วิทยานิพนธ์

การพัฒนาวงจรเข้ารหัสและถอดรหัสแบบเทอร์โบด้วย FPGA

A Development of Turbo Encoder and Decoder on FPGA

นายชินพจน์ วงศ์ศรีพิสันต์

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์

พ.ศ. 2549

ใบรับรองวิทยานิพนธ์

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์

วิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมไฟฟ้า)

ปริญญา

วิศวกรรมไฟฟ้า	วิศวกรรมไฟฟ้า
สาขา	**ภาควิชา**

เรื่อง การพัฒนาวงจรเข้ารหัสและถอดรหัสแบบเทอร์โบด้วย FPGA

A Development of Turbo Encoder and Decoder on FPGA

นามผู้วิจัย นายชินพจน์ วงศ์ศรีพิสันต์

ได้พิจารณาเห็นชอบโดย

ประธานกรรมการ .....
(รองศาสตราจารย์มงคล รักษาพัชรวงศ์, Ph.D.)

กรรมการ .....
(ผู้ช่วยศาสตราจารย์วัชรี วีรคเชนทร์, M.S.)

กรรมการ .....
(ผู้ช่วยศาสตราจารย์ศรีจิตรา มหาประคุณชัย, Ph.D.)

หัวหน้าภาควิชา .....
(อาจารย์ชูเกียรติ การะเกตุ, Ph.D.)

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์รับรองแล้ว

.....
(รองศาสตราจารย์วินัย อาจคงหาญ, M.A.)

คณบดีบัณฑิตวิทยาลัย

วันที่ 30 **เดือน** มีนาคม **พ.ศ.** 2549

วิทยานิพนธ์

เรื่อง

การพัฒนาวงจรเข้ารหัสและถอดรหัสแบบเทอร์โบด้วย FPGA

A Development of Turbo Encoder and Decoder on FPGA

โดย

นายชินพจน์ วงศ์ศรีพิสันต์

เสนอ

บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์
เพื่อความสมบูรณ์แห่งปริญญาวิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมไฟฟ้า)
พ.ศ. 2549

ISBN 974-16-1407-1

ชินพจน์ วงศ์ศรีพิสันต์ 2549: การพัฒนาวงจรเข้ารหัสและถอดรหัสแบบเทอร์โบ
ด้วย FPGA ปริญญาวิศวกรรมศาสตรมหาบัณฑิต (วิศวกรรมไฟฟ้า)
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
ประธานกรรมการที่ปรึกษา: รองศาสตราจารย์มงคล รักษาพัชรวงศ์, Ph.D. 42 หน้า
ISBN 974-16-1407-1

วิทยานิพนธ์ฉบับนี้เสนอการวิจัยและพัฒนาวงจรเข้ารหัสและวงจรถอดรหัสของรหัสแบบเทอร์โบมาพัฒนาบนบอร์ด FPGA ด้วยภาษา VHDL เพื่อสามารถนำวงจรเข้ารหัสและวงจรถอดรหัสที่ได้พัฒนาขึ้นมานั้นไปประยุกต์ใช้กับระบบการสื่อสารรูปแบบอื่น ๆ เช่น ระบบโทรศัพท์เคลื่อนที่ยุคที่ 3 การทดสอบวงจรเข้ารหัสและถอดรหัสนั้นทำได้โดยใช้วิธีการทดสอบการประมวลผลแบบวนกลับ เพื่อทำการตรวจสอบความถูกต้องของวงจรเข้ารหัสและถอดรหัส นอกจากนี้ยังได้สร้างช่องสัญญาณจำลองแบบ Binary Symmetric Channel (BSC) ขึ้นมาเพื่อทดสอบประสิทธิภาพของตัวถอดรหัส และเพื่อเป็นการยืนยันว่าวงจรเข้ารหัสและวงจรถอดรหัสของรหัสแบบเทอร์โบสามารถที่จะทำงานได้จริง

ชินพจน์ วงศ์ศรีพิสันต์ — ลายมือชื่อนิสิต

ลายมือชื่อประธานกรรมการ — 24 / 3 / 49

Chinnapot Wongsripisant 2006: A Development of Turbo Encoder and Decoder on FPGA. Master of Engineering (Electrical Engineering), Major Field: Electrical Engineering, Department of Electrical Engineering. Thesis Advisor: Associate Professor Mongkol Raksapatcharawong, Ph.D. 42 pages.
ISBN 974-16-1407-1

This research developes turbo encoder and decoder for communication systems on FPGA board using VHDL. This functional block development can apply to many communication systems such as a 3G cellular system. We test all block function to check the correction of encoding and decoding by loopback with encoder and decoder block. At last we implement channel for a Binary Symmetric Channel (BSC) in order to test performance of turbo decoder and verify functional block in real communication systems.

Chinnapot Wongsripisant — Student's signature

Thesis Advisor's signature — 24 / 3 / 49

กิตติกรรมประกาศ

ข้าพเจ้าขอกราบขอบพระคุณ รองศาสตราจารย์ ดร. มงคล รักษาพัชรวงศ์ ประธานกรรมการที่ปรึกษา ผู้ช่วยศาสตราจารย์ วัชรี วีรคเชนทร์ และ ผู้ช่วยศาสตราจารย์ ดร. ศรีจิตรา มหาประคุณชัย ที่ได้ให้ความช่วยเหลือในการวางแผนงานวิจัยในวิทยานิพนธ์ฉบับนี้ ตลอดจนการให้คำปรึกษาแนะนำและตรวจแก้ไขข้อบกพร่องในวิทยานิพนธ์ให้สำเร็จลุล่วงไปด้วยดี

ข้าพเจ้าขอกราบขอบพระคุณ คุณพ่อ คุณแม่ ที่คอยให้ความรักและดูแลเอาใจใส่ ขอบคุณเพื่อน ๆ ที่คอยให้กำลังใจ และขอบคุณพี่น้องในห้อง SCORPion Lab ทุกคนโดยเฉพาะ นายศิริชัย แซ่หว่อง ที่มีส่วนสำคัญในการช่วยแนะนำและแก้ไขในการทำวิทยานิพนธ์ฉบับนี้ให้สำเร็จลุล่วงไปได้ด้วยดี และหากวิทยานิพนธ์ฉบับนี้มีข้อบกพร่องประการใด ข้าพเจ้ายินดีรับข้อเสนอแนะและขออภัยมา ณ ที่นี้ด้วย

ชินพจน์ วงศ์ศรีพิสันต์

มีนาคม 2549

สารบัญ

หน้า

สารบัญตาราง

สารบัญภาพ

สารบัญภาพ(ต่อ)

การพัฒนาวงจรเข้ารหัสและถอดรหัสแบบเทอร์โบด้วย FPGA

A Development of Turbo Encoder and Decoder on FPGA

คำนำ

การเข้ารหัสแบบเทอร์โบโค้ดนั้นเป็นการเข้ารหัสข้อมูลในระบบการสื่อสารรูปแบบหนึ่ง ที่มีวัตถุประสงค์ในการทำงาน เพื่อที่จะลดความผิดพลาดของข้อมูลที่เกิดขึ้นระหว่างการส่งข้อมูลผ่านระบบสื่อสาร ที่มีผลมาจากการถูกสัญญาณรบกวนต่างๆที่เกิดขึ้นในระบบสื่อสาร ผลที่ได้จากการเข้ารหัสข้อมูลแบบเทอร์โบโค้ดในระบบสื่อสาร สามารถที่ทำให้ข้อมูลต่าง ๆ ที่ถูกส่งผ่านในระบบสื่อสารนั้น มีอัตราการเกิดความผิดพลาดที่น้อยลง ซึ่งส่งผลให้สามารถส่งข้อมูลโดยใช้กำลังส่งที่ลดลงได้ ซึ่งความสัมพันธ์ระหว่างค่าอัตราส่วนระหว่างกำลังส่งข้อมูลต่อกำลังของสัญญาณรบกวน หรือ S/N และสำหรับระบบสื่อสารที่มีการนำหลักการของ เทอร์โบโค้ดมาทำการใช้งานนั้นจะสามารถลดความต้องการค่า S/N ของภาคถอดรหัสลงได้จนมีค่าที่ใกล้เคียงกับค่าในทฤษฎีของ Shannon โดยในการเข้ารหัสข้อมูลแบบเทอร์โบโค้ดจะเป็นการเข้ารหัสข้อมูลที่มีการนำหลักการเกี่ยวกับการเข้ารหัสในรูปแบบของ Parallel Concatenated และมีการนำวิธีการถอดรหัสแบบ Iterative decoding มาใช้ในการทำงาน

ในรายงานการวิจัยฉบับนี้จะเป็นการบรรยายเกี่ยวกับหลักการพื้นฐาน และการพัฒนาวงจรเข้ารหัสและวงจรถอดรหัสของรหัสเทอร์โบบนบอร์ด FPGA ด้วยภาษา VHDL เพื่อที่จะสามารถนำส่วนของวงจรเข้ารหัสและวงจรถอดรหัสแบบเทอร์โบที่ได้พัฒนาขึ้นนั้น นำไปประยุกต์ใช้งานได้จริงในงานด้านการสื่อสารข้อมูลรูปแบบอื่น ๆ ต่อไป

วัตถุประสงค์

1. เพื่อพัฒนาฟังก์ชันการทำงานของส่วนต่างๆ ด้วยภาษา VHDL เพื่อสามารถนำไปใช้กับการสื่อสารดิจิตอลไร้สายแบบอื่น ๆ ได้

2. เพื่อพัฒนาและปรับให้บล็อกการถอดรหัสเทอร์โบโค้ดมีประสิทธิภาพในการทำงานที่สูงขึ้น

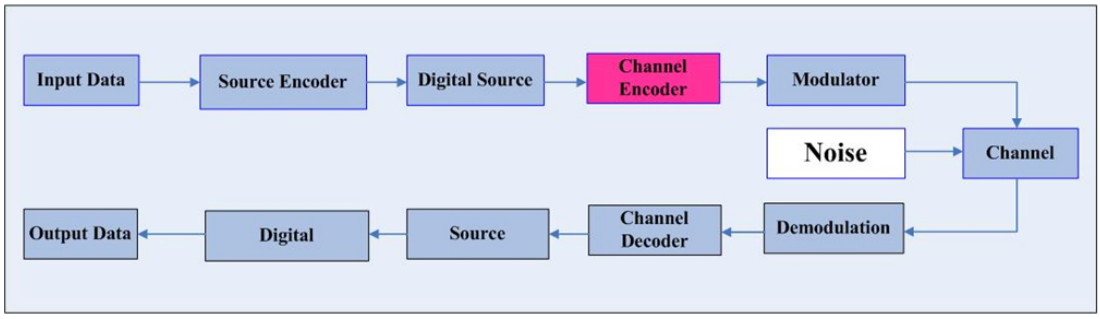
3. ประหยัดค่าใช้จ่ายในการที่จะต้องซื้อ ip core ของส่วนเข้ารหัสและถอดรหัสของเทอร์โบที่มีราคาสูงเพราะสามารถพัฒนาเพื่อนำมาใช้งานเองได้

4. สามารถนำวงจรเข้ารหัสและวงจรถอดรหัสของรหัสเทอร์โบโค้ดไปประยุกต์ใช้งานจริงกับระบบการสื่อสารรูปแบบอื่น ๆ ได้

การตรวจเอกสาร

1. การเข้ารหัสช่องสัญญาณในระบบสื่อสาร (Channel Coding)

ในการออกแบบหรือใช้งานระบบสื่อสารแบบดิจิตอลนั้น จำเป็นจะต้องมีการพิจารณาองค์ประกอบในหลาย ๆ ส่วนด้วยกัน โดยสิ่งหนึ่งที่จะต้องมีการพิจารณาก็คือ ข้อมูลดิบที่ถูกส่งจากต้นทางไปถึงปลายทางนั้นมีข้อมูลที่เกิดความผิดพลาดนั้นหรือไม่ ที่เกิดจากสาเหตุต่าง ๆ หลายสาเหตุด้วยกัน โดยที่สาเหตุหลักที่จะทำให้เกิดความผิดพลาดดังกล่าวก็คือ การที่ระบบสื่อสารนั้นถูกรบกวนจากสัญญาณรบกวนต่าง ๆ ถ้าหากว่าขนาดของสัญญาณรบกวนที่เกิดขึ้นในระบบสื่อสารนั้นมีค่าที่สูง จะส่งผลให้อัตราการเกิดความผิดพลาดของข้อมูล (Bit Error Rate) ที่เกิดขึ้นมีค่าสูงตามไปด้วย ในการที่จะลดอัตราการเกิดความผิดพลาดของข้อมูลให้มีค่าที่ลดลงนั้น สามารถทำได้หลายรูปแบบด้วย เช่น เพิ่มกำลังของเครื่องส่ง, การทำให้ขนาดของสัญญาณรบกวนมีค่าน้อยลง หรือการเข้ารหัสช่องสัญญาณ เป็นต้น



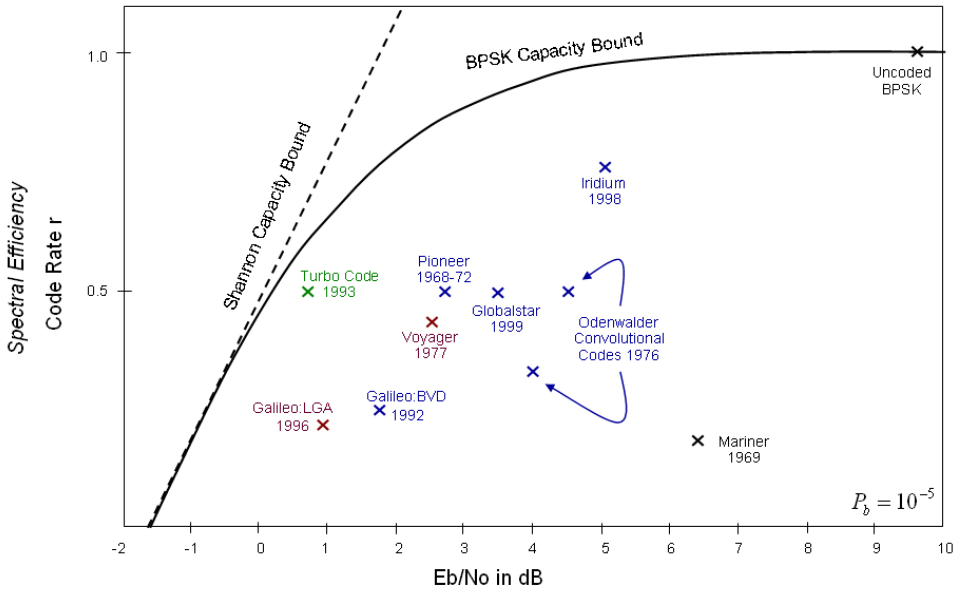
ภาพที่ 1 โครงสร้างพื้นฐานของการสื่อสารข้อมูล

สำหรับการลดอัตราการเกิดความผิดพลาดที่เกิดขึ้นในการส่งข้อมูลด้วยวิธีการเข้ารหัส (Coding) นั้น เป็นการนำข้อมูลดิบที่จะทำการส่งผ่านระบบสื่อสารที่เป็นข้อมูลแบบดิจิตอลมาทำกระบวนการเข้ารหัส (Encoding) เพื่อเปลี่ยนรูปแบบของข้อมูลที่จะถูกส่งผ่านระบบสื่อสารให้อยู่ในรูปแบบที่สามารถนำข้อมูลมาทำการแก้ไขความผิดพลาดของข้อมูลที่เกิดขึ้นเนื่องจากการถูกรบกวนจากสัญญาณรบกวนต่าง ๆ ได้ โดยที่ข้อมูลที่ได้จากการทำงานนั้นจะเป็นข้อมูลที่จะถูกส่งออกไปผ่านระบบสื่อสาร และเมื่อข้อมูลดังกล่าวถูกส่งมาถึงปลายทางจะมีการนำข้อมูลที่รับได้นั้นมาผ่านกระบวนการถอดรหัส (Decoding) เพื่อเปลี่ยนรูปแบบของข้อมูลที่รับได้ให้กลับมาอยู่ในรูปของ

ข้อมูลดิบพร้อมทั้งทำการแก้ไขข้อมูลที่คาดว่าจะเกิดความผิดพลาดขึ้นให้มีค่าที่ถูกต้องโดยที่รูปแบบหรือวิธีการที่ใช้ในการเข้ารหัสข้อมูลนั้นจะมีอยู่หลายวิธีด้วยกัน ซึ่งก่อนที่จะมีการค้นพบการเข้ารหัสแบบเทอร์โบวิธีการเข้ารหัสแบบ Convolution Codes จะเป็นรูปแบบในการเข้ารหัสข้อมูลที่มีการนิยมใช้งานมากที่สุด โดยจะมีการนิยมใช้วิธีแบบ Viterbi Decoding ในการถอดรหัสข้อมูลสำหรับ Convolution Codes

2. รหัสเทอร์โบ (Turbo Codes)

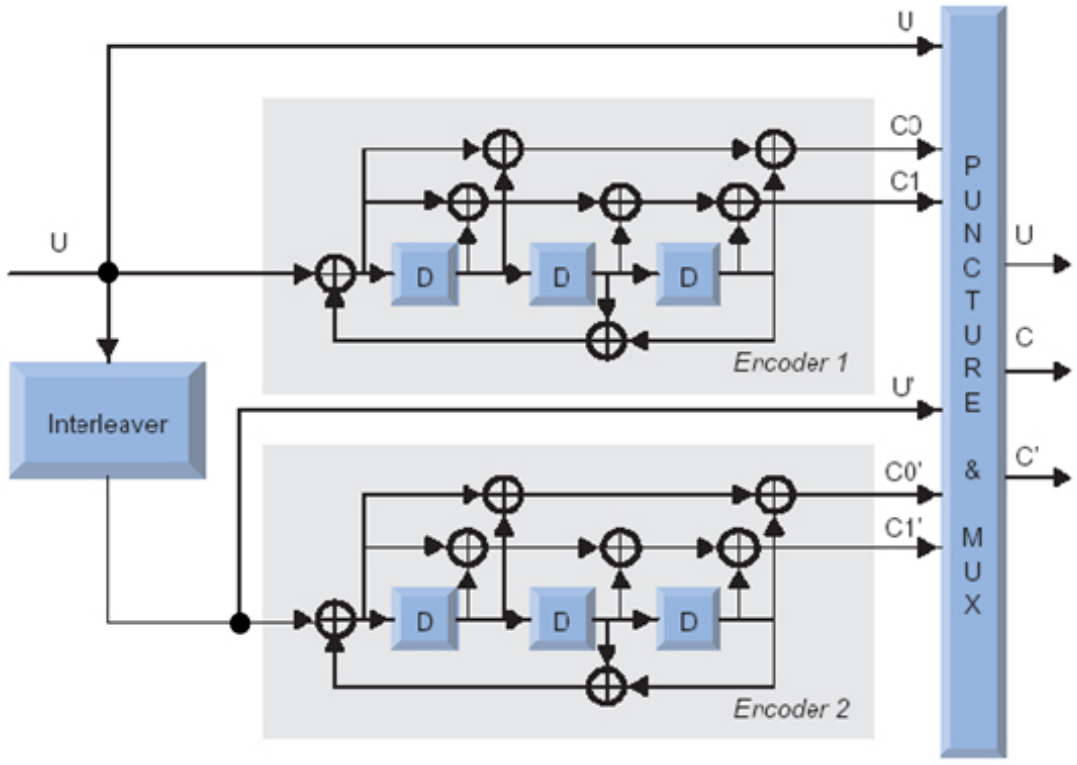
รหัสเทอร์โบเป็นหนึ่งในวิธีการเข้าและถอดรหัสช่องสัญญาณการสื่อสาร (Channel Coding) ซึ่งมีจุดประสงค์เพื่อแก้ไขความผิดพลาดในการส่งข้อมูลเนื่องจากผลกระทบของสัญญาณรบกวน ที่มีประสิทธิภาพสูงมาก โดยได้มี การค้นพบและพัฒนาขึ้นในปี ค.ศ. 1993 โดย Claude Berrou, Alian Glavieux และ Punya Thitimajshima สามารถที่จะให้คุณภาพของการสื่อสารใกล้เคียงกับค่าที่กำหนดจากทฤษฎีของ Shannon มากที่สุด เมื่อเปรียบกับรหัสแบบอื่น ๆ ดังแสดงในภาพที่ 2 นอกจากนี้ยังสามารถนำไปสร้างวงจรเข้ารหัสและถอดรหัสด้วยวิธีการที่ไม่ซับซ้อนมาก และสามารถนำไปประยุกต์ใช้ในงานในด้านระบบการสื่อสารที่มีการรับส่งข้อมูลแบบดิจิตอลแบบต่าง ๆ ได้อย่างกว้างขวาง



ภาพที่ 2 คุณภาพของการสื่อสารของรหัสเทอร์โบเปรียบเทียบกับค่าที่กำหนดจากทฤษฎีของ Shannon

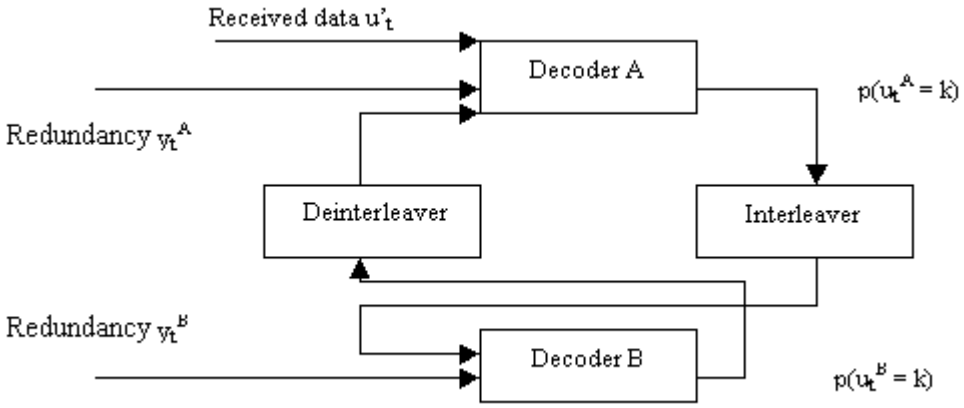
รหัสเทอร์โบมีหลักการและคุณสมบัติที่แตกต่างจากวิธีการเข้าและถอดรหัสช่องสัญญาณในแบบอื่น ๆ ดังนี้

การเข้ารหัสของรหัสเทอร์โบเป็นการนำเอาวงจรเข้ารหัสซิสเต็มเมติกส์คอนโวลูชันแบบรีเคอร์ซีฟ (Recursive Systematic Convolution Codes: RSC) จำนวนตั้งแต่ 2 ชุดขึ้นไปมาต่อกันแบบขนาน โดยที่วงจรเข้ารหัสนั้นจะถูกต่อร่วมกับอินเตอร์ลีฟเวอร์ (Interleaver) ดังตัวอย่างที่แสดงตามภาพที่ 3 ซึ่งเป็นการเข้ารหัสของรหัสเทอร์โบที่มีอัตราการเข้ารหัสเท่ากับ 1/5 ด้วยการใช้วงจรเข้ารหัส RSC จำนวน 2 ตัวมาต่อกัน



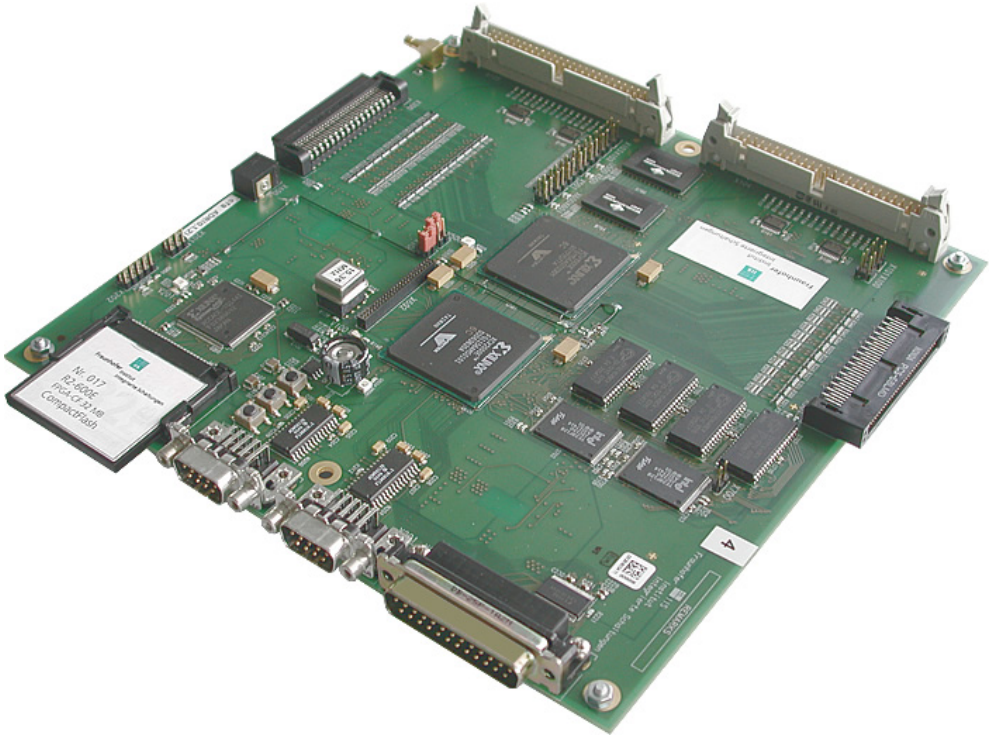
ภาพที่ 3 วงจรเข้ารหัสแบบ Turbo Codes

การถอดรหัสของรหัสเทอร์โบจะมีลักษณะการทำงานเป็นการถอดรหัสแบบป้อนกลับและมีการวนซ้ำ (Iterative Decoding) โดยนำผลการคำนวณส่วนหนึ่งมาใช้เป็นข้อมูลสำหรับการคำนวณในรอบต่อ ๆ ไป ทำให้ข้อมูลที่ได้จากการถอดรหัสมีความถูกต้องมากยิ่งขึ้น ดังตัวอย่างที่แสดงตามภาพที่ 4



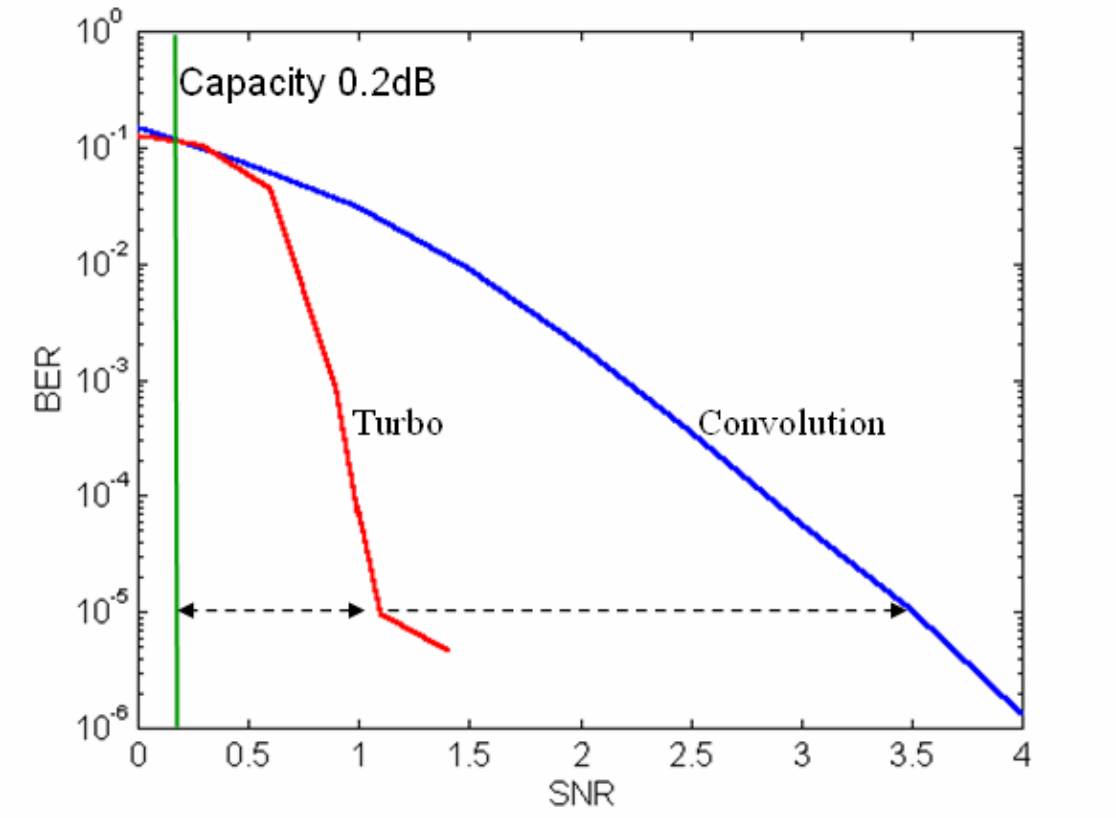
ภาพที่ 4 วงจรถอดรหัส Turbo Codes

การสร้างวงจรถอดรหัสของรหัสเทอร์โบเพื่อนำไปใช้งานสามารถทำได้ด้วยวิธีการที่ไม่ซับซ้อนมากนัก ด้วยการนำไปสร้างเป็นโมดูลย่อย ๆ และนำมาต่อกัน โดยที่แต่ละโมดูลสามารถสร้างวงจรสำเร็จรูปได้อย่างง่าย โดยการนำไปสร้างด้วยอุปกรณ์ที่สามารถโปรแกรมได้ เช่น FPGA



ภาพที่ 5 บอร์ด FPGA

สำหรับประสิทธิภาพของรหัสเทอร์โบนั้นสามารถแสดงได้ด้วยผลการจำลองค่าความผิดพลาดของบิตข้อมูล (Bit Error Rate: BER) กับอัตราส่วนของกำลังของสัญญาณต่อกำลังของสัญญาณรบกวน (Eb/No) ดังแสดงในภาพที่ 6



ภาพที่ 6 เปรียบเทียบประสิทธิภาพของรหัสเทอร์โบกับรหัสคอนโวลูชัน

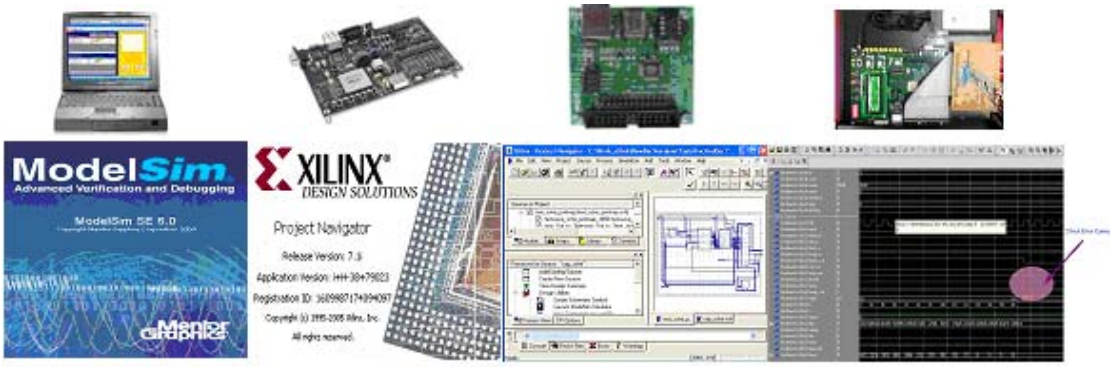
เห็นได้ว่ารหัสเทอร์โบมีค่าของความผิดพลาดของบิตข้อมูล (BER) ห่างจากค่าที่กำหนดจากทฤษฎีของ Shannon น้อยมาก ซึ่งถือได้ว่าเข้าใกล้ค่าที่กำหนดจากทฤษฎีมากที่สุดเท่าที่เคยมีมาก่อน

ด้วยประสิทธิภาพของรหัสเทอร์โบดังกล่าว ปัจจุบันมีการนำรหัสเทอร์โบไปประยุกต์ใช้งานในด้านการสื่อสารหลายประเภท ไม่ว่าจะเป็นการสื่อสารทางสาย การสื่อสารผ่านดาวเทียม และระบบโทรศัพท์เคลื่อนที่ยุคที่ 3 เป็นต้น

อุปกรณ์และวิธีการ

อุปกรณ์

1. คอมพิวเตอร์ส่วนบุคคล Pentium 4 , 2.4 GHz , 1GB DDR-RAM, 60 GB HDD
2. โปรแกรม Xilinx
3. โปรแกรม ModelSim
4. โปรแกรม Matlab
5. บอร์ดพัฒนา FPGA xc4vlx25 ของบริษัท Xilinx



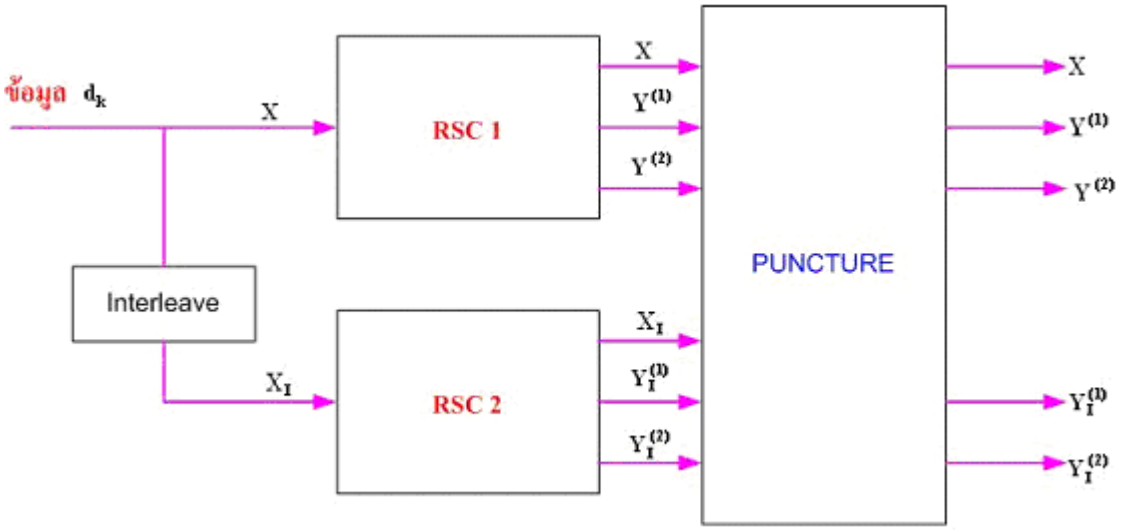
ภาพที่ 7 ฮาร์ดแวร์และซอฟต์แวร์ที่ใช้งาน

วิธีการ

งานวิจัยชิ้นนี้ทำการทดสอบแนวคิดการเข้ารหัสและการถอดรหัสแบบเทอร์โบด้วยการเขียนโค้ดด้วยภาษา VHDL แล้วทำการจำลอง (simulation) บนเครื่องคอมพิวเตอร์ ด้วยโปรแกรม Model Sim เมื่อโค้ดที่เขียนขึ้นทำการเข้ารหัสและถอดรหัสได้ถูกต้องแล้วก็ทำการนำโมดูลต่าง ๆ ที่ได้พัฒนานั้นไปประมวลผลในฮาร์ดแวร์ จากนั้นทำการทดสอบประสิทธิภาพของส่วนการถอดรหัสแบบเทอร์โบแบบผ่านช่องสัญญาณจำลองที่ได้สร้างขึ้นซึ่งเป็นแบบ Binary Symmetric Channel (BSC) โดยการป้อนอินพุตผ่านส่วนเชื่อมต่อกับโมดูลส่งสัญญาณเสียงผ่านโปรแกรม Matlab แล้วทำการวัดค่า Bit Error Rate (BER) ที่ได้เมื่อเปลี่ยนค่า Probability of error ที่ช่องสัญญาณ

1. การเข้ารหัส Turbo Code

การเข้ารหัสเทอร์โบที่นำมาพัฒนาในงานวิจัยนี้มีอัตราเข้ารหัสเทอร์โบเท่ากับ 1/5 ตามภาพที่ 8 ประกอบด้วย 2 ส่วนหลัก ๆ คือ ส่วนการเข้ารหัสบิตข้อมูล และส่วนการสลับบิตของเทอร์โบ

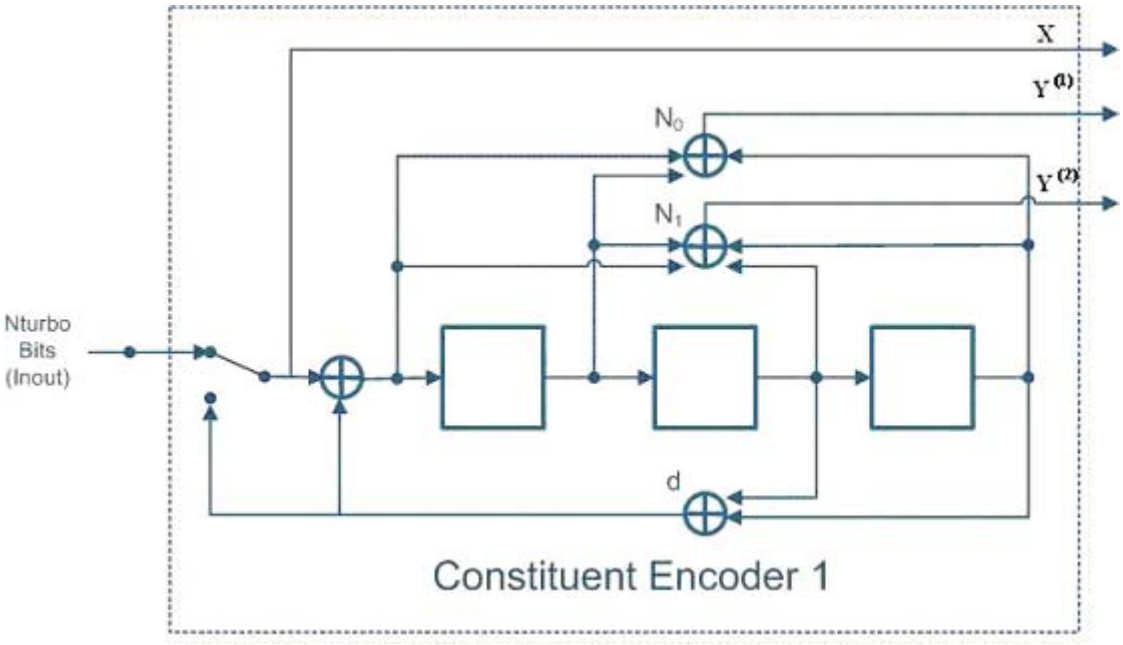


ภาพที่ 8 วงจรเข้ารหัสแบบเทอร์โบที่มีอัตราเข้ารหัสเทอร์โบเท่ากับ 1/5

1.1 ส่วนการเข้ารหัสบิตข้อมูล

สำหรับการเข้ารหัสบิตข้อมูลนั้นจะใช้การเข้ารหัสข้อมูลแบบ Recursive Systematic Convolution Code (RSC) ซึ่งเป็นรูปแบบของการเข้ารหัสข้อมูลที่มีการนำคุณสมบัติของการเข้ารหัสแบบ Nonsystematic และ Systematic Convolution Codes รวมเข้าด้วยกันเพื่อให้ข้อมูลที่ได้จากการเข้ารหัสนั้นมีลักษณะของข้อมูลเป็นแบบ Systematic ที่มีการส่งข้อมูลดิบไปพร้อมกับข้อมูลที่ผ่านการเข้ารหัส แต่ยังคงมีค่าความแตกต่างของข้อมูล (distance) ซึ่งเป็นค่าที่แสดงถึงความสามารถในการป้องกันความผิดพลาดของข้อมูลมีค่าเท่ากับกรณีของ Nonsystematic ซึ่งผลที่ได้นั้นจะทำให้ความสามารถในการป้องกันความผิดพลาดของข้อมูลในกรณีของการเข้ารหัสแบบ RSC นั้นมีค่ามากกว่ากรณีของ Nonsystematic Convolution Code ณ สภาวะของระบบสื่อสารที่มีค่า S/N ต่ำ และมีการเข้ารหัสด้วยอัตราการเข้ารหัสที่สูง สำหรับวงจรที่ใช้ในการเข้ารหัสแบบ RSC จะเป็นวงจรเข้ารหัสที่มีการดัดแปลงมาจากวงจรเข้ารหัสแบบ Nonsystematic Convolution Codes โดยการนำ

สัญญาณข้อมูลที่ได้จากการเข้ารหัสสัญญาณใดสัญญาณหนึ่งป้อนกลับ (Feedback) มาเป็นข้อมูลอินพุทของข้อมูลเข้ารหัส

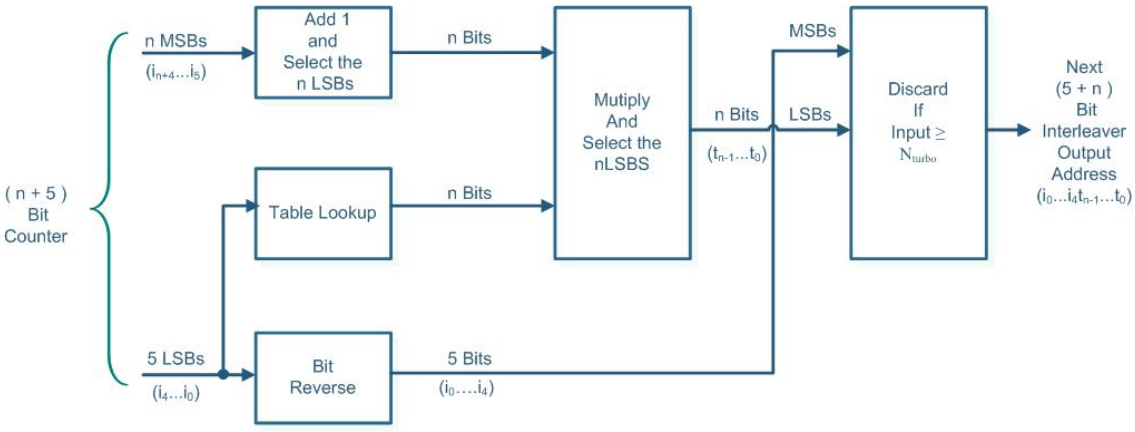


ภาพที่ 9 วงจรเข้ารหัสบิตข้อมูล

1.2 การสลับบิตของเทอร์โบ (Interleave)

การ Interleave ข้อมูลนั้นเป็นกระบวนการในการเปลี่ยนแปลงการจัดเรียงข้อมูลดิจิตอลให้มีลักษณะที่แตกต่างออกไปจากเดิม ซึ่งโดยปกติแล้วจะถูกนำมาใช้ในระบบสื่อสารเพื่อป้องกันการผิดพลาดของข้อมูลในรูปแบบของ Burst Error แต่สำหรับกรณีของวงจรเข้ารหัสแบบเทอร์โบนั้น จะมีการนำ Interleave มาใช้งานโดยมีจุดประสงค์เพื่อทำให้ข้อมูลที่ได้จากการเข้ารหัสจากวงจรเข้ารหัสต่าง ๆ นั้นมีลักษณะของข้อมูลที่ไม่มีความสัมพันธ์ซึ่งกันและกัน โดยจะเป็นการนำข้อมูลดิบที่จะทำการเข้ารหัสในวงจรเข้ารหัสต่าง ๆ มาผ่านการ Interleave เพื่อทำการเปลี่ยนแปลงรูปแบบของข้อมูลเพื่อให้ข้อมูลที่ถูกเข้ารหัสนั้นมีลักษณะที่แตกต่างกัน

สำหรับวิธีการคำนวณหาตำแหน่งที่ใช้ในการสลับบิตในโมดูล Interleave นั้นมีอยู่หลายวิธีด้วยกันแต่สำหรับงานวิจัยนี้จะคำนวณหาตำแหน่งที่ใช้ในการสลับบิตในโมดูล Interleave ดังแสดงได้ในภาพที่ 10



ภาพที่ 10 ส่วนสร้างตำแหน่งสลับบิตในโมดูล Interleave

จากรูปค่า n จะแปรค่าตามค่าของ Turbo Interleave Block Size (N_{turbo}) ซึ่งค่า N_{turbo} ที่เลือกใช้คือ 3858 ดังนั้นค่า n ก็จะมีค่าเท่ากับ 7 ดังตารางที่ 1

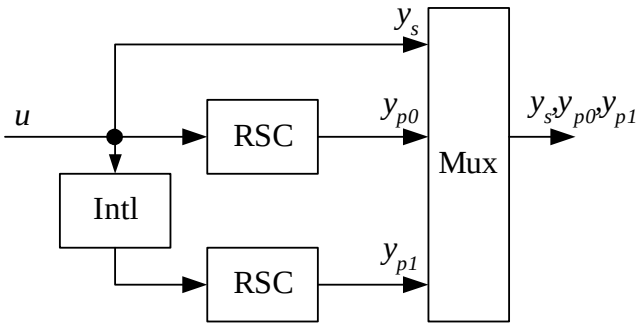
ตารางที่ 1 ค่า Block size และ Turbo Interleave parameter

Turbo Interleaver Block Size	**Turbo Interleave Parameter**
N_{turbo}	**n**
210	3
402	4
786	5
1,146	6
3,066	7
3,090	7
3,858	7
4,602	8
6,138	8
9,210	9
12,282	9
20,730	10

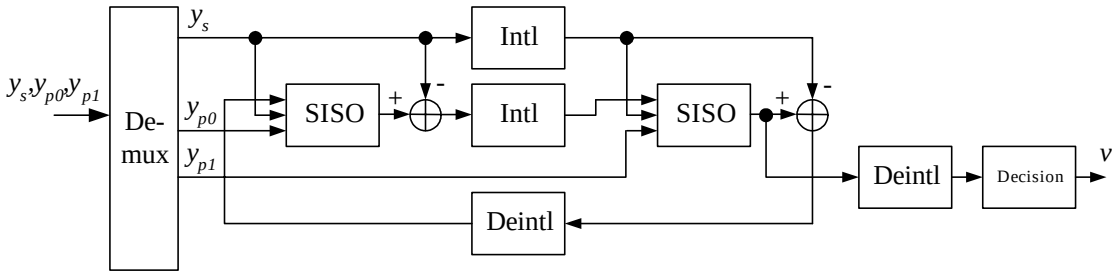
2. <u>การถอดรหัส Turbo Code</u>

การเข้ารหัสแบบเทอร์โบโค้ดเป็นการเข้ารหัสที่มีการใช้ส่วนเข้ารหัสย่อยมากกว่าหนึ่งตัวมาสร้างคำรหัส (code word) โดยรหัสข้อมูลที่ส่งให้แก่ส่วนเข้ารหัสย่อยแต่ละตัวจะพยายามให้มีความสัมพันธ์กันต่ำ ซึ่งโดยปกตินิยมใช้การสลับบิตข้อมูลแบบกึ่งสุ่ม (pseudo-random Interleaver) คำรหัสที่ได้ เป็นรหัสที่ได้จากสายรหัสข้อมูลต้นฉบับ รวมเข้ากับสายรหัสพาริตี้ที่สร้างได้จากส่วนเข้ารหัสย่อยทั้งหมด ลักษณะการเข้ารหัสที่กล่าวมานี้จะใช้วิธีการถอดรหัสแบบวนรอบ (iteration) เพื่อดึงประสิทธิผลที่มีทั้งหมดออกมา วิธีการถอดรหัสแบบเทอร์โบเป็นการใช้ส่วนถอดรหัสย่อยหลาย ๆ ตัวมาช่วยในการถอดคำรหัสที่รับเข้ามา และแลกเปลี่ยนความผิดพลาดระหว่างส่วนถอดรหัสย่อยต่างๆโดยผลที่ได้จะนำมาใช้ในการตัดสินใจเลือกรหัสข้อมูลที่ถูกต้องในภายหลัง

การเข้ารหัสเทอร์โบในงานวิจัยนี้ เป็นการเข้ารหัสโดยมีส่วนเข้ารหัสย่อยแบบ RSC (Recursion Systematic Convolutional) จำนวน 2 ตัว และมีส่วนสลับบิตข้อมูลแบบกึ่งสุ่มจำนวน 1 ตัว ดังแสดงในภาพที่ 11 การถอดรหัสเทอร์โบโค้ดนี้จึงจำเป็นต้องใช้ส่วนถอดรหัสย่อยแบบ SISO (Soft in Soft out) จำนวน 2 ตัว เพื่อใช้ถอดรหัสที่สร้างได้จากส่วนเข้ารหัสย่อยในแต่ละตัว และแลกเปลี่ยนข้อมูลเกี่ยวกับความผิดพลาดระหว่างสายรหัส ในทางปฏิบัติเราสามารถสร้างส่วนถอดรหัสเพียงตัวเดียวให้ทำงานเป็นส่วนถอดรหัสย่อยแต่ละตัวในเวลาต่างกันได้ ซึ่งทำให้สามารถใช้ประโยชน์จากวงจรได้อย่างสูงสุด

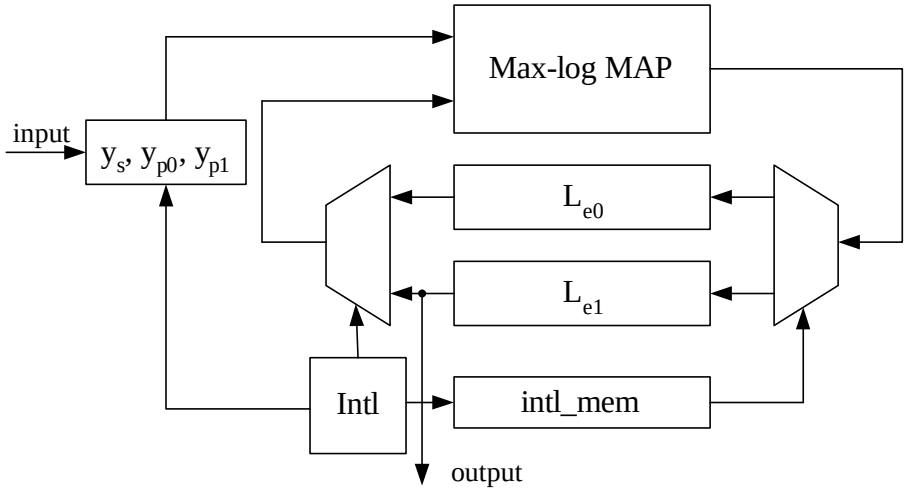


<u>ภาพที่ 11</u> แผนภาพแสดงการทำงานของส่วนเข้ารหัสเทอร์โบโค๊ดแบบใช้ส่วนเข้ารหัสย่อยแบบ RSC จำนวน 2 ตัว



ภาพที่ 12 แผนภาพแสดงการทำงานของส่วนถอดรหัสเทอร์โบโค๊ดที่สร้างได้จากการเข้ารหัส

วงจรถอดรหัสเทอร์โบโค้ดที่ออกแบบมีโครงสร้างโดยรวมดังแสดงแสดงในภาพที่ 12 ซึ่งมีส่วนประกอบหลักของการถอดรหัสเทอร์โบโค้ด คือ ส่วนถอดรหัสย่อยซึ่งในส่วนเราจะใช้การถอดรหัสแบบ Max-Log MAP (Maximum A Posteriori), ส่วนสร้างตำแหน่งการสลับบิตข้อมูล หน่วยความจำ extrinsic หน่วยความจำคำรหัส และส่วนจักรกลสถานะ (Main State Machine)



ภาพที่ 13 โครงสร้างของวงจรถอดรหัส

ส่วนสร้างตำแหน่งการสลับบิตข้อมูลเริ่มการคำนวณหาพารามิเตอร์ที่ใช้ในการสร้างตำแหน่งในการสลับบิตข้อมูล หลังจากที่คำรหัสที่รับเข้ามาถูกนำมาเก็บไว้ในหน่วยความจำคำรหัสเรียบร้อยแล้ว สายรหัสต้นฉบับ ys และสายรหัสพาริตี้ yp0 จะถูกลำเลียงให้แก่ส่วนถอดรหัสแบบ Max-Log MAP หลังจากคำนวณหาค่าแล้วจะลำเลียงนำค่าไปจัดเก็บในหน่วยความจำ extrinsic Le0 โดยตำแหน่งที่จัดเก็บจะได้จากส่วนสร้างตำแหน่งการสลับบิตข้อมูล เมื่อเสร็จสิ้นสายรหัสต้นฉบับ ys และสายรหัสพาริตี้ yp1พร้อมค่าที่จัดเก็บในหน่วยความจำ extrinsic Le0 จะถูกลำเลียงให้แก่ส่วน

ถอดรหัสแบบ Max-Log MAP โดยตำแหน่งจะขึ้นกับส่วนสร้างตำแหน่งการสลับบิตข้อมูล หลังจากคำนวณหาค่าแล้วจะลำเลียงมาจัดเก็บในหน่วยความจำ extrinsic Le1 โดยตำแหน่งที่อยู่ใน intl_mem เมื่อทำจนเสร็จสิ้นจะถือว่าเป็นหนึ่งรอบของการถอดรหัส หลังจากนั้นจะเริ่มรอบที่สองในลักษณะเดียวกันนี้ เมื่อจำนวนรอบของการถอดรหัสเพิ่มขึ้น ค่าอัตราบิตผิดพลาด (BER) มักมีค่าต่ำลง แต่ในทางกลับกันอัตราการไหลของข้อมูลที่ผ่านส่วนถอดรหัสก็มีค่าต่ำลง

2.1 ส่วนถอดรหัสแบบ Max-Log MAP (Maximum A Posteriori)

ส่วนถอดรหัสแบบ Max-Log MAP เป็นวิธีที่ปรับปรุงจาก MAP โดยแปลงให้อยู่ในโดเมนของล็อการิทึมซึ่งการกระทำคูณและหาร จะแปลงเป็นการกระทำบวกและลบ ทำให้มีความซับซ้อนน้อยลง แต่ก็ต้องแลกกับการสูญเสียค่าประสิทธิผล วิธีในการถอดรหัสมีหลักการง่าย ๆ คือ การหาค่าโอกาสที่ข้อมูลที่ตำแหน่งนั้นมีค่าเป็นข้อมูลอะไร ซึ่งเมื่อเทียบกับการถอดรหัสด้วยวิธี Viterbi ซึ่งเป็นการหาคำรหัสที่ใกล้เคียงที่สุดแล้ว การถอดรหัสแบบ MAP จะให้ค่าอัตราบิตผิดพลาดต่ำกว่าแบบ Viterbi แต่ก็จำเป็นต้องใช้หน่วยความจำในการเก็บค่าความน่าจะเป็นเป็นจำนวนมาก

2.2 ขั้นตอนวิธีการทำงาน (Algorithm)

การทำงานจะใช้ประโยชน์จากร่างแห (Trellis) ของรหัส ร่วมกับการหาค่าความน่าจะเป็นแอลฟ่า, เบต้า และแกมม่า มาช่วยในการถอดรหัส โดยเราสามารถสรุปใจความได้ว่า

ค่าความน่าจะเป็นแอลฟ่า (α-Probability) หมายถึง ค่าความน่าจะเป็นที่โหนดนั้นๆ มีโอกาสปรากฏขึ้นเมื่อทราบว่าโหนดก่อนหน้าปรากฏ มีสมการในการคำนวณดังนี้

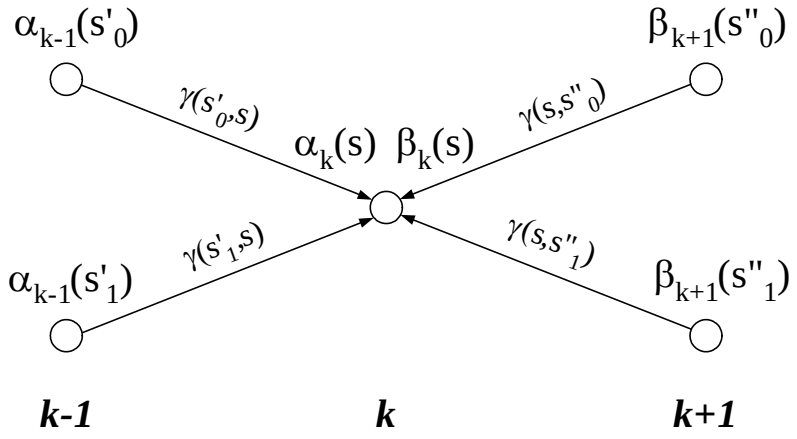
$$\alpha_k(s) = \sum_{all\ s'} \alpha_{k-1}(s') \cdot \gamma_k(s', s)$$

ค่าความน่าจะเป็นเบต้า (β-Probability) หมายถึง ค่าความน่าจะเป็นที่โหนดนั้นๆ มีโอกาสปรากฏขึ้นเมื่อทราบว่าโหนดหลังจากนี้ปรากฏ มีสมการในการคำนวณดังนี้

$$\beta_{k-1}(s') = \sum_{all\ s} \beta_k(s) \cdot \gamma_k(s', s)$$

ค่าความน่าจะเป็นแกมม่า (γ-Probability) หมายถึง ค่าความน่าจะเป็นที่การเปลี่ยนจากโหนดก่อนหน้าไปยังอีกโหนดหนึ่งเมื่อมีข้อมูลอินพุตเข้ามา

$$\gamma_k(s', s) = P(\underline{y}_k \mid \underline{x}_k) \cdot P(u_k)$$



ภาพที่ 14 แผนภาพร่างแหบางส่วนที่แสดงถึงค่าความน่าจะเป็นในโหนด และเส้นทางต่างๆ ที่ตำแหน่ง k-1, k และ k+1

จะเห็นว่าสมการในการหาค่าความน่าจะเป็นแอลฟ่าและเบต้ามีลักษณะเป็นแบบเวียนบังเกิด (Recursion) โดยค่าความน่าจะเป็นของโหนดในตำแหน่งเริ่มต้นจะต้องถูกกำหนดมา ซึ่ง ค่าความน่าจะเป็นเริ่มต้นของแอลฟ่าและเบต้าจะมีค่าเป็น 1 ที่โหนด 0 และมีค่าเป็น 0 ในโหนดอื่นๆ การคำนวณหาค่าความน่าจะเป็นจะทำในทุก ๆ โหนด ซึ่งแต่ละโหนดจะมีทั้งค่าความน่าจะเป็นแอลฟ่า และเบต้า

หลังจากคำนวณค่าความน่าจะเป็นเหล่านี้เรียบร้อยแล้ว จะมีการคำนวณค่าอัตราส่วนความคล้ายคลึงเชิงล็อกการิทึม (Log-likelihood Ratio) ในตำแหน่งต่าง ๆ โดยสมการของค่าอัตราส่วนความคล้ายคลึงเชิงล็อการิทึม คือ

$$L(u_k \mid \underline{y}) = \log\left(\frac{\sum_{u_k=+1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}{\sum_{u_k=-1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}\right)$$

ค่าของอัตราส่วนความคล้ายคลึงเชิงล็อการิทึมเป็นค่าที่บ่งบอกถึงความโน้มเอียงของข้อมูลในแต่ละตำแหน่งมีทิศทางไปทางใด เช่น ถ้ามีค่าอัตราส่วนเป็นค่าบวกมาก ๆ แสดงให้เห็นว่าความโน้มเอียงที่ข้อมูลในตำแหน่งนั้นมีค่าเป็นบิต 1 สูง หรือในทางกลับกันค่าอัตราส่วนนั้นเป็นค่าลบมาก ๆ ก็แสดงให้เห็นว่าความโน้มเอียงที่ข้อมูลในตำแหน่งนั้นมีค่าเป็นบิต 0 สูง ส่วนในกรณีที่ค่าอัตราส่วนมีค่าใกล้เคียง 0 มากจะบ่งบอกว่าข้อมูลในตำแหน่งนั้นมีโอกาสที่จะเป็นบิต 0 และบิต 1 ใกล้เคียงกัน

จากวิธีการคำนวณข้างต้นจะเห็นว่าการกระทำส่วนใหญ่จะเป็นการกระทำคูณ ซึ่งทำให้ความซับซ้อนของวงจรสูง จึงได้มีการปรับปรุงวิธีการโดยแปลงให้อยู่ในโดเมนของล็อกการิทึม เป็นผลให้การคำนวณหาค่าความน่าจะเป็นต่างๆ ในโดเมนของล็อกการิทึมเป็นดังนี้

ค่าความน่าจะเป็นแอลฟ่าในโดเมนของล็อกการิทึม

$$A_k(s) \square \log\left(\alpha_k(s)\right)$$
$$= \log\left(\exp\left[A_{k-1}(s') + \Gamma_k(s',s)\right]_{u_k=+1} + \exp\left[A_{k-1}(s') + \Gamma_k(s',s)\right]_{u_k=-1}\right)$$

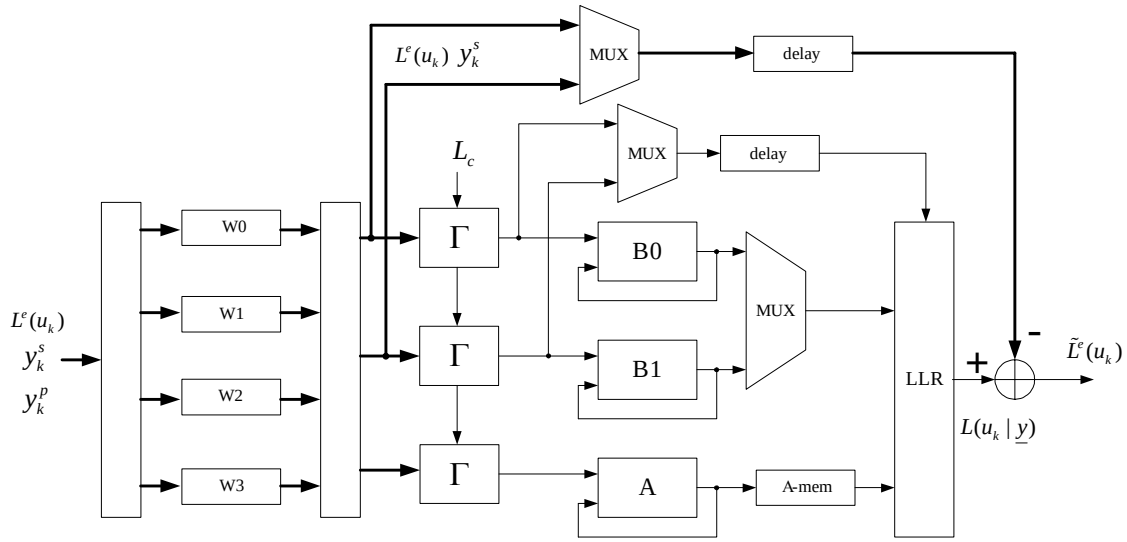
ค่าความน่าจะเป็นเบต้าในโดเมนของล็อการิทึม

$$B_{k-1}(s') \square \log\left(\beta_k(s)\right)$$
$$= \log\left(\exp\left[B_k(s) + \Gamma_k(s',s)\right]_{u_k=+1} + \exp\left[B_k(s) + \Gamma_k(s',s)\right]_{u_k=-1}\right)$$

ค่าความน่าจะเป็นแกมม่าในโดเมนของล็อการิทึม

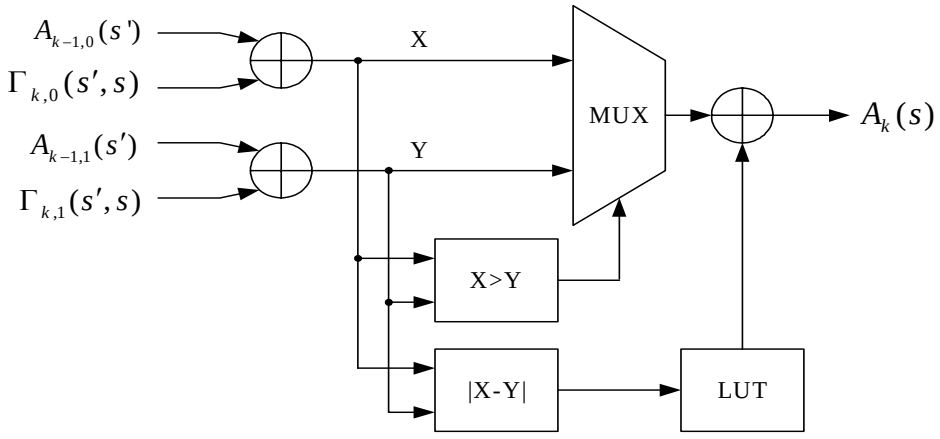
$$\Gamma_k(s',s) \square \log\left(\gamma_k(s',s)\right) = C + \frac{1}{2}u_k L(u_k) + \frac{L_c}{2}\sum_{l=1}^{n} y_{kl}x_{kl}$$

และใช้ความสัมพันธ์ของ $\log\left(e^A + e^B\right) = \max(A,B) + \log\left(1 + e^{-|A-B|}\right)$ ร่วมด้วย ก็จะได้วิธีการ Log MAP และหากเลือกใช้เฉพาะค่า $\max(A,B)$ ก็จะเป็นวิธีการ Max-Log MAP



<u>ภาพที่ 15</u> โครงสร้างของวงจรถอดรหัสแบบ Max-Log MAP ที่ออกแบบ

วงจรคำนวณหาค่าแอลฟ่า (A) และค่าเบต้า (B0, B1) นั้นจะมีวงจรย่อยที่ใช้คำนวณค่าเหมือนกันซึ่งมีชื่อว่า ACS (Add Compare and Select) เป็นวงจรที่มีหน้าที่หลัก คือการเพิ่มค่า, เปรียบเทียบค่า และคัดเลือกค่าที่เหมาะสม ซึ่งสอดคล้องกับสมการหาค่าความน่าจะเป็นแอลฟ่าและเบต้าในโดเมนของล็อการิทึมดังแสดงในภาพที่ 15

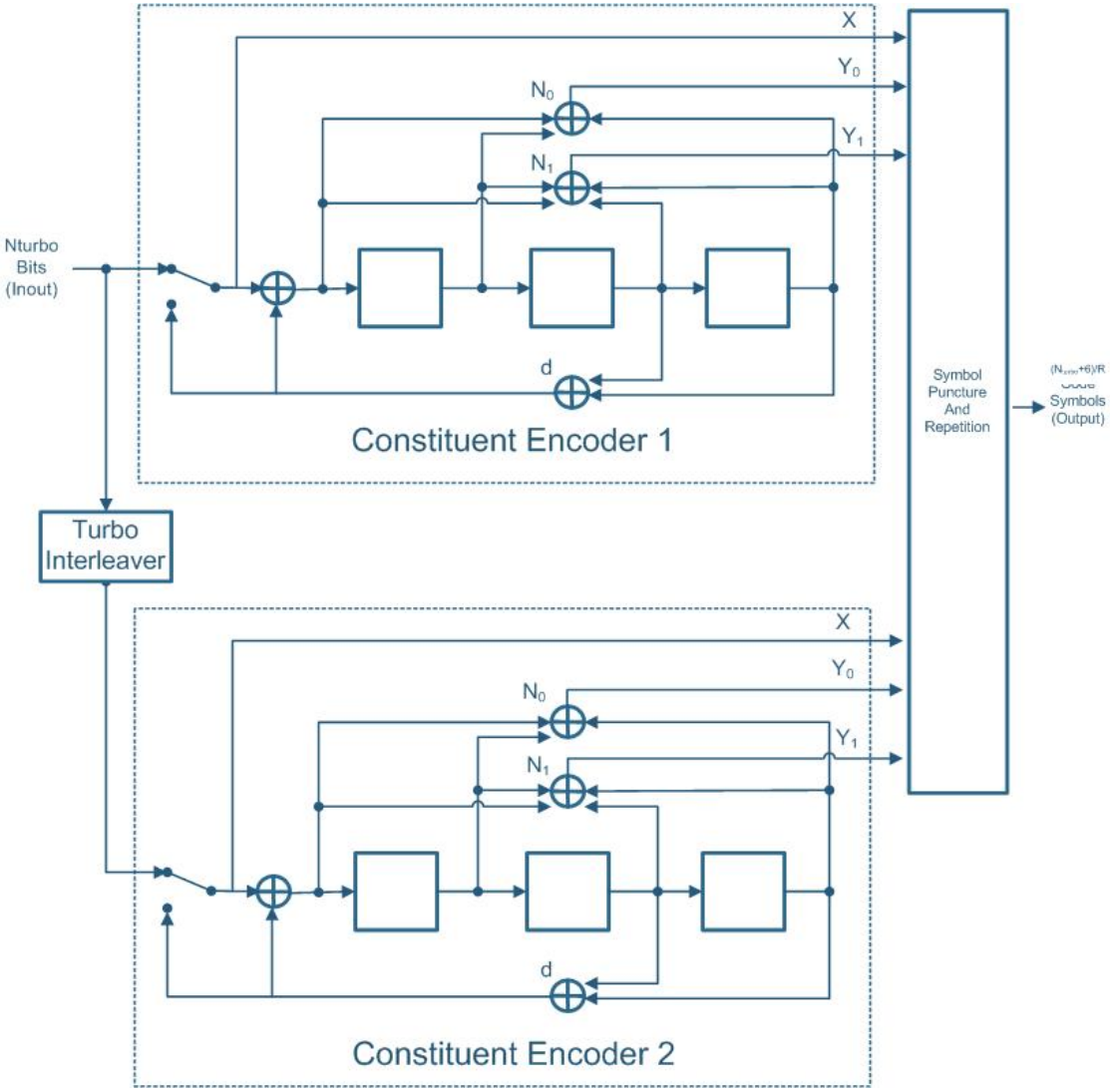


<u>ภาพที่ 16</u> โครงสร้างของวงจร ACS ที่ใช้ในวงจรคำนวณหาค่าแอลฟ่า (A) และค่าเบต้า (B0, B1)

3. การทำงานของซอฟต์แวร์ในส่วนของวงจรการเข้ารหัส

การเข้ารหัสเทอร์โบในงานวิจัยนี้จะใช้อัตราการเข้ารหัสเทอร์โบเท่ากับ 1/5 และขนาดของบิตข้อมูลที่จะเข้ามายังบล็อกตัวเข้ารหัส (frame size) มีค่าเท่ากับ 3858 บิต

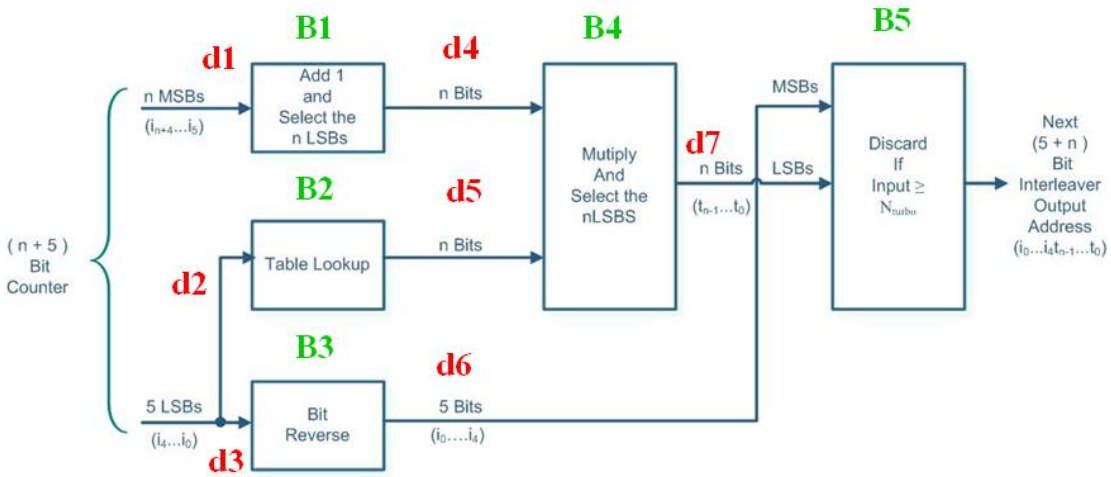
3.1 ส่วนภาคเข้ารหัสบิตข้อมูล



ภาพที่ 17 ส่วนภาคเข้ารหัสบิตข้อมูลของ Turbo Encode Rate 1/5

3.2 ส่วนการสลับบิตของเทอร์โบ

เมื่อเริ่มทำงานส่วนสร้างตำแหน่งการสลับบิตข้อมูลเริ่มการคำนวณหาพารามิเตอร์ที่ใช้ในการสร้างตำแหน่งในการสลับบิตข้อมูล ดังแสดงตามตัวอย่างตามภาพที่ 18 แล้วเก็บค่าดังกล่าวไว้ในหน่วยความจำ ซึ่งวิธีการสร้างตำแน่งการสลับบิตมีขั้นตอนดังต่อไปนี้



ภาพที่ 18 ส่วนสร้างตำแหน่งสลับบิตของรหัสเทอร์โบ

1. นำบิตบนจำนวน n บิต (n MSBs) จาก bit counter มาเข้าบล็อก B1 ซึ่งจะเป็นการรับเอาอินพุท (d1) ที่เข้ามาทำการบวกด้วยค่าหนึ่งจากนั้นก็จะเลือกเอาค่าจากบิตล่างจำนวน n บิต (n LSBs) ออกเป็น เอาพุท ซึ่งในกรณีนี้ค่า N_{turbo} มีค่าเท่ากับ 3858 ดังนั้นเราสามารถหาค่า n ได้เท่ากับ 7 จากตารางที่ 2

2. นำบิตล่างจำนวน 5 บิต (5 LSBs) จาก bit counter มาเข้า บล็อก B2 ซึ่งจะเป็นการรับเอาอินพุท (d2) ที่เข้ามาทำการ Lookup Table จากตารางที่ 3 จากนั้นก็จะเลือกข้อมูลจำนวน n บิต ออกเป็น เอาพุท

ตารางที่ 2 ความสัมพันธ์ระหว่างค่า Block size และ Turbo Interleave parameter

Turbo Interleaver Block Size	**Turbo Interleave Parameter**
$\mathbf{N_{turbo}}$	**n**
210	3
378	4
402	4
570	5
762	5
786	5
1,146	6
1,530	6
1,554	6
2,298	7
2,322	7
3,066	7
3,090	7
3,858	7
4,602	8
6,138	8
9,210	9
12,282	9
20,730	10

ตารางที่ 3 การคำนวณค่า Turbo Interleaver Lookup Table Definition

Table Index	Entries: n=7
0	15
1	127
2	89
3	1
4	31
5	15
6	61
7	47
8	127
9	17
10	119
11	15
12	57
13	123
14	95
15	5
16	85
17	17
18	55
19	57
20	15
21	41
22	93
23	87
24	63
25	15
26	13
27	15
28	81
29	57
30	31
31	69

3. นำบิตล่างจำนวน 5 บิต (5 LSBs) จาก bit counter มาเข้า บล็อก B3 ซึ่งจะเป็นการรับเอาอินพุท (d3) ที่เข้ามาทำการกลับบิต ดังภาพที่ 19

ภาพที่ 19 ขั้นตอนการสลับบิต (Bit Reverse)

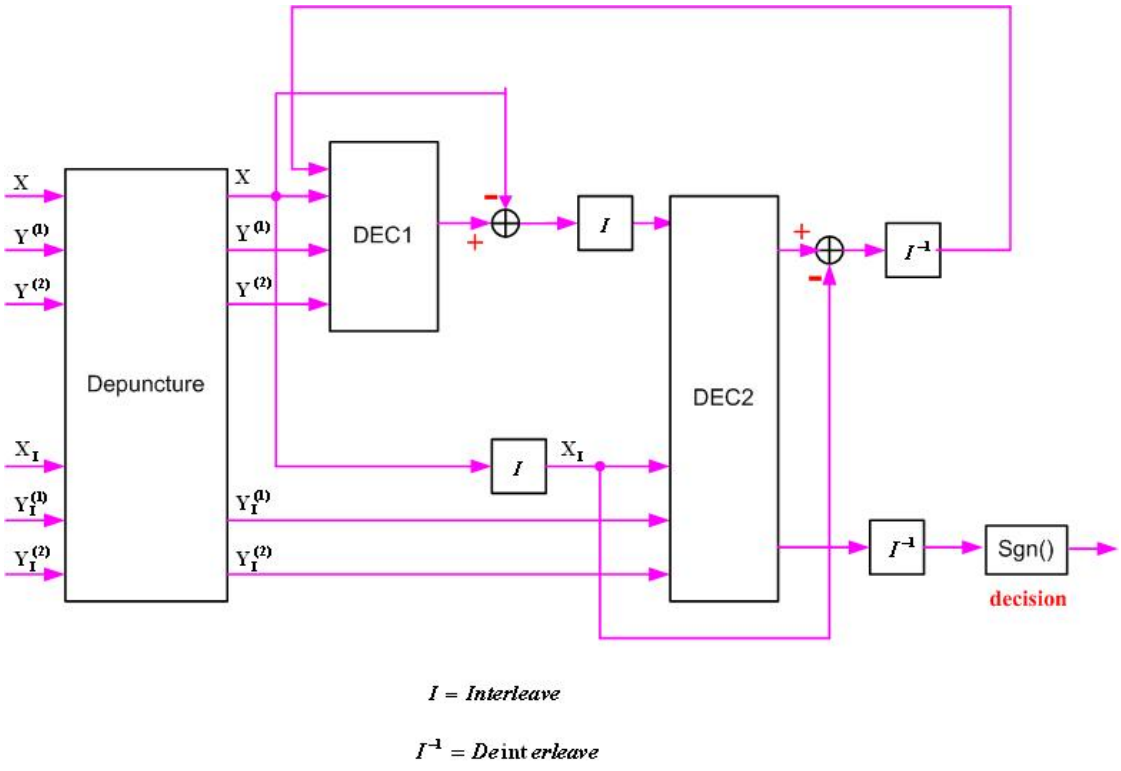
4. นำ เอาพุท (n bits) จาก บล็อก B1 (d4) และ เอาพุท (n bits) จาก บล็อก B2 (d5) มาเข้า บล็อก B4 มาทำการคูณกันจากนั้นก็เลือกบิตล่างจำนวน n บิต (n LSBs) มาออกเป็น เอาพุท (d7)

5. นำ เอาพุทจาก บล็อก B3 และเอาพุทจาก บล็อก B4 มาเข้าบล็อก B5 มาต่อกัน โดยให้ข้อมูลเส้น d6 เป็นบิตบน (MSBs) และข้อมูลเส้น d7 เป็นบิตล่าง(LSBs) จากนั้นมาทำการตรวจสอบว่าค่าที่ได้น้อยกว่าค่า Nturbo หรือไม่ ถ้าปรากฏว่าน้อยกว่าก็จะเก็บค่านั้นไว้โดยจะเก็บค่าดังกล่าวไว้ที่ RAM แต่ถ้ามากกว่าหรือเท่ากับค่านั้นก็จะทิ้งไป

6. ทำการวนรอบทำไปเรื่อย ๆ ตั้งแต่ 0 ถึง $2^{(n+5)}$-1

เมื่อบิตข้อมูลเข้ามาที่บล็อกเข้ารหัสก็จะถูกเก็บไว้ที่หน่วยความจำก่อนเพื่อรอให้ส่วนสร้างตำแหน่งสลับบิตทำงานเสร็จก่อน เมื่อส่วนสร้างตำแหน่งสลับบิตทำงานเสร็จแล้ว ก็จะอ่านข้อมูลจากหน่วยความจำโดยเริ่มจากตำแหน่งแรกไปเรื่อย ๆ เพื่อมาเข้าส่วนเข้ารหัสข้อมูลตัวที่ 1 และ ตัวที่ 2 (โดยที่ก่อนที่จะเข้าส่วนเข้ารหัสตัวที่ 2 ต้องผ่านบล็อก Interleave เพื่อทำการสลับบิตข้อมูลก่อน) ซึ่งจะได้ เอาพุทออกจากตัวเข้ารหัสทั้งหมด 6 เส้น ซึ่งได้แก่ X, Y(1), Y(2), XI, YI(1), YI(2) เมื่ออ่านข้อมูลจากหน่วยความจำจนถึงตำแหน่งสุดท้ายเสร็จสิ้นวงจรเข้ารหัสข้อมูลจะทำการวนรอบการทำงานอีกทั้งหมด 3 ครั้งเพื่อทำให้ค่าที่อยู่ในรีจิสเตอร์ เป็น 0 ทั้งหมด ดังนั้นข้อมูลแต่ละเส้นจะมีความยาวเท่ากับ 2861 บิต (2858+3) ซึ่งข้อมูลทั้ง 6 เส้นจะถูกส่งไปยังวงจรพังค์เชอร์ ต่อไป

4. <u>การทำงานของซอฟต์แวร์ในส่วนของวงจรการถอดรหัส</u>



<u>ภาพที่ 20</u> ขั้นตอนการทำงานของวงจรถอดรหัสของเทอร์โบ

ข้อมูลที่เข้ามาในบล็อกถอดรหัสจะถูกทำ ดีพังค์เชอร์ แล้วเก็บค่าไว้ในหน่วยความจำ RAM ทั้ง 5 ตัว จากนั้นทำการเริ่มถอดรหัสข้อมูลโดยจะอ่านข้อมูลจาก RAM มาเข้าตัวถอดรหัสตัวที่ 1 (DEC1) ซึ่งจะมีการคำนวณค่าต่าง ๆ ต่อไปนี้

1. ค่าความน่าจะเป็นแกมม่า (γ-Probability) หมายถึง ค่าความน่าจะเป็นที่การเปลี่ยนจากโหนดก่อนหน้าไปยังอีกโหนดหนึ่งเมื่อมีข้อมูลอินพุตเข้ามา

$$\gamma_k(s',s) = P(\underline{y}_k \mid \underline{x}_k) \cdot P(u_k)$$

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	L(uk)	0	0	0	0	0	0	0	0	0	0	0	
2	xk	3	-3	3	-3	3	-3	3	-3	3	-3	3	
3	yk	3	3	-3	-3	-3	-3	3	-3	-3	3	3	
4	yk2	3	3	3	-3	-3	3	-3	3	3	3	-3	
5													
6	g00	-18	-6	-6	18	6	6	-6	6	-6	-6	-6	
7	g14	-18	-6	-6	18	6	6	-6	6	-6	-6	-6	
8	g25	-6	6	-18	6	-6	-6	6	-6	-18	6	6	
9	g31	-6	6	-18	6	-6	-6	6	-6	-18	6	6	
10	g42	6	18	-6	-6	-18	6	-6	6	-6	18	-6	
11	g56	6	18	-6	-6	-18	6	-6	6	-6	18	-6	
12	g67	-6	6	6	6	-6	18	-18	18	6	6	-18	
13	g73	-6	6	6	6	-6	18	-18	18	6	6	-18	
14	g04	18	6	6	-18	-6	-6	6	-6	6	6	6	
15	g10	18	6	6	-18	-6	-6	6	-6	6	6	6	
16	g21	6	-6	18	-6	6	6	-6	6	18	-6	-6	
17	g35	6	-6	18	-6	6	6	-6	6	18	-6	-6	
18	g77	6	-6	-6	-6	6	-18	18	-18	-6	-6	18	
19	g63	6	-6	-6	-6	6	-18	18	-18	-6	-6	18	
20	g52	-6	-18	6	6	18	-6	6	-6	6	-18	6	
21	g46	-6	-18	6	6	18	-6	6	-6	6	-18	6	
22													

ภาพที่ 21 ค่าความน่าจะเป็นแกมม่า (γ-Probability) ของตัวถอดรหัสตัวที่ 1

2. ค่าความน่าจะเป็นแอลฟ่า (α-Probability) หมายถึง ค่าความน่าจะเป็นที่โหนดนั้นๆ มีโอกาสปรากฏขึ้นเมื่อทราบว่าโหนดก่อนหน้าปรากฏ มีสมการในการคำนวณดังนี้

$$\alpha_k(s) = \sum_{all\ s'} \alpha_{k-1}(s') \cdot \gamma_k(s', s)$$

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	L(uk)	0	0	0	0	0	0	0	0	0	0	0	
2	xk	3	-3	3	-3	3	-3	3	-3	3	-3	3	
3	yk	3	3	-3	-3	-3	-3	3	-3	-3	3	3	
4	yk2	3	3	3	-3	-3	3	-3	3	3	3	-3	
5													
6	A0	0	-18	-24	-30	36	42	48	66	12	30	72	
7	A1	-100	-94	-88	54	0	30	60	78	24	66	60	
8	A2	-100	-94	36	-18	24	54	36	54	0	42	84	
9	A3	-100	-94	-88	-6	12	18	72	90	84	54	48	
10	A4	-100	18	-12	-18	72	30	36	54	24	18	60	
11	A5	-100	-94	-88	18	-12	18	48	66	36	102	48	
12	A6	-100	-94	0	-6	12	90	24	42	12	30	120	
13	A7	-100	-94	-88	6	0	6	108	126	48	42	36	
14													

ภาพที่ 22 ค่าความน่าจะเป็นแอลฟ่า (α-Probability)ของตัวถอดรหัสตัวที่ 1

3. ค่าความน่าจะเป็นเบต้า (β-Probability) หมายถึง ค่าความน่าจะเป็นที่โหนดนั้นๆ มีโอกาสปรากฏขึ้นเมื่อทราบว่าโหนดหลังจากนี้ปรากฏ มีสมการในการคำนวณดังนี้

$$\beta_{k-1}(s') = \sum_{all\ s} \beta_k(s) \cdot \gamma_k(s', s)$$

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	L(uk)	0	0	0	0	0	0	0	0	0	0	0	
2	xk	3	-3	3	-3	3	-3	3	-3	3	-3	3	
3	yk	3	3	-3	-3	-3	-3	3	-3	-3	3	3	
4	yk2	3	3	3	-3	-3	3	-3	3	3	3	-3	
5													
6	B0		13	19	85	19	13	67	49	43	1	43	
7	B1		25	31	121	31	25	55	37	31	-11	55	
8	B2		25	79	73	31	1	55	37	55	13	31	
9	B3		37	43	61	43	13	43	25	91	1	43	
10	B4		97	7	49	103	25	79	61	31	37	55	
11	B5		61	19	37	67	37	67	49	19	73	67	
12	B6		37	-5	61	43	85	91	73	19	25	115	
13	B7		49	7	49	55	49	127	109	7	13	79	
14													

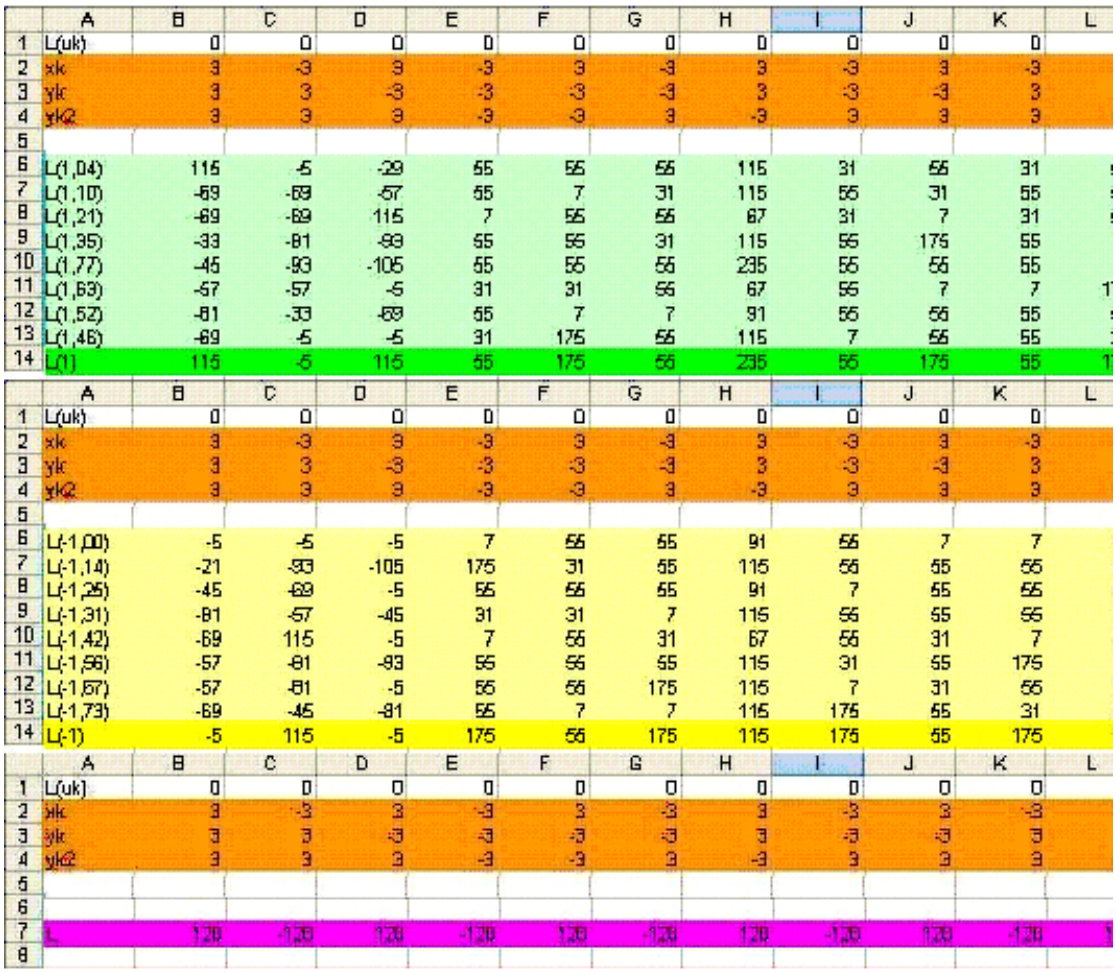
ภาพที่ 23 ค่าความน่าจะเป็นเบต้า (β-Probability) ของตัวถอดรหัสตัวที่ 1

4. ค่าอัตราส่วนความคล้ายคลึงเชิงล็อการิทึม (Log-likelihood Ratio) ในตำแหน่งต่างๆ โดยสมการของค่าอัตราส่วนความคล้ายคลึงเชิงล็อการิทึม คือ

$$L(u_k | \underline{y}) = \log\left(\frac{\sum_{u_k=+1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}{\sum_{u_k=-1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}\right)$$

5. นำค่า LLR ที่คำนวณดังกล่าวได้เก็บไว้ใน RAM

6. หาค่า Extrinsic (Lef) โดยมีสมการคือ Lef = LLR-Xk

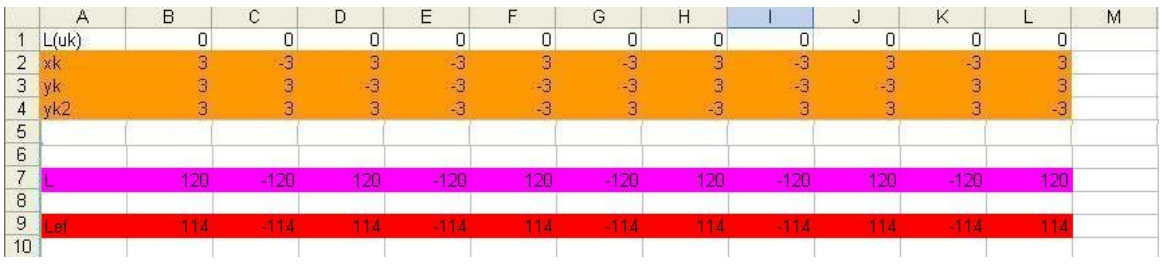


	A	B	C	D	E	F	G	H	I	J	K	L
1	L(uk)	0	0	0	0	0	0	0	0	0	0	
2	xk	3	-3	3	-3	3	-3	3	-3	3	-3	
3	yk	3	3	-3	-3	-3	-3	3	-3	-3	3	
4	yk2	3	3	3	-3	-3	3	-3	3	3	3	
5												
6	L(1,04)	115	-5	-29	55	55	55	115	31	55	31	
7	L(1,10)	-69	-69	-57	55	7	31	115	55	31	55	
8	L(1,21)	-69	-69	115	7	55	55	67	31	7	31	
9	L(1,35)	-33	-81	-69	55	55	31	115	55	175	55	
10	L(1,77)	-45	-93	-105	55	55	55	235	55	55	55	
11	L(1,63)	-57	-57	-5	31	31	55	67	55	7	7	1
12	L(1,52)	-81	-33	-69	55	7	7	91	55	55	55	
13	L(1,46)	-69	-5	-5	31	175	55	115	7	55	55	
14	L(1)	115	-5	115	55	175	55	235	55	175	55	1

	A	B	C	D	E	F	G	H	I	J	K	L
1	L(uk)	0	0	0	0	0	0	0	0	0	0	
2	xk	3	-3	3	-3	3	-3	3	-3	3	-3	
3	yk	3	3	-3	-3	-3	-3	3	-3	-3	3	
4	yk2	3	3	3	-3	-3	3	-3	3	3	3	
5												
6	L(-1,00)	-5	-5	-5	7	55	55	91	55	7	7	
7	L(-1,14)	-21	-93	-105	175	31	55	115	55	55	55	
8	L(-1,25)	-45	-69	-5	55	55	55	91	7	55	55	
9	L(-1,31)	-81	-57	-45	31	31	7	115	55	55	55	
10	L(-1,42)	-69	115	-5	7	55	31	67	55	31	7	
11	L(-1,56)	-57	-81	-93	55	55	55	115	31	55	175	
12	L(-1,67)	-57	-81	-5	55	55	175	115	7	31	55	
13	L(-1,73)	-69	-45	-81	55	7	7	115	175	55	31	
14	L(-1)	-5	115	-5	175	55	175	115	175	55	175	

	A	B	C	D	E	F	G	H	I	J	K	L
1	L(uk)	0	0	0	0	0	0	0	0	0	0	
2	xk	3	-3	3	-3	3	-3	3	-3	3	-3	
3	yk	3	3	-3	-3	-3	-3	3	-3	-3	3	
4	yk2	3	3	3	-3	-3	3	-3	3	3	3	
5												
6												
7	L	120	-120	120	-120	120	-120	120	-120	120	-120	1
8												

ภาพที่ 24 ค่าอัตราส่วนความคล้ายคลึงเชิงล็อการิทึม (LLR) ที่คำนวณได้ของตัวถอดรหัสตัวที่ 1



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	L(uk)	0	0	0	0	0	0	0	0	0	0	0	
2	xk	3	-3	3	-3	3	-3	3	-3	3	-3	3	
3	yk	3	3	-3	-3	-3	-3	3	-3	-3	3	3	
4	yk2	3	3	3	-3	-3	3	-3	3	3	3	-3	
5													
6													
7	L	120	-120	120	-120	120	-120	120	-120	120	-120	120	
8													
9	Lef	114	-114	114	-114	114	-114	114	-114	114	-114	114	
10													

ภาพที่ 25 ค่า Lef ที่คำนวณได้ของตัวถอดรหัสตัวที่ 1

7. เขียนค่า Lef ลงใน RAM

8. อ่านค่า Lef และ X_k จาก RAM เพื่อนำข้อมูลเข้าตัว Interleave ทั้ง 2 เส้น

จากนั้นทำการอ่านข้อมูลจาก RAM เพื่อนำข้อมูลเข้าตัวถอดรหัสตัวที่ 2 ซึ่งก็มีขั้นตอนการทำงานเช่นเดียวกันกับตัวถอดรหัสตัวที่ 1 คือ

1. ค่าความน่าจะเป็นแกมม่า (γ-Probability) หมายถึง ค่าความน่าจะเป็นที่การเปลี่ยนจากโหนดก่อนหน้าไปยังอีกโหนดหนึ่งเมื่อมีข้อมูลอินพุตเข้ามา

$$\gamma_k(s', s) = P(\underline{y}_k \mid \underline{x}_k) \cdot P(u_k)$$

85											
86	L(uk)	-20	20	20	20	20	20	-20	-20	-20	-20	20
87	2xk	-3	3	3	3	3	3	-3	-3	-3	-3	3
88	2yk	-3	3	-3	3	-3	-3	3	3	-3	-3	-3
89	2yk2	-3	3	-3	-3	3	-3	-3	3	3	3	3
90												
91												
92	g00	38	-38	-14	-26	-26	-14	26	14	26	26	-26
93	g14	38	-38	-14	-26	-26	-14	26	14	26	26	-26
94	g25	26	-26	-26	-14	-38	-26	38	26	14	14	-38
95	g31	26	-26	-26	-14	-38	-26	38	26	14	14	-38
96	g42	14	-14	-38	-26	-26	-38	26	38	26	26	-26
97	g56	14	-14	-38	-26	-26	-38	26	38	26	26	-26
98	g67	26	-26	-26	-38	-14	-26	14	26	38	38	-14
99	g73	26	-26	-26	-38	-14	-26	14	26	38	38	-14
100	g04	-38	38	14	26	26	14	-26	-14	-26	-26	26
101	g10	-38	38	14	26	26	14	-26	-14	-26	-26	26
102	g21	-26	26	26	14	38	26	-38	-26	-14	-14	38
103	g35	-26	26	26	14	38	26	-38	-26	-14	-14	38
104	g77	-26	26	26	38	14	26	-14	-26	-38	-38	14
105	g63	-26	26	26	38	14	26	-14	-26	-38	-38	14
106	g52	-14	14	38	26	26	38	-26	-38	-26	-26	26
107	g46	-14	14	38	26	26	38	-26	-38	-26	-26	26
108												

ภาพที่ 26 ค่าความน่าจะเป็นแกมม่า (γ-Probability) ของตัวถอดรหัสตัวที่ 2

2. ค่าความน่าจะเป็นแอลฟ่า (α-Probability) หมายถึง ค่าความน่าจะเป็นที่โหนดนั้นๆ มีโอกาสปรากฏขึ้นเมื่อทราบว่าโหนดก่อนหน้าปรากฏ มีสมการในการคำนวณดังนี้

$$\alpha_k(s) = \sum_{all\ s'} \alpha_{k-1}(s') \cdot \gamma_k(s', s)$$

85												
86	L(uk)	-20	20	20	20	20	20	-20	-20	-20	-20	20
87	2xk	-3	3	3	3	3	3	-3	-3	-3	-3	3
88	2yk	-3	3	-3	3	-3	-3	3	3	-3	-3	-3
89	2yk2	-3	3	-3	-3	3	-3	-3	3	3	3	3
90												
91												
92	A0	0	38	0	-74	-60	-42	8	-26	-64	-38	-72
93	A1	-100	-74	-60	-86	-8	-6	-44	10	-76	-26	-20
94	A2	-100	-86	-52	-22	-68	-70	108	-54	-12	-10	-84
95	A3	-100	-74	-60	-58	92	-58	-56	-42	-48	26	80
96	A4	-100	-38	76	-46	-48	-94	-20	-78	-36	-50	-60
97	A5	-100	-74	-48	-94	-36	70	-32	86	-88	2	-48
98	A6	-100	-86	-24	54	-20	-82	32	-66	64	-62	-32
99	A7	-100	-74	-48	-82	16	-30	-4	-14	-100	102	4
100												

ภาพที่ 27 ค่าความน่าจะเป็นแอลฟ่า (α-Probability)ของตัวถอดรหัสตัวที่ 2

3. ค่าความน่าจะเป็นเบต้า (β-Probability) หมายถึง ค่าความน่าจะเป็นที่โหนดนั้นๆ มีโอกาสปรากฏขึ้นเมื่อทราบว่าโหนดหลังจากนี้ปรากฏ มีสมการในการคำนวณดังนี้

$$\beta_{k-1}(s') = \sum_{all\ s} \beta_k(s) \cdot \gamma_k(s', s)$$

85												
86	L(uk)	-20	20	20	20	20	20	-20	-20	-20	-20	20
87	2xk	-3	3	3	3	3	3	-3	-3	-3	-3	3
88	2yk	-3	3	-3	3	-3	-3	3	3	-3	-3	-3
89	2yk2	-3	3	-3	-3	3	-3	-3	3	3	3	3
90												
91												
92	B0		59	-83	-9	-23	-49	-31	-65	-19	-45	-11
93	B1		-17	-55	3	-75	-77	21	-93	-7	-57	-63
94	B2		-81	-31	-61	-11	-13	109	-29	-71	-81	1
95	B3		-29	-19	-33	65	-25	33	-41	-43	-109	77
96	B4		-105	81	-37	-35	11	-3	-5	-47	-33	-23
97	B5		-77	5	15	-47	87	9	71	5	-85	-35
98	B6		-53	-59	103	-71	-1	-55	-17	93	-21	-59
99	B7		-41	-7	27	-99	-53	-27	-69	17	55	-87
100												

ภาพที่ 28 ค่าความน่าจะเป็นเบต้า (β-Probability) ของตัวถอดรหัสตัวที่ 2

4. ค่าอัตราส่วนความคล้ายคลึงเชิงล็อการิทึม (Log-likelihood Ratio) ในตำแหน่งต่างๆ โดยสมการของค่าอัตราส่วนความคล้ายคลึงเชิงล็อการิทึม คือ

$$L(u_k \mid \underline{y}) = \log\left(\frac{\sum_{u_k=+1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}{\sum_{u_k=-1} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}\right)$$

L(-1,00)	97	-83	-23	-123	-135	-87	-31	-31	-83	-23	-135
L(-1,14)	-167	-31	-111	-147	-23	-23	-23	-23	-83	-23	-23
L(-1,25)	-151	-107	-63	-83	-19	-87	217	-23	-83	-31	-23
L(-1,31)	-91	-155	-83	-147	-23	-63	-111	-23	-91	-23	-23
L(-1,42)	-167	-83	-23	-83	-87	-23	-23	-111	-91	-23	-87
L(-1,56)	-139	-147	17	-191	-63	-23	-23	217	-83	-31	-63
L(-1,67)	-115	-119	-23	-83	-87	-135	-23	-23	157	-111	-87
L(-1,73)	-103	-119	-107	-55	-23	-23	-31	-31	-171	217	-23
L(-1)	97	-31	17	-55	-19	-23	217	217	157	217	-23
L(1,04)	-143	157	-23	-83	-23	-31	-23	-87	-123	-87	-23
L(1,10)	-79	-119	-55	-83	-31	-23	-135	-23	-147	-63	-31
L(1,21)	-143	-115	-23	-83	-107	-23	-23	-87	-83	-87	-111
L(1,35)	-203	-43	-19	-91	217	-23	-23	-63	-147	-23	217
L(1,77)	-167	-55	5	-143	-23	-31	-87	-23	-83	-23	-23
L(1,63)	-155	-79	-31	157	-31	-23	-23	-135	-83	-23	-31
L(1,52)	-195	-91	-71	-79	-23	217	-87	-23	-195	-23	-23
L(1,46)	-167	-83	217	-91	-23	-111	-63	-23	-83	-135	-23
L(1)	-79	157	217	157	217	217	-23	-23	-83	-23	217
L	-176	188	200	212	236	240	-240	-240	-240	-240	240

ภาพที่ 29 ค่าอัตราส่วนความคล้ายคลึงเชิงล็อการิทึม (LLR) ที่คำนวณได้ของตัวถอดรหัสตัวที่ 2

5. นำค่า LLR ที่คำนวณดังกล่าวได้เก็บไว้ใน RAM

6. หาค่า Extrinsic (Lef) โดยมีสมการคือ Lef = LLR-X_k

85											
86	L(uk)	-20	20	20	20	20	20	-20	-20	-20	-20	20
87	2xk	-3	3	3	3	3	3	-3	-3	-3	-3	3
88	2yk	-3	3	-3	3	-3	-3	3	3	-3	-3	-3
89	2yk2	-3	3	-3	-3	3	-3	-3	3	3	3	3
90												
91												
92	Lef	-170	182	194	206	230	234	-234	-234	-234	-234	234
93												

ภาพที่ 30 ค่า Lef ที่คำนวณได้ของตัวถอดรหัสตัวที่ 2

7. เขียนค่า Lef ลงใน RAM

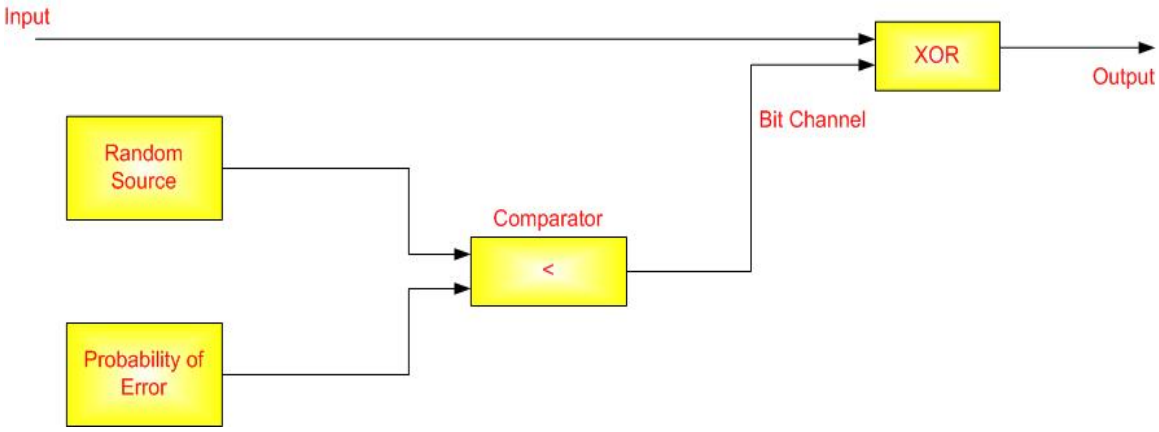
8. อ่านค่า Lef และ X_k จาก RAM เพื่อนำข้อมูลเข้าตัว DeInterleave

จากนั้นนำค่า Lef ที่ได้วนกลับไปป้อนให้ตัวถอดรหัสตัวที่ 1 อีกครั้ง ทำเช่นนี้ไปเรื่อยจนครบจำนวนรอบตามค่าที่เราตั้งไว้ (ประมาณ 4 รอบ) ยิ่งทำหลาย ๆ รอบก็จะทำให้ความถูกต้องในการถอดรหัสมีมากยิ่งขึ้นแต่ก็จะทำให้วงจรถอดรหัสนั้นมีความซับซ้อนมากยิ่งขึ้นด้วย เมื่อครบจำนวนรอบแล้วก็จะส่งค่าที่ได้ไปเข้าส่วนที่ใช้ทำการตัดสินใจ (decision) ว่าควรจะถอดรหัสได้บิต 0 หรือ บิต 1 ซึ่งถือว่าเป็น เอาพุทของส่วนถอดรหัสข้อมูลของรหัสเทอร์โบ

5. การจำลองช่องสัญญาณ

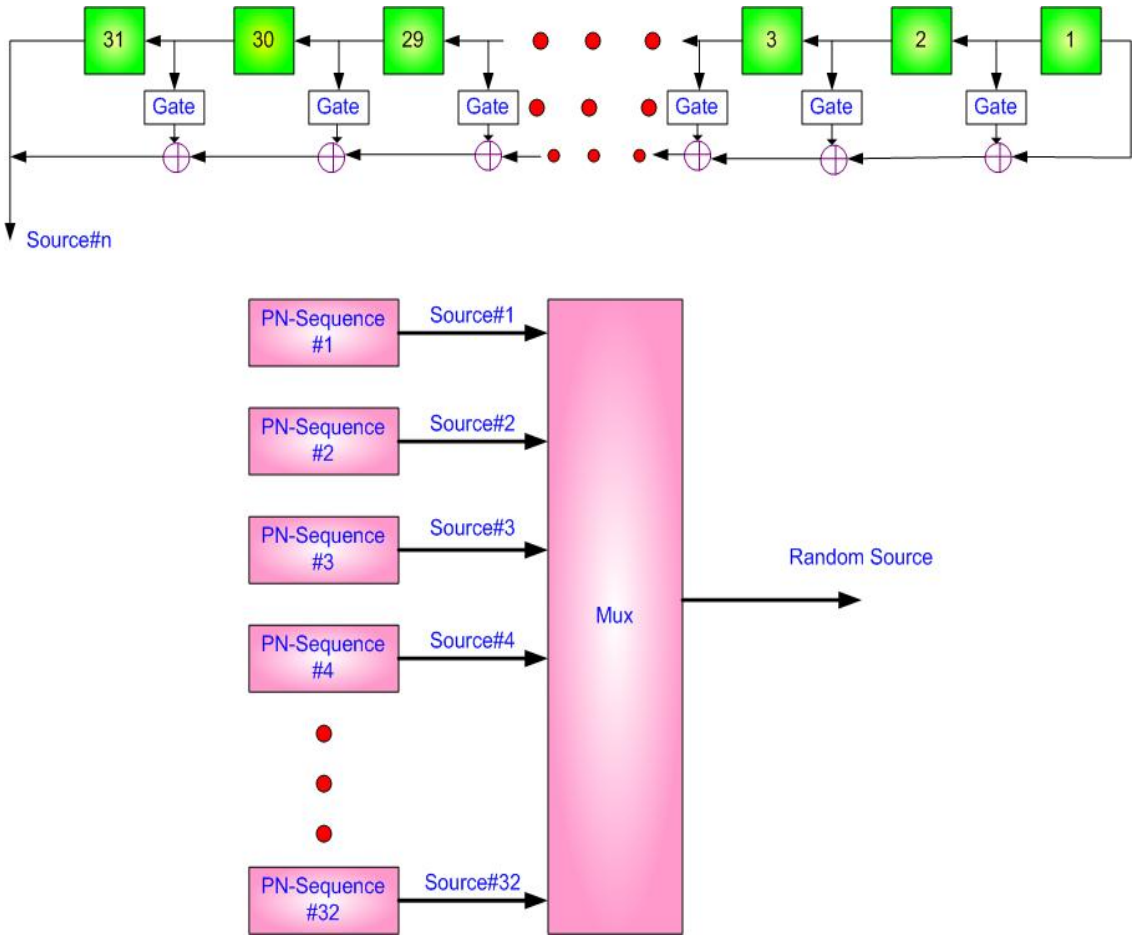
เนื่องจากระบบที่ได้ทำการสร้างขึ้นมานั้นเป็นการส่งข้อมูลที่เรียกว่าเป็นพวก ฮาร์ดบิต (hard bit) คือบิต “0” หรือ “1” เท่านั้น ดังนั้นในการจำลองช่องสัญญาณขึ้นมานั้นก็จะใช้ช่องสัญญาณประเภท BSC (Binary Symmetric Channel) ซึ่งเหมาะสมกับการส่งข้อมูลประเภทนี้ โดยจะมีโครงสร้างเป็นไปตามภาพที่ 31

โดยข้อมูลที่ใช้เป็นอินพุทจะอยู่ในส่วนของเอาพุทที่ออกจาก Interleaver และจะทำการ XOR กับบิต “0” หรือ “1” ซึ่งได้มาจากการเปรียบเทียบของ Random Source กับ Probability of Error



ภาพที่ 31 โครงสร้างของช่องสัญญาณแบบ BSC

การสร้าง Random Source จะใช้การสร้าง PN-Sequence 32 สายมาต่อเข้าด้วยกัน (เพื่อให้ค่าที่ได้มีความละเอียดและสามารถตัดสินใจได้ถูกต้องมากขึ้น) ตามภาพที่ 32 โดย PN-Sequence แต่ละเส้นจะสร้างจาก Primitive Polynomial Over GF(2) ค่า N = 31 และ M = 10 เป็นไปตามตารางที่ 4 และค่าเริ่มต้นของแต่ละเส้นก็จะ Random ขึ้นมา เพื่อให้ข้อมูลเกิดการกระจายตัวกันให้มากที่สุด



ภาพที่ 32 โครงสร้างของ Random Source

โดยข้อมูลที่ได้มาจาก Random Source จะคล้าย ๆ กับการจำลองพื้นที่ใต้กราฟ ของการกระจายแบบ Gaussian โดยค่า Probability of Error จะเป็นตัวตัดสินใจในการบ่งบอกถึงการ Error ที่จะเกิดขึ้นได้ในระบบ

ตารางที่ 4 ค่า Primitive Polynomials สำหรับการสร้าง PN-Sequence

PN-Sequence # n	Polynomials
1	$x^{31} + x^{3} + 1$
2	$x^{31} + x^{3} + x^{2} + x^{1} + 1$
3	$x^{31} + x^{13} + x^{8} + x^{3} + 1$
4	$x^{31} + x^{27} + x^{23} + x^{19} + x^{15} + x^{11} + x^{7} + x^{3} + 1$
5	$x^{31} + x^{21} + x^{12} + x^{3} + x^{2} + x^{1} + 1$
6	$x^{31} + x^{20} + x^{18} + x^{7} + x^{5} + x^{3} + 1$
7	$x^{31} + x^{20} + x^{15} + x^{5} + x^{4} + x^{3} + 1$
8	$x^{31} + x^{25} + x^{19} + x^{14} + x^{7} + x^{3} + x^{2} + x^{1} + 1$
9	$x^{31} + x^{16} + x^{8} + x^{4} + x^{3} + x^{2} + 1$
10	$x^{31} + x^{23} + x^{22} + x^{15} + x^{14} + x^{7} + x^{4} + x^{3} + 1$
11	$x^{31} + x^{26} + x^{19} + x^{9} + x^{7} + x^{6} + x^{3} + x^{2} + 1$
12	$x^{31} + x^{15} + x^{13} + x^{12} + x^{7} + x^{5} + x^{4} + x^{3} + 1$
13	$x^{31} + x^{15} + x^{14} + x^{9} + x^{8} + x^{6} + x^{2} + x^{1} + 1$
14	$x^{31} + x^{23} + x^{17} + x^{15} + x^{6} + x^{5} + x^{3} + x^{1} + 1$
15	$x^{31} + x^{29} + x^{25} + x^{17} + x^{16} + x^{13} + x^{9} + x^{3} + 1$
16	$x^{31} + x^{30} + x^{29} + x^{25} + x^{23} + x^{16} + x^{6} + x^{4} + 1$
17	$x^{31} + x^{25} + x^{24} + x^{21} + x^{15} + x^{13} + x^{8} + x^{5} + 1$
18	$x^{31} + x^{19} + x^{18} + x^{15} + x^{11} + x^{2} + 1$
19	$x^{31} + x^{29} + x^{26} + x^{20} + x^{18} + x^{15} + x^{6} + x^{1} + 1$
20	$x^{31} + x^{14} + x^{12} + x^{11} + x^{9} + x^{8} + x^{3} + x^{2} + 1$
21	$x^{31} + x^{29} + x^{28} + x^{24} + x^{21} + x^{16} + x^{11} + x^{3} + 1$
22	$x^{31} + x^{18} + x^{17} + x^{11} + x^{8} + x^{6} + x^{4} + x^{2} + 1$
23	$x^{31} + x^{30} + x^{26} + x^{24} + x^{22} + x^{19} + x^{14} + x^{7} + 1$
24	$x^{31} + x^{27} + x^{26} + x^{25} + x^{24} + x^{21} + x^{20} + x^{10} + 1$
25	$x^{31} + x^{27} + x^{22} + x^{11} + x^{9} + x^{8} + x^{6} + x^{5} + 1$
26	$x^{31} + x^{30} + x^{28} + x^{25} + x^{22} + x^{14} + x^{13} + x^{4} + 1$
27	$x^{31} + x^{19} + x^{15} + x^{11} + x^{9} + x^{8} + x^{7} + x^{1} + 1$
28	$x^{31} + x^{30} + x^{28} + x^{19} + x^{18} + x^{13} + x^{9} + x^{8} + 1$
29	$x^{31} + x^{28} + x^{25} + x^{20} + x^{18} + x^{12} + x^{10} + x^{5} + 1$
30	$x^{31} + x^{27} + x^{23} + x^{19} + x^{15} + x^{12} + x^{7} + x^{4} + 1$
31	$x^{31} + x^{27} + x^{20} + x^{18} + x^{17} + x^{16} + x^{9} + x^{4} + 1$
32	$x^{31} + x^{25} + x^{24} + x^{23} + x^{21} + x^{17} + x^{11} + x^{8} + 1$

ผลและวิจารณ์

แนวทางการทดสอบการทำงานของส่วนเข้ารหัสและถอดรหัสของเทอร์โบนั้นของวงจรที่ได้พัฒนาขึ้นนั้นเป็นดังนี้

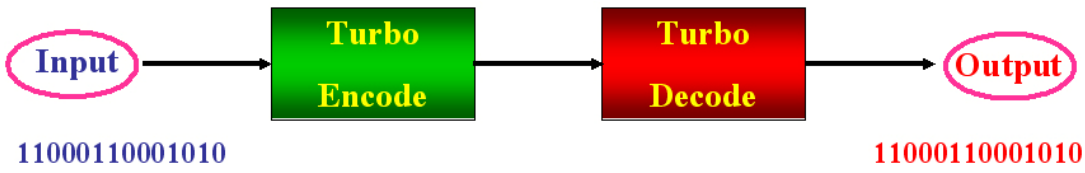
1. ทดสอบด้วยวิธีการ simulation โดยใช้โปรแกรม ที่มีชื่อว่า Modelsim V6.0

1.1 ทดสอบฟังก์ชันการทำงานของโมดูลย่อย

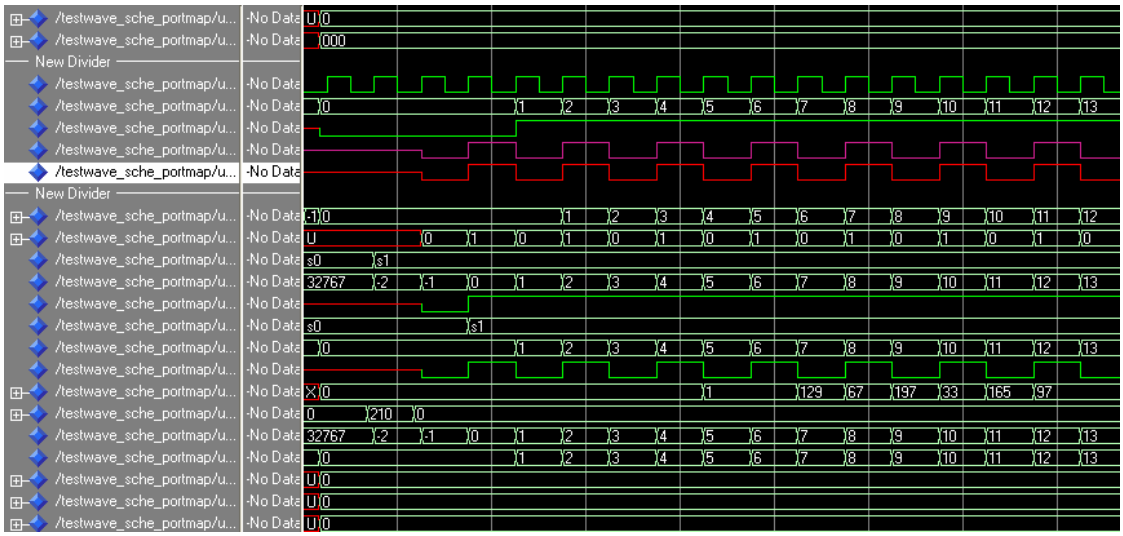
เป็นการทดสอบการทำงานเบื้องต้น โดยเป็นการทดสอบฟังก์ชันการทำงานของแต่ละโมดูลทั้งที่อยู่ในส่วนการเข้ารหัสและการถอดรหัส เช่น วงจร RSC, interleaver, MAP Decoder เป็นต้น

1.2 ทดสอบฟังก์ชันการทำงานของวงจรเข้ารหัสและถอดรหัสเทอร์โบ

ทำการรวมโมดูลย่อย ๆ ของส่วนเข้ารหัสและส่วนถอดรหัส จากนั้นทำการ simulation โดยนำเอาวงจรเข้ารหัสและวงจรถอดรหัสมาเชื่อมต่อกันโดยตรง จากนั้นทำการการป้อนอินพุตให้กับวงจรเข้ารหัสเพื่อตรวจสอบว่าวงจรเข้ารหัสและวงจรถอดรหัสที่ได้พัฒนาขึ้นทำงานถูกต้องหรือไม่



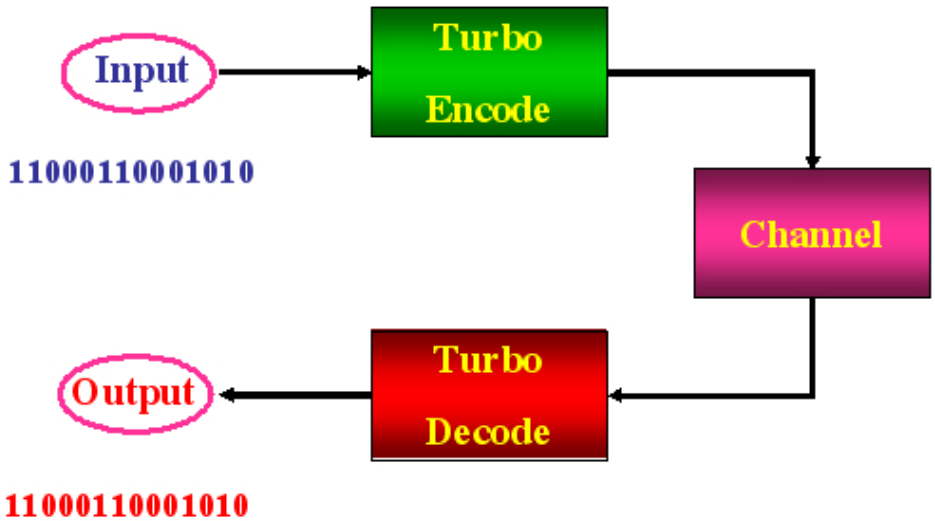
ภาพที่ 33 การเชื่อมต่อบล็อกตัวเข้ารหัสและบล็อกตัวถอดรหัสของเทอร์โบ



ภาพที่ 34 ผลการ Simulation บนโปรแกรม Model Sim

1.3 ทดสอบฟังก์ชันการทำงานของวงจรถอดรหัสเทอร์โบเมื่อผ่านช่องสัญญาณจำลอง

ทำการทดสอบการทำงานของโมดูลช่องสัญญาณจำลอง BSC ก่อนว่าทำงานถูกหรือไม่ โดยการ Simulation เมื่อโมดูลดังกล่าวทำงานถูกต้องแล้ว จากนั้นทำการเชื่อมต่อโมดูลช่องสัญญาณจำลองเข้าไปในระบบตามภาพที่ 35 เพื่อดูสัญญาณข้อมูลที่ได้จากการถอดรหัสต่อไป



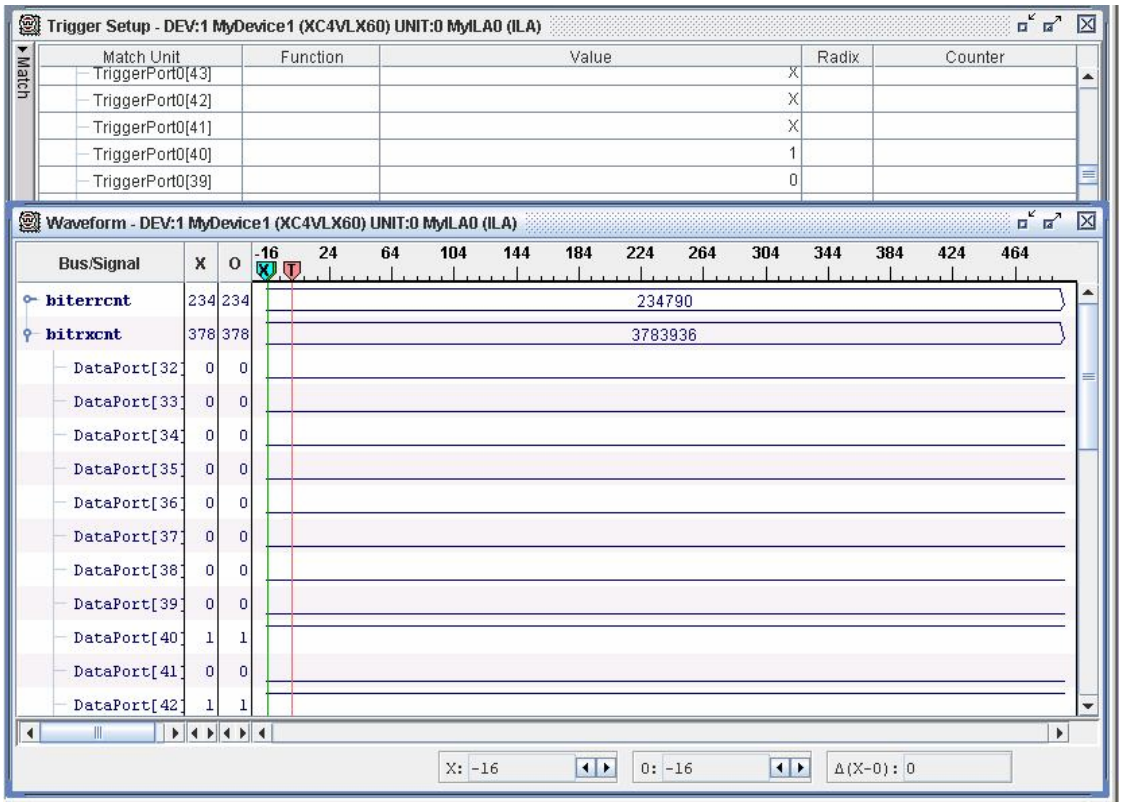
ภาพที่ 35 การทดสอบโดยผ่านช่องสัญญาณจำลองที่สร้างขึ้น

2. <u>ทดสอบการทำงานของฮาร์ดแวร์โดยใช้โปรแกรม ChipScope Pro 7.1</u>

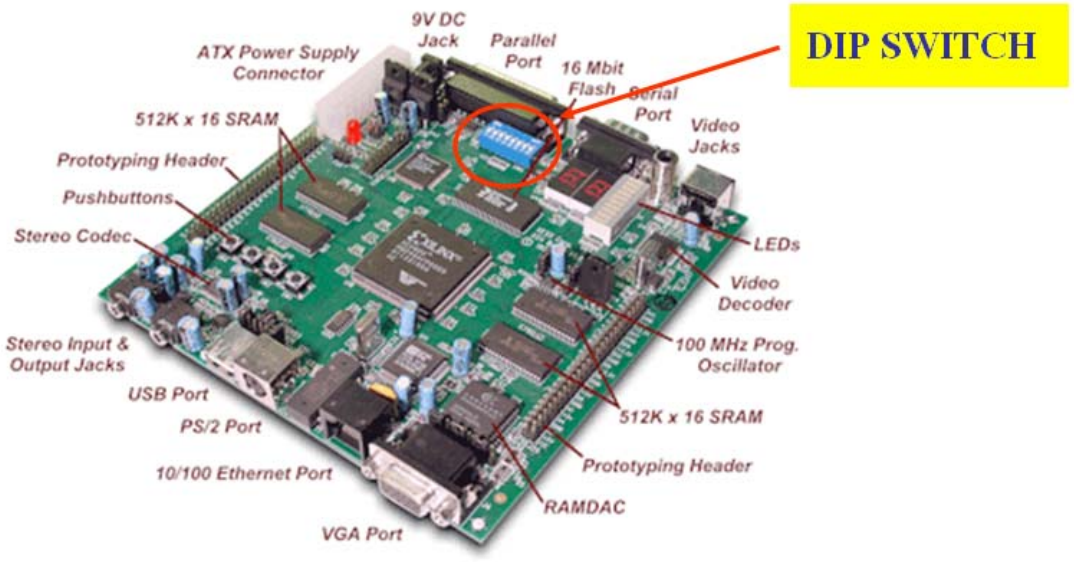
นำโมดูลของวงจรเข้ารหัสและถอดรหัสของเทอร์โบที่ผ่านการตรวจสอบความถูกต้องในการทำงานจากการ Simulation ด้วยโปรแกรม Modelsim V6.0 มาประมวลผลในฮาร์ดแวร์ ซึ่งเราจะสามารถดูค่าสัญญาณต่าง ๆ ได้โดยใช้โปรแกรม ChipScope Pro 7.1 ซึ่งสามารถดึงสัญญาณจากฮาร์ดแวร์ มาตรวจสอบเพื่อตรวจสอบความถูกต้องได้

3. <u>การทดสอบและการคำนวณหาค่า Bit Error Rate (BER)</u>

ทำการสร้างโมดูลสำหรับนำจำนวนบิตข้อมูลที่ได้ทำการถอดรหัสและจำนวนบิตข้อมูลที่ถอดรหัสแล้วนำค่าทั้ง 2 มาคำนวณหาค่า BER ด้วยสมการคือ BER = Bit Error / Bit Receive ดังภาพที่ 36 จากนั้นทำการปรับค่า Probability of error ที่ช่องสัญญาณ BSC (Binary Symmetric Channel) ที่ Dip Switch ของบอร์ด FPGA ดังภาพที่ 37 แล้วทำการบันทึกค่า BER ดังตารางที่ 5



<u>ภาพที่ 36</u> ค่าจำนวนบิตที่รับได้และจำนวนบิตที่ถอดรหัสผิดบนโปรแกรม Chip Scope



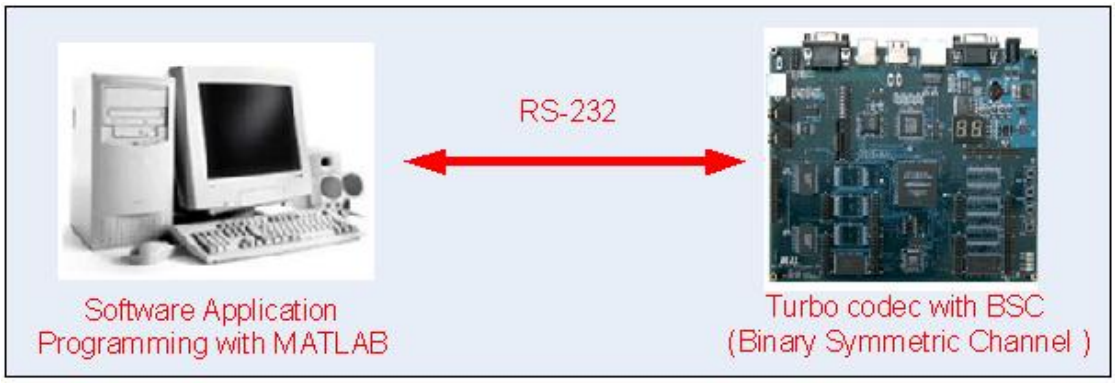
ภาพที่ 37 ตำแหน่งของ Dip Switch บนบอร์ด FPGA

ตารางที่ 5 แสดงค่าความสัมพันธ์ระหว่าง Probability of error กับค่า BER

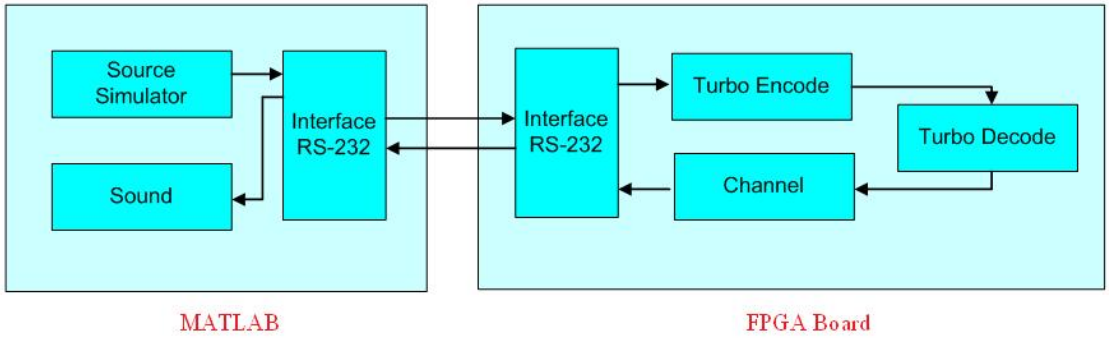
Probability of error	Bit Error (bit)	Bit Receive (bit)	BER
0.01359	2	17755392	1.13 * 10-7
0.01584	6	19501824	3.07* 10-7
0.01847	9	21830400	4.12* 10-7
0.02928	16	19501824	8.20* 10-7
0.03415	424	12516096	3.39*10-5
0.03981	894	3783936	2.36*10-4
0.08576	493999	3783936	0.13
0.10000	315791	2037504	0.15

จากตารางที่ 5 แสดงให้เห็นว่าเมื่อค่า Probability of error ที่ช่องสัญญาณ BSC เพิ่มขึ้นจะพบว่าจำนวนบิตข้อมูลที่วงจรถอดรหัสเทอร์โบถอดรหัสออกมาได้นั้นจะมีจำนวนบิตที่ผิดเพิ่มขึ้นเรื่อย ๆ ซึ่งก็จะทำให้ค่า BER เพิ่มขึ้น

4. การทดสอบกับโปรแกรม Matlab



ภาพที่ 38 การเชื่อมต่อระหว่างคอมพิวเตอร์กับ FPGA ผ่าน RS-232



ภาพที่ 39 รูปการติดต่อระหว่าง MATLAB และ FPGA Board

ขั้นตอนการทดสอบ

1. ส่งข้อมูลเสียงจากฟังก์ชันบน MATLAB ไปยัง FPGA Board ด้วย RS-232
2. FPGA รับข้อมูลแล้วทำการแปลงสัญญาณจาก RS-232 ไปเป็นสัญญาณที่จะนำไปเข้าวงจรเข้ารหัสเทอร์โบ
3. ข้อมูลที่ผ่านการเข้ารหัสแล้วจะถูกส่งผ่านช่องสัญญาณแบบ BSC (Binary Symmatric Channel) ซึ่งสามารถปรับค่า Probability of error ได้ จากนั้นจะส่งข้อมูลต่อไปยังวงจรถอดรหัส

4. เมื่อวงจรถอดรหัสทำการถอดรหัสข้อมูลเสร็จเรียบร้อยแล้ว จากนั้นทำการส่งข้อมูลจาก FPGA กลับไปยัง MATLAB โดยผ่าน RS-232

5. เมื่อ MATLAB ได้รับข้อมูลจาก RS-232 ก็จะทำการส่งข้อมูลเสียงออกมาที่ลำโพง

6. ทำการเปรียบเทียบคุณภาพเสียงของข้อมูลที่ส่งไปเข้าวงจรเข้ารหัสและหลังจากวงจรถอดรหัส

จากการทดสอบจะเห็นได้ว่าเมื่อปรับค่า Probability of error ของช่องสัญญาณ BSC ให้มีค่าเท่ากับ 0 ผลปรากฏว่าข้อมูลเสียงที่ได้จากการถอดรหัสเทอร์โบจะเหมือนกับข้อมูลเสียงต้นฉบับทุกประการ จากนั้นทำการค่อย ๆ เพิ่มค่าของ Probability of error ให้มากขึ้นเรื่อย ๆ สัญญาณเสียงที่ได้จากการถอดรหัสก็จะมีความเพี้ยนมากขึ้นเรื่อย ๆ เช่นเดียวกัน

ปรับ Probability of error ของช่องสัญญาณ BSC ให้มีค่าประมาณ 0.04 แล้วเปรียบเทียบข้อมูลเสียงที่ผ่านการถอดรหัสจะเริ่มเห็นความแตกต่างของเสียงมากขึ้นจนเริ่มรู้สึกได้ และจะยิ่งแตกต่างมากยิ่งขึ้นเรื่อย ๆ จนแทบจะฟังไม่รู้เรื่องเมื่อปรับค่า Probability of error ให้มีค่าสูง ๆ

5. ทรัพยากรที่ใช้ในแต่ละโมดูล

5.1 ฟังก์ชันบล็อก Interleave

ทรัพยากรที่ใช้

Number of Slices:	196	out of	10752	1%
Number of Slice Flip Flops:	107	out of	21504	0%
Number of 4 input LUTs:	345	out of	21504	1%
Number of bonded IOBs:	24	out of	242	9%
Number of FIFO16/RAMB16s:	4	out of	72	5%
Number used as RAMB16s:	4			
Number of GCLKs:	1	out of	32	3%

Minimum period: 7.031ns (Maximum Frequency: 142.236MHz)

Minimum input arrival time before clock: 5.642ns

Maximum output required time after clock: 5.723ns

5.2 ฟังก์ชันบล็อก Turbo Encoder

ทรัพยากรที่ใช้

Number of Slices:	766	out of	10752	7%
Number of Slice Flip Flops:	408	out of	21504	1%
Number of 4 input LUTs:	1249	out of	21504	5%
Number of bonded IOBs:	27	out of	242	11%
Number of FIFO16/RAMB16s:	11	out of	72	15%
Number used as RAMB16s:	11			
Number of GCLKs:	1	out of	32	3%

Minimum period: 7.841ns (Maximum Frequency: 127.538MHz)

Minimum input arrival time before clock: 5.752ns

Maximum output required time after clock: 4.001ns

5.3 ฟังก์ชันบล็อก MAP Decoder

ทรัพยากรที่ใช้

Number of Slices:	4125	out of	10752	38%
Number of Slice Flip Flops:	1073	out of	21504	4%
Number of 4 input LUTs:	7299	out of	21504	33%
Number of bonded IOBs:	150	out of	242	61%
Number of FIFO16/RAMB16s:	34	out of	72	47%
Number used as RAMB16s:	34			
Number of GCLKs:	2	out of	32	6%

Minimum period: 14.964ns (Maximum Frequency: 66.825MHz)

Minimum input arrival time before clock: 14.668ns

Maximum output required time after clock: 4.001ns

5.4 ฟังก์ชันบล็อก Deinterleave

ทรัพยากรที่ใช้

Number of Slices:	196	out of	10752	1%
Number of Slice Flip Flops:	107	out of	21504	0%
Number of 4 input LUTs:	345	out of	21504	1%
Number of bonded IOBs:	24	out of	242	9%
Number of FIFO16/RAMB16s:	4	out of	72	5%
Number used as RAMB16s:	4			
Number of GCLKs:	1	out of	32	3%

Minimum period: 7.031ns (Maximum Frequency: 142.236MHz)

Minimum input arrival time before clock: 5.642ns

Maximum output required time after clock: 5.723ns

5.5 ฟังก์ชันบล็อก Turbo Decoder

ทรัพยากรที่ใช้

Number of Slices:	5381	out of	10752	50%
Number of Slice Flip Flops:	1881	out of	21504	8%
Number of 4 input LUTs:	9363	out of	21504	43%
Number of bonded IOBs:	37	out of	242	15%
Number of FIFO16/RAMB16s:	63	out of	72	87%
Number used as RAMB16s:	63			
Number of GCLKs:	2	out of	32	6%

Minimum period: 14.401ns (Maximum Frequency: 69.440MHz)

Minimum input arrival time before clock: 7.039ns

Maximum output required time after clock: 5.723ns

สรุป

1. การพัฒนาฟังก์ชันบล็อกการเข้ารหัสของรหัสเทอร์โบและบล็อกถอดรหัสของรหัสเทอร์โบซึ่งภายในประกอบด้วยฟังก์ชันบล็อกย่อย ๆ ด้วยภาษา VHDL และได้ทำการทดสอบการทำงานของแต่ละฟังก์ชันบล็อกผลปรากฏว่าแต่ละฟังก์ชันบล็อกสามารถทำงานได้ถูกต้อง

2. การทดสอบฟังก์ชันการทำงานของระบบ โดยการรวมฟังก์ชันบล็อกการเข้ารหัสและบล็อกการถอดรหัสของรหัสเทอร์โบ แล้วนำสัญญาณเอาพุทที่ได้จากการเข้าแบบรหัสเทอร์โบมาป้อนเป็นอินพุทให้กับบล็อกการถอดรหัสของเทอร์โบ เพื่อทดสอบบล็อกการถอดรหัสว่าทำงานถูกต้องหรือไม่ จากนั้นทำการทดสอบโดยการจำลองช่องสัญญาณขึ้นมา โดยนำสัญญาณเอาพุทที่ได้จากการเข้ารหัสเทอร์โบมาป้อนเป็นอินพุทของช่องสัญญาณ BSC ที่สามารถปรับค่า Probability of error ได้จากนั้นนำเอาพุทที่ผ่านช่องสัญญาณมาป้อนเป็นอินพุทให้กับบล็อกการถอดรหัสของเทอร์โบเพื่อทดสอบบล็อกการถอดรหัสอีกครั้งว่าทำงานถูกต้องหรือไม่ ซึ่งผลปรากฏว่าทั้งสองกรณีที่ได้ทำการทดลองนั้นตัวถอดรหัสของเทอร์โบสามารถทำงานได้ถูกต้อง

3. การทดสอบประสิทธิภาพของตัวถอดรหัสของรหัสเทอร์โบด้วยการคำนวณหาค่า Bit Error Rate (BER) ที่ Probability of error ค่าต่าง ๆ ผลปรากฏว่าประสิทธิภาพของตัวถอดรหัสอยู่ในเกณฑ์ที่ดี จะเห็นได้ว่าด้วยประสิทธิภาพต่าง ๆ ดังกล่าว บล็อกฟังก์ชันของส่วนเข้ารหัสและส่วนถอดรหัสของรหัสเทอร์โบที่ได้พัฒนาขึ้นสามารถที่จะนำไปประยุกต์ใช้ในระบบการสื่อสารรูปแบบอื่น ๆ ได้

เอกสารและสิ่งอ้างอิง

ชัยวัฒน์ แก้วสาย, ดิสพล ฉ่ำเฉียวกุล และเกียรติศักดิ์ ศรีพิมานวัฒน์. 2545. การเข้ารหัสแบบ Turbo Codes, น. 1-10. **รายงานการการวิจัย.** ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ.

Benedetto, S. and G. Montorsi. 1996. Design of parallel concatenated codes. **IEEE Trans.Comm**.

Berrou , C., A. Glavieux and P. Thitimajshima. 1993. Near Shannon limit error-correcting coding and decoding:Turbo codes. **Proc.1993Int.** 1: 1064-1070.

Hagenauer, J. and P. Hoeher. 1989. A Viterbi algorithm with soft-decision outputs and its applications. **Proc.GlobeCom**. 1: 1680-1686.

Ohtsuki, Tomoaki and Joseph M. Kahn . 2000. BER Performance of Turbo-Coded PPM CDMA Systems on Optical Fiber. **Journal of lightwave technology**.

Robertso, P. 1994. Illuminating the structure of code and decode of parallel concatenated recursive systematic(turbo) codes. **Proc.GlobeCom**. 1: 1298-1303.

Ungerboeck, G. 1982. Channel coding with multilevel/phase signals. **IEEE Trans.Inf.Theory**.

Valenti, M.C. and J. Sun. 2001. The UMTS Turbo Code and an Efficient Decoder Implementation Suitable for Software-Defined Radios. **International Journal of Wireless Information Networks.**