซอฟท์แวร์จะทำการส่งข้อมูลขนาด 64 บิต ไปยังฮาร์ดแวร์ จากนั้นฮาร์ดแวร์จะทำการเข้ารหัสและส่งค่ากลับไปยังซอฟท์แวร์ เนื่องด้วยไมโครคอนโทรลเลอร์มีหน่วยความจำที่จำกัดไม่สามารถที่จะรับข้อมูลได้มากจึงทำให้ใช้เวลาการเข้ารหัสลับมากขึ้น และยังไม่สามารถใช้อัลกอริทึมอื่นๆ ในการเข้ารหัสได้ เนื่องจากมีฟังก์ชั่นในการทำงานที่ซับซ้อนมาก ยากต่อการเขียนโปรแกรมในฮาร์ดแวร์

การซ่อนข้อมูล (Steganography) จะใช้เวลาในการซ่อนข้อมูลนานมาก ถ้าภาพมีขนาดใหญ่ เนื่องจากมีทำงานที่ซับซ้อนมากทำให้แต่ละรอบในการทำงานจะใช้เวลานาน

5.3 แนวทางในการพัฒนา

แนวทางในการพัฒนาอุปกรณ์เข้ารหัสลับแบบผสมจะมีในส่วนของซอฟท์แวร์และฮาร์ดแวร์ ซึ่งมีรายละเอียดดังต่อไปนี้

5.3.1 ทำการเข้ารหัสลับด้วยอัลกอริทึมที่มีซับซ้อนมากขึ้น เพื่อยากต่อการถอดรหัส

5.3.2 นำอัลกอริทึมที่มีเหมาะสมกับฮาร์ดแวร์มาใช้ในการเข้ารหัสลับ เพื่อทำให้ระบบมีความแข็งแกร่งมากขึ้น ยากต่อการปลอมแปลงฮาร์ดแวร์

5.3.3 สามารถนำไปเข้ารหัสลับกับไฟล์ในรูปแบบต่างๆ

5.3.4 ใช้เวลาในการเข้ารหัสลับน้อยลง

5.3.5 พัฒนาอุปกรณ์เข้ารหัสลับแบบผสมให้ขนาดเล็กลงง่ายต่อการพกพา

บรรณานุกรม

[1] วรเทพ ไพบูลย์รัตนากร, บุญอนันต์ เกียงเอีย, "สัมผัสโลก ยูเอสบี ด้วย Ezy ยูเอสบี Module" บริษัท แอสทรอน ลอจิก รีเสิร์ชแอนด์ดิวิลอปเมนต์ จำกัด, 2537

[2] กฤดากร กล่อมการ, "Data Communication & Network" lecture Note ภาควิชาวิศวกรรมสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2546

[3] นิรุธ อำนวยศิลป์, "คู่มือการเขียนโปรแกรม Microsoft Visual C++ 6.0 ฉบับเพื่อการใช้งานจริง" บริษัท ซัคเซส มีเดีย จำกัด, 2542

[4] James L. Massey, "SAFER K-64 A-Byte-Oriented Block-Ciphering Algorithm" Signal and Information Processing Laboratory Swiss Federal Institute of technology, 1994

[5] David J. Wheeler, Roger M. Needham "TEA, a Tiny Encryption Algorithm" Computer Laboratory Cambridge University England

[6] Blum-Blum-Shub one-time pad.
http://math.boisestate.edu/~marion/teaching/crypto1f03/bbs_the_system.htm, 2003

[7] RFC 1321-The MD5 Message Digest Algorithm. http://www.faqs.org/rfcf/rfc1321.html, 1992