ส่วนที่ 2 **รายงานผลการวิจัยฉบับสมบูรณ์**

โครงการวิจัยทุนอุดหนุนวิจัย มก. ปีงบประมาณ 2558

โครงการวิจัยรหัส ว-ท(ด)22.58

ระบบรหัสนอนไบนารีสำหรับการสื่อสารข้อมูลสำคัญแบบไร้สายเคลื่อนที่ซึ่งสามารถสร้างอุปกรณ์ได้

Implementable nonbinary coding systems for important data transfer via mobile wireless channel

รศ. ดร. อุศนา ตัณฑุลเวศม์[(1)] นายไกรวีร์ ลิ้มชัยกิจ[(1)]
นายนิธิ หลิ่มโตประเสริฐ[(1)] นายนภทีป์ ไชยยงค์[(1)] นายธนากิต มั่นคง[(1)]
Usana Tuntoolavest[(1)], Kraiwee Limchaikit[(1)],
Nithi Limtoprasert[(1)], Nopatee Chaiyong[(1)], Tanakit Munkong[(1)]

[(1)] *ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์*

Electrical Engineering Department, Faculty of Engineering

หมายเหตุ : ให้ส่งรายงานผลการวิจัยฉบับสมบูรณ์ (ฉบับร่าง) จำนวน 3 ชุด ก่อน โดยสถาบันวิจัยและพัฒนาจะส่งให้ผู้ทรงคุณวุฒิ ประเมิน / วิจารณ์ หากไม่มีการแก้ไข จะแจ้งให้ส่งเพิ่ม แต่หากมีความเห็นข้อเสนอแนะจากผู้ทรงคุณวุฒิให้ปรับแก้ไข จะแจ้งให้ดำเนินการแก้ไข และให้ส่งรายงานผลงานวิจัยฉบับสมบูรณ์ (ฉบับจริง) **จำนวน 12 ชุด** พร้อม Diskette ต่อไป

บทคัดย่อ

งานวิจัยนี้นำเสนอวิธีการใหม่ ในการนำรหัสรีดโซโลมอนมาใช้เป็นรหัสภายในในระบบรหัสคอนคาทิเนตแทนที่จะใช้เป็นรหัสภายนอก ส่วนรหัสภายนอกใช้เป็นรหัสคอนโวลูชันแบบนอนไบนารี ด้วยวิธีการนี้ระบบรหัสที่เสนอสามารถแก้ไขความผิดพลาดแบบเบริสต์ที่ยาวมากๆ(ระดับหลักพันบิต)ได้ นอกจากนั้นยังสามารถแก้ไขความผิดพลาดแบบสุ่มและความผิดพลาดแบบเบริสต์ที่ขนาดสั้นกว่า(ระดับหลักร้อยบิต)ได้ ซึ่งเป็นความสามารถในแก้ไขความผิดพลาดของระบบรหัสมาตรฐานที่ใช้รหัสรีดโซโลมอนเป็นรหัสภายนอกและรหัสคอนโวลูชันเป็นรหัสภายใน ทั้งนี้ในช่องสัญญาณไร้สายเคลื่อนที่ ความผิดพลาดแบบเบริสต์จะยาวขึ้นเมื่ออัตราความเร็วการส่งข้อมูลเพิ่มขึ้นในช่วงระยะเวลาการจางหายที่เท่ากัน ผลของงานวิจัยนี้แบ่งออกเป็น 2 ส่วน ส่วนของการจำลองการทำงานและส่วนของชิ้นงานฮาร์ดแวร์ สำหรับผลของการจำลองการทำงาน มุ่งเพื่อใช้วิเคราะห์ผลกระทบของรหัสภายในเมื่อนำเทคนิคสเกลลิงมาใช้กับระบบรหัสที่นำเสนอ นอกจากนั้นการถอดรหัสร่วมกับการลบสัญลักษณ์สำหรับรหัสรีดโซโลมอนภายในถูกพิจารณาในการพยายามเพิ่มประสิทธิภาพของรหัสที่นำเสนอ ผลการจำลองแสดงให้เห็นว่าประสิทธิภาพของระบบรหัสคอนคาทิเนตจะดีขึ้น เมื่อปรับขนาดสัญลักษณ์ให้ใหญ่ขึ้นตามความผิดพลาดแบบเบริสต์ที่ยาวขึ้น ตัวอย่างเช่น สัญลักษณ์ขนาด 1,784 บิต ซึ่งใช้รหัสรีดโซโลมอน(255,223) เป็นรหัสภายใน มีความน่าจะเป็นในการถอดรหัสผิดพลาดเท่ากับ 2×10^{-5} สำหรับช่องสัญญาณจางหายแบบเรย์ลีย์ที่มีความถี่ดอปเปอร์เท่ากับ 255.56 เฮิร์ตที่เอสเอ็นอาร์ 22 เดซิเบล ขณะที่สัญลักษณ์ขนาด 330 บิตและ 223 บิต ที่ใช้รหัสรีดโซโลมอน(63,55) และบีซีเอช(255,223) เป็นรหัสภายในมีความน่าจะเป็นในการถอดรหัสผิดพลาดเท่ากับ 9×10^{-5} และ 2.3×10^{-4} ตามลำดับ การถอดรหัสร่วมกับการลบสัญลักษณ์ของรหัสรีดโซโลมอนภายในโดยใช้ค่าขีดแบ่ง 9% ช่วยลดความน่าจะเป็นในการถอดรหัสผิดพลาดจาก 4.4×10^{-5} เป็น 1.5×10^{-5} ที่เอสเอ็นอาร์ 21 เดซิเบล สำหรับส่วนของฮาร์ดแวร์ ตัวเข้ารหัสคอนโวลูชันภายนอก ตัวเข้าและถอดรหัสรีดโซโลมอนภายในได้ถูกนำไปสร้างบนนาโนบอร์ด3000 ตัวถอดรหัสวีเอสดีซึ่งเป็นตัวถอดรหัสภายนอกได้ถูกแก้ไขให้เป็นโปรแกรมภาษาซีที่ใช้เฉพาะฟังก์ชันที่สามารถนำไปสร้างบนนาโนบอร์ดเดียวกันนี้ได้

คำสำคัญ: รหัสช่องสัญญาณ, รหัสนอนไบนารี, รหัสภายในแบบรีดโซโลมอน, การแก้ไขความผิดพลาดแบบเบริสต์, การถอดรหัสเวกเตอร์ซิมโบล, การถอดรหัสแบบลบสัญลักษณ์, ช่องสัญญาณเคลื่อนที่, บอร์ดทดลองอิเล็กทรอนิกส์

ABSTRACT

This paper proposes a new way of using a Reed-Solomon (RS) code as an inner code of a concatenated code instead of an outer code. The outer code is changed to a nonbinary convolutional code. By doing this, the proposed concatenated code can correct very long burst errors (in the range of thousands of bits) in addition to the random errors and the relatively short burst errors (in the range of hundreds of bits) that the standard RS outer-convolutional inner code can correct. For the same fading duration in the mobile channel, longer burst errors occur when the data rate is increased. The results are divided into two parts, which are the simulation part and the hardware part. For the simulation results, the proposed concatenated code is analyzed to investigate the effect of the inner coding system when the scaling technique is used. In addition, the erasure decoding for RS inner codes is considered in an attempt to further improve the performance of the proposed concatenated coding system. Results show that the performance of the concatenated coding system is better when the symbol size is increased to accommodate the longer burst. For example, the 1,784 bits/symbol which uses RS(255,223) as the inner code has the decoding failure probability of $2x10^{-5}$ for the Rayleigh fading channel with Doppler frequency 255.56 Hz at SNR of 22 dB. While the 330 bits/symbol and 223 bits/symbol which uses RS(63,55) and BCH(255,223) as the inner code have the decoding failure probability of $9x10^{-5}$ and $2.3x\ 10^{-4}$ respectively. Erasure decoding of RS inner code with the threshold of 9% helps lower the inner decoding failure probability from $4.4x10^{-5}$ to $1.5x10^{-5}$ at SNR of 21 dB. For the hardware, the outer convolutional encoder, the inner RS encoder and the inner RS decoder have been implemented on a Nanoboard3000. The Vector Symbol Decoder (VSD), which is the outer decoder, has been modified to be in C programming language that uses only the functions that can be implemented on this Nanoboard.

Key words: channel coding, nonbinary code, burst error correcting, Reed-Solomon Inner code, vector symbol decoding, erasure decoding, Mobile channel, Field programmable gate array, FPGA

บทนำ

เทคโนโลยีของการสื่อสารข้อมูลแบบไร้สายโดยเฉพาะแบบเคลื่อนที่ได้กลายมาเป็นส่วนหนึ่งของชีวิตประจำวัน ของเราและได้รับความนิยมมากขึ้นเรื่อยๆ ดังจะเห็นได้จากการนิยมให้อุปกรณ์สื่อสารแบบพกพาได้ ทั้งแทบเล็ต (Tablet) โทรศัพท์ สมาร์ตโฟน (smart phone) รวมทั้งอุปกรณ์อื่นๆที่สะดวกกับการใช้งานมากขึ้น เช่น Google glass [1] ที่เป็นอุปกรณ์ในรูปแบบแว่นตา หรือ Sumsumg Galaxy Gear [2] ซึ่งเป็นอุปกรณ์ในรูปของนาฬิกาข้อมือซึ่งเพิ่งเปิดตัวไปเมื่อต้นเดือนกันยายน 2556 เป็นต้น อุปกรณ์ต่างๆเหล่านี้ถูกออกแบบให้สามารถเชื่อมต่อกับอินเตอร์เน็ตได้ โดยอาจใช้หลักการของ cloud computer เพื่อเก็บข้อมูลรวมทั้งโปรแกรมต่างๆนอกตัวอุปกรณ์

(a) (b)

รูปที่ 1 อุปกรณ์ Google glass (a) ตัวอุปกรณ์ (b) ตัวอย่างการใช้งาน [1]

รูปที่ 2 อุปกรณ์ Samsung Galaxy gear [2]

รูปที่ 3 อุปกรณ์ tablet iPad [3]

รูปที่ 4 อุปกรณ์ tablet Galaxy Tab [4]

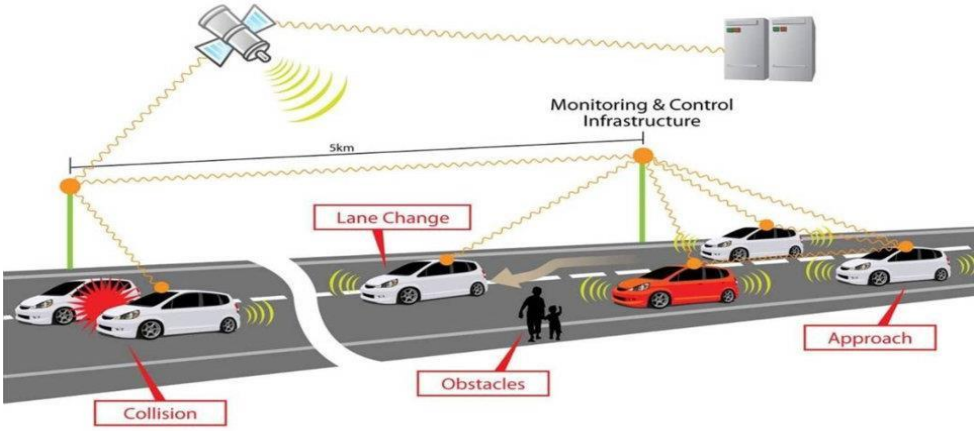
รูปที่ 5 Smart phone Sony Xperia [5]

รูปที่ 6 iPhone [6]

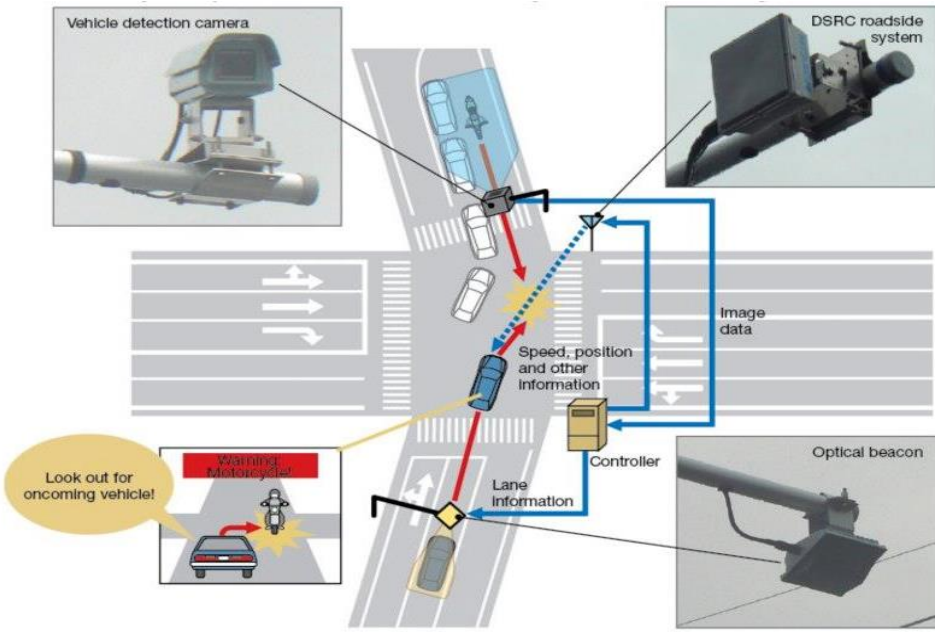
การสื่อสารข้อมูลเหล่านี้ไร้สายเหล่านี้ มีทั้งที่ต้องการความถูกต้องเชื่อถือได้ในระดับปกติเช่น กรณีที่ส่งข้อมูลในเครือข่าย Social network หรือการติดต่อผ่านแอพพิเคชันเช่น “Line” เป็นต้น และยังมีการใช้งานที่ต้องการความเชื่อถือได้สูงมากเช่น การทำธุรกรรมการเงินผ่านธนาคารบนอินเทอร์เน็ต (Internet Banking) และผ่านโทรศัพท์มือถือ(Mobile Banking) เป็นต้น

นอกจากนั้นยังมีแนวโน้มของการใช้งานอินเทอร์เน็ตเพื่อการติดต่อสื่อสารระหว่างอุปกรณ์แบบอัตโนมัติ (Internet of Things) [7] ซึ่งในเบื้องต้นจะเป็นการสื่อสารไร้สายระหว่างอุปกรณ์ที่อยู่กับที่ เช่นเครื่องใช้ไฟฟ้าอัจฉริยะต่างๆในบ้านที่สามารถทำการสั่งซื้ออาหารหรือวัสดุสิ้นเปลืองมาเพิ่มเติมทางออนไลน์ได้เอง และรวมไปถึงการติดต่อสื่อสารกันเองระหว่างอุปกรณ์ที่เคลื่อนที่ เช่น ระบบเพิ่มความปลอดภัยในการขับรถยนต์ที่เรียกว่า Vehicle-to-Vehicle (V2V) [8] ที่มีการแลกเปลี่ยนข้อมูลระหว่างรถยนต์ในขณะขับขี่ ทำการเตือนการเหยียบเบรคและการเสียหลักของรถคันอื่น เป็นต้น และเทคโนโลยีนี้ได้ถูกนำมาทดลองใช้จริง โดยเพิ่มเติมการทำงานร่วมกับกล้องตรวจการจราจร เพื่อใช้ในการแจ้งเตือนผู้ขับขี่ในบริเวณที่อาจเป็นมุมอับสายตา ดังรูปที่ 8(b) จึงเป็นที่คาดหมายได้ว่าในอนาคต วิถีชีวิตประจำวันของมนุษย์จะสัมพันธ์กับเทคโนโลยีการสื่อสารไร้สายและเคลื่อนที่อย่างยิ่ง

รูปที่ 7 Internet of Things [7]



(a) ตัวอย่างการสื่อสารในการขับขี่รถยนต์



(b) ตัวอย่างกล้องตรวจการจราจรที่นำมาใช้

รูปที่ 8 ระบบเพิ่มความปลอดภัยในการขับรถยนต์ที่ (V2V) [8]

ช่องสัญญาณที่ใช้ในการสื่อสารไร้สายนั้นเป็นช่องสัญญาณที่มีคุณภาพต่ำกว่าช่องสัญญาณแบบใช้สาย เนื่องจากเป็นการส่งผ่านอากาศจึงถูกสัญญาณแทรกสอด(interference) และรบกวนได้ง่ายเมื่อมีการเคลื่อนที่ของอุปกรณ์ที่ใช้ส่งและรับสัญญาณก็จะยิ่งทำให้ช่องสัญญาณมีคุณภาพต่ำลง เนื่องจากเกิดปรากฏการณ์ดอปเพลอร์ (Doppler Effect) ช่องสัญญาณไร้สายแบบเฟดนั้น มีความผิดพลาดแบบเบริสต์ (Burst error) ซึ่งการเกิดความผิดพลาดแบบติดกันหลายบิต เพิ่มจากความผิดพลาดแบบสุ่ม (Random error) ซึ่งเกิดขึ้นในทุกช่องสัญญาณ

การเพิ่มความเชื่อถือได้ของการสื่อสารข้อมูลสามารถทำได้โดยการใช้รหัสช่องสัญญาณ ซึ่งรหัสช่องสัญญาณตัวหนึ่งที่เหมาะกับการแก้ไขความผิดพลาดแบบเบริสต์นั้นคือรหัส Reed-Solomon (RS) รหัสนี้มีการใช้งานอย่างแพร่หลายทั้งในการสื่อสารห้วงอวกาศลึก (Deep Space) ตามมาตรฐานของ CCSDS (The Consultative Committee for Space Data System [9] ใช้เป็นมาตรฐานใน Compact Disc (CD) [10] เป็นต้น รหัส RS เป็นรหัสนอนไบนารี (nonbinary code) ที่แก้ไขความผิดพลาดในระดับสัญลักษณ์ซึ่งมีขนาดมากกว่า 1 บิต เมื่อทำการแก้ไขสัญลักษณ์จะแก้ไขความผิดพลาดทุกบิตในแต่ละสัญลักษณ์ ดังนั้นจึงเหมาะกับการแก้ไขความผิดพลาดที่มีลักษณะเป็นเบริสต์ และเมื่ออนุญาตให้มีการลบสัญลักษณ์ที่มีโอกาสผิดพลาดสูงออกไป (erasure symbols) จะทำให้สามารถถอดรหัสได้มีประสิทธิภาพมากขึ้นด้วย โดยวิธีการลบสัญลักษณ์นี้มีความสำคัญต่อประสิทธิภาพในการถอดรหัสด้วย

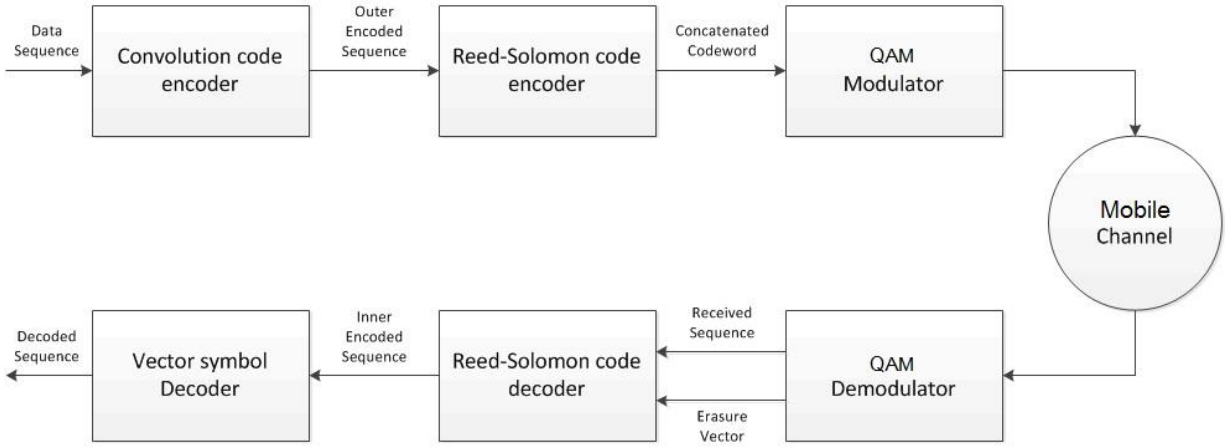
Berlekamp-Massey [11-13] ได้เสนอวิธีการถอดรหัสรีดโซโลมอนโดยการตัดสินใจแบบฮาร์ด (hard decision decoding) ซึ่งมีการใช้กันอย่างแพร่หลาย นอกจากนั้นยังมีความพยายามถอดรหัสแบบซอฟลิส (soft list Reed-Solomon decoding) ของ Guruswami-Sudan(GS) [14] ที่แก้ไขสัญลักษณ์ที่ผิดพลาดได้สูงกว่า t สัญลักษณ์ ซึ่งเป็นขอบเขตตามปกติของรหัส RS แต่โดยทั่วไปการใช้ GS หรือวิธีการถอดรหัส RS แบบซอฟอื่นๆ มักจะมีความซับซ้อนสูง ซึ่งพบว่าการถอดรหัสแบบซอฟลิส มีความซับซ้อนสูงกว่าแบบ Berlekamp-Massey มาก แม้แต่ในช่วงที่จำนวนความผิดพลาดยังไม่เกินความสามารถที่ Berlekamp-Massey จะถอดรหัสได้ จนทำให้มีนักวิจัยหลายคน [15,16] ระบุว่าการถอดรหัสแบบ Guruswami-Sudan นั้นยากที่จะใช้งานได้จริง

โครงการวิจัยนี้จะเสนอวิธีการใหม่ในการเพิ่มประสิทธิภาพของรหัสรีดโซโลมอนโดยการนำรหัสรีดโซโลมอนมาใช้เป็นรหัสภายในของรหัสคอนคาทิเนตที่มีโครงสร้างแบบ Forney [17] โดยจะพิจารณาเปรียบเทียบกรณีที่มีสัญลักษณ์ผิดพลาดอย่างเดียว และกรณีที่มีการผสมกันของสัญลักษณ์ผิดพลาดและสัญลักษณ์ที่ถูกลบ ในส่วนรหัสภายนอกของรหัสคอนคาทิเนตนั้นจะใช้รหัสคอนโวลูชันที่ใช้สัญลักษณ์นอนไบนารีขนาดใหญ่ (32 บิต/สัญลักษณ์ หรือมากกว่า) ซึ่งสามารถถูกถอดรหัสด้วยวิธีถอดรหัสแบบเวกเตอร์ซิมโบล (Vector symbol decoding) ซึ่งผู้วิจัยได้พัฒนาขึ้น การใช้โครงสร้างนี้จะทำให้สามารถแก้ไขความผิดพลาดได้มากกว่าการใช้รหัสรีดโซโลมอนเพียงรหัสเดียว และคาดว่าจะถอดรหัสได้ดีกว่าการใช้รหัสภายในเป็นรหัสไบนารี โดยเฉพาะสำหรับช่องสัญญาณไร้สายเคลื่อนที่ ระบบรหัสที่เสนอนี้สามารถถูกนำไปประยุกต์ใช้งานได้จริงเนื่องจากหลักการถอดรหัสรีดโซโลมอนแบบนี้มีการใช้งานจริงแล้ว และตัวถอดรหัสเวกเตอร์ซิมโบลนั้นได้มีการทดลองสร้างลงบนบอร์ดอิเล็กทรอนิกส์ FPGA (Field Programmable Gate Array) ในงานวิจัยของผู้วิจัยซึ่งได้รับทุนในปีงบประมาณ 2556

โดยสรุปในโครงการวิจัยนี้จะเสนอระบบรหัสช่องสัญญาณแบบใหม่ที่ให้ความเชื่อถือได้สูงและเหมาะกับช่องสัญญาณไร้สายแบบเคลื่อนที่ซึ่งเป็นพื้นฐานในการสื่อสารของอนาคต โดยผู้วิจัยจะออกแบบโครงสร้างรหัสที่เหมาะสม รวมทั้งขั้นตอนวิธีของตัวเข้ารหัส และตัวถอดรหัสทุกตัวที่ใช้ในระบบรหัสใหม่นี้ รวมทั้งทำการวิเคราะห์องค์ประกอบของระบบ โดยระบบรหัสใหม่นี้สามารถนำไปประยุกต์สร้างอุปกรณ์เข้ารหัสถอดรหัสได้ต่อไป งานวิจัยนี้เป็นการพัฒนาต่อยอดจากงานวิจัยที่ได้รับทุกอุดหนุนวิจัยในปีงบประมาณ 2556 และ 2557 โดยขอบเขตงานวิจัยที่เสนอส่วนหนึ่งจะมาจากขอบเขตของโครงการวิจัยเดิม เนื่องจากในโครงการวิจัยปี 2556 และ 2557 นั้นได้รับอนุมัติงบประมาณเพียง 70% และ 60% จึงจำเป็นต้องลดขอบเขตลงบางส่วนตามข้อเสนอโครงการฉบับปรับปรุงแก้ไข และได้รวมไว้เป็นส่วนหนึ่งของโครงการวิจัยที่ได้รับงบประมาณในปี 2558 นี้

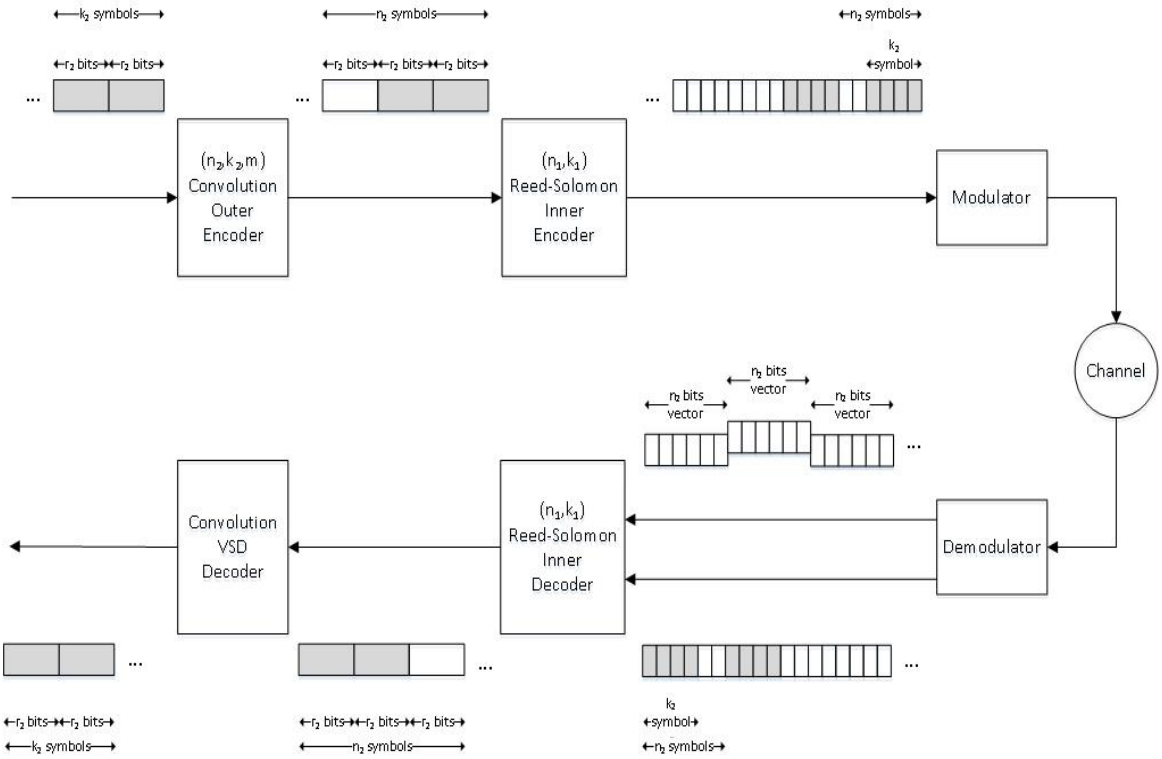
กรอบแนวความคิดของโครงการวิจัย

โครงการวิจัยนี้นำเสนอระบบรหัสคอนคาทิเนตแบบลำดับ (serial concatenated code) ซึ่งประกอบด้วยรหัสสองชั้น รหัสภายนอกที่เลือกใช้คือรหัสคอนโวลูชัน และรหัสภายในที่เลือกใช้คือรหัส Reed-Solomon (RS) ตามรูปที่ 9 ทั้งรหัสภายนอกและรหัสภายในจะใช้สัญลักษณ์นอนไบนารี รหัสคอนโวลูชันภายนอกนั้นจะใช้สัญลักษณ์ขนาดใหญ่มาก (ตั้งแต่ 32 บิต/สัญลักษณ์หรือมากกว่า) ส่วนรหัส Reed-Solomon (RS) ภายในจะใช้สัญลักษณ์ขนาดไม่เกิน 8 บิต ซึ่งแตกต่างการจากรหัสคอนคาทิเนตพื้นฐานที่มักใช้รหัสบล็อกแบบไบนารีเป็นรหัสภายในและมักใช้รหัส Reed-Solomon เป็นรหัสภายนอก เพื่อให้รหัสภายในแก้ไขความผิดพลาดแบบสุ่มและรหัสภายนอกแก้ไขความผิดพลาดแบบเบริสต์ ตัวอย่างเช่นรหัส RS ที่ใช้อย่างแพร่หลายที่สุดคือรหัส (255, 223) ที่ใช้สัญลักษณ์ขนาด 8 บิตตามมาตรฐาน CCSDS [9]

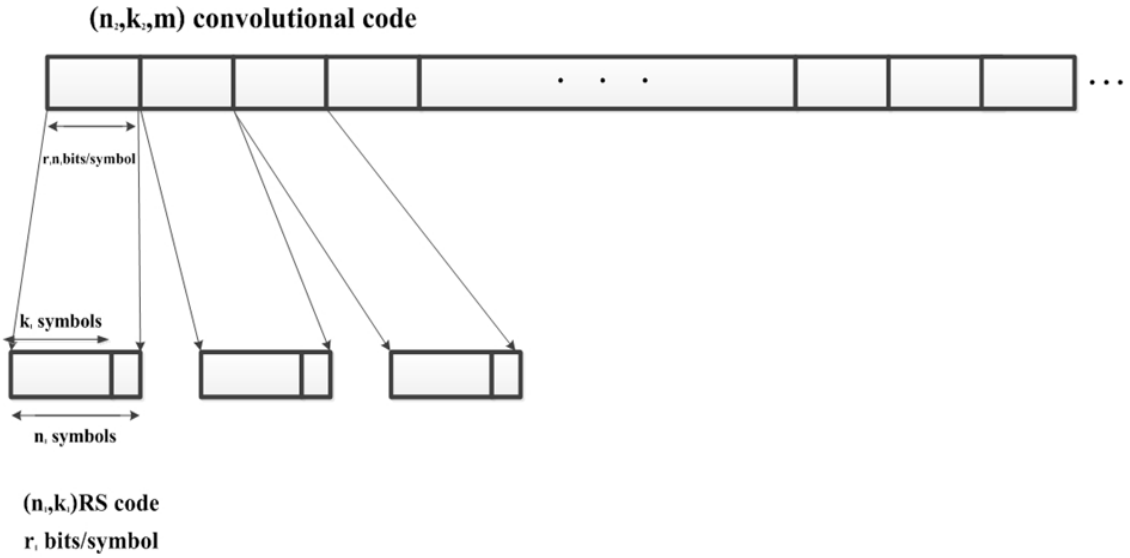


รูปที่ 9 แผนภาพของระบบที่นำเสนอ

การออกแบบระบบรหัสคอนคาทิเนตแบบใหม่ที่นำ RS มาใช้เป็นรหัสภายใน โดยใช้ร่วมกับการลบสัญลักษณ์ (erasure) บางส่วนที่มีความน่าจะเป็นที่จะผิดพลาดสูงตามรูปที่ 9 ตัวดีมอดูเลเตอร์ (demodulator) จะส่งทั้งสัญลักษณ์ที่ได้รับ (received sequence) และเวกเตอร์ระบุตำแหน่งสัญลักษณ์ที่ถูกลบ (erasure vector) ให้กับตัวถอดรหัส Reed-Solomon ซึ่งจะประยุกต์วิธีการถอดรหัสแบบฮาร์ด (hard decision decoding) ของ Berlekamp-Massey [18,19] ซึ่งจะอธิบายในหัวข้อทฤษฎี จากรูปที่ 10 ซึ่งแสดงรายละเอียดของจำนวนสัญลักษณ์ที่เข้าและออกจากตัวเข้ารหัสถอดรหัสทั้งภายในและภายนอกนั้น ความสัมพันธ์ของจำนวนบิต/สัญลักษณ์ถูกขยายให้เห็นชัดเจนขึ้นในรูปที่ 11



รูปที่ 10 แผนภาพระบบรหัสที่เสนอซึ่งแสดงรายละเอียดของจำนวนสัญลักษณ์



รูปที่ 11 ความสัมพันธ์ระหว่างขนาดของสัญลักษณ์ภายนอกและรหัสภายใน

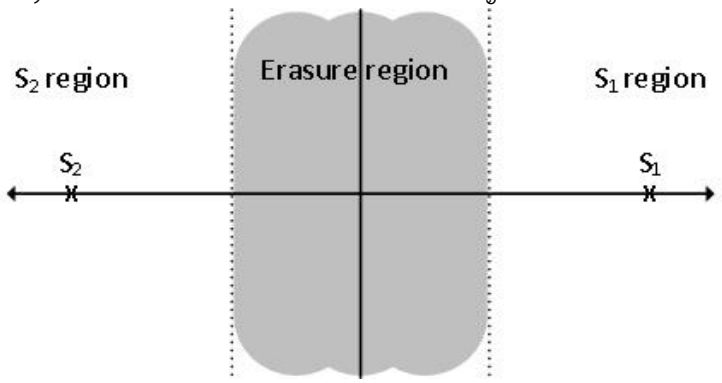
ระบบที่เสนอนี้เป็นการออกแบบเพื่อรองรับอัตราการส่งข้อมูลที่มีแนวโน้มเพิ่มขึ้นเรื่อยๆอย่างก้าวกระโดดในอนาคตอันใกล้ ของการสื่อสารไร้สาย และการสื่อสารไร้สายแบบเคลื่อนที่ ตัวอย่างเช่นจากมาตรฐาน 3G ที่อัตราการส่งข้อมูลได้ถึง 2 Mb/s ไปยังมาตรฐาน 4G ที่จะมีอัตราการส่งข้อมูลสูงสุด 100 Mb/s ในสภาวะที่ผู้ใช้มีการเคลื่อนที่ด้วยความเร็ว เมื่อใช้งานในยานพาหนะที่เคลื่อนที่ และอัตราการส่งข้อมูลที่สูงถึง 1 Gb/s สำหรับผู้ใช้งานที่เดินเท้าหรืออยู่กับที่ [20]

การเพิ่มขึ้นของอัตราการส่งข้อมูลนี้ทำให้ความผิดพลาดแบบเบริสต์ที่เกิดขึ้นในการสื่อสารไร้สายมีความยาวเบริสต์ (burst length) ที่เพิ่มขึ้นอย่างก้าวกระโดดเช่นกัน ดังนั้นเพื่อให้เหมาะกับความผิดพลาดแบบเบริสต์ที่

มีค่าเฉลี่ยที่ยาวขึ้น โครงการวิจัยนี้เสนอให้เปลี่ยนรหัสภายในของรหัสคอนคาทิเนตจากรหัสไบนารีที่แก้ความผิดพลาดแบบสุ่ม มาเป็นรหัสนอนไบนารีที่ใช้สัญลักษณ์ขนาดกลาง (ไม่เกิน 8 บิต/สัญลักษณ์) ที่มีประสิทธิภาพสูง คือรหัส Reed-Solomonและเพื่อเพิ่มประสิทธิภาพของการถอดรหัสจะใช้การตัดสินใจลบสัญลักษณ์บางส่วนที่มีโอกาสผิดพลาดสูง (Erasure decoding) ร่วมด้วย โดยจะพิจารณาทั้งกรณีที่ใช้ตัวถอดรหัสภายในแก้ไขความผิดพลาดเท่านั้น กรณีที่ใช้เติมสัญลักษณ์ที่ถูกลบไปเท่านั้น และกรณีที่ใช้ทั้งแก้ไขความผิดพลาดและเติมสัญลักษณ์ที่ถูกลบไป สำหรับความผิดพลาดที่ไม่สามารถแก้ไขได้ด้วยรหัสภายในนั้นจะถูกส่งต่อไปให้ตัวถอดรหัสภายนอกต่อไป

แนวคิดในถอดรหัสด้วยสัญลักษณ์ที่ถูกลบนี้เนื่องมาจากการที่ตัวถอดรหัสจะแก้ไขจำนวนสัญลักษณ์ที่ถูกลบได้มากกว่าสัญลักษณ์ที่ผิดพลาด ทั้งนี้เพราะตัวถอดรหัสมีสารสนเทศ (information) เกี่ยวกับสัญลักษณ์ที่ถูกลบมากกว่า สารสนเทศของสัญลักษณ์ที่ผิด เพราะตัวถอดรหัสจะทราบตำแหน่งของสัญลักษณ์ที่ถูกลบ แต่ไม่ทราบตำแหน่งของสัญลักษณ์ที่ผิดพลาด ซึ่งจะต้องมีการจัดการในการลบสัญลักษณ์อย่างเหมาะสมในขั้นตอนของการดีมอดูเลต โดยจะพิจารณาสัญลักษณ์ที่มีความน่าจะเป็นที่จะผิดพลาดได้สูง

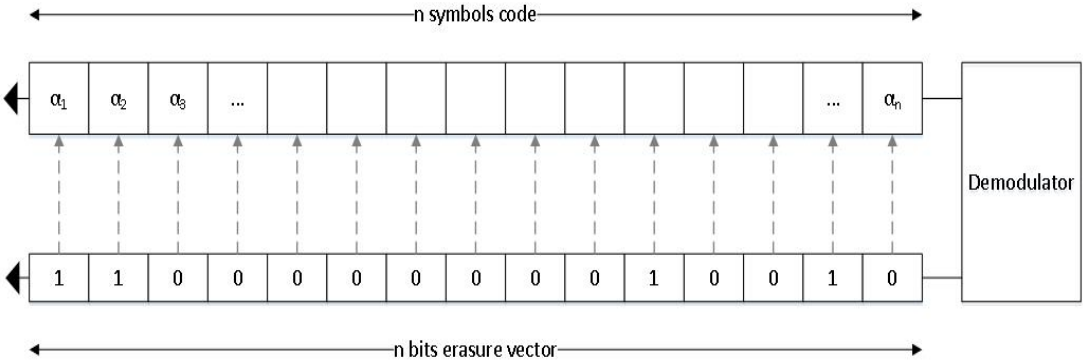
วิธีการที่ง่ายที่สุดในการระบุตำแหน่งของสัญลักษณ์ที่จะลบ คือ การใช้ค่าขีดแบ่ง (threshold) ในการตัดสินใจ ซึ่งอาจตัดสินใจจากกำลังสัญญาณที่รับมาต่ำเกินกว่าค่าที่กำหนด หรือตัดสินใจจากการที่สัญญาณที่รับมาอยู่บริเวณพื้นที่ขอบ (boundary area) ที่กำหนด ซึ่งคาบเกี่ยวระหว่างสัญลักษณ์อย่างน้อย 2 สัญลักษณ์ก็ได้



รูปที่ 12 แนวความคิดพื้นฐานของการกำหนดบริเวณ (region) ที่จะถูกลบ

จากรูปที่ 12 คือภาพของการใช้การมอดูเลตแบบ Pulse Amplitude Modulation (PAM) โดยในขั้นตอนการดีมอดูเลตโดยปกตินั้น ภาครับจะตัดสินใจว่าได้รับสัญญาณใดมาโดยดูจากว่าสัญญาณอยู่ในบริเวณใด แต่การระบุสัญลักษณ์ที่ถูกลบโดยใช้ค่าขีดแบ่งนั้น จะมีการระบุบริเวณในปริภูมิสัญญาณที่สัมพันธ์กับสัญลักษณ์ที่จะถูกลบ (Erasure Region) โดยขอบเขตนี้ขึ้นกับค่าขีดแบ่งที่กำหนด แต่การกำหนดค่าขีดแบ่งที่เหมาะสมนั้นจะต้องใช้การทดสอบและบันทึกผล เนื่องจากค่าขีดแบ่งที่เหมาะสมจะแตกต่างกันตามคุณสมบัติของช่องสัญญาณ [21] และลักษณะของพื้นที่ก็จะแตกต่างตามวิธีการที่ใช้ในการมอดูเลต

ระหว่างการดีมอดูเลต หากพบสัญญาณที่อยู่ในพื้นที่สัญลักษณ์ที่ถูกลบ ในกรณีทั่วไปตัวดีมอดูเลตจะแทนที่สัญลักษณ์นั้นด้วย 0 และสร้างข้อมูลใหม่ที่เรียกว่าเวกเตอร์สัญลักษณ์ที่ถูกลบคู่ไปด้วยกัน แต่วิธีการที่จะใช้นั้นไม่ได้สร้างตัวดีมอดูเลตขึ้นมาใหม่ จึงเพิ่มเติมเฉพาะในส่วนของการหาเวกเตอร์สัญลักษณ์ที่ถูกลบซึ่งทำงานคู่กับตัวดีมอดูเลตเดิม ผลคือจะได้รับสัญลักษณ์ที่ดีมอดูเลตได้จริงโดยไม่ถูกแทนที่ด้วย 0 ซึ่งไม่ส่งผลกระทบใดกับการถอดรหัส และสามารถนำไปใช้ได้เลยในกรณีที่จะทำการถอดรหัสโดยไม่ใช้สัญลักษณ์ที่ถูกลบ ตามรูปที่ 13 ขนาดของเวกเตอร์สัญลักษณ์ที่ถูกลบจะมีความสัมพันธ์กับความยาวของคำรหัส โดยใช้ข้อมูลเพียง 1 บิทต่อสัญลักษณ์เพื่อระบุว่าสัญลักษณ์ใดมีความน่าเชื่อถือต่ำ



รูปที่ 13 สัญลักษณ์ที่ได้รับและเวกเตอร์สัญลักษณ์ที่ถูกลบ

สำหรับรหัสภายนอกนั้นควรสามารถใช้สัญลักษณ์ที่มีขนาดใหญ่มากได้ เพราะขนาดสัญลักษณ์ควรจะขึ้นกับค่าเฉลี่ยความยาวของเบริสต์ในแต่ละสภาวการณ์ รหัส RS จึงไม่เหมาะที่ใช้เป็นรหัสภายนอกสำหรับการสื่อสารไร้สายความเร็วสูง เพราะรหัสนี้มักใช้สัญลักษณ์ขนาดไม่ใหญ่นัก รหัสคอนโวลูชันที่เลือกใช้เป็นรหัสภายนอก จะเป็นรหัสที่ใช้สัญลักษณ์ขนาดใหญ่ได้ง่าย โดยในโครงการวิจัยนี้จะใช้ได้ถึง 1,784 บิต/สัญลักษณ์ ซึ่งในด้านการถอดรหัสคอนโวลูชันด้วยวิธีเวกเตอร์ซิมโบล (vector symbol decoding –VSD) นั้น ขนาดของสัญลักษณ์ที่ใหญ่ขึ้นจะไม่ส่งผลต่อความซับซ้อนของการถอดรหัสมากนัก และยังเพิ่มโอกาสที่ความผิดพลาดในแต่ละสัญลักษณ์ มีความเป็นอิสระเชิงเส้นต่อกัน ซึ่งเป็นเงื่อนไขหนึ่งที่จะทำให้การถอดรหัสด้วย VSD ประสบความสำเร็จ วิธีการถอดรหัสแบบ VSD จะอธิบายในหัวข้อทฤษฎี ระบบที่สร้างขึ้นนี้มุ่งเน้นไปที่การนำไปใช้ในระบบสื่อสารแบบไร้สายและช่องสัญญาณไร้สายเคลื่อนที่ ช่องสัญญาณที่นำมาทดสอบจะใช้เป็นช่องสัญญาณที่มีการเฟดดิง โดยเฉพาะช่องสัญญาณแบบเรย์ลีย์ และแบบไรเซียนซึ่งเป็นช่องสัญญาณที่มีคุณภาพต่ำบางเวลา และพบได้ทั่วไปในการสื่อสารแบบไร้สาย รวมถึงการมีปรากฏการณ์ดอปเพลอร์ (Doppler effect) เนื่องจากการเคลื่อนที่ของเครื่องส่งและ/หรือเครื่องรับอีกด้วย

สมมุติฐาน

สมมุติฐานที่ตั้งไว้คือการใช้รหัสภายในแบบนอนไบนารีที่มีประสิทธิภาพในการแก้ไขความผิดพลาดสูงและมีขนาดสัญลักษณ์ 6 บิตและ 8 บิตจะทำให้ตัวถอดรหัสสามารถแก้ไขความผิดพลาดแบบสุ่มได้บาง

ส่วนและแก้ไขความผิดพลาดแบบเบริสต์ได้บางส่วนด้วย ซึ่งจะทำให้ตัวถอดรหัสภายนอกต้องแก้ไขเฉพาะกรณีที่มีความผิดพลาดแบบเบริสต์ขนาดใหญ่เท่านั้น โดยถ้าช่องสัญญาณมีคุณภาพต่ำในช่วง 1-2 สัญลักษณ์ภายนอก จะทำให้ตัวถอดรหัสภายนอกแบบ VSD ทำงานได้ดี โดยเมื่อตัวถอดรหัสภายในมีการเติมสัญลักษณ์ที่ถูกลบ (erasure decoding) ผู้วิจัยคาดว่าจะทำให้การถอดรหัสโดยรวมของระบบมีประสิทธิภาพดีขึ้นเมื่อคุณภาพช่องสัญญาณไม่ดีนัก ในขณะที่การถอดรหัสภายในแบบแก้ไขความผิดพลาดเท่านั้น อาจจะมีประสิทธิภาพที่ดีกว่าถ้าช่องสัญญาณมีคุณภาพดี ซึ่งจะต้องทำการทดลองต่อไป

ทฤษฎี

ในส่วนนี้จะเน้นอธิบายหลักการพื้นฐานของการเข้ารหัสถอดรหัสช่องสัญญาณที่ใช้ ซึ่งจะประกอบด้วยรหัสคอนโวลูชันและรหัส Reed-Solomon โดยทั้งสองรหัสนี้จะใช้กับสัญลักษณ์นอนไบนารี รหัสคอนโวลูชันนั้นปกติมักใช้เป็นรหัสไบนารี แต่ในระบบที่เสนอถูกเลือกให้เป็นรหัสภายนอกของรหัสคอนคาทิเนตจึงต้องใช้เป็นรหัสนอนไบนารีที่มีขนาดใหญ่ โดยการถอดรหัสคอนโวลูชันแบบนี้ไม่สามารถถอดรหัสด้วยวิธีวิเทอร์บิ (Viterbi decoding) ซึ่งเป็นการถอดรหัสแบบ maximum likelihood decoding ที่มักถูกใช้ในการถอดรหัสคอนโวลูชันแบบไบนารีได้ เนื่องจากจำนวนการคำนวณและจำนวนสถานะ (state) จะเพิ่มขึ้นอย่างทวีคูณ จึงต้องใช้การถอดรหัสที่เรียกว่า vector symbol decoding (VSD) ที่เหมาะกับการถอดรหัสเชิงเส้นที่ใช้สัญลักษณ์นอนไบนารีขนาดใหญ่

สำหรับรหัส Reed-Solomon นั้นมีวิธีการถอดรหัสมากมาย เดิมเป็นการถอดรหัสโดยการตัดสินใจแบบฮาร์ด(hard decision) ต่อมามีผู้พัฒนาการถอดรหัสแบบมีสัญลักษณ์ที่ถูกลบไป (erasure) และยังมีการถอดรหัสที่มีการตัดสินใจแบบซอฟ (soft decision) และการถอดรหัสที่มีให้ผลลัพธ์มากกว่าหนึ่งผลลัพธ์ที่เรียกว่าการถอดรหัสแบบลิส (List decoding) ซึ่งในโครงการวิจัยจะใช้การถอดรหัสแบบฮาร์ด ร่วมกับแบบมีสัญลักษณ์ที่ถูกลบไป ซึ่งจะอธิบายโดยสังเขปต่อไป

1. รหัสคอนโวลูชันเชิงเส้น (Linear Convolutional Code) [18]

รหัสคอนโวลูชันเชิงเส้นนั้นจะถูกระบุด้วยพารามิเตอร์ (n,k,m) และ ν โดยที่ n คือจำนวนบิตของคำรหัส k คือจำนวนบิตของข้อมูลที่ต้องการเข้ารหัส m คือระดับหน่วยความจำว่าข้อมูลมีความสัมพันธ์ย้อนหลังกี่หน่วยเวลา และ ν คือความยาวคอนสแตน (constraint length) ซึ่งเท่ากับจำนวนรีจิสเตอร์ (register) ที่มีการใช้งานในการเข้ารหัส ซึ่งค่านี้ว่า ตัวเข้ารหัสสามารถเขียนเป็นวงจรรีจิสเตอร์แบบเลื่อน (Shift Register Circuit) จากเมตริกตัวกำเนิด (Generator Matrix) ได้

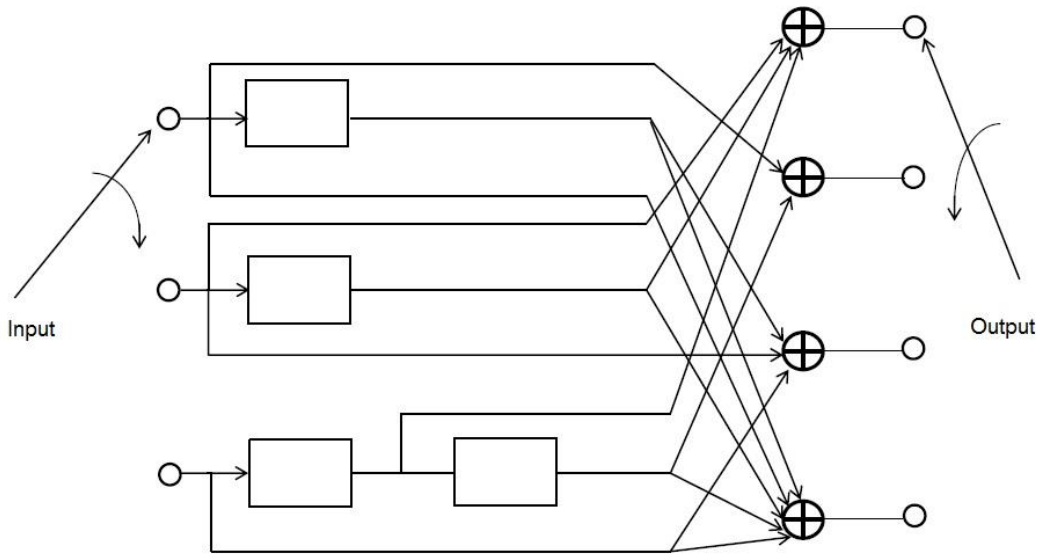
ตัวอย่าง รหัสคอนโวลูชัน (4,3,2) และ ν = 4 มีเมตริกตัวกำเนิดในรูปแบบทรานฟอร์มโดเมนแปลงคือ

$$G(D) = \begin{pmatrix} 0 & 1 & D & D+1 \\ D+1 & 0 & 1 & D \\ D & D^2 & 1 & D^2+1 \end{pmatrix} \quad (1)$$

ในรูป เลขฐาน 8 และฐาน 2 ตามสมการ(2)

$$\mathbf{G} = \begin{pmatrix} 0_8 & 1_8 & 2_8 & 3_8 \\ 3_8 & 0_8 & 1_8 & 2_8 \\ 2_8 & 4_8 & 1_8 & 5_8 \end{pmatrix} = \begin{pmatrix} 000_2 & 001_2 & 010_2 & 011_2 \\ 011_2 & 000_2 & 001_2 & 010_2 \\ 010_2 & 100_2 & 001_2 & 101_2 \end{pmatrix} \quad (2)$$

รหัสนี้สามารถเขียนเป็นวงจรรีจิสเตอร์แบบเลื่อนได้ดังรูปที่ 14



รูปที่ 14 วงจรรีจิสเตอร์แบบเลื่อนได้ของรหัสคอนโวลูชัน (4,3,2) และ ν = 4

ในรูปที่ 14 จะเห็นได้ว่า m = 2 เพราะ ข้อมูลมีความสัมพันธ์ย้อนหลังมากที่สุด 2 หน่วยเวลา ในขณะที่ ν = 4 เนื่องจากในเส้นของข้อมูล u1 และ u2 จะใช้รีจิสเตอร์อย่างละหนึ่งตัว ในขณะที่เส้นข้อมูล u3 จะใช้รีจิสเตอร์สองตัว

<u>การเข้ารหัสคอนโวลูชันแบบนอนไบนารี</u>

การเข้ารหัสคอนโวลูชันแบบนอนไบนารีทำได้โดยการป้อนข้อมูลเข้าไปในวงจรรีจิสเตอร์แบบเลื่อนทีละสัญลักษณ์ตัวอย่างเช่น ให้ขนาดสัญลักษณ์เท่ากับ 2 บิตจะได้ Input ขาเข้าตามสมการที่ (3)

$$\mathbf{u_1} = ((u_1^{(0)}, u_1^{(1)}), (u_1^{(2)}, u_1^{(3)}), ...)$$
$$\mathbf{u_2} = ((u_2^{(0)}, u_2^{(1)}), (u_2^{(2)}, u_2^{(3)}), ...)$$
$$\mathbf{u_3} = ((u_3^{(0)}, u_3^{(1)}), (u_3^{(2)}, u_3^{(3)}), ...) \quad (3)$$

ชุดบิตข้อมูลที่จะเข้ารหัสทั้งหมดเขียนได้ตามสมการที่ (4)

$$\mathbf{U} = ((u_1^{(0)}, u_1^{(1)}), (u_2^{(0)}, u_2^{(1)}), (u_3^{(0)}, u_3^{(1)}), ...)$$
$$= [\, u_1\ u_2\ u_3\,] \quad (4)$$

โดยในแต่ละรอบของการเลื่อนของวงจรรีจิสเตอร์แบบเลื่อน Input จะเลื่อนเข้ามายังวงจรทีละ 1 ชุด ยกตัวอย่างเช่นรอบแรกของการเลื่อน Input คือ $((u_1^{(0)}, u_1^{(1)}), (u_2^{(0)}, u_2^{(1)}), (u_3^{(0)}, u_3^{(1)}))$ และจะได้ Output ออกมาเป็นรหัสที่ถูกเข้ารหัสแล้วคือ $((v_1^{(0)}, v_1^{(1)}), (v_2^{(0)}, v_2^{(1)}), (v_3^{(0)}, v_3^{(1)}), (v_4^{(0)}, v_4^{(1)}))$ ดังนั้นจะได้คำรหัสที่ผ่านการเข้ารหัสตามสมการที่ (5)

$$\mathbf{v_1} = ((v_1^{(0)}, v_1^{(1)}), (v_1^{(2)}, v_1^{(3)}), ...)$$
$$\mathbf{v_2} = ((v_2^{(0)}, v_2^{(1)}), (v_2^{(2)}, v_2^{(3)}), ...)$$
$$\mathbf{v_3} = ((v_3^{(0)}, v_3^{(1)}), (v_3^{(2)}, v_3^{(3)}), ...)$$
$$\mathbf{v_4} = ((v_4^{(0)}, v_4^{(1)}), (v_4^{(2)}, v_4^{(3)}), ...) \quad (5)$$

และคำรหัสที่ได้ออกมาตามสมการที่ (6)

$$\mathbf{V} = ((v_1^{(0)}, v_1^{(1)}), (v_2^{(0)}, v_2^{(1)}), (v_3^{(0)}, v_3^{(1)}), (v_4^{(0)}, v_4^{(1)}), ...) = [\mathbf{v_1}\ \mathbf{v_2}\ \mathbf{v_3}\ \mathbf{v_4}\,] \quad (6)$$

การเข้ารหัสคอนโวลูชันนั้นจะเข้ารหัสต่อเนื่องไปเรื่อยๆ แต่ถ้าข้อมูลสิ้นสุดแล้วจะยังต้องมีการป้อนข้อมูลที่เป็นศูนย์หมดเข้าไปในวงจรเพื่อให้ข้อมูลส่วนที่อยู่ในหน่วยความจำออกมาเป็นส่วนหนึ่งของคำรหัสด้วย โดยจำนวนศูนย์ที่ป้อนเข้าไปนี้จะเท่ากับจำนวนของระดับหน่วยความจำ m ของรหัสคอนโวลูชัน (n,k,m) ดังนั้นสามารถคำนวณจำนวนบิตในแต่ละคำรหัสได้ตามสมการที่ (7)

จำนวนบิตคำรหัส = จำนวนสัญลักษณ์ของคำรหัส × ขนาด ของสัญลักษณ์

$$= (((N/k) + m) \times n) \times r \quad (7)$$

โดย N = จำนวนบิตข้อมูลทั้งหมดที่ต้องการเข้ารหัส

r = จำนวนบิตต่อสัญลักษณ์

2. การถอดรหัสเวกเตอร์ซิมโบล (VSD) [18]

วิธีการถอดรหัสแบบเวกเตอร์ซิมโบลมีสมมุติฐานว่าความผิดพลาด(Error) ที่เกิดขึ้นในแต่ละสัญลักษณ์จะต้องเป็นอิสระเชิงเส้นต่อกัน(Linearly independent) ซึ่งสมมุติฐานนี้จะใกล้เคียงความจริงเมื่อสัญลักษณ์ที่ใช้มีขนาดใหญ่พอ ซึ่งในโครงการวิจัยนี้คาดว่าจะใช้ 32-128 บิต/สัญลักษณ์ ดังนั้นการถอดรหัสนี้จึงใช้มีความแตกต่างจากการถอดรหัสโดยทั่วไปอย่างมากเนื่องจากรหัสส่วนใหญ่จะใช้สัญลักษณ์ที่มีขนาดเล็กโดยเฉพาะ 1 บิต/สัญลักษณ์

การใช้สัญลักษณ์ขนาดใหญ่ทำให้การถอดรหัสสามารถใช้หลักการพิสูจน์หาสัญลักษณ์ที่ถูกต้อง (Verification)ได้ดี และแนวคิดของการถอดรหัสแบบนี้มีความยืดหยุ่นสูงคือสามารถนำไปประยุกต์ใช้กับรหัสต่างๆ ได้มากมาย ทั้งรหัสแบบบล็อกและรหัสแบบคอนโวลูชันเพียงแต่ดัดแปลงให้รหัสเหล่านั้นใช้สัญลักษณ์ขนาดใหญ่ในการเข้ารหัส

การใช้งานของการถอดรหัสนี้เหมาะสมกับช่องสัญญาณแบบไร้สายเนื่องจากมีความสามารถในการแก้ไขข้อผิดพลาดที่ติดๆกัน(Burst errors) ได้ดี และเหมาะกับการเป็นตัวถอดรหัสภายนอกของรหัสคอนคาทิเนต โดยเฉพาะรหัสคอนคาทิเนตที่มีรหัสภายในเป็นรหัสนอนไบนารี เนื่องจาก VSD เป็นตัวถอดรหัสนอนไบนารีที่สามารถประยุกต์ใช้กับรหัสที่มีขนาดใหญ่เพิ่มขึ้นได้ไม่จำกัด การถอดรหัสแบบ VSD ยังเหมาะกับกรณีที่เครื่องรับอาจได้รับข้อมูลที่ถูกส่งมาหลายครั้งหรือหลายชุด ซึ่งอาจเกิดจากการใช้ตัวถอดรหัสภายในที่ให้ผลลัพธ์มากกว่าหนึ่งแบบหรือมีการใช้เครื่องรับแบบที่ได้รับสัญญาณซ้ำทางปริภูมิ(space diversity) เช่นการใช้สายอากาศหลายอันหรือการรับสัญญาณซ้ำทางความถี่(frequency diversity) ในการส่งสัญญาณแบบจุดต่อจุดรวมทั้งการรับแพคเกตซ้ำในเครื่องข่ายสื่อสารข้อมูล วิธีการถอดรหัสนี้จะสามารถใช้ชุดข้อมูลสำรองนี้มาช่วยในการแก้ไขข้อมูลได้มากขึ้นและง่ายลง เนื่องจากสามารถนำตัวเลือกอื่นๆ(หรือชุดข้อมูลสำรอง) ที่ถูกต้องมาแทนลงในตัวเลือกที่หนึ่ง(หรือชุดข้อมูลหลัก) ที่พบว่าผิดพลาดได้

อย่างไรก็ตามการใช้รหัส RS เป็นรหัสภายในตามโครงสร้างที่นำเสนอนั้น ตัวถอดรหัส RS โดยทั่วไปไม่สามารถให้ผลลัพธ์มากกว่าหนึ่งแบบได้ ดังนั้นถ้าไม่มีชุดข้อมูลสำรองจากการใช้เครื่องรับที่ได้รับสัญญาณซ้ำทางปริภูมิหรือทางความถี่ จะตั้งสมมุติฐานว่าไม่มีชุดข้อมูลสำรอง ถึงแม้ว่าจะไม่มีชุดข้อมูลสำรองแต่การที่รหัส RS เป็นรหัสที่มีประสิทธิผลสูงโดยเฉพาะในการเติมเต็มส่วนที่หายไป (erasure) จึงคาดได้ว่าการถอดรหัส VSD ในกรณีนี้ยังจะให้ประสิทธิผลสูง ในการประยุกต์ใช้งานในอนาคตสามารถนำเครื่องรับแบบที่ได้รับสัญญาณซ้ำมาใช้ร่วมกับโครงสร้างรหัสที่เสนอเพื่อประสิทธิผลที่ดียิ่งขึ้นและความซับซ้อนในการถอดรหัส VSD ที่ลดลง

การเข้ารหัสช่องสัญญาณ (channel coding) สามารถตรวจสอบสัญลักษณ์ที่ผิดพลาด (error symbol) ของสัญลักษณ์ที่รับได้ที่ฝั่งรับ (Received symbol) ได้ โดยการคำนวณค่าซินโดรม (syndrome) ซึ่งหนึ่งกลุ่มสัญลักษณ์ที่ได้รับ จะถูกนำไปคำนวณหาค่าซินโดรม เพื่อใช้ในการถอดรหัสได้ 1 ตัว โดยจำนวนสัญลักษณ์ในกลุ่มขึ้นอยู่กับรหัสที่ใช้ สำหรับรหัสคอนโวลูชัน (3, 2, 2) จะมี 3 สัญลักษณ์ในแต่ละกลุ่ม

ค่าของซินโดรมจะสามารถบอกได้ว่าสัญลักษณ์ที่รับได้(received symbols) นั้นมีความผิดพลาด (error) เกิดขึ้นหรือไม่ โดยการหาค่าซินโดรมนี้จะทำสำหรับสัญลักษณ์ที่ได้รับในชุดหลัก (First choice) เท่านั้น จะไม่มีการหาสำหรับชุดข้อมูลสำรอง (Second choice) การคำนวณค่าซินโดรมทำได้ตาม สมการ (8)

$$\mathbf{S} = \mathbf{H}\,\mathbf{Y} = \mathbf{H}\,\mathbf{E} \qquad (8)$$

เมื่อ **S** = Syndrome Matrix, **H** = Parity Check Matrix, **Y** = Received Matrix และ **E** = Error Matrix ซึ่งค่าของเมทริกซ์ **H** ก็ขึ้นกับรหัสที่เลือกใช้

โดยฟังก์ชันหลักของการถอดรหัสเวกเตอร์ซิมโบลคือการถอดรหัสด้วยชุดการรวมเป็นศูนย์ดังนี้

3. การถอดรหัสด้วยชุดการรวมเป็นศูนย์ [18]

การถอดรหัสด้วยชุดการรวมเป็นศูนย์ (Correct with Null Combination) มีหลักการถอดรหัสจะใช้หลักการกรองสัญลักษณ์ที่ถูกต้องออกมา หรือที่อาจเรียกว่าการถอดรหัสโดยการทวนสอบสัญลักษณ์ที่ถูกต้อง (verification based decoding) บางครั้งจึงเรียกขั้นตอนนี้ว่า ขั้นตอนแก้ไขโดยใช้การรวมกันทางพีชคณิตได้ (เวกเตอร์) ศูนย์ เป็นวิธีถอดรหัสที่มีขั้นตอนที่ซับซ้อนกว่าการถอดรหัสแบบใช้ชุดข้อมูลสำรอง หรือชุดสัญลักษณ์อีกชุดที่ได้เครื่องถอดรหัสอาจได้รับมาซึ่งเป็นคำรหัสที่อาจประกอบด้วยบางสัญลักษณ์ที่ผิดพลาด และเป็นชุด

สัญลักษณ์ที่มีความน่าจะเป็นที่จะถูกต้องรองจากชุดข้อมูลหลัก แต่ก็มีความสามารถในแก้ไขความผิดพลาดที่ไม่สามารถแก้ได้เนื่องจากข้อมูลสำรองไม่ถูกต้อง หรือไม่มีข้อมูลสำรองที่ตำแหน่งนั้นๆ

การระบุตำแหน่งของความผิดพลาดเพื่อทำการแก้ไขหลังจากที่พบว่าเกิดความผิดพลาดขึ้นมีวิธีการดังนี้

ให้ $\mathbf{s}_i$ และ $\mathbf{s}_j$ เป็นเวกเตอร์ของซินโดรมย่อยที่อยู่ใน **S** และมีค่าไม่เป็น 0 โดย $i \neq j$ สมมติให้ $\mathbf{s}_i + \mathbf{s}_j = 0$ ซึ่งจะเรียกว่าชุดการรวมเป็นศูนย์ (Null Combination) จะได้ว่า

$$s_i + s_j = (h_i + h_j) \bullet E \qquad (9)$$

เมื่อชุดคำรหัสที่ใช้ในการเข้ารหัสเป็นอิสระเชิงเส้นแล้ว $\mathbf{h}_i + \mathbf{h}_j \neq 0$ เสมอเมื่อ $i \neq j$ ดังนั้นสามารถสรุปได้ว่าตำแหน่งที่ผิดพลาดคือตำแหน่งบิตที่รวมกันแล้วไม่เป็น 0

ขั้นตอนวิธีการถอดรหัสด้วยวิธีเวกเตอร์ซิมโบลด้วยชุดการรวมเป็นศูนย์

ขั้นตอนการถอดรหัสมีรายละเอียดต่อไปนี้

1. ตัวถอดรหัสคำนวณค่าซินโดรมโดยสำหรับรหัสคอนโวลูชัน (n,k,m) แต่ละกลุ่มของ n สัญลักษณ์ที่เข้ามาจะได้ค่าซินโดรมหนึ่งเวกเตอร์
2. ถ้าค่าซินโดรมแรกที่คำนวณได้มีค่าเท่ากับเวกเตอร์ศูนย์ตัวถอดรหัสจะถือว่าสัญลักษณ์ n ตัวแรกถูกต้องและไม่ต้องทำการแก้ไขสามารถส่งออกไปเป็นส่วนหนึ่งของคำรหัสที่ถูกต้องได้เลย
3. จากนั้นจะคำนวณค่าซินโดรมของสัญลักษณ์ n ตัวถัดไปโดยนำข้อมูลของสัญลักษณ์ที่จำเป็นก่อนหน้ามาช่วยคำนวณด้วยถ้าได้ค่าเป็นศูนย์อีกก็จะทำการคำนวณสัญลักษณ์กลุ่มถัดไปเรื่อยๆจนหมดสัญลักษณ์ที่ได้รับ
4. ถ้าค่าซินโดรมใดมีค่าไม่เท่ากับศูนย์แสดงว่ามีความผิดพลาดเกิดขึ้นในสัญลักษณ์กลุ่มนั้น ตัวถอดรหัสจะทำการคำนวณซินโดรมของสัญลักษณ์กลุ่มต่อไปเพื่อนำมาช่วยในการแก้ไขความผิดพลาดจำนวนซินโดรมที่จำเป็นต้องใช้ขึ้นกับตำแหน่งและรูปแบบของความผิดพลาดที่เกิดขึ้นโดยความผิดพลาดที่เกิดขึ้นในสัญลักษณ์ที่ติดกันจะแก้ไขยากกว่าความผิดพลาดในสัญลักษณ์ที่อยู่ห่างกัน
5. ตัวถอดรหัสจะใช้วิธี Correct with Null Combination โดยจะทำการคำนวณหาชุดของเวกเตอร์ซินโดรมที่บวกกันแบบมอดูโล-2 (XOR) ได้เท่ากับเวกเตอร์ศูนย์ ซึ่งเรียกว่า null indicator จากค่า null indicator ตัวถอดรหัสจะหาค่า null combination จากการนำแถวของเมทริกซ์ตรวจสอบภาวะคู่คี่ (H) ที่สัมพันธ์กันมา (XOR) กัน ซึ่ง null combination แต่ละตัวจะแสดงสัญลักษณ์ที่ถูกทวนสอบว่าถูกต้องในรูปของบิต “1” และสัญลักษณ์ที่ไม่สามารถทวนสอบในรูปบิต “0”
6. ตัวถอดรหัสจะทำการหาค่าเวกเตอร์แสดงตำแหน่งสัญลักษณ์ที่ผิดพลาด (error locating vector) โดยการนำ null combination ทั้งหมดมา OR กัน เนื่องจากสัญลักษณ์ที่ถูกทวนสอบว่าถูกต้องโดยอย่างน้อย null combination ตัวหนึ่งแล้ว ตัวถอดรหัสจะถือว่าไม่มีความผิดพลาด
7. เมื่อทราบตำแหน่งความผิดพลาดแล้วตัวถอดรหัสสามารถคำนวณหาค่าความผิดพลาดได้จากการแก้สมการหาค่า **E** ตามสมการที่ (10)

$$S_{sub} = H_{sub} \bullet E \qquad (10)$$

เมื่อ $\mathbf{S}_{sub}$ = เมทริกซ์ซินโดรมโดยเลือกเฉพาะแถวที่เป็นอิสระเชิงเส้นต่อกัน

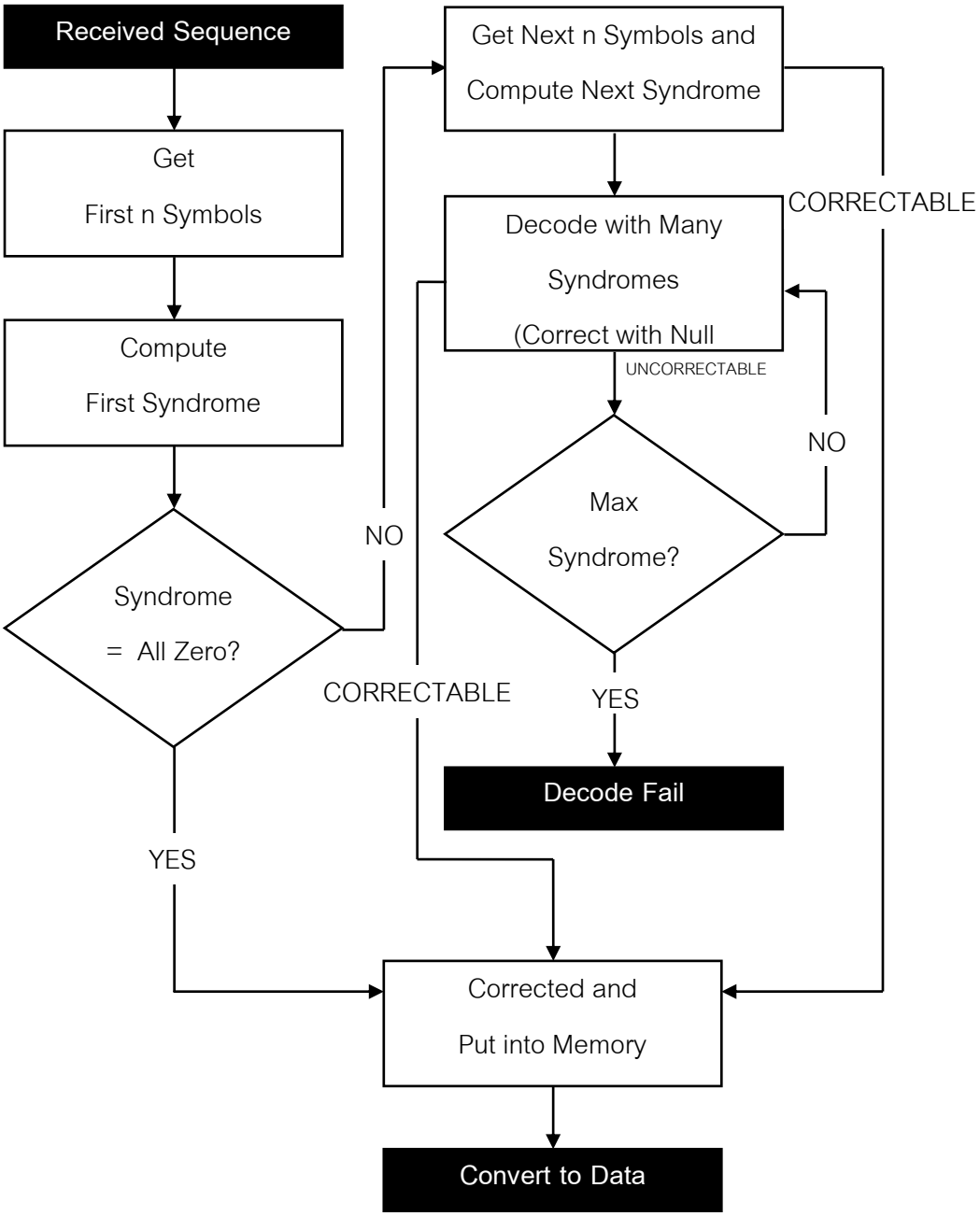
$\mathbf{H}_{sub}$ = เมทริกซ์ตรวจสอบภาวะคู่คี่โดยเลือกแถวที่สัมพันธ์กับแถวของ $\mathbf{S}_{sub}$ และคอลัมน์ที่สัมพันธ์กับตำแหน่งของสัญลักษณ์ผิดพลาด

8. เมื่อได้ทั้งตำแหน่งและค่าความผิดพลาดแล้วตัวถอดรหัสก็นำค่าความผิดพลาดที่คำนวณได้ไปลบ (XOR) ออกจากตำแหน่งที่ผิด เมื่อแก้ไขแล้วกลุ่มข้อมูลทั้งหมดที่เกี่ยวข้องก็จะสามารถถูกส่งออกไปได้เลย

9. ตัวถอดรหัสทำการคำนวณค่าซินโดรมสำหรับกลุ่มสัญลักษณ์ถัดไปและดำเนินการตามข้อ 3 หรือข้อ 4 เช่นนี้เรื่อยๆจนได้รับข้อมูลครบ

10. ถ้าตัวถอดรหัสไม่สามารถแก้ไขข้อมูลที่ได้รับมาได้โดยค่าซินโดรมที่ใช้เกินจำนวนที่กำหนดไว้ตัวถอดรหัสจะรายงานว่า ข้อมูลทีความผิดพลาดแต่ไม่สามารถแก้ไขได้

โดยขั้นตอนวิธีการถอดรหัสเวกเตอร์ซิมโบลแบบไม่มีชุดข้อมูลสำรองข้างต้นสามารถเขียนเป็นแผนภาพแสดงขั้นตอนการถอดรหัสได้ดังแสดงในรูปที่ 15



รูปที่ 15 แผนผังขั้นตอนการถอดรหัสเวกเตอร์ซิมโบลด้วยชุดการรวมเป็นศูนย์

4. รหัส Reed-Solomon ซึ่งถูกเลือกใช้เป็นรหัสภายใน [18]

รหัสนี้คิดค้นโดย Irving Reed และ Gustave Solomon ในปีค.ศ. 1960 [22] จัดอยู่ในกลุ่มรหัสวน (Cyclic code) รหัสรีดโซโลมอนนับว่าเป็นรหัสบีซีเอชที่เป็นแบบนอนไบนารี ซึ่งมีคุณสมบัติพิเศษ คือ เป็นรหัสที่

อยู่ในกลุ่มที่มีระยะแฮมมิงต่ำสุด (d_{min}) สูงสุดเท่าที่จะเป็นไปได้ ทำให้มีประสิทธิภาพในการแก้ไขความผิดพลาดสูง สามารถแก้ไขได้ทั้งความผิดพลาดแบบสุ่มและความผิดพลาดแบบเบริสต์ขนาดสั้นๆได้ โครงสร้างของรหัสรีดโซโลมอน จะเขียนในรูปของ (n,k) หรือ (n,k,t) โดย

$n = 2^m - 1$ คือ ความยาวของคำรหัส มีหน่วยเป็นสัญลักษณ์
k คือ ข้อมูลทั้งหมด มีหน่วยเป็นสัญลักษณ์
m คือ ขนาดของแต่ละสัญลักษณ์ มีหน่วยเป็นบิต
t คือ จำนวนความผิดพลาดที่สามารถแก้ไขได้ มีหน่วยเป็นสัญลักษณ์

โดยมี d_{min} = n-k+1 ซึ่งทำให้สามารถแก้ไขความผิดพลาดทั้งหมดโดยทั่วไปได้ $t = (n-k)/2$ สัญลักษณ์

นอกจากนี้ รหัสรีดโซโลมอนยังเป็นรหัสที่เหมาะสมที่สุด (Optimal code) [23] ในการเติมเต็มสัญลักษณ์ที่ถูกลบ ซึ่งเราจะทราบตำแหน่งของสัญลักษณ์ที่ถูกลบ แต่ไม่ทราบขนาดของข้อมูลและมักจะถูกเขียนแทนที่ด้วย 0 โดยความสามารถในการแก้ไขสัญลักษณ์ที่ผิดพลาดและสัญลักษณ์ที่ถูกลบ เป็นไปตามสมการ

$$e + \frac{s}{2} \le \left(\frac{n-k}{2}\right)$$

หรือ

$$2e + s \le n - k \qquad (11)$$

เมื่อ e คือจำนวนสัญลักษณ์ที่ผิดพลาด
s คือจำนวนสัญลักษณ์ที่ถูกลบ

จากสมการที่ (1) จะพบว่าเมื่อพิจารณาเฉพาะสัญลักษณ์ที่ถูกลบ (e=0) รหัสจะสามารถแก้ไขความผิดพลาดได้มากเป็น 2 เท่า ของรหัสที่พิจารณาเฉพาะสัญลักษณ์ที่ผิดพลาด (s=0)

4.1 การเข้ารหัสรีดโซโลมอน [18]

เนื่องจากรหัสรีดโซโลมอนเป็นรหัสวน เราจึงสามารถสร้างรหัสที่เป็นระบบได้โดยใช้วิธีการเดียวกับรหัสวนทั่วๆไป วิธีการเข้ารหัสรีดโซโลมอนที่ง่ายที่สุด เริ่มต้นจากการสร้างพหุนามตัวกำเนิด โดยสมการพหุนามตัวกำเนิดของรหัสรีดโซโลมอน จะสามารถเขียนอยู่ในรูปทั่วไปได้เป็น

พหุนามตัวกำเนิด $$g(x) = \prod_{i=b}^{b+2t-1} (x - \alpha^i) \qquad (12)$$

โดย α^i เป็นสมาชิกในฟิลด์กาลัว (Galois field) $GF(q)$

ปกติจะให้ b=1 เรียกว่า แนโรว์เซนส์ (narrow sense) แต่บางกรณีจะกำหนดให้ไม่เท่ากับ 1 ก็ได้ จากนั้นจึงนำพหุนามตัวกำเนิดและชุดข้อมูลมาสร้างเป็นคำรหัส โดยคำรหัสจะอยู่ในรูป

$$c(x) = x^{(n-k)} m(x) + p(x) \qquad (13)$$

เมื่อ $c(x)$ คือ คำรหัสที่ได้
$m(x)$ คือ ชุดข้อมูล
และ $p(x) = x^{(n-k)} m(x) \mod g(x)$ หรือคือ เศษที่ได้จากการหาร $x^{(n-k)} m(x)$ ด้วย $g(x)$

4.2 การถอดรหัสรีดโซโลมอน

รหัสรีดโซโลมอน สามารถถอดรหัสได้หลายวิธี โดยนิยามแล้ว ตัวถอดรหัสจะพยายามค้นหาคำรหัส โดยใช้คำที่ได้รับมาเป็นศูนย์กลาง และมีรัศมีในการค้นหาเท่ากับ t โดยในการถอดรหัสซึ่งมีการตัดสินใจแบบฮาร์ด (Hard decision) นั้น หากการค้นหานั้นพบคำรหัส ตัวถอดรหัสจะใช้คำรหัสนั้นเป็นคำตอบ หรือถ้าไม่พบก็จะส่งข้อผิดพลาดกลับมา และเนื่องจากระยะห่างระหว่างแต่ละคำรหัสมีค่าเท่ากับ $2t+1$ ในการค้นหานั้น จะพบคำรหัสที่เป็นคำตอบเพียงค่าเดียวเสมอ

วิธีการถอดรหัสแบบ Berlekamp-Massey ถูกนำเสนอโดย Elwyn Berlekamp [13] ซึ่งคิดค้นวิธีการถอดรหัสบีซีเอช ต่อมาภายหลัง James Massey [12] พบว่าวิธีการดังกล่าวคือการทำชิฟรีจิสเตอร์ป้อนกลับแบบเชิงเส้น (Linear Feedback Shift Registers: LFSR) และได้ปรับให้วิธีการง่ายลงและเป็นที่รู้จักในชื่อวิธีการ Berlekamp-Massey งานวิจัยนี้ได้ใช้โปรแกรม MATLAB ในการจำลองการเข้ารหัสและถอดรหัสรีดโซโลมอน ซึ่งในส่วนของการถอดรหัสนั้น โปรแกรม MATLAB ก็ได้ใช้วิธีการของ Berlekamp-Massey ซึ่งจะประกอบด้วย 4 ขั้นตอนโดยสรุป ดังนี้ [19]

1) คำนวณหาค่าซินโดรม

การหาค่าซินโดรมจะถูกใช้เป็นขั้นตอนแรกในการตรวจสอบความถูกต้องของข้อมูลที่ได้รับ โดยดูจากผลการคำนวณหาค่าซินโดรม หากค่าซินโดรมที่ได้มีค่าเท่ากับ 0 ทั้งหมด ตัวถอดรหัสจะเข้าใจว่าข้อมูลที่ได้รับมาถูกต้อง แต่หากค่าซินโดรมมีค่าใดค่าหนึ่งหรือทั้งหมดไม่เป็น 0 ก็จะแสดงว่าเกิดความผิดพลาดขึ้นกับข้อมูลที่ได้รับ

เนื่องจากคำรหัสเกิดจากพหุนามตัวกำเนิดซึ่งมี α^i ซึ่งเป็นรากของพหุนาม ดังนั้น α^i จึงเป็นรากของคำรหัสด้วย เมื่อนำ α^i ไปทดสอบกับพหุนามคำรหัสจะต้องได้ผลเป็นศูนย์ โดยซินโดรมที่ได้เกิดจาก

$$s_i = r(\alpha^i) = e(\alpha^i) \qquad (14)$$

โดย s_i คือ ค่าซินโดรมตัวที่ i โดย i = 1,2,3,...,2v

และ v คือจำนวนความสามารถในการแก้ไขความผิดพลาดของรหัส

$r(\alpha^i)$ และ $e(\alpha^i)$ คือ พหุนามสัญลักษณ์ที่ได้รับและพหุนามความผิดพลาดตามลำดับ และ $\alpha^i \in GF(q)$

ซินโดรมทั้งหมดจะมีจำนวน $2v$ สัญลักษณ์ หากมีตัวใดมีค่าไม่เท่ากับศูนย์ ก็จะเข้าสู่กระบวนการแก้ไขความผิดพลาดต่อไป

2) แก้สมการหาพหุนามระบุตำแหน่งความผิดพลาด $\sigma(x)$

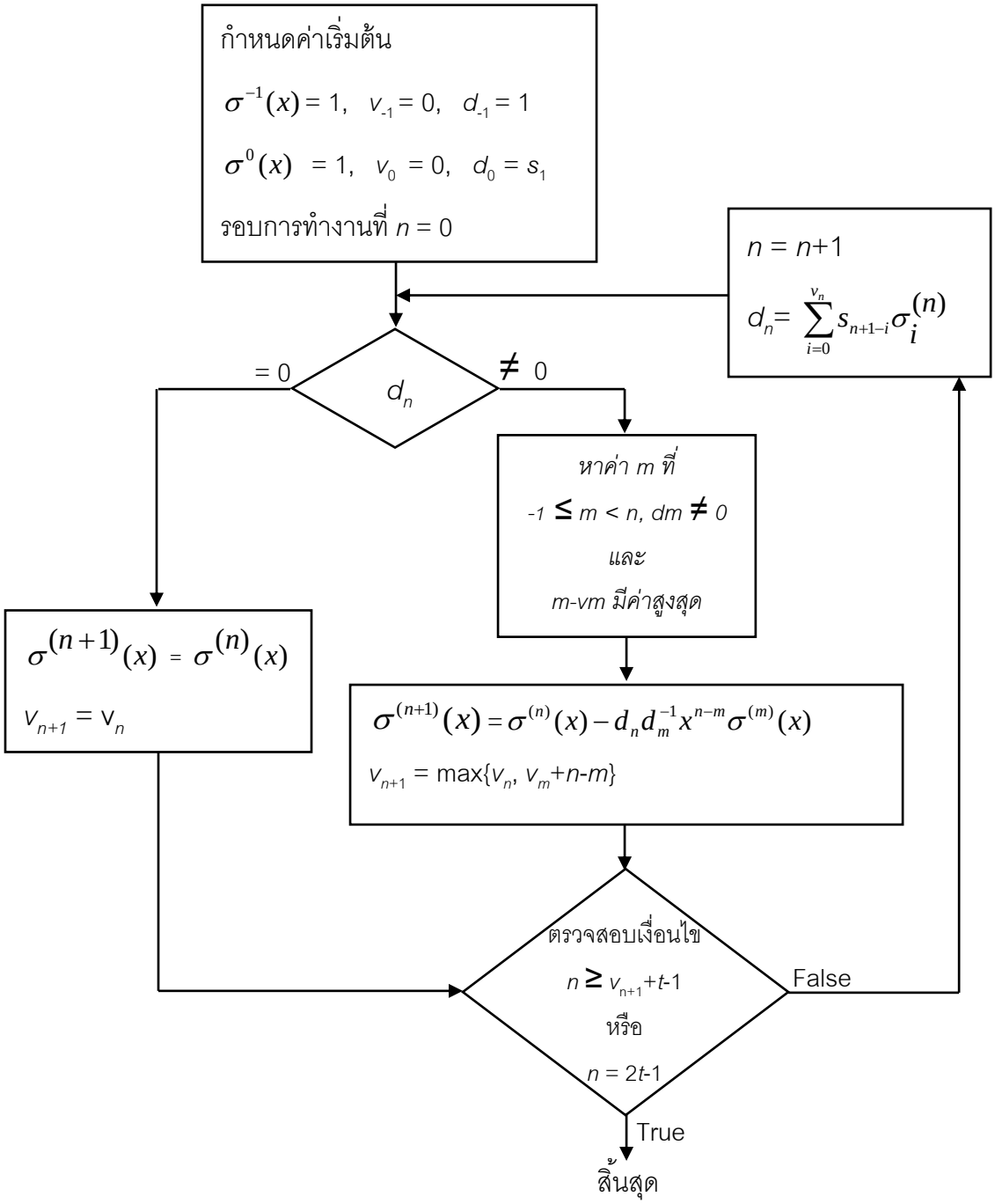
ในการหาตำแหน่งของความผิดพลาด มีผู้คิดค้นวิธีการต่างๆเป็นจำนวนมากวิธีการ สำหรับวิธีการ Berlekamp-Massey สามารถจัดรูปการแก้สมการซินโดรมได้เป็น

$$s_i = -\sigma_1 s_{i-1} - \sigma_2 s_{i-2} - \ldots - \sigma_v s_{i-v} \ , \ v+1 \le i \le 2t \qquad (15)$$

ซึ่งก็คือสมการของชิฟรีจิสเตอร์ที่มี $\sigma(x)$ เป็นสัมประสิทธิ์การป้อนกลับ ซึ่งสุดท้ายสิ่งที่ต้องการ คือ รีจิสเตอร์ที่มีความยาวต่ำสุดพร้อมสัมประสิทธิ์ที่ยังคงทำให้สมการเป็นจริง โดยอาศัยการวนทำซ้ำจำนวนไม่เกิน $2t$ รอบ และในทุกๆรอบจะมีการคำนวณเพื่อปรับปรุงค่าก่อนทำซ้ำในรอบต่อๆไป เมื่อวนจนครบจำนวนก็จะได้คำตอบเป็นพหุนามระบุตำแหน่งความผิดพลาด โดยวิธีการของ Berlekamp-Massey สามารถเขียนเป็นผังขั้นตอนได้ตามรูปที่ 16 [18]

3) หาตำแหน่งของความผิดพลาด โดยในขั้นตอนนี้ค่อนข้างเป็นวิธีการที่ตายตัว แม้ว่าในขั้นตอนการหาพหุนามระบุตำแหน่งความผิดพลาดจะมีวิธีที่สามารถเลือกใช้ได้หลายวิธี แต่ตั้งแต่ในขั้นตอนนี้มักจะใช้วิธีลักษณะเดียวกัน โดยการหารากทั้ง v ตัวของพหุนาม $\sigma(x)$ ซึ่งส่วนกลับของรากที่ได้จะเรียกว่าค่าระบุตำแหน่งของความ

ผิดพลาด (β_λ) และจะสามารถคำนวณตำแหน่งของความผิดพลาดได้โดยใช้ความสัมพันธ์ $\beta_\lambda = \alpha^{j_\lambda}$ เมื่อ $1 \le \lambda \le v$ และ j_λ เป็นตำแหน่งที่เกิดความผิดพลาดของสัญลักษณ์ที่ได้รับ



รูปที่ 16 ผังขั้นตอนของวิธีการ Berlekamp-Massey

4) หาค่าความผิดพลาดและแก้ไข โดยใช้วิธีของ Forney เช่นเดียวกับขั้นตอนที่ 3 ที่ขั้นตอนนี้มีวิธีที่ค่อนข้างตายตัว โดยจะสามารถหาค่าความผิดพลาดได้จากสมการที่ (16) และ (17)

$$\Omega(x) = \sigma(x)\left[1 + S(x)\right] \qquad (16)$$

และ

$$e_{j_\lambda} = \frac{\Omega(\beta_\lambda^{-1})}{\prod_{i \neq \lambda}(1 - \beta_i \beta_\lambda^{-1})} \quad (17)$$

โดย $S(x)$ คือ พหุนามซินโดรม

และ e_{j_λ} คือ ค่าความผิดพลาดที่ตำแหน่ง j_λ

จากนั้น เมื่อนำค่าความผิดพลาดที่คำนวณได้ไปรวมกับข้อมูลที่ได้รับ ก็จะได้เป็นคำรหัสที่ถูกต้องซึ่งก็คือการแก้ไขความผิดพลาด

4.3 การถอดรหัส Reed-Solomon เมื่อใช้ร่วมกับสัญลักษณ์ที่ถูกลบ

แนวคิดในการนำรหัสที่ถูกลบมาใช้ในการเพิ่มประสิทธิภาพการถอดรหัส เนื่องจากในการแก้ไขสัญลักษณ์ที่ถูกลบนั้น สามารถที่จะแก้ไขจำนวนสัญลักษณ์ได้มากกว่า แต่ส่วนสำคัญในการใช้สัญลักษณ์ที่ถูกลบ คือ การระบุตำแหน่งของสัญลักษณ์ที่ถูกลบ ตั้งแต่ในขั้นตอนของการแยกสัญญาณ (Demodulate) ระหว่างการดีมอดูเลต หากพบสัญญาณที่ระบุสัญลักษณ์ที่ถูกลบ ในกรณีทั่วไปตัวแยกสัญญาณจะแทนที่สัญลักษณ์นั้นด้วย 0 และสร้างข้อมูลใหม่ที่เรียกว่าเวกเตอร์สัญลักษณ์ที่ถูกลบคู่ไปด้วยกัน เพื่อใช้เป็นข้อมูลให้กับตัวถอดรหัสเพื่อระบุตำแหน่งของสัญลักษณ์ที่ถูกลบ ซึ่งหากสามารถระบุตำแหน่งได้อย่างถูกต้องหรือเป็นตำแหน่งเดียวกันกับตำแหน่งที่เกิดความผิดพลาดในสัญลักษณ์ จะเป็นการเพิ่มจำนวนความผิดพลาดที่สามารถแก้ไขได้ แต่ในทางกลับกันหากตำแหน่งที่ระบุไม่ถูกต้อง ก็จะลดจำนวนสัญลักษณ์ที่สามารถแก้ไขได้ซึ่งส่งผลให้ประสิทธิภาพในการถอดรหัสลดลงได้

พิจารณารหัสรีด-โซโลมอน(7,3)

กรณีที่ 1 การใช้ตัวถอดรหัสที่แก้ไขความผิดพลาดเพียงอย่างเดียว จากสมการ (1) แสดงให้เห็นว่ารหัสนี้จะสามารถแก้ไขความผิดพลาดได้สูงสุด 2 สัญลักษณ์

กรณีที่ 2 การใช้ตัวถอดรหัสที่เติมสัญลักษณ์ที่ถูกลบเพียงอย่างเดียว จากสมการ (1) รหัสนี้จะสามารถเติมสัญลักษณ์ที่ถูกลบได้สูงสุดถึง 4 ตำแหน่ง ซึ่งหากตำแหน่งของสัญลักษณ์ที่ถูกลบเป็นตำแหน่งเดียวกันกับตำแหน่งของสัญลักษณ์ที่ผิดพลาด จะเป็นการเพิ่มประสิทธิภาพของตัวถอดรหัสได้

กรณีที่ 3 การใช้ตัวถอดรหัสที่แก้ไขความผิดพลาดร่วมกับสัญลักษณ์ที่ถูกลบ โดยสามารถระบุตำแหน่งของสัญลักษณ์ที่ถูกลบได้อย่างถูกต้อง โดยในกรณีนี้ หากสามารถระบุตำแหน่งของสัญลักษณ์ที่ถูกลบให้เป็นตำแหน่งเดียวกันกับตำแหน่งของสัญลักษณ์ที่ผิดพลาดได้อย่างถูกต้อง 1-2 ตำแหน่ง ตัวถอดรหัสจะยังคงเหลือความสามารถที่จะแก้ไขความผิดพลาดที่ไม่รู้ตำแหน่งได้อีก 1 สัญลักษณ์ ซึ่งก็เป็นการเพิ่มประสิทธิภาพในการถอดรหัสได้เช่นกัน

กรณีที่ 4 การใช้ตัวถอดรหัสที่แก้ไขความผิดพลาดร่วมกับสัญลักษณ์ที่ถูกลบ โดยระบุตำแหน่งของสัญลักษณ์ที่ถูกลบผิดพลาด ในกรณีนี้ หากทำการระบุตำแหน่งสัญลักษณ์ที่ถูกลบผิดพลาด 1 ตำแหน่ง ตัวถอดรหัสจะเหลือความสามารถที่จะแก้ไขความผิดพลาดได้อีกเพียง 1 สัญลักษณ์ หากรหัสนี้เกิดความผิดพลาด 2 สัญลักษณ์ ตัวถอดรหัสจะไม่สามารถแก้ไขความผิดพลาดได้ แม้ว่าโดยปกติจะเป็นจำนวนความผิดพลาดที่รหัสนี้สามารถแก้ไขความผิดพลาดได้ก็ตาม

ซึ่งการเกิดลักษณะตามกรณีที่ 4 จะเป็นจุดที่ทำให้ความสามารถในการแก้ไขความผิดพลาดลดลง การใช้ประโยชน์จากตัวถอดรหัสรีดโซโลมอนที่แก้ไขความผิดพลาดร่วมกับสัญลักษณ์ที่ถูกลบ จึงเน้นไปที่วิธีการระบุ

ตำแหน่งของสัญลักษณ์ที่ถูกลบให้มีความถูกต้องแม่นยำเป็นสำคัญ วิธีการที่ง่ายที่สุดในการระบุตำแหน่งของสัญลักษณ์ที่ถูกลบ คือ การใช้ค่าขีดแบ่ง (threshold) ในการตัดสินใจ แต่การกำหนดค่าขีดแบ่งที่เหมาะสมนั้นจะต้องใช้การทดสอบและบันทึกผล เนื่องจากค่าขีดแบ่งที่เหมาะสมจะแตกต่างกันตามคุณสมบัติของช่องสัญญาณ [21,24]

4.3.1 การระบุตำแหน่งสัญลักษณ์ที่ถูกลบ

ในการระบุตำแหน่งของสัญลักษณ์ที่ถูกลบมีวิธีอย่างง่าย คือ การกำหนดค่าขีดแบ่ง โดยทั่วไปแล้วสัญญาณที่ฝั่งรับจะมีลักษณะ

$$\tilde{y} = \alpha \tilde{x} + n \quad (18)$$

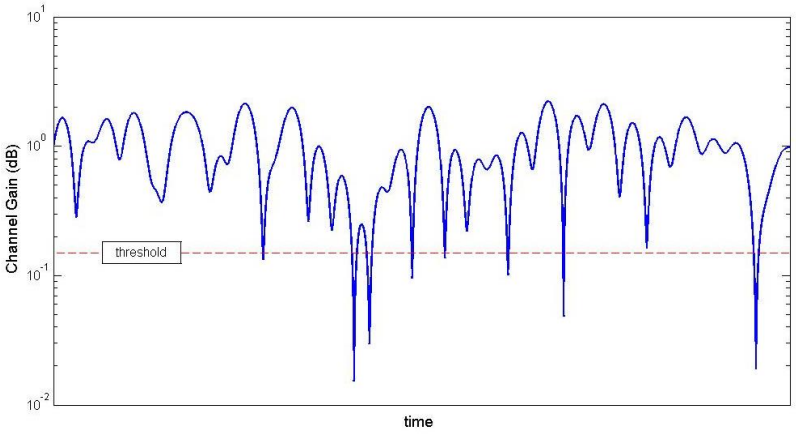
$\tilde{y}$ คือ สัญญาณที่ฝั่งรับ

$\tilde{x}$ คือ สัญญาณส่ง

α คือ กำลังขยายช่องสัญญาณ

n คือ สัญญาณรบกวน

ซึ่งในการใช้ค่าขีดแบ่ง คือ เมื่อกำลังขยายช่องสัญญาณ α มีค่าต่ำกว่าค่าขีดแบ่งที่กำหนด ตำแหน่งของสัญลักษณ์นั้นจะถูกระบุเป็นสัญลักษณ์ที่ถูกลบ โดยตั้งสมมติฐานว่า เมื่อกำลังขยายช่องสัญญาณมีขนาดน้อยๆหรือเกิดการจางหาย จะมีโอกาสที่เกิดความผิดพลาดที่ตำแหน่งนั้นๆได้มาก การระบุตำแหน่งของสัญลักษณ์ที่ถูกลบที่ตำแหน่งนั้น จึงมีความเป็นไปได้สูงที่จะเป็นตำแหน่งเดียวกันกับสัญลักษณ์ที่เกิดความผิดพลาด โดยในทางปฏิบัติค่าขีดแบ่งที่เหมาะสมจะขึ้นกับช่องสัญญาณ, ค่าเอสเอ็นอาร์ (Signal to noise Ratio: SNR) รวมทั้งวิธีการกล้ำสัญญาณ (modulate)



รูปที่ 17 การใช้ค่าขีดแบ่งเพื่อระบุตำแหน่งสัญลักษณ์ที่ถูกลบ

4.3.2 การปรับปรุงวิธีการถอดรหัสเมื่อใช้ร่วมกับสัญลักษณ์ที่ถูกลบ [19]

สำหรับการปรับปรุงวิธี Berlekamp-Massey เพื่อให้สามารถแก้ไขได้ทั้งสัญลักษณ์ที่ผิดพลาดและสัญลักษณ์ที่ถูกลบได้ด้วยนั้น จะมีขั้นตอนโดยย่อ ดังนี้

1) คำนวณหาค่าซินโดรม

ในขั้นตอนนี้มีความแตกต่างจากปกติเล็กน้อย โดยตำแหน่งของสัญลักษณ์ที่ได้รับซึ่งถูกระบุเป็นสัญลักษณ์ที่ถูกลบจะถูกแทนที่ด้วย 0 ก่อน แล้วจึงจะทำการคำนวณซินโดรมเหมือนปกติ

2) คำนวณพหุนามระบุตำแหน่งสัญลักษณ์ที่ถูกลบ Ψ (erasure locator polynomial)

เป็นขั้นตอนที่เพิ่มขึ้นมาใหม่ โดยจะคำนวณได้จากสมการ (19)

$$\Psi(x) = \prod_{l=1}^{s} (1 - \alpha^{i_l} x) \tag{19}$$

เมื่อ s คือ จำนวนของสัญลักษณ์ที่ถูกลบ

และ i_l คือตำแหน่งของสัญลักษณ์ที่ถูกลบ

3) หาค่าซินโดรมใหม่

$$[1 + S'(x)] = [1 + S(x)]\,\Psi(x) \bmod x^{2t+1} \tag{20}$$

โดย $S'(x)$ คือ พหุนามซินโดรมใหม่

4) หาค่าพหุนามระบุตำแหน่งสัญลักษณ์ผิดพลาดและสัญลักษณ์ที่ถูกลบ (error-erasure location polynomial) $\bar{\sigma}(x)$

ขั้นตอนนี้เป็นการรวมขั้นตอนที่ 2 และ 3 ในวิธี Berlekamp-Massey ตามปกติมารวมกัน โดยจะใช้ค่าซินโดรมใหม่ที่ได้ในขั้นตอนที่ 1) ในการคำนวณพหุนามระบุตำแหน่งความผิดพลาด $\sigma(x)$ แล้วจึงนำมาหารากซึ่งก็คือการหาตำแหน่งของความผิดพลาด (หาเพียงตำแหน่งของความผิดพลาดก็เพียงพอ เนื่องจากเราทราบตำแหน่งของสัญลักษณ์ที่ถูกลบอยู่ก่อนแล้ว) จากนั้นพหุนามระบุตำแหน่งสัญลักษณ์ผิดพลาดและสัญลักษณ์ที่ถูกลบ $\bar{\sigma}(x)$ สามารถคำนวณได้จากสมการ (21)

$$\bar{\sigma}(x) = \sigma(x)\Psi(x) \tag{21}$$

โดย $\bar{\sigma}(x)$ คือ พหุนามระบุตำแหน่งสัญลักษณ์ผิดพลาดและสัญลักษณ์ที่ถูกลบ

5) หาค่าความผิดพลาดและแก้ไข

เช่นเดียวกับขั้นตอนปกติ ซึ่งใช้วิธีของ Forney ในการหาค่าความผิดพลาด โดยมีความแตกต่างคือ จะใช้พหุนามซินโดรมใหม่หรือพหุนามระบุตำแหน่งสัญลักษณ์ผิดพลาดและสัญลักษณ์ที่ถูกลบในการคำนวณแทน

$$\Omega(x) = \bar{\sigma}(x)\left[1 + S(x)\right] = \sigma(x)\left[1 + S'(x)\right] \tag{22}$$

และ

$$e_{j_\lambda} = \frac{\Omega(\beta_\lambda^{-1})}{\prod_{i \neq \lambda} (1 - \beta_i \beta_\lambda^{-1})} \tag{23}$$

5. ช่องสัญญาณ

5.1 ช่องสัญญาณเกาท์เซียนขาวแบบบวก (AWGN)

เป็นช่องสัญญาณพื้นฐานที่นิยมใช้ในการจำลองช่องสัญญาณในงานวิจัย เกิดจากการบวกเชิงเส้นของสัญญาณรบกวนขาว (White Noise) ที่มีความหนาแน่นของสเปกทรัล (Spectral) คงที่ กับการแจงแจงแบบเกาท์

เซียน (Gaussian Distribution) ของแอมพลิจูด (Amplitude) ซึ่งสัญญาณรบกวนเกาท์เซียนขาวแบบบวกสามารถสร้างจากฟังก์ชันความหนาแน่นของความน่าจะเป็น (Probability Density Function: PDF) ที่มีค่าเฉลี่ย (Mean) เป็นศูนย์ ได้ดังนี้

$$p(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{x^2}{2\sigma^2}} \quad (24)$$

โดยที่ σ^2 คือค่าความแปลปรวน (Variance), σ คือค่าเบี่ยงเบนมาตราฐาน และ x คือค่าตัวแปรสุ่ม (Random Variable)

5.2 ช่องสัญญาณที่มีการจางหาย (Fading Channel) [25]

ช่องสัญญาณที่มีการจางหายสามารถแบ่งได้หลายแบบเช่น

1. ช่องสัญญาณที่มีการจางหายแบบช้า (slow fading) และการจางหายแบบเร็ว (fast fading)
2. ช่องสัญญาณที่มีการจางหายแบบราบ (flat fading) และการจายหายแบบเลือกความถี่ (frequency-selective fading)
3. ช่องสัญญาณที่มีการจางหายมัลติพาท (multipath fading หรือ small-scale fading) และแบบเกิดเงา (shadowing หรือ large-scale fading)

โดยรูปแบบการจำลองช่องสัญญาณการจางหายที่นิยมใช้คือการจางหายแบบเรย์ลี (Rayleigh Fading) และการจางหายแบบไรเซียน (Rician Fading)

5.3 ช่องสัญญาณเรย์ลี

ช่องสัญญาณเรย์ลีเป็นรูปแบบของการกระจายสัญญาณในพื้นที่ที่มีสิ่งกีดขวางจำนวนมาก เช่น ในเขตเมือง ทำให้ไม่มีสัญญาณหลักที่มีกำลังมากกว่าสัญญาณที่กระจัดกระจายอื่นๆ หรือไม่มีเส้นทางที่ปราศจากสิ่งกีดขวาง (Line of Sight: LOS) ระหว่างตัวส่งและตัวรับ ฟังก์ชันความหนาแน่นของความน่าจะเป็นจะเท่ากับ

$$p(x) = \frac{x}{\sigma^2}e^{-\frac{x^2}{2\sigma^2}} \quad (25)$$

ช่องสัญญาณเรย์ลีเป็นช่องสัญญาณที่มีคุณภาพต่ำ เนื่องจากมีความน่าจะเป็นสูงที่กำลังขยายช่องสัญญาณจะมีค่าน้อย

5.4 ช่องสัญญาณไรเซียน

ช่องสัญญาณไรเซียนเป็นแบบของการกระจายสัญญาณ ซึ่งมีเส้นทางหนึ่งที่สัญญาณมีกำลังมากกว่าสัญญาณอื่นๆที่กระจัดกระจายออกไป หรือมีเส้นทางที่ปราศจากสิ่งกีดขวางระหว่างตัวส่งและตัวรับ ฟังก์ชันความหนาแน่นของความน่าจะเป็นจะเท่ากับ

$$p(x) = \frac{x}{\sigma^2}e^{-\frac{(x^2+v^2)}{2\sigma^2}}I_0\left(\frac{xv}{\sigma^2}\right) \quad (26)$$

โดยมี

I_0 คือ ฟังก์ชันเบสเซล (Bessel function)

v คือ สัดส่วนของกำลังสัญญาณเทียบที่สัญญาณกระจัดกระจาย

สำหรับช่องสัญญาณไรเซียนเมื่อสัญญาณหลักมีค่ามากกว่าสัญญาณอื่นมากๆ(vมากขึ้น) กำลังขยายช่องสัญญาณก็จะมีค่ามากขึ้นไปด้วย

5.5 ปรากฏการณ์ดอปเพลอร์ (Doppler effect) [25]

ปรากฏการณ์ดอปเพลอร์เกิดจากการเคลื่อนที่ของตัวส่งสัญญาณหรือตัวรับสัญญาณ หรือทั้งสองอุปกรณ์ ซึ่งจะทำให้มุมของสัญญาณที่มาถึงเปลี่ยน และทำให้คลื่นสัญญาณที่มีถึงมีความถี่เปลี่ยนไป โดยค่าความถี่ดอปเพลอร์สามารถหาได้จาก

$$f_d = \frac{v}{c_0} f_0 \cos\alpha \qquad (27)$$

เมื่อ f_d คือความถี่ดอปเพลอร์

v คือความเร็วสัมพัทธ์ระหว่างเครื่องรับกับเครื่องส่ง (เมตรต่อวินาที)

c_0 คือความเร็วแสง มีค่าเท่ากับ 3×10^8 (เมตรต่อวินาที)

f_0 คือความถี่คลื่นพาห์

α คือมุมของสัญญาณที่มาถึง

ค่าความถี่ดอปเพลอร์จะมีค่ามากที่สุดเมื่อ α มีค่าเท่ากับศูนย์

6. บอร์ดทดลอง [26]

Field-Programmable Gate Array (FPGA) เป็นวงจรรวมดิจิตอลที่ถูกออกแบบให้ผู้ใช้สามารถโปรแกรมการทำงานลงไปในตัวชิพได้โดยการใช้ภาษา Hardware Description Language (HDL) ในการอธิบายการทำงานของวงจรภายใน ส่วนประกอบของ FPGA ปะกอบไปด้วย Configurable Logic Blocks (CLBS) ทำหน้าที่สร้างการทำงานลอจิกแบบต่างๆตามการออกแบบของผู้ใช้งาน ความซับซ้อนและประสิทธิภาพของระบบส่วนหนึ่งขึ้นอยู่กับการออกแบบ CLBs ของผู้ผลิต และส่วนประกอบของ FPGA อีกส่วนคือ I/O pins (Input Output pins) ซึ่งเป็นส่วนของการเชื่อมต่อไปยังอุปกรณ์ภายนอก

ปัจจุบันได้มีการพัฒนาวิธีการใช้งาน FPGA อีกรูปแบบหนึ่ง คือ การใช้งานซอฟท์คอร์โปรเซสเซอร์ (soft core processor) บน FPGA โดยวิธีการนี้เป็นที่นิยมเพิ่มขึ้น เนื่องจากผู้ใช้สามารถสร้างวงจรรวมได้หลายชุด ในชิพ FPGA 1 ตัว ซึ่งขึ้นอยู่กับปริมาณความจุของลอจิกเกตภายใน FPGA ทำให้สามารถลดขนาดอุปกรณ์และราคาลงได้ สำหรับทางด้านการใช้งานซอฟท์คอร์โปรเซสเซอร์จะสามารถใช้โปรแกรมภาษา C เพื่อควบคุมการทำงานของชิพ FPGA

6.1 บอร์ดทดลองรุ่น Altium Nanoboard 3000

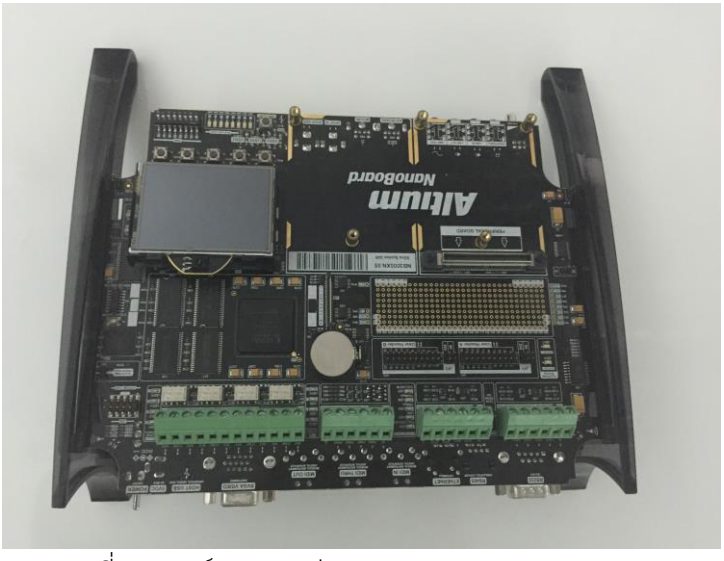
บอร์ดทดลอง Altium Nanoboard 3000 ซึ่งเป็นบอร์ดทดลองที่มีชิพ FPGA ของบริษัท Xilinx รุ่น Spartan-3AN XCS1400AN-4FGG676C ที่มีความจุวงจรเท่ากับ 1,400,000 เกต ใช้โปรแกรม Altium Designer ในการออกแบบวงจรภายใน อุปกรณ์ที่อยู่บนบอร์ดทดลองประกอบไปด้วย

1. จอ TFT LCD (Thin Film Transistor Liquid Crystal Display) ขนาด 240x320 พิกเซล
2. USB hubs ที่รองรับการเชื่อมต่อแบบ USB ได้ 3 ช่อง
3. พอร์ต SVGA (Super Video Graphics Array) ขนาด 24บิต ความถี่ 80 MHz,

4. พอร์ตการเชื่อมต่อมาตรฐานอย่าง เช่น RS-232, RS-485, Ps/2, 10/100 fast Ethernet, USB2.0 และ S/PDIF MIDI

5. มีความถี่นาฬิกาที่สามารถโปรแกรมได้ตั้งแต่ 6 MHz ถึง 200 MHz และความถี่นาฬิกาสำหรับบอร์ดทดลองที่มีความเร็ว 20 MHz

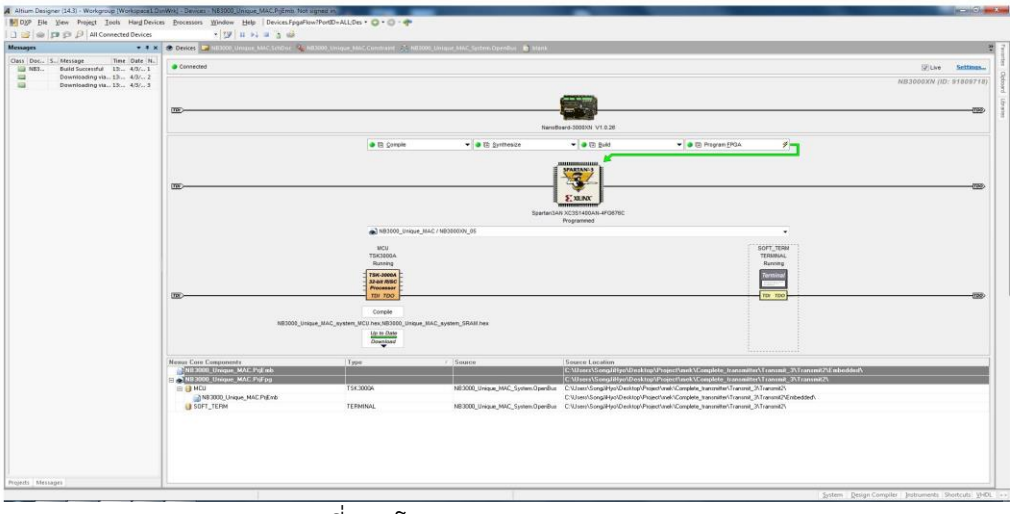
6. หน่วยความจำบนบอร์ดทดลองที่สามารถใช้ได้ ประกอบด้วย SRAM (Static Random Access Memory) ขนาด 512 KB จำนวน 2 ชิ้น รวม 1 MB, SDRAM (Synchronous Dynamic Random Access Memory) ขนาด 64 MB, Flash Memory ขนาด 16 MB



รูปที่ 18 บอร์ดทดลองรุ่น Altium Nanoboard 3000

6.2 โปรแกรม Altium Designer

โปรแกรม Altium Designer เป็นโปรแกรมที่ใช้สำหรับการออกแบบ FPGA บนบอร์ดทดลองแบบกราฟฟิกของบริษัท Altium ทำให้ง่ายต่อการใช้งานและสามารถเห็นการจัดวางออปเจค (object) ต่างๆได้สะดวก Altium Designer ใช้โค้ดภาษา C ในการเขียนโปรแกรม และมีตัวแปลโปรแกรม (compiler) ก่อนที่จะทำการโหลดโปรแกรมไปยังชิพ การออกแบบจะแบ่งออกเป็น 2 ส่วนใหญ่ ๆ คือ การออกแบบฮาร์ดแวร์ของระบบและการออกแบบซอฟต์แวร์ของระบบ โดยต้องเริ่มจากการออกแบบระบบ Open bus สำหรับเชื่อมต่อระหว่างอุปกรณ์บนบอร์ดทดลองและชิพ FPGA จากนั้นจึงกำหนดองค์ประกอบของส่วนหน่วยความจำ ส่วนการออกแบบซอฟต์แวร์ ระบบจะสร้างวงจรลอจิกจากโปรแกรมภาษา C ที่ผู้ใช้สร้างขึ้น



รูปที่ **19** โปรแกรม Altium Designer

7. การทบทวนวรรณกรรม/สารสนเทศ (information) ที่เกี่ยวข้อง

ในปี ค.ศ. 1948 Shannon [27] ได้เสนอทฤษฎีเข้ารหัส (Shannon's coding theorem) ที่กล่าวว่าเมื่อมีการเข้ารหัสช่องสัญญาณที่เหมาะสมแล้ว จะสามารถลดความผิดพลาดที่เกิดขึ้นจากช่องสัญญาณให้ต่ำได้เท่าที่ต้องการ ซึ่งงานของ Shannon ทำให้เกิดทฤษฎีรหัสช่องสัญญาณ และทำให้เกิดงานวิจัยที่เกี่ยวข้องจำนวนมากเกี่ยวกับการรหัสช่องสัญญาณที่ดี ตัวอย่างเช่น รหัสบล็อก รหัสคอนโวลูชัน รหัสวน (cyclic code) รหัส BCH รหัส LDPC [28] รหัส Turbo [29] เป็นต้น รหัสช่องสัญญาณถูกนำไปใช้เพื่อให้การสื่อสารข้อมูลมีความเชื่อถือได้มากขึ้น เพราะสามารถลดความผิดพลาดที่เกิดขึ้นจากการส่งข้อมูล ทั้งจากช่องสัญญาณใช้สายและช่องสัญญาณไร้สาย รวมทั้งเครือข่ายสื่อสารข้อมูลก็มีการใช้โปรโตคอลที่บรรจุรหัสเพื่อแก้ไขความผิดพลาดตามมาตรฐานของ CCITT (International Telephone and Telegraph Consultative Committee) เช่น CRC-16 และ CRC-32 เป็นต้น

การสื่อสารแบบใช้สายมักจะมีความผิดพลาดไม่มากนักและความผิดพลาดที่เกิดจะเป็นแบบสุ่ม (random errors) แต่ในปัจจุบันการสื่อสารไร้สายได้เติบโตขึ้นมากจากความต้องการของผู้ใช้งานที่จะทำการติดต่อสื่อสารกันตลอดเวลา (stay connected) รวมทั้งการเข้าถึงอินเทอร์เน็ตจากอุปกรณ์ไร้สายต่างๆช่องสัญญาณไร้สายโดยเฉพาะแบบที่มีการจางหายของสัญญาณนั้น ความผิดพลาดหลักจะเป็นแบบเบริสต์ (burst errors) ซึ่งเป็นการผิดพลาดหลายบิตติดต่อกัน การสื่อสารเคลื่อนที่ที่ผู้รับมีการเคลื่อนที่เร็วได้รับความสนใจเพิ่มขึ้นดังจะเห็นได้จากมาตรฐาน 4G หรือ LTE [30] ที่จะมีอัตราการส่งข้อมูลสูงสุด 100 Mb/s ในสภาวะที่ผู้ใช้มีการเคลื่อนที่ด้วยความเร็วเช่นอยู่ในยานพาหนะ นอกจากนั้นการสื่อสารสำหรับระบบรางเพื่อผู้รับที่อยู่ในรถไฟความเร็วสูงก็ได้รับความสนใจมากขึ้น [31,32]

ในการแก้ไขความผิดพลาดแบบเบริสต์นั้นมีหลายวิธีเช่นวิธีอินเทอร์ลีฟ (Interleaving) [33] และการใช้รหัสนอนไบนารี เป็นต้น นอกจากความผิดพลาดแบบเบริสต์ซึ่งเป็นความผิดพลาดหลักแล้ว บางครั้งยังมีความผิดพลาดแบบสุ่มเกิดขึ้นในช่องสัญญาณไร้สายด้วย ดังนั้นจึงมักใช้โครงสร้างรหัสที่เรียกว่ารหัสคอนคาทิเนตซึ่งสามารถแก้ไขได้ทั้งความผิดพลาดแบบสุ่มและแบบเบริสต์ รหัสแบบคอนคาทีเนตได้ถูกนำเสนอเป็นครั้งแรกโดย Forney [17] ในปีค.ศ. 1966 โครงสร้างของรหัสแบบนี้ประกอบด้วยรหัสภายในซึ่งเป็นรหัสที่ใช้สัญลักษณ์ไบนารีที่ถูกออกแบบให้แก้ไขความผิดพลาดแบบสุ่มและรหัสภายนอกซึ่งเป็นรหัสที่ใช้สัญลักษณ์นอนไบนารีที่ถูกออกแบบให้แก้ไขความผิดพลาดแบบเบริสต์ โครงสร้างแบบนี้มีข้อดีคือสามารถสร้างรหัสที่มีประสิทธิผลในการแก้ไขความ

ผิดพลาดสูงได้ง่ายโดยมีความซับซ้อนไม่มากนัก รหัสภายนอกที่ใช้กันอย่างแพร่หลายทั้งในการเก็บข้อมูล (data storage) เช่น แผ่น CD การส่งข้อมูลผ่านดาวเทียม เป็นต้น คือรหัส Reed-Solomon [18,19] ซึ่งเป็นรหัสแบบบล็อกที่ใช้สัญลักษณ์นอนไบนารี เพราะมีประสิทธิภาพสูง

รหัสแบบคอนโวลูชันที่ใช้สัญลักษณ์นอนไบนารีเดิมนั้นไม่เป็นที่นิยมเพราะไม่เหมาะที่จะใช้กับการถอดรหัสด้วยขั้นตอนวิธีของวิเทอร์บิ (Viterbi decoding) [34] ซึ่งเป็นขั้นตอนวิธีที่ใช้กันทั่วไปสำหรับถอดรหัสคอนโวลูชันแบบไบนารี อย่างไรก็ดีในปี คศ. 2002 ผู้วิจัยร่วมกับ Metzner ได้เสนอขั้นตอนวิธีการถอดรหัสสำหรับรหัสนอนไบนารีคอนโวลูชันที่ไม่เป็นระบบ (Nonsystematic convolutional codes) [35] การถอดรหัสนี้เรียกว่าการถอดรหัสเวกเตอร์ซิมโบล (Vector Symbol Decoding หรือ VSD) ซึ่งต่อยอดมาจากงานของ Metzner และ Kapturowski [36] ที่ได้เสนอวิธีการถอดรหัส VSD สำหรับรหัสเชิงเส้นใดๆที่ใช้สัญลักษณ์นอนไบนารี หลักการถอดรหัส VSD ถูกนำเสนอในปีค.ศ. 1999 อีกครั้งโดย Haslach และ Vinck [37] ในชื่อว่ารหัสแอเรย์ (Array Codes) ในปี 2003 Metzner ยังแสดงให้เห็นว่าการถอดรหัส VSD กับรหัสบล็อกที่ถูกเลือกมาแบบสุ่มนั้นให้ประสิทธิภาพในการถอดรหัสสูงกว่าค่าประกันการแก้ไขความผิดพลาดของรหัส Reed-Solomon เมื่อใช้สัญลักษณ์ขนาดใหญ่เช่น 32 บิตต่อสัญลักษณ์โดยไม่มีการทำอินเทอร์ลีฟ [38]

รหัส RS (Reed-Solomon) มีความเหมาะสมที่จะนำมาใช้เป็นรหัสภายในเพื่อใช้งานร่วมกับวิธีการถอดรหัส VSD (Vector Symbol Decoding) ที่ใช้เป็นตัวถอดรหัสภายนอก เนื่องจากคุณสมบัติของรหัส RS ซึ่งเป็นรหัส นอน-ไบนารี ที่มีระยะแฮมมิงที่สูง จึงทำให้มีความสามารถในการแก้ไขสัญลักษณ์ที่ผิดพลาดได้มาก นอกจากนั้นรหัส RS ยังเป็นรหัสที่เหมาะสมที่สุดในการเติมสัญลักษณ์ที่ถูกลบ (Erasure Symbol) [23] ซึ่งจะช่วยเพิ่มความสามารถในการแก้ไขจำนวนสัญลักษณ์ได้ตามอสมการ $2e+s \leq (n-k)$ เมื่อ e คือจำนวนสัญลักษณ์ที่ผิดพลาดและ s คือจำนวนสัญลักษณ์ที่ถูกลบ

ดังนั้นในงานวิจัยจะพิจารณาการเพิ่มประสิทธิภาพของระบบโดยการลบสัญลักษณ์ที่มีโอกาสผิดพลาดสูงร่วมกับการถอดรหัสแบบฮาร์ด เนื่องจากตัวถอดรหัสสามารถแก้ไขสัญลักษณ์ที่ถูกลบได้มากกว่าสัญลักษณ์ที่ผิดพลาด ซึ่งหากสามารถมีวิธีการลบสัญลักษณ์ที่เหมาะสมสำหรับช่องสัญญาณต่างๆ [24] จะให้ผลเสมือนว่ารหัสสามารถที่จะแก้ไขสัญลักษณ์ที่ผิดพลาดได้สูงกว่า t สัญลักษณ์ตามปกติ ในขณะที่ความซับซ้อนเพิ่มขึ้นไม่มาก วิธีการที่ง่ายที่สุด คือ ใช้ค่าขีดแบ่ง (threshold) ในการตัดสินใจในการระบุตำแหน่งของสัญลักษณ์ที่ถูกลบ แต่การนำมาใช้จริงก็ต้องมีกระบวนการในจัดการที่ดีพอสมควร เช่น ในช่องสัญญาณที่คุณภาพต่ำมากๆ อาจมีจำนวนสัญลักษณ์ที่มีค่าต่ำกว่าค่าขีดแบ่ง มากเกิน 2t สัญลักษณ์ซึ่งอาจจะทำให้ไม่สามารถทำการถอดรหัสได้ สำหรับแนวความคิดในการนำสัญลักษณ์ที่ถูกลบมาใช้นั้นก็มีผู้นำเสนอวิธีการอยู่พอสมควรเช่นกัน ดังเช่น [39,40]

อย่างไรก็ตามการที่ตัวถอดรหัส RS ซึ่งใช้เป็นรหัสภายในยังต้องทำงานร่วมกับ VSD ซึ่งใช้เป็นตัวถอดรหัสภายนอก ขั้นตอนในการระบุตำแหน่งสัญลักษณ์ที่ถูกลบ จึงไม่ควรที่จะมีความซับซ้อนมาก ซึ่งอาจส่งผลให้เกิดความล่าช้าเพิ่มขึ้น เช่น วิธี Generalized Minimum Distance (GMD) ซึ่งจะค่อยๆเพิ่มจำนวนสัญลักษณ์ที่ถูกลบทีละ 2 สัญลักษณ์ ทุกครั้งที่เกิดความผิดพลาดในการถอดรหัส ซึ่งในกรณีที่มีสัญลักษณ์ที่ผิดพลาดมากๆ อาจจะทำให้เกิดความล่าช้าในการถอดรหัสและสิ้นเปลืองการคำนวณสูงมาก วิธีการที่ใช้จึงควรที่จะเป็นวิธีการที่เรียกว่า single-trial หรือ one-pass ซึ่งก็คือการระบุจำนวน และตำแหน่งของสัญลักษณ์ที่ถูกลบและนำไปใช้ในการถอดรหัสให้เสร็จภายในครั้งเดียว ตามตัวอย่าง [41]

ในการสร้างชิ้นงานต้นแบบ บนบอร์ดทดลอง FPGA เป็นการพัฒนาเพิ่มเติมจาก [26] ซึ่งรองรับรหัสคอนโวลูชันภายนอกขนาดเล็กกว่าและมีขนาดไม่เกิน 128 บิตต่อสัญลักษณ์ โดยมีส่วนสำคัญอยู่ที่การปรับปรุงให้ตัวถอดรหัส VSD ให้สามารถรองรับสัญลักษณ์ขนาดใหญ่ได้ถึง 1,784 บิตต่อสัญลักษณ์ เนื่องจากในส่วนอื่นๆ เช่น การเข้ารหัสถอดรหัส RS แบบฮาร์ด สามารถศึกษาจากงานวิจัยอื่นซึ่งมีผู้ทดลองบนบอร์ด FPGA มาแล้วจำนวนมาก [42-44] และในส่วนของการเข้าคอนโวลูชันให้มีขนาดสัญลักษณ์ใหญ่ขึ้นก็สามารถแก้ไขได้โดยง่าย ตัวถอดรหัส VSD จึงเป็นส่วนที่เหลือซึ่งต้องการการปรับปรุงเพื่อให้สามารถทำงานได้อย่างถูกต้อง

สำหรับโครงการวิจัยนี้จะเสนอการสร้างระบบรหัสคอนคาทิเนตแบบใหม่ที่เหมาะสมกับการสื่อสารไร้สายเคลื่อนที่โดยการนำรหัส RS มาใช้เป็นรหัสภายใน เพราะรหัส RS มีคุณภาพดีและมีความยืดหยุ่นสูงในการปรับขนาดความยาวของรหัส [19] โดยจะใช้ขนาดสัญลักษณ์ 6-8 บิต เพื่อแก้ไขความผิดพลาดที่มีความยาวไม่มากและเติมเต็มสัญลักษณ์ที่หายไป (erasure symbols) ได้ และใช้รหัสนอนไบนารีคอนโวลูชันที่ใช้สัญลักษณ์ขนาดใหญ่ได้ถึง 1,784 บิตต่อสัญลักษณ์มาเป็นรหัสภายนอก ซึ่งตัวถอดรหัสภายในจะใช้การถอดรหัสของ RS และตัวถอดรหัสภายนอกจะใช้ VSD ระบบนี้เหมาะสมกับการสื่อสารไร้สายเคลื่อนที่ และรองรับอัตราการส่งข้อมูลมีการเปลี่ยนแปลงเพิ่มขึ้นได้โดยการปรับขนาดสัญลักษณ์ที่ใช้ การถอดรหัส VSD ที่สามารถถอดรหัสนอนไบนารีขนาดใหญ่ได้ง่ายและสามารถขยายขนาดสัญลักษณ์ได้ไม่จำกัด จึงมีความน่าสนใจมาก และการใช้รหัสคอนโวลูชันแทนรหัสแบบบล็อกเป็นรหัสภายนอก เพราะ VSD สำหรับรหัสคอนโวลูชันนั้นสามารถเริ่มทำการถอดรหัสได้เมื่อได้รับจำนวนสัญลักษณ์เพียงไม่กี่สัญลักษณ์แทนที่จะต้องรอให้ได้รับสัญลักษณ์ครบทั้งหมดดังในกรณีของ VSD สำหรับรหัสแบบบล็อก ลักษณะเฉพาะนี้ทำให้การถอดรหัสนอนไบนารีคอนโวลูชันมีความซับซ้อนน้อยกว่ารหัสแบบบล็อก [45]

วิธีวิจัย

1. ระบบรหัสคอนคาทิเนตที่นำเสนอ

ระบบรหัสที่นำเสนอประกอบด้วย รหัสภายนอกซึ่งใช้รหัสคอนโวลูชันแบบนอนไบนารีและใช้ตัวถอดรหัสเวกเตอร์ซิมโบลในการถอดรหัส ส่วนรหัสภายในมีการเปรียบเทียบการใช้รหัสภายในแบบต่างๆ เช่น รหัส BCH, รหัส RS รวมทั้งรหัส RS ที่ใช้การถอดรหัสร่วมกับสัญลักษณ์ที่ถูกลบ ผ่านการจำลองช่องสัญญาณที่มีการจางหายและเกิดปรากฏการณ์ดอปเปอร์ การจำลองระบบใช้โปรแกรมภาษา C++ ในส่วนของตัวถอดรหัส VSD โดยมีการปรับปรุงโปรแกรม C++ เดิม เพื่อให้สามารถรองรับขนาดสัญลักษณ์ที่ใหญ่ขึ้นได้ รวมถึงการปรับปรุงให้มีการใช้งานหน่วยความจำน้อยลง และมีการใช้ค่าซินโดรมสูงสุดที่ดีที่สุดเพื่อลดการคำนวณที่มากเกินความจำเป็น และใช้ MATLAB ในการจำลองส่วนอื่นๆที่เหลือ

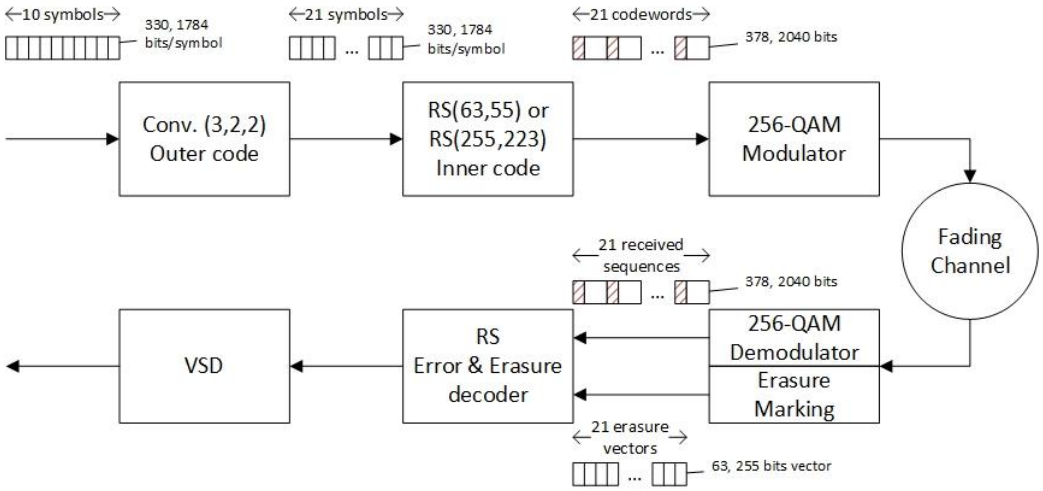
ในการทดลองแต่ละรอบ จะมีสัญลักษณ์อินพุตจำนวน 10 สัญลักษณ์ ซึ่งจะเข้ารหัสได้เป็นรหัสภายนอกขนาด 21 สัญลักษณ์ และจะถูกเข้ารหัสเป็นรหัสภายในแบบหนึ่งต่อหนึ่ง คือ 1 สัญลักษณ์ของรหัสภายนอกจะถูกเข้ารหัสเป็น 1 คำรหัสภายใน และจะถูกส่งผ่านช่องสัญญาณ ที่ฝั่งรับชุดสัญลักษณ์ที่ได้รับ(received sequence) จะถูกถอดรหัสโดย VSD

ระบบรหัสที่นำเสนอใช้รหัสคอนโวลูชัน(3,2,2) ซึ่งมีระยะคอนสเทรนท์เท่ากับ 4 เป็นรหัสภายนอก โดยใช้การเข้ารหัสแบบนอนไบนารี ขนาดสัญลักษณ์ของรหัสภายนอกจะเปลี่ยนแปลงตามรหัสภายในที่เลือกใช้ ซึ่งขนาดสัญลักษณ์สามารถรองรับรหัส RS(255,223) ซึ่งเป็นรหัสมาตรฐาน NASA [9] ได้ การทดสอบนี้ มีการเปรียบเทียบการใช้รหัสภายในแบบต่างๆ ซึ่งมีอัตรารหัส(code rate) ที่เท่ากันหรือใกล้เคียงกัน โดย

- รหัส RS(255,223) มีอัตรารหัส 0.875 ขนาดสัญลักษณ์รหัสภายนอก 1784 bits
- รหัส RS(63,55) มีอัตรารหัส 0.873 ขนาดสัญลักษณ์รหัสภายนอก 330 bits
- รหัส BCH(255,223) มีอัตรารหัส 0.875 ขนาดสัญลักษณ์รหัสภายนอก 223 bits

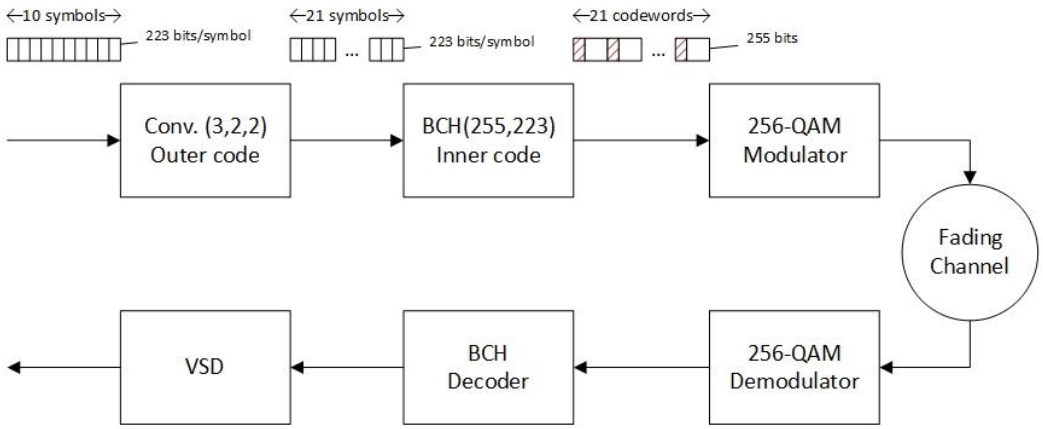
เพื่อให้สมเหตุสมผลในการใช้กับระบบที่มีอัตราการส่งข้อมูลที่สูง การกล้ำสัญญาณและการแยกสัญญาณ (modulation/demodulation) ใช้เทคนิคในเชิงสัญลักษณ์โดยเลือกใช้ 256-QAM ซึ่งสามารถกล้ำสัญญาณได้ครั้งละ 8 บิต

ตัวถอดรหัส RS ใช้การตัดสินใจแบบฮาร์ด โดยใช้ขั้นตอนวิธีของ Berlekamp-Massey ซึ่งสามารถใช้คุณสมบัติการแก้ไขสัญลักษณ์ที่ผิดพลาดร่วมกับสัญลักษณ์ที่หายไปเพื่อเพิ่มประสิทธิภาพในการแก้ไขได้ การระบุตำแหน่งสัญลักษณ์ที่ถูกลบใช้การกำหนดค่าขีดแบ่ง โดยทำในขั้นตอนการแยกสัญญาณ ตามสมการ (18) เมื่อกำลังช่องสัญญาณมีค่าต่ำกว่าค่าที่กำหนด จะทำการระบุตำแหน่งเป็นสัญลักษณ์ที่ถูกลบ แผนภาพระบบที่ใช้รหัส RS เป็นรหัสภายในสามารถเขียนได้ดังรูปที่ 20



รูปที่ 20 ระบบคอนคาทิเนตที่ใช้รหัสภายในแบบ RS

สำหรับการใช้รหัส BCH เป็นรหัสภายใน ถูกนำมาใช้เพื่อเปรียบเทียบประสิทธิภาพของระบบซึ่งมีสัญลักษณ์ขนาดเล็กและใช้รหัสภายในแบบไบนารี การถอดรหัสโดยใช้การตัดสินใจแบบฮาร์ดเช่นเดียวกันการใช้รหัส RS แต่จะไม่มีการถอดรหัสโดยใช้ร่วมกับสัญลักษณ์ที่ถูกลบ สามารถเขียนเป็นแผนภาพได้ดังรูปที่ 21

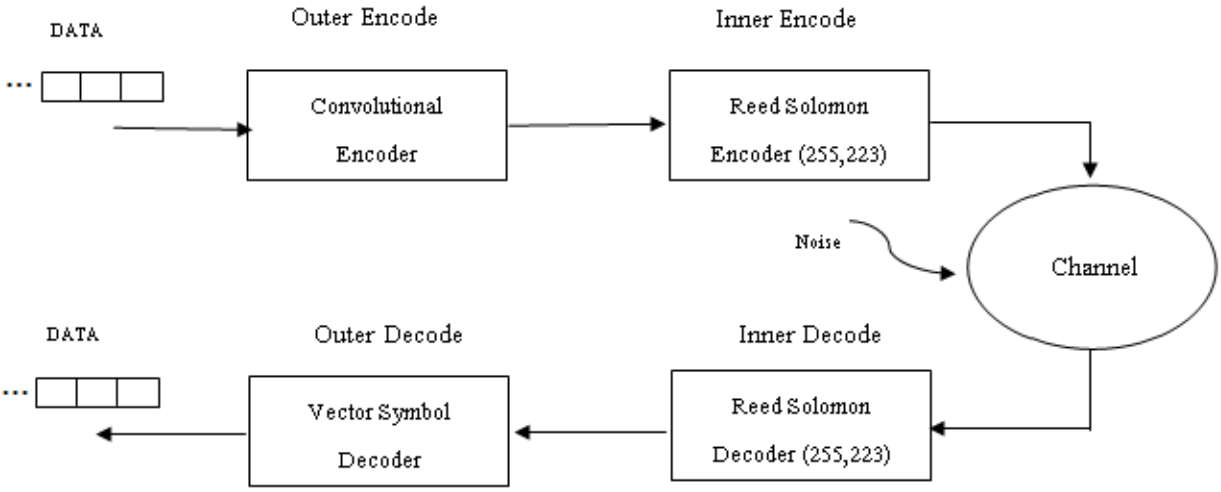


รูปที่ 21 ระบบคอนคาทิเนตที่ใช้รหัสภายในแบบ BCH

2. การออกแบบระบบภายในบอร์ดทดลอง

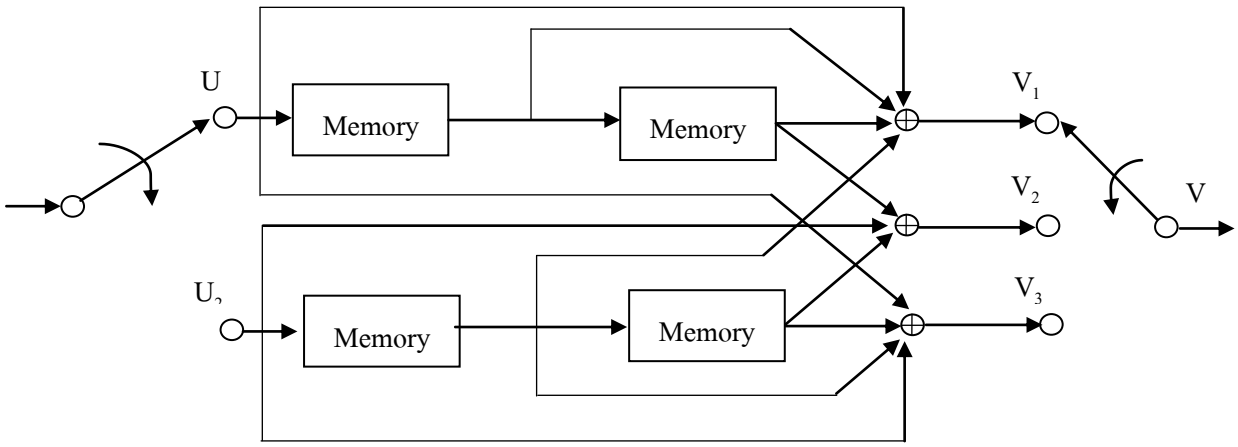
ระบบรหัสที่ออกแบบและนำมาใช้บนบอร์ดทดลอง ประกอบไปด้วย 5 ส่วน ซึ่งเป็นการนำองค์ประกอบของแต่ละส่วนของรูปที่ มาสร้างเป็นชุดวงจรรวม ได้แก่

1. วงจรเข้ารหัสภายนอกคอนโวลูชัน(3,2,2) แบบนอนไบนารี ขนาดสัญลักษณ์ 1,784 บิต
2. วงจรเข้ารหัสภายใน RS(255,223)
3. วงจรถอดรหัสภายใน RS(255,223) โดยใช้วิธี Berlakamp-Massey
4. วงจรถอดรหัสภายนอก VSD
5. ส่วนจำลองการเกิดความผิดพลาด โดยกำหนดตำแหน่งและค่าความผิดพลาดด้วยตัวเอง



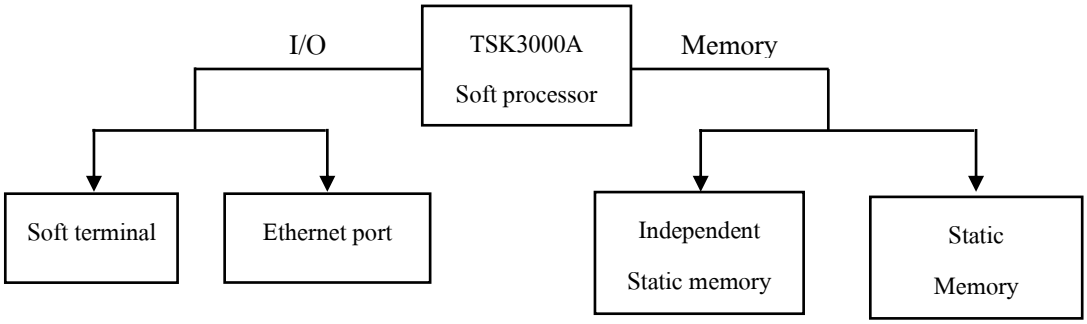
รูปที่ 22 ภาพรวมในการทำการเข้ารหัสและถอดรหัส

สำหรับรหัสคอนโวลูชัน จะใช้วงจรที่แสดงตามรูปที่23



รูปที่ 23 วงจรการเข้ารหัสของ Convolutional Encoder (3,2,2) [26]

การออกแบบ Open bus หรือการเชื่อมต่อต่างๆสำหรับบอร์ดทดลอง จะมีลักษณะ ดังรูปที่24

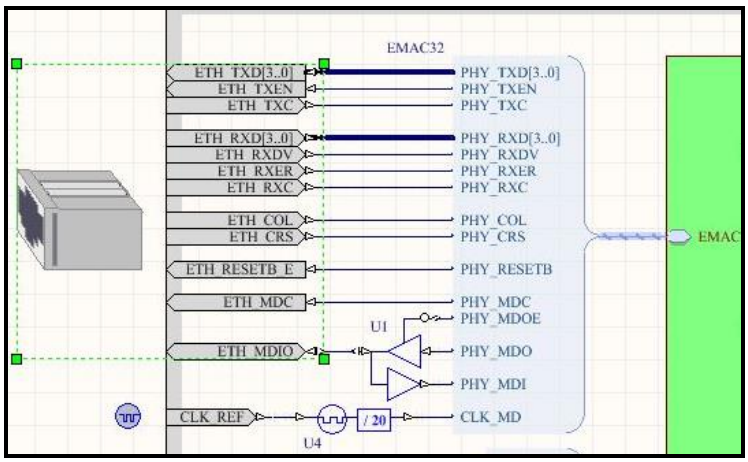


รูปที่ 24 การเชื่อมต่อระหว่างอุปกรณ์บนบอร์ดทดลองและชิพ FPGA

การเชื่อมต่ออินพุตเอาต์พุต (I/O) ประกอบไปด้วย

1. Soft terminal เป็นส่วนที่ถูกจำลองขึ้นเพื่อเป็นส่วนแสดงผลลัพธ์ที่ได้จากการประมวลผลภายในบอร์ดทดลอง ซึ่งผลลัพธ์ที่ได้จะสามารถดูได้ผ่านโปรแกรม Altium Designer บนเครื่องคอมพิวเตอร์

2. Ethernet port การเชื่อมต่อกับพอร์ตอีเทอร์เนตบนบอร์ดทดลอง จะใช้เพื่อรับข้อมูลที่จะถูกนำมาเข้ารหัส และใช้ส่งข้อมูลหลังจากถอดรหัสเรียบร้อยแล้ว

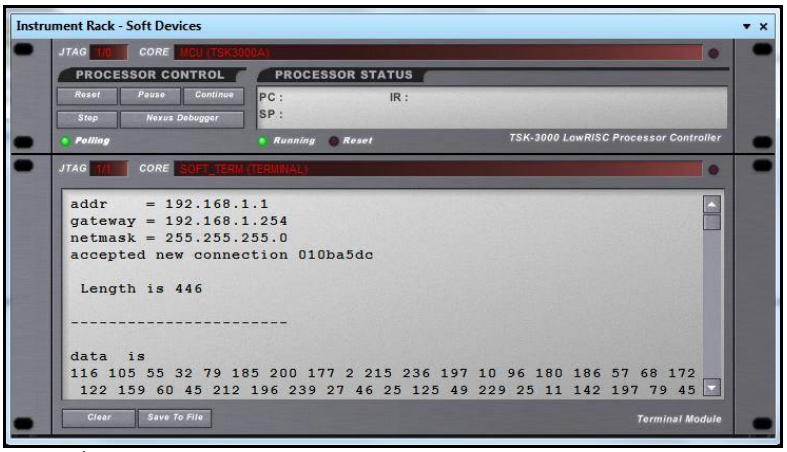


รูปที่ 25 การเชื่อมต่อระหว่างส่วนประมวลผลและอุปกรณ์อีเทอร์เนต [26]

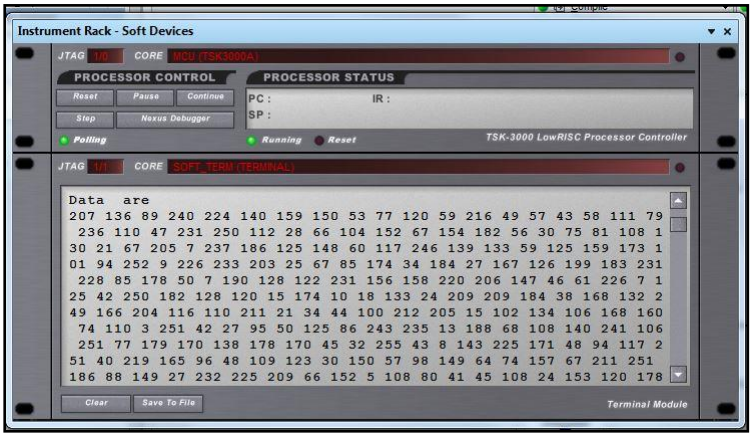
การกำหนดค่าคุณสมบัติของอีเทอร์เนต ใช้การเขียนโปรแกรม Matlab เพื่อรับส่งข้อมูลจากคอมพิวเตอร์ โดยจำลองว่ามีการส่งข้อมูลจากที่หนึ่งไปยังอีกที่หนึ่ง คำสั่งที่ใช้คือ

T = tcpip ('192.168.1.1', 9090,'LocalPort', 9090);

เป็นคำสั่งสำหรับการเชื่อมต่อ โดยเป็นการตั้งค่า บอร์ดให้มีค่า IP = 192.168.1.1 และใช้ Port 9090 ซึ่งต้องตั้งค่าของบอร์ดนี้ในโปรแกรม Altium ให้เป็นค่าเดียวกัน เมื่อเริ่มทำการทำงานจะใช้คำสั่ง fopen() ในการเปิดค่าตัวแปรและส่งค่าผ่านตัวแปรโดยใช้คำสั่ง fwrite() เมื่อสามารถทำให้บอร์ดและคอมพิวเตอร์เชื่อมต่อและส่งข้อมูลกันได้แล้ว ก็จะส่งข้อมูลขนาด 446 ไบต์หรือเท่ากับ 2 สัญลักษณ์ของรหัส ไปทำการเข้ารหัส การเชื่อมต่อได้อย่างสมบูรณ์จะมีการแสดงผลดังรูปที่ 26 และ 27 หลังจากที่ส่งข้อมูลเสร็จแล้ว เมื่อต้องการปิดการเชื่อมต่อ จะใช้คำสั่ง fclose()

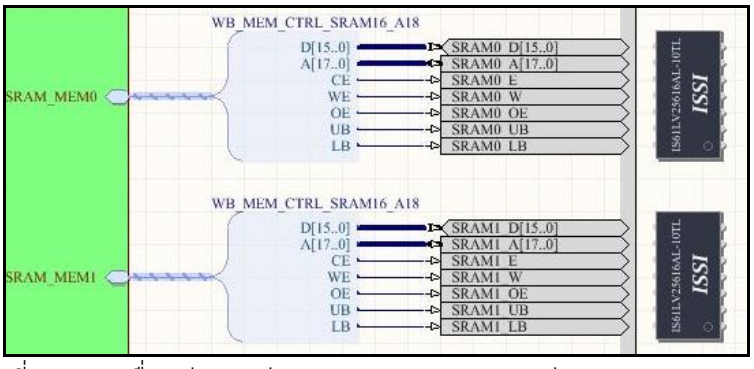


รูปที่ 26 แสดงการเชื่อมต่อสมบูรณ์และมีการส่งข้อมูลตัวเลขจำนวน 446 ไบต์ บน Soft terminal

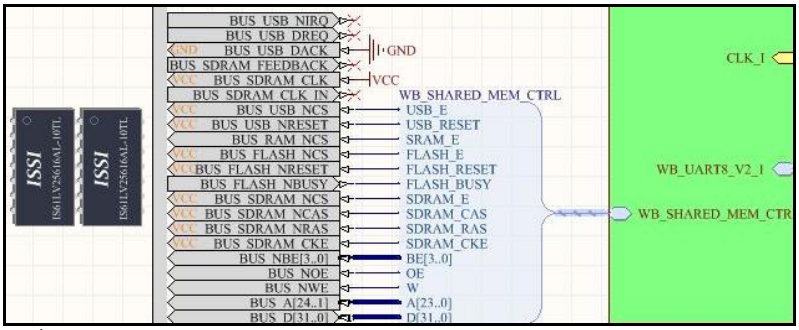


รูปที่ 27 แสดงข้อมูลที่ได้รับเป็นตัวเลข บน Soft terminal

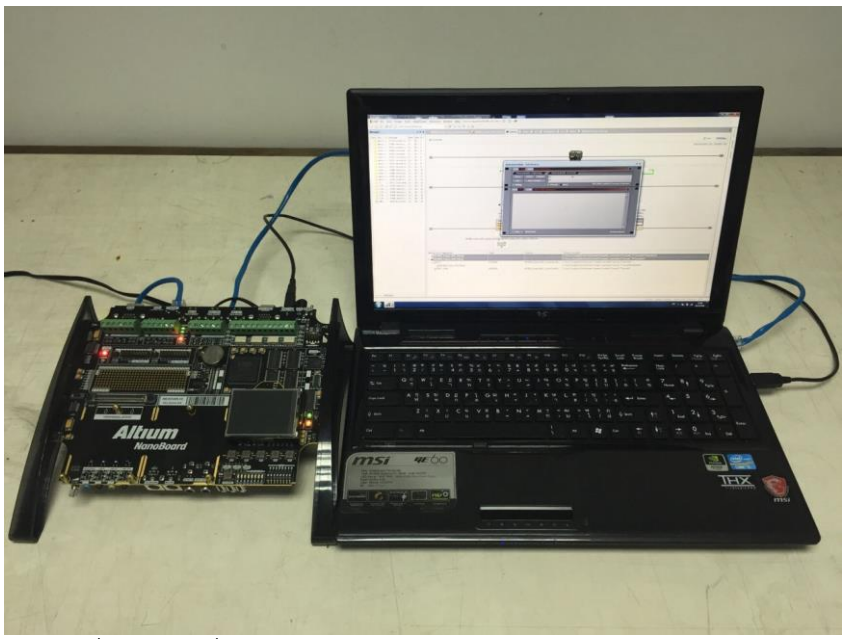
การเชื่อมต่อกับหน่วยความจำ ประกอบด้วย 2 อุปกรณ์คือ SRAM และ หน่วยความจำร่วม (Shared Memory) โดยแสดงการเชื่อมต่อ ดังรูปที่ 28 และ 29



รูปที่ 28 การเชื่อมต่อระหว่างเอสแรม (SRAM) และส่วนประมวลผล [26]



รูปที่ 29 การเชื่อมต่อระหว่างหน่วยความจำร่วม (Shared memory) และส่วนประมวลผล [26]

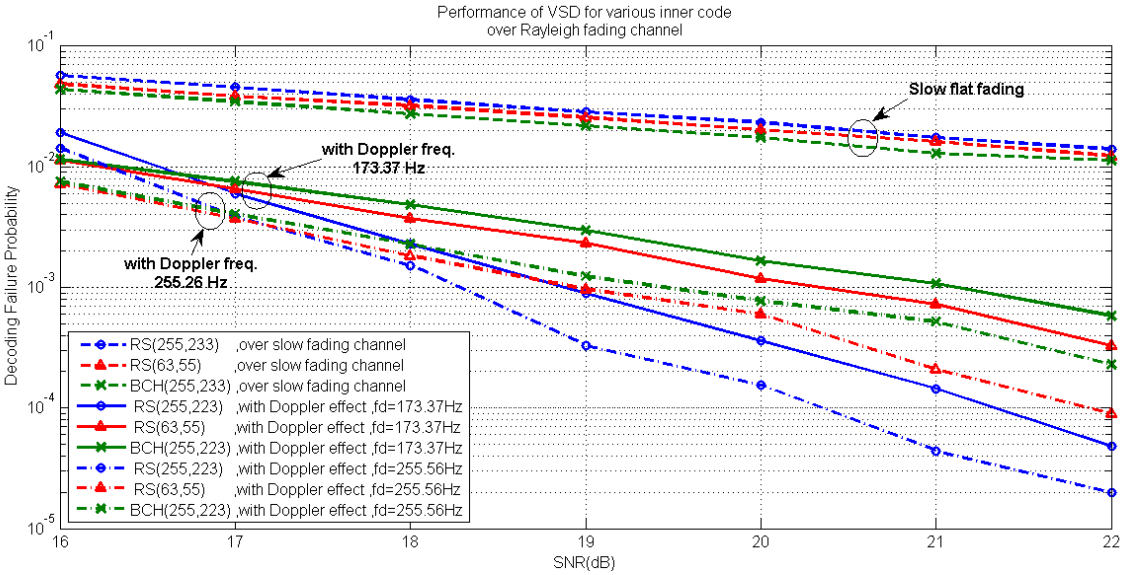


รูปที่ 30 การเชื่อมต่อระหว่างบอร์ดทดลองและอุปกรณ์คอมพิวเตอร์

ผลและวิจารณ์

1. ประสิทธิภาพของระบบรหัสที่ไม่มีการใช้รหัสที่ถูกลบ

การทดลองใช้รหัสภายใน 3 ชนิด ได้แก่ รหัส RS(255,223), RS(63,51) และ BCH(255,223) ซึ่งเป็นรหัสนอนไบนารีขนาดใหญ่ รหัสนอนไบนารีขนาดเล็ก และรหัสไบนารีตามลำดับ และถอดรหัสภายในด้วยวิธีเชิงพีชคณิต(Algebraic) ด้วยการตัดสินใจแบบฮาร์ด จากนั้นถอดรหัสภายนอกด้วย VSD เปรียบเทียบประสิทธิภาพของ VSD เมื่อมีการใช้รหัสภายในแบบต่างๆ ผ่านช่องสัญญาณที่มีการจางหายแบบ Rayleigh โดยทำการทดลองจำนวนอย่างน้อย 100,000 รอบ



รูปที่ 31 ผลการเปรียบเทียบประสิทธิภาพการถอดรหัสคอนคาทิเนตโดยใช้รหัส RS และรหัส BCH ภายในและ VSD ภายนอก

จากรูปที่ ผลการทดสอบแบ่งออกเป็น 3 กลุ่ม คือ ประสิทธิภาพเมื่อใช้งานผ่านช่องสัญญาณมีการจางหายแบบราบที่เปลี่ยนแปลงอย่างช้า (slow flat fading) ประสิทธิภาพเมื่อใช้งานผ่านช่องสัญญาณที่เกิดปรากฏการณ์ดอปเปอร์ที่มีความถี่ 173.37 Hz ซึ่งเกิดเมื่อมีการเคลื่อนที่ที่ความเร็ว 80 km/h และประสิทธิภาพเมื่อใช้งานผ่านช่องสัญญาณที่เกิดปรากฏการณ์ดอปเปอร์ที่มีความถี่ 255.56 Hz ซึ่งเกิดเมื่อมีการเคลื่อนที่ที่ความเร็ว 120 km/h

สำหรับช่องสัญญาณมีการจางหายแบบราบที่เปลี่ยนแปลงอย่างช้า ในกรณีที่ช่องสัญญาณมีคุณภาพต่ำจะทำให้เกิดความผิดพลาดจำนวนมากในทุกสัญลักษณ์ของรหัสภายนอก หรือเกิดความผิดพลาดแบบเบริสต์ยาวตลอดรหัสภายนอก ซึ่งจะทำให้ระบบรหัสนี้ไม่สามารถแก้ไขความผิดพลาดได้ แต่ในกรณีที่ช่องสัญญาณมีคุณภาพดี ความผิดพลาดที่เกิดขึ้นจะเป็นความผิดพลาดแบบสุ่ม ซึ่งการใช้รหัสภายนอกที่มีสัญลักษณ์ขนาดใหญ่จะได้รับผลกระทบจากความผิดพลาดแบบสุ่มมากกว่า ประสิทธิภาพของรหัสที่ใช้สัญลักษณ์ขนาดเล็กจึงมีประสิทธิภาพที่ดีกว่า โดยระบบรหัสที่ใช้รหัส BCH(255,223) เป็นรหัสภายในจะมีประสิทธิภาพที่ดีที่สุด และระบบรหัสที่ใช้รหัส RS(255,223) เป็นรหัสภายในซึ่งมีขนาดสัญลักษณ์ใหญ่ที่สุด จะมีประสิทธิภาพต่ำที่สุด

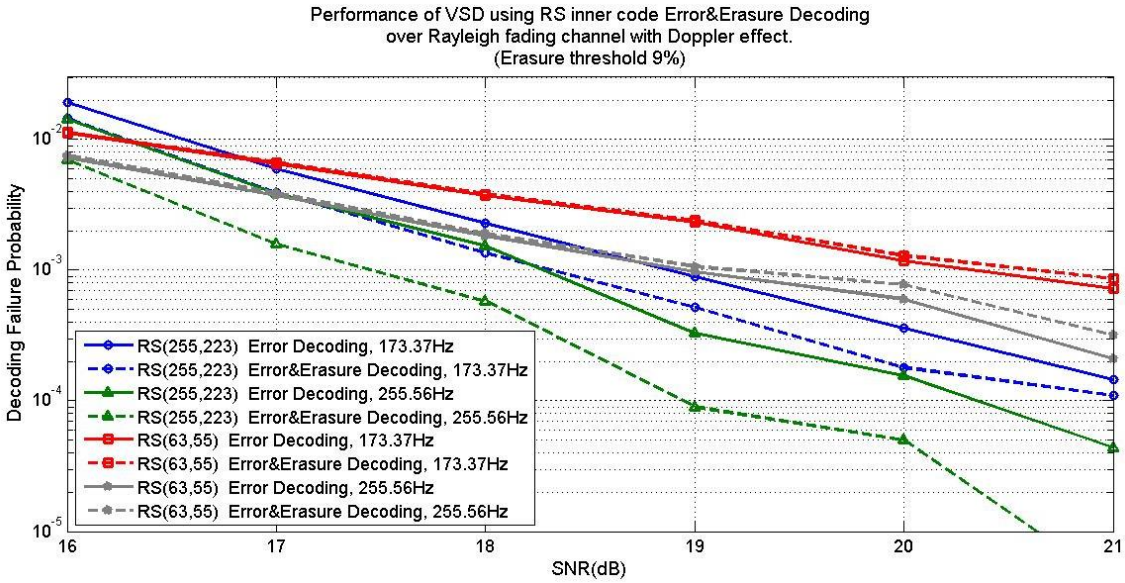
แต่สำหรับช่องสัญญาณที่เกิดปรากฏการณ์ดอปเปอร์ ทำให้ช่องสัญญาณเกิดการเปลี่ยนแปลง และมีความหลากหลาย (diversity) ซึ่งจะทำให้เกิดความผิดพลาดแบบเบริสต์ที่บางจุดในรหัส การใช้ระบบรหัสที่มีสัญลักษณ์ขนาดใหญ่จะให้ประสิทธิภาพที่ดีกว่า โดยระบบรหัสที่ใช้รหัส RS(255,223) เป็นรหัสภายในจะมีประสิทธิภาพที่ดี

ที่สุด และระบบรหัสที่ใช้รหัส BCH(255,223) เป็นรหัสภายใน จะมีประสิทธิภาพต่ำที่สุดเมื่อค่า SNR สูงขึ้น สำหรับจุดที่ค่า SNR ต่ำ ซึ่งเกิดความผิดพลาดแบบสุ่มได้ง่าย ผลกระทบจากการใช้สัญลักษณ์ขนาดใหญ่จะเหมือนกับบนช่องสัญญาณมีการจางหายแบบราบที่เปลี่ยนแปลงอย่างช้า

นอกจากนั้น ประสิทธิภาพของระบบรหัสจะดีกว่าเมื่อความผิดพลาดแบบเบริสต์มีขนาดสั้น โดยที่ความถี่ดอปเปอร์ 255.56 Hz ช่องสัญญาณมีการเปลี่ยนแปลงที่รวดเร็วกว่า ทำให้ความผิดพลาดแบบเบริสต์มีขนาดที่สั้นกว่าที่ความถี่ดอปเปอร์ 177.73 Hz ประสิทธิภาพของระบบรหัสที่ความถี่ดอปเปอร์ 255.56 Hz จึงมีประสิทธิภาพที่ดีกว่าที่ความถี่ดอปเปอร์ 177.73 Hz

2. ประสิทธิภาพของระบบรหัสที่มีการใช้ร่วมกับรหัสที่ถูกลบ

ในการใช้ตัวถอดรหัสร่วมกับสัญลักษณ์ที่ถูกลบ จะทดสอบกับระบบที่ใช้รหัส RS เป็นรหัสภายในเท่านั้น เนื่องจากเป็นรหัสที่เติมเต็มสัญลักษณ์ที่ถูกลบได้ดีที่สุด โดยการทดลองจะใช้ผลลัพธ์ที่ได้จากการทดลองที่แล้ว หรือใช้ผลลัพธ์ที่ได้ตามรูปที่ 31 นำมาถอดรหัสโดยใช้ร่วมกับสัญลักษณ์ที่ถูกลบ โดยกำหนดค่าขีดแบ่งที่ 9%



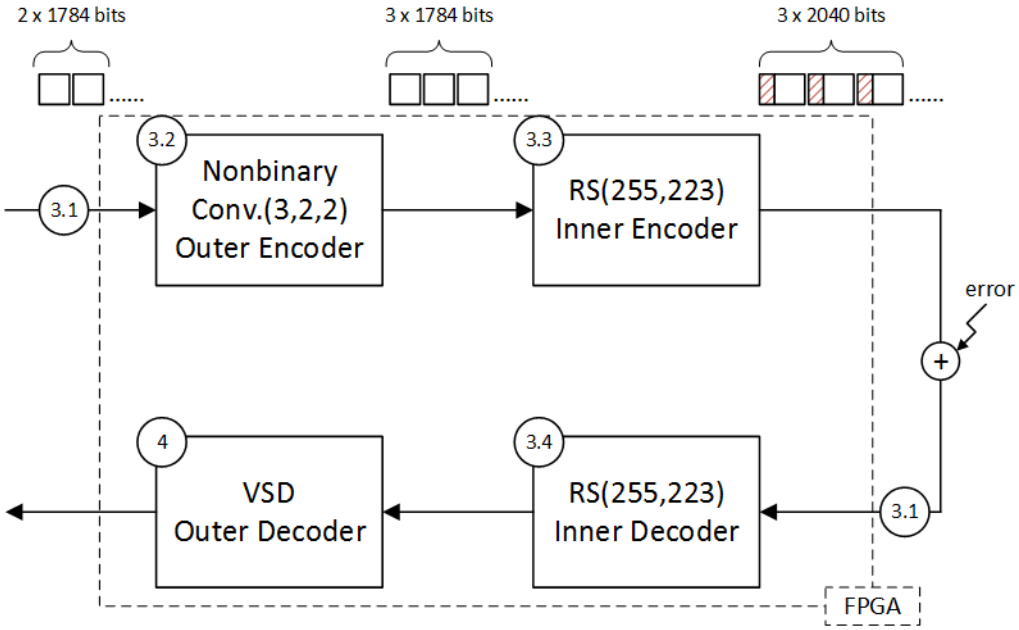
รูปที่ 32 ผลการเปรียบเทียบประสิทธิภาพการถอดรหัสคอนคาทิเนตโดยใช้รหัส RS ภายในซึ่งใช้ตัวถอดรหัสร่วมกับสัญลักษณ์ที่ถูกลบและ VSD ภายนอก

จากรูปที่ แสดงให้เห็นว่าในช่วง SNR ที่สนใจ คือที่ 16-21 dB การกำหนดค่าขีดแบ่งที่ 9% สำหรับรหัส RS(255,223) จะให้ประสิทธิภาพที่ดี โดยประสิทธิภาพที่ดีขึ้นสามารถเห็นได้ชัดเจนจากกราฟในรูปที่ 32 แต่ในทางตรงกันข้าม รหัส RS(63,55) เมื่อกำหนดค่าขีดแบ่งที่ 9% เช่นเดียวกัน กลับได้ประสิทธิภาพที่ไม่ดี โดยจะบางช่วง SNR ที่ให้ผลที่ใกล้เคียงกัน แต่ในบางช่วง SNR ก็จะมีประสิทธิภาพที่ด้อยกว่าอย่างเห็นได้ชัด

การระบุตำแหน่งสัญลักษณ์ที่ถูกลบโดยการกำหนดค่าขีดแบ่งที่เหมาะสมสามารถให้ประสิทธิภาพที่ดีกว่าไม่ใช้สัญลักษณ์ที่ถูกลบได้ เช่น การกำหนดค่าขีดแบ่งที่ 9% สำหรับรหัส RS(255,223) ระบบจะประสิทธิภาพที่ดีกว่าตลอดช่วงที่ให้ความสนใจ แต่ในทางกลับกันสำหรับรหัส RS(63,55) การใช้ตัวถอดรหัสร่วมกับสัญลักษณ์ที่ถูกลบจะให้ประสิทธิภาพที่ไม่ดี

3. การทำงานของบอร์ดทดลอง

การสร้างชิ้นงานบอร์ดทดลองจะถูกแบ่งออกเป็นส่วนๆตามรูปที่ 33



รูปที่ 33 แผนภาพโครงสร้างของบอร์ดทดลอง

จากรูปที่ 33 ส่วนที่ดำเนินการสร้างและทำการทดสอบบนบอร์ดทดลองเรียบร้อยแล้วประกอบด้วย

3.1 ส่วนของการนำข้อมูลเข้าสู่บอร์ดทดลอง

3.2 ส่วนวงจรเข้ารหัสภายนอก

3.3 ส่วนวงจรเข้ารหัสภายใน

3.4 ส่วนของวงจรถอดรหัสรหัสภายใน

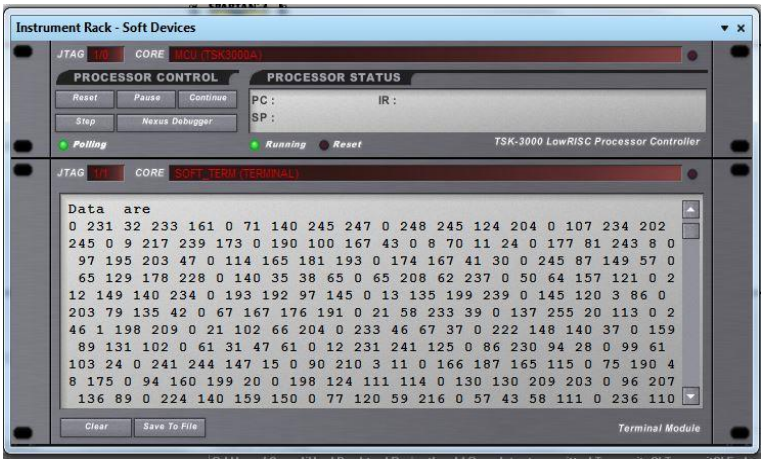
สำหรับตัวถอดรหัส VSD ได้มีการแก้ไขโปรแกรมให้เป็นภาษา C โดยใช้เฉพาะฟังก์ชันที่บอร์ดยอมรับแล้ว และ ได้จำลองการทำงานในคอมพิวเตอร์แล้ว ยังขาดการทดสอบบนบอร์ดทดลองจริง จึงได้แยกเป็นหัวข้อที่ 4

ในการทำงานของบอร์ดทดลอง ข้อมูลสุ่มจะถูกสร้างทีละ 2 สัญลักษณ์ของรหัสคอนโวลูชันภายนอก โดยมีขนาดสัญลักษณ์ 1,784 บิตต่อสัญลักษณ์ จะถูกเข้ารหัสภายนอกตามขั้นตอนที่ 3.2 ได้เป็น 3 สัญลักษณ์ จากนั้นแต่ละสัญลักษณ์จะถูกเข้ารหัส RS ภายในตามขั้นตอนที่ 3.3 ซึ่งแต่ละสัญลักษณ์จะมีขนาดเพิ่มเป็น 2,040 บิตต่อสัญลักษณ์ ด้านการถอดรหัส ขั้นตอนที่ 3.4 จะรับข้อมูลและทำการถอดรหัสทีละ 3 สัญลักษณ์ และส่งต่อให้ตัวถอดรหัส VSD ซึ่งในการทำงานต้องการข้อมูลเข้าเป็นชุด ชุดละ 3 สัญลักษณ์ได้พอดี

3.1 ส่วนของการนำข้อมูลเข้าสู่บอร์ดทดลอง

ในขั้นตอนการทดสอบวงจรเข้ารหัส จะทำการสร้างข้อมูลแบบสุ่มจากคอมพิวเตอร์ จากนั้นใช้โปรแกรม Matlab ในการส่งข้อมูลสุ่มนั้นไปยังบอร์ดทดลอง ข้อมูลที่ส่งจะมีขนาด 446 ไบต์หรือเท่ากับ 2 สัญลักษณ์ของ

รหัสคอนโวลูชันภายนอก หรือแต่ละสัญลักษณ์มีขนาด 223 ไบต์หรือ 1,784 บิตต่อสัญลักษณ์ จากนั้นแต่ละสัญลักษณ์จะถูกนำไปเข้ารหัสภายใน RS(255,223)



รูปที่ 34 ข้อมูลส่งเข้าบอร์ดทดลองแสดงบน Soft terminal [46]

เนื่องจากหน้าจอ Soft terminal มีขนาดเล็ก จึงได้ทำการบันทึกข้อมูล นำมาแสดงบน Notepad ดังรูปที่ 35

```
Data  are
0 231 32 233 161 0 71 140 245 247 0 248 245 124 204 0 107 234 202
245 0 9 217 239 173 0 190 100 167 43 0 8 70 11 24 0 177 81 243 8
0 97 195 203 47 0 114 165 181 193 0 174 167 41 30 0 245 87 149 57
0 65 129 178 228 0 140 35 38 65 0 65 208 62 237 0 50 64 157 121 0
212 149 140 234 0 193 192 97 145 0 13 135 199 239 0 145 120 3 86
0 203 79 135 42 0 67 167 176 191 0 21 58 233 39 0 137 255 20 113
0 246 1 198 209 0 21 102 66 204 0 233 46 67 37 0 222 148 140 37 0
159 89 131 102 0 61 31 47 61 0 12 231 241 125 0 86 230 94 28 0 99
61 103 24 0 241 244 147 15 0 90 210 3 11 0 166 187 165 115 0 75
190 48 175 0 94 160 199 20 0 198 124 111 114 0 130 130 209 203 0
96 207 136 89 0 224 140 159 150 0 77 120 59 216 0 57 43 58 111 0
236 110 47 231 0 112 28 66 104 152 67 154 182 56 30 75 81 108 130
21 67 205 7 237 186 125 148 60 117 246 139 133 59 125 159 173 101
94 252 9 226 233 203 25 67 85 174 34 184 27 167 126 199 183 231
228 85 178 50 7 190 128 122 231 156 158 220 206 147 46 61 226 7
125 42 250 182 128 120 15 174 10 18 133 24 209 209 184 38 168 132
249 166 204 116 110 211 21 34 44 100 212 205 15 102 134 106 168
160 74 110 3 251 42 27 95 50 125 86 243 235 13 188 68 108 140 241
106 251 77 179 170 138 178 170 45 32 255 43 8 143 225 171 48 94
117 251 40 219 165 96 48 109 123 30 150 57 98 149 64 74 157 67
211 251 186 88 149 27 232 225 209 66 152 5 108 80 41 208 24 40 36
167 193 180 210 112 125 70 127 192 245 215 89 90 73 19 33 41 154
115 211 27 222 110 34 218 19 106 125 199 33 60 43 140 46 237 78
164 240 53 49 79 250 0
```

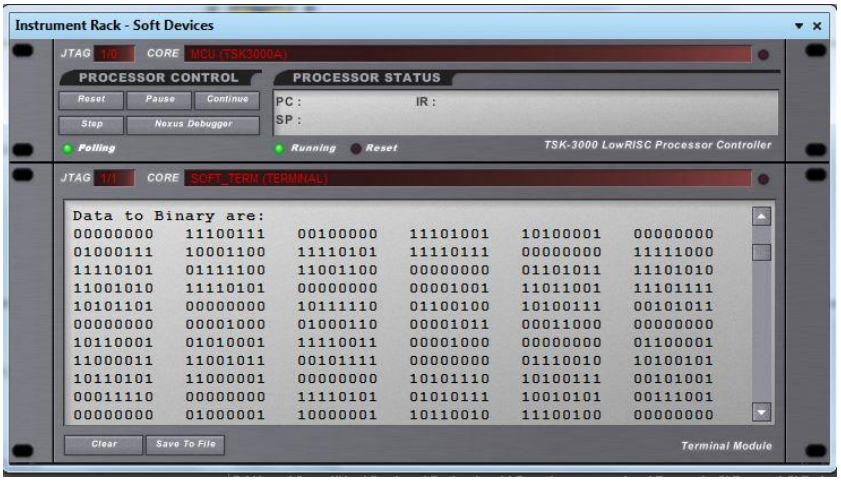
รูปที่ 35 ข้อมูลส่งเข้าบอร์ดทดลองแสดงบน Notepad [46]

รูปที่ 34 และ 35 แสดงให้เห็นว่าบอร์ดทดลองสามารถรับข้อมูลจากภายนอกได้ โดยแสดงผลข้อมูลแต่ละไบต์ในรูปแบบของเลขฐานสิบ

3.2 ส่วนวงจรเข้ารหัสภายนอก

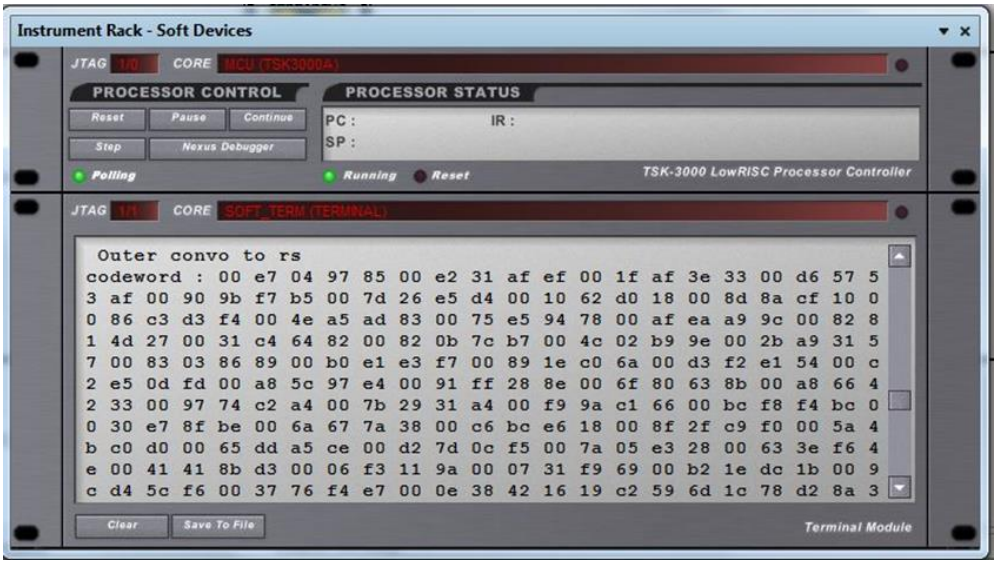
จากข้อมูลที่ได้รับขนาด 446 ไบต์ จะถูกนำมาจัดเรียงเป็น 2 สัญลักษณ์ สัญลักษณ์ละ 223 ไบต์หรือ 1,784 บิต และเข้ารหัสด้วยรหัสคอนโวลูชันภายนอก(3,2,2) แบบนอนไบนารี ก่อนการเข้ารหัสข้อมูลจะถูกเตรียม

ความพร้อมโดยแปลงข้อมูลให้เป็นไบนารี ดังรูปที่ 36 และทำการส่งเข้าไปยังตัวเข้ารหัสคอนโวลูชันภายนอก (3,2,2) ได้ผลลัพธ์เป็นคำรหัสจำนวน 3 สัญลักษณ์ สัญลักษณ์ละ 223 ไบต์หรือ 1,784 บิตเช่นกัน



รูปที่ 36 ข้อมูลก่อนการเข้ารหัสภายนอก ที่แสดงบน Soft terminal [46]

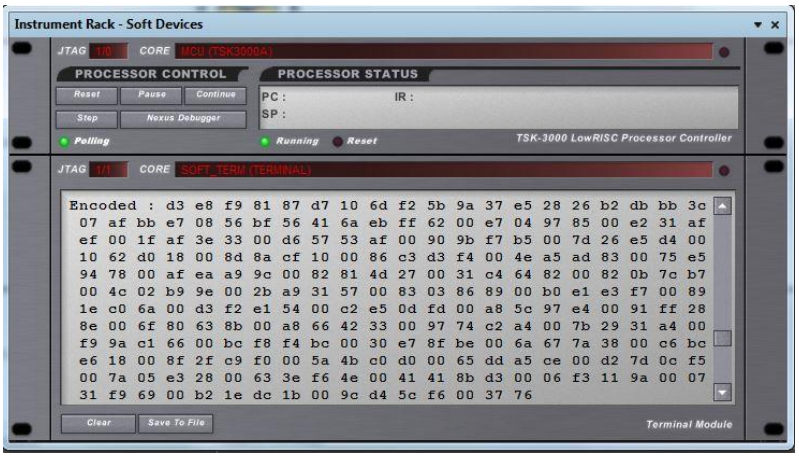
ข้อมูลที่ผ่านการเข้ารหัสภายนอกแล้ว จะถูกจัดกลุ่มใหม่อีกครั้ง เป็นกลุ่มละ 8 บิต เพื่อให้สะดวกต่อการเข้ารหัสภายในแบบ RS (255,223) ในภายหลัง เนื่องจากรหัส RS (255,223) มีขนาดสัญลักษณ์ 8 บิตต่อสัญลักษณ์ ซึ่งข้อมูลบน Soft terminal จะแสดงในรูปแบบเลขฐานสิบหก เมื่อทำการเข้ารหัสภายนอกเสร็จสิ้นแล้ว ผลที่ได้จะแสดงตามรูปที่ 37



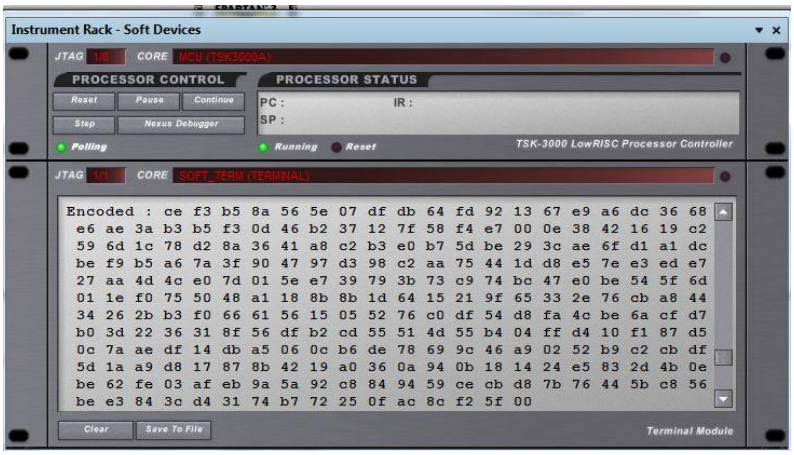
รูปที่ 37 ข้อมูลหลังจากเข้ารหัสภายนอกแสดงในรูปเลขฐานสิบหกบน Soft terminal [46]

3.3 ส่วนวงจรเข้ารหัสภายใน

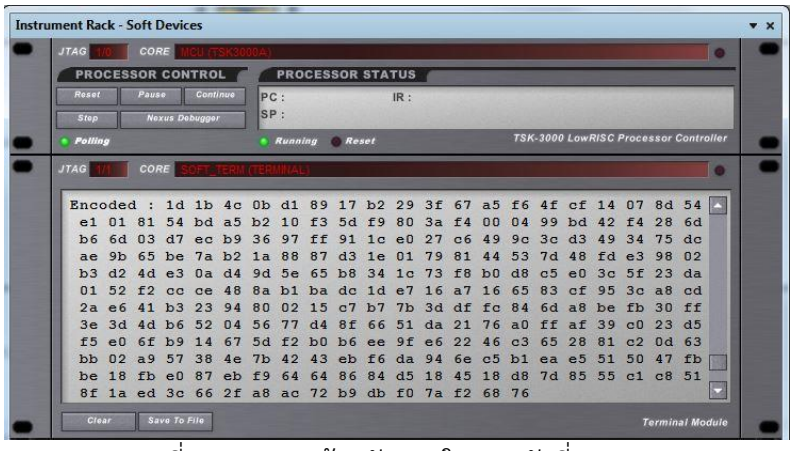
การเข้ารหัสภายในจะใช้รหัส RS (255,223) โดยรหัสภายนอกทั้ง 3 สัญลักษณ์จะถูกนำไปเข้ารหัสภายในแบบหนึ่งต่อหนึ่ง โดยทำทีละรหัสอย่างต่อเนื่องกัน ได้เป็นคำรหัส RS จำนวน 3 รหัส ดังรูปที่ 38, 39 และ 40



รูปที่ 38 ผลการเข้ารหัสภายใน RS ตัวที่ 1 [46]



รูปที่ 39 ผลการเข้ารหัสภายใน RS ตัวที่ 2 [46]

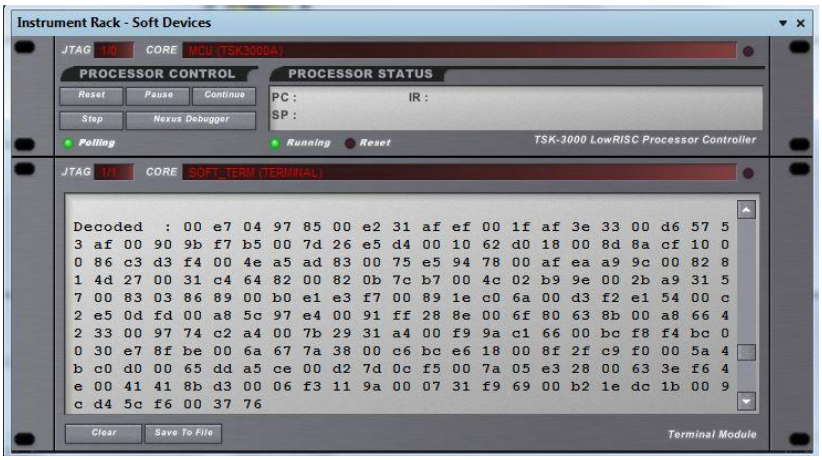


รูปที่ 40 ผลการเข้ารหัสภายใน RS ตัวที่ 3 [46]

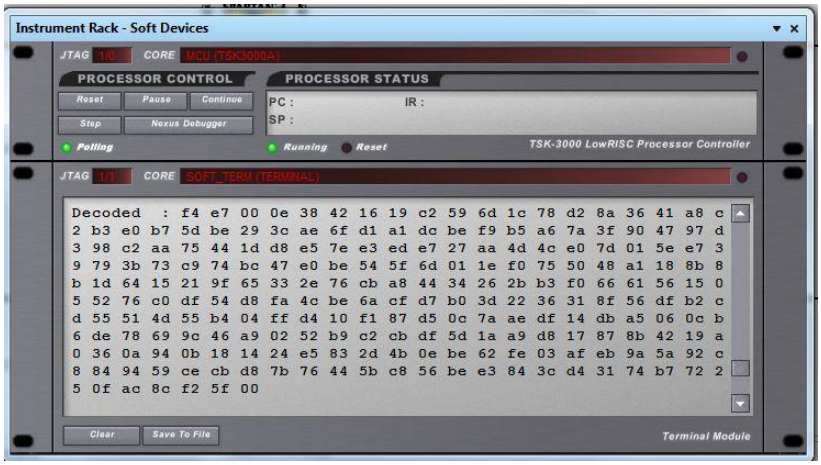
3.4 ส่วนของวงจรถอดรหัสรหัสภายใน

คำรหัสที่ได้รับหลังจากได้รับหลังจากการเข้ารหัส RS (255,223) จะเป็นคำรหัสจำนวน 3 ชุดด้วยกัน ในขั้นตอนการถอดรหัสก็จะรับข้อมูลมาครั้งละ 3 ชุดและทำการถอดรหัสทีละตัวอย่างต่อเนื่องกันเช่นกัน เพื่อที่ในขั้นตอนต่อไป ตัวถอดรหัส VSD จะสามารถนำไปใช้งานได้ทันที คำรหัสเดิมที่ได้จากขั้นตอนที่ 3.3 จะถูกกำหนดให้เกิดความผิดพลาดที่บางตำแหน่ง วงจรถอดรหัส RS ภายใน จะใช้ขั้นตอนวิธีของ Berlakamp-Massey และ

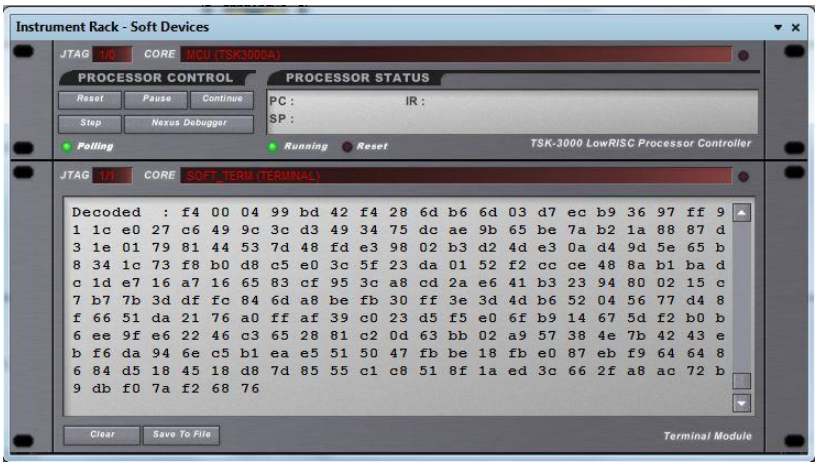
เนื่องจากรหัส RS (255,223) มีความสามารถในการแก้ไขความผิดพลาดได้ไม่เกิน 16 สัญลักษณ์ ในกรณีที่จำนวนความผิดพลาดไม่เกินความสามารถของรหัส ตัวถอดรหัสจะสามารถแก้ไขความผิดพลาดนั้นได้อย่างถูกต้อง ข้อมูลที่ได้หลังการถอดรหัสภายในจะแสดงดังรูป 41, 42 และ 43



รูปที่ 41 ผลการถอดรหัสภายใน RS ตัวที่ 1 [46]



รูปที่ 42 ผลการถอดรหัสภายใน RS ตัวที่ 2 [46]



รูปที่ 43 ผลการถอดรหัสภายใน RS ตัวที่ 3 [46]

จะเห็นว่าผลการถอดรหัส ตามรูปที่ 41 จะได้ผลที่ตรงกับข้อมูลก่อนการเข้ารหัสตามรูปที่ 37

4. การแปลงโปรแกรมตัวถอดรหัสภายนอก VSD เพื่อใช้บนบอร์ดทดลอง

สำหรับตัวถอดรหัสภายนอก VSD นั้น เดิมมีตัวโปรแกรมจำลองการทำงานที่เป็นภาษา C++ อยู่แล้ว แต่ในการไปใช้บนบอร์ดทดลอง FPGA “Altium Nanoboard 3000” จะต้องทำการแปลงจากภาษา C++ ให้เป็นภาษา C ที่สามารถใช้งานได้บนบอร์ดทดลองก่อน ซึ่งภาษา C ที่บอร์ดทดลองนี้รองรับจะมีชุดคำสั่งที่จำกัดกว่าภาษา C ทั่วไป นอกจากนั้นยังปรับปรุงแก้ไขโปรแกรม VSD ที่รองรับสัญลักษณ์ที่มีขนาดไม่ใหญ่มาก ให้สามารถรองรับสัญลักษณ์ขนาด 1,784 บิตต่อสัญลักษณ์ได้ โดยขั้นตอนในการแก้ไขโปรแกรมจะมีขั้นตอนดังนี้

4.1 การแก้ไขชุดคำสั่งที่ทำงานเหมือนกันแต่เขียนต่างกัน เช่น fout() ของ C++ กับ printf() ของ C

4.2 การแก้ไขรูปแบบต่างๆที่ภาษา C++ สามารถทำงานได้แต่ภาษา C ไม่สามารถทำงานได้ เช่น การส่งผ่านตัวแปร array หลายมิติใน C++ ไม่จำเป็นต้องระบุขนาดของแถวลำดับ ส่วนในภาษา C การส่งผ่านค่าตัวแปรนั้นจะต้องกำหนดขนาดของแถวลำดับที่เป็นคอลัมน์ดังรูปที่ 44

maincpp.cpp

```
46  void correct_with_second_choice(bool& all_zero_synd,int max_H[][48],int syndrome[][32],int mod_syndrome[][32],int guass_mod_sydrome[][32],int rec_sym[][32],int
47
48  //Function to decode with two syndromes. If the first and the second syndrome are the same, we get a null commination.
49  bool decode_many_syndrome(bool& made_correction,int num_syndrome,int max_H[][48],int mod_syndrome[][32],int syndrome[][32],int second_syndrome[32],int rec_sym[
50
51  //Function to find null combinations.
52  void find_null_comb(int guass_mod_syndrome[][32],unsigned int col[],unsigned int col2[],int num_syndrome,int null_comb[][16],int null_row[],int& null_count);
53
54  //Function to make correction using the null combination
55  bool correct_with_null_comb(bool& made_correction,int max_H[][48],int mod_syndrome[][32],int rec_sym[][32],int corrected_sym[][32],int num_syndrome,int null_co
56
```

รูปที่ 44 การประกาศฟังก์ชันที่มีการส่งผ่าน array หลายมิติบน C++ [47]

นอกจากนั้นยังมีรูปแบบของการเขียนชุดคำสั่งแบบลัด เช่น การเพิ่มค่าตัวแปรทีละ 1 บน C++ สามารถเขียนในรูปแบบของ *param2 =*param2++ ; ได้ แต่บน C การเขียนคำสั่งลักษณะนี้อาจไม่สามารถใช้ได้ในทุกๆ กรณีจึงต้องเขียนในรูปแบบเต็มให้เป็น *param2 =*param2 +1 ; จากโค้ด testparameter1 ทำให้รู้ว่าควรที่จะไปแก้ บรรทัดที่ 1440 ในรูปที่ 44 จาก *null_count = *null_count++; ให้กลายเป็น *null_count = *null_count +1;

maincpp.cpp

```
1423  //Function to find null combinations.
1424  void find_null_comb(int guass_mod_syndrome[][32],unsigned int col[],unsigned int col2[],int num_syndrome,int null_comb[][16],int null_row[],int& null_count)
1425  {
1426      for(int i = 0; i < 16;i++)
1427      {
1428          null_row[i] = -100;
1429      }
1430
1431      guass_jordan(guass_mod_syndrome,col,col2,num_syndrome,4*num_syndrome,null_row,null_count);
1432      int k = 0;
1433      null_count = 0;
1434      while((null_row[k] >= 0) && (k < 16))
1435      {
1436          //If row 0 is a dependent row, it must be an all zero row. Thus we find a null combination.
1437          if(null_row[k] == 0)
1438          {
1439              null_comb[0][0] = 0;
1440              null_count++;
1441              k++;
1442              continue;
1443          }
1444
```

รูปที่ 45 ตัวแปร null_count ที่ยังไม่ได้แก้ไขให้เป็นลักษณะภาษา C [47]

4.3 ความแตกต่างในการส่งค่าผ่านฟังก์ชันบน C++ และ C

ข้อควรระวังในการเขียนการใช้งานฟังก์ชันบน C คือ ต้องระบุให้ชัดเจนว่า ค่าที่ส่งผ่านมาหรือค่าที่จะนำไปใช้นั้น เป็น ข้อมูลจริงหรือเป็น pointer ที่ชี้ตำแหน่งของข้อมูล ซึ่งสำหรับบน C++ complier จะมีความยืดหยุ่นมากกว่า

4.4 การขยายให้รองรับสัญลักษณ์ขนาด 1,784 บิต

เมื่อโปรแกรมสามารถรันโปรแกรมได้อย่างถูกต้องในขนาด 32 บิตต่อสัญลักษณ์ จึงทำการขยายขนาดสัญลักษณ์จาก 32 บิตต่อสัญลักษณ์ ให้กลายเป็น 1,784 บิตต่อสัญลักษณ์ ในการโค้ดจะใช้การประกาศตัวแปร global ขึ้นมา หนึ่งตัวชื่อ symbol_size ดังในรูปที่ 46 บรรทัดที่ 19 เพื่อความสะดวก และรวดเร็วในการแก้ไขขนาดของสัญลักษณ์ ในการประกาศฟังก์ชันนั้น จะนำตัวแปร symbol_size มาแทนที่ค่าที่เคยถูกระบุเป็น 32 ในโปรแกรม แต่มีจุดต้องระวังว่าค่าที่จะแทนที่ด้วยตัวแปร symbol_size เกี่ยวข้องกับขนาดของสัญลักษณ์จริงหรือไม่

```
18
19  #define symbol_size 1784
20
```

รูปที่ 46 ประกาศตัวแปรชนิด global ชื่อ symbol_size [47]

ต่อมาเมื่อมีการเพิ่มขนาดสัญลักษณ์หรือปรับค่าตัวแปร symbol_size ให้เพิ่มมากขึ้นก็จะทำให้ต้องมีการประกาศตัวแปรเพิ่มในบางฟังก์ชัน ที่มีเรียกใช้จากหลายๆช่องทาง แต่มีจำนวนรอบในการวน for ไม่เท่ากัน จึงทำให้ต้องสร้างตัวแปรมารับค่าในการวน for ที่ถูกต้องเพื่อให้โปรแกรมยังทำงานได้ถูกต้อง

ในการขยายขนาดสัญลักษณ์ของตัวแปรในโปรแกรมให้เป็นขนาด 1,784 บิตต่อสัญลักษณ์ ซึ่งมีขนาดใหญ่มาก เมื่อตัวแปรมีขนาดใหญ่มาก อาจจะเกิดการจองหน่วยความจำทับกันเกิดขึ้นจึงต้องมีการประการตัวแปรเป็น static เพื่อให้จองหน่วยความจำได้ถูกต้อง ดังรูปที่ 47

[*] 1784.c

```
111     static int h[11][3] = {0};                //The repeated block that forms the H matrix
112     static int max_H[16][48] = {0};         //max_H is the maximum size H matrix that we will use.
113     int num_syndrome = 1;
114     //int input_sym[14][32] = {0};
115     static int encoded_sym[66][symbol_size] ={0};
116     static int rec_sym[66][symbol_size] = {0};          //each row contains a outer symbols. There are actually 21 outer symbols with 32 bit each
117                                         //However,we extend the length by 21 all zero symbols so that we can decode the end of the
118                                         //block up to 16 syndromes.
119     static int second_rec_sym[66][symbol_size] ;    // The second choice
120
```

รูปที่ 47 การประกาศตัวแปรแบบ static บนโปรแกรมภาษา C [47]

เมื่อทำครบตามขั้นตอนทั้งหมด และครบทุกๆบรรทัดในโปรแกรมภาษา C และรันโปรแกรมทดสอบบนคอมพิวเตอร์ ซึ่งสามารถทำงานได้อย่างถูกต้องแล้ว จึงนำไปสู่ขั้นตอนที่จะนำไปใช้งานบนบอร์ดทดลองต่อไป

สรุปและเสนอแนะ

ในงานวิจัยนี้ ได้แสดงถึงประสิทธิภาพของระบบรหัสที่นำเสนอในการจัดการกับความผิดพลาดแบบเบริสต์ โดยเฉพาะในช่องสัญญาณไร้สายแบบเคลื่อนที่ที่มักจะเกิดความผิดพลาดแบบเบริสต์ได้ง่าย ซึ่งผลที่ได้แสดงให้เห็นว่าการใช้สัญลักษณ์ขนาดใหญ่สามารถให้ประสิทธิภาพที่ดีกว่าการใช้สัญลักษณ์ขนาดเล็ก และยังได้แสดงถึงการเพิ่มประสิทธิภาพของระบบรหัสโดยภาพรวมจากการเพิ่มประสิทธิภาพตัวถอดรหัสภายในด้วยการใช้ตัวถอดรหัสร่วมกับสัญลักษณ์ที่ถูกลบ ซึ่งเมื่อตัวถอดรหัสภายในสามารถแก้ไขความผิดพลาดได้มากขึ้น ประสิทธิภาพของระบบรหัสโดยรวมก็จะดีขึ้นตามไปด้วย ผลการทดลองที่ได้แสดงให้เห็นว่า การถอดรหัสร่วมกับสัญลักษณ์ที่ถูกลบโดยการใช้ค่าขีดแบ่ง สามารถให้ประสิทธิภาพที่ดีขึ้นได้ แต่มีเงื่อนไขว่า จำเป็นจะต้องระบุค่าขีดแบ่งให้เหมาะสม ซึ่งการระบุค่าขีดแบ่งได้เหมาะสมจะทำให้ประสิทธิภาพของระบบรหัสสูงขึ้น แต่ในทางตรงกันข้ามการระบุค่าขีดแบ่งไม่เหมาะสม จะทำให้ประสิทธิภาพด้อยลงได้เช่นกัน การถอดรหัสร่วมกับสัญลักษณ์ที่ถูกลบโดยการใช้ค่าขีดแบ่ง จึงเป็นทางเลือกที่ควรใช้อย่างระมัดระวัง

สำหรับในการสร้างชิ้นงานต้นแบบบนบอร์ดทดลอง มีส่วนที่สามารถดำเนินการแล้วเสร็จและทำการทดสอบบนบอร์ดทดลองแล้ว ได้แก่ วงจรเข้ารหัสคอนโวลูชันภายนอก และวงจรเข้าและถอดรหัส RS ภายใน สำหรับวงจรถอดรหัสภายนอก VSD ได้ดำเนินการสร้างโปรแกรมภาษา C ที่พร้อมจะนำไปใช้บนบอร์ดทดลองต่อไป

ข้อเสนอแนะ

1. นำโปรแกรมถอดรหัสภายนอก VSD ไปสร้างเป็นวงจรเพื่อทดสอบการใช้งานบนบอร์ดทดลองจริง ซึ่งอาจพบความผิดพลาดที่ต้องตรวจสอบและแก้ไขต่อไป
2. เพิ่มประสิทธิภาพการทำงานของบอร์ดทดลอง เช่น เพิ่มความรวดเร็วในการทำงานของบอร์ดทดลองด้านการถอดรหัสโดยเฉพาะเมื่อต้องทำงานกับข้อมูลขนาดใหญ่ และปรับปรุงอินเตอร์เฟสในการรับ-ส่งข้อมูลให้มีความเร็วสูงขึ้น
3. นำไปใช้งานในช่องสัญญาณอื่นๆนอกเหนือจากช่องสัญญาณไร้สายเคลื่อนที่ เช่น ช่องสัญญาณผ่านสายไฟฟ้า (Power Line Communications) เป็นต้น

เอกสารอ้างอิง

[1] http://www.google.com/glass/ (access Sept. 9, 2013)

[2] http://www.engadget.com/2013/09/04/galaxy-gear-hands-on/ (access Sept. 9, 2013)

[3] www.apple.com/ipad (access Sept. 9, 2013)

[4] http://www.samsung.com/global/microsite/galaxytab/10.1/index.html (access Sept. 9, 2013)

[5] http://www.sonymobile.com/ (access Sept. 9, 2013)

[6] www.apple.com/iphone (access Sept. 9, 2013)

[7] http://www.livescience.com/38562-internet-of-things.html (access Sept. 9, 2013)

[8] http://automobileinnovations.blogspot.com/2013/06/v2v-vehicle-to-vehicle-communication.html (access Sept. 9, 2013)

[9] CCSDS 130.1-G-1, “TM Synchronization and Channel Coding Summary of Concept and Rationale,” Green book, June 2006.

[10] ISO/IEC 10149:1995, Information technology-Data interchange on read-only 120 mm optical data disks (CD-ROM), 46 pages, 1995.

[11] E.R. Berlekamp, “Nonbinary BCH decoding,” International Symposium on Information Theory, 1967, Italy.

[12] J. L. Massey, “Shift-register synthesis and BCH decoding,” IEEE Trans. Inform. Theory, vol.15, issue 1, pp. 122-127, Jan. 1969.

[13] E.R. Berlekamp, Algebraic Coding Theory, McGraw-Hill, New York, 1968.

[14] V. Guruswami and M. Sudan, “Improved decoding of Reed-Solomon codes and algebraic geometry codes,” IEEE Trans. Inform. Theory, vol. 45, no. 6, pp. 1757–1767, Sept. 1999.

[15] R. Koetter, “On optimal weight assignments for multivariate interpolation list-decoding,” IEEE Information Theory Workshop, pp. 37-41, Mar. 2006, Uruguay.

[16] L. Chen, R.A. Carrasco, E.G. Chester, “Performance of Reed–Solomon codes using the Guruswami–Sudan algorithm with improved interpolation efficiency,” IEEE Comm., vol. 1, issue 2, pp. 241-250, Apr. 2007.

[17] G.D. Forney, Jr., Concatenated Codes, MIT Press, Cambridge, 1966.

[18] พิสิฐ วนิชชานันท์, ปิยะ โควินท์ทวีวัฒน์, อุศนา ตัณฑุลเวศม์ และคณะ, ทฤษฎีรหัสช่องสัญญาณ (Channel Coding Theory), published by TRIDI (Telecommunications Research and Industrial Development Institute), 392 pages, 2009.

[19] S. Lin and D. J. Costello, Jr., Error Control Coding, 2nd edition, Prentice Hall, 1260 pages, 2004.

[20] M.K. Simon and M. Alouini, Digital Communication over Fading Channels, 2nd edition, Wiley-IEEE Press, 936 pages, 2004.

[21] N.F. Zein and T.G. Clarkson, “Optimum erasure threshold in Reed-Solomon coded differential 16 star-QAM modulation,” Electronics Letters, vol. 29, issue. 20, pp. 1746-1748, Sep 1993.

[22] I.S. Reed and G. Solomon, “Polynomial codes over certain finite fields,” SIAM J. on Applied Math., vol. 8, pp. 300-304, 1960.

[23] R.C. Singleton, “Maximum distance q-nary codes,” IEEE Trans. Inf. Theory, vol. 10, issue 2, pp. 116–118, Apr 1964.

[24] J. Jin and T. Le-Ngoc, “RS-coded M-QAM Schemes Using Error-and-erasure Decoding over Wireless Fading Channels,” 8th Communication Networks and Services Research Conference, pp. 33-39, 11-14 May 2010, Montreal, Canada.

[25] วศิน สุขตลอดชีพ, ระบบบรหัสภายในของรหัสคอนคาทิเนตที่ใช้ตัวถอดรหัสภายนอกแบบเวกเตอร์ซิมโบลสำหรับช่องสัญญาณเคลื่อนที่, วิทยานิพนธ์ปริญญาโท, มหาวิทยาลัยเกษตรศาสตร์, 91 หน้า, 2556

[26] จตุพล ท่นไชย, ตัวตนแบบระบบบรหัสชองสัญญาณแบบต่อรวมที่ใชการถอดรหัสแบบเวกเตอรซิมโบลบนซอฟทคอรโปรเซสเซอร์, วิทยานิพนธ์ปริญญาโท, มหาวิทยาลัยเกษตรศาสตร์, 88 หน้า, 2556

[27] C.E. Shannon, “A Mathematical Theory of Communications,” Bell Syst. Tech. J., pp. 379-423 (Part 1); pp.623-56 (Part 2), July 1948.

[28] G.G. Robert, Low Density Parity Check Codes, M.I.T. Press, 1963. (Retrieved Sept. 10, 2013)

[29] K. Sripimanwat, Turbo Code Applications: A Journey from a Paper to Realization, Springer, 386 pages, 2005.

[30] S. Sesia, I. Toufik, M. Baker, Long Term Evolution: From Theory to Practice, Wiley, 752 pages, 2011.

[31] T. Qin, D. Jin, G. Zheng, J. Wang, “A novel Doppler Effect testing approach for highspeed railway wireless communication systems,” Proceedings of Int. Conf. on Electronics, Communications and Control (ICECC), pp.657-660, September 2011, Ningbo, China.

[32] G. Tingting, S. Bin, “A high-speed railway mobile communication system based on LTE,” Proceedings of Int. Conf. on Electronics and Information Engineering (ICEIE), vol. 1, pp.414-417, August, 2010, Kyoto, Japan.

[33] G. Schmidt, V.R. Sidorenko, and M. Bossert, “Collaborative Decoding of Interleaved Reed–Solomon Codes and Concatenated Code Designs,” IEEE Trans. Inform. Theory, vol. 55, Issue 7, pp. 2991 – 3012, 2009.

[34] A.J. Viterbi, “Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm,” IEEE Trans. Inform. Theory, IT-13, pp. 260-269, April 1967.

[35] U. Tuntoolavest and J.J. Metzner, “Vector symbol decoding with list symbol decisions and outer convolutional codes for reliable communications,” Integrated Computer-Aided Engineering Journal, vol. 9, no. 2, pp. 101-116, 2002.

[36] J.J. Metzner and E.J. Kapturowski, “A general decoding technique applicable to replicated file disagreement location and concatenated code decoding,” IEEE Trans. Inform. Theory, vol. 36, pp. 911-917, July 1990.

[37] C. Haslach and A.J. Han Vinck, “A decoding algorithm with restrictions for array codes,” IEEE Trans. Inform. Theory, vol. 45, no. 7, pp. 2339-2344, November 1999.

[38] J.J. Metzner, “Vector symbol decoding with list inner symbol decisions,” IEEE Trans Comm., vol. 51, No. 3, March 2003.

[39] E. Hadad, L.Goldfeld, V. Lyandres, “Adaptive Erasure Decoder for Reed-Solomon code and its Performance in Rayleigh Fading Channel,” Electrical and Electronics Engineers in Israel, pp. 157-161, Dec. 2008, Israel.

[40] S.H. Jamali and T. Le-Ngoc. 1993, “Successive-erasure Decoding of RS-coded MPSK Schemes in a Rayleigh Fading Channel,” IEE Proceedings Communications, vol. 140, issue 2, pp. 152-156, 1993.

[41] C. Senger, V.R. Sidorenko, S. Schober, M. Bossert, V.V. Zyablov, “Adaptive Single–Trial Error/Erasure Decoding of Reed–Solomon Codes,” 12th Canadian Workshop Information Theory 2011, pp. 47-51, May 2011.

[42] Y.H. Chen, C.L. Chu, C.C. Yehk and K.F. Lin, “FPGA Implementation and Verification of Reed Solomon (63,47,8) Code in SDR System,” 2nd International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC), pp. 100-103, 8-10 December 2012, Harbin.

[43] L. Biard and D. Noguet, “Choice and Implementation of a reed Solomon Code for low power low data rate communication systems,”, IEEE Radio and Wireless Symposium, pp. 365-368, 9-11 January 2007, Long Beach, CA.

[44] T. Mizuochi, Y. Konishi, Y. Miyata and et al., “Experimental Demonstration of Concatenated LDPC and RS codes by FPGA emulator,” IEEE Photonics Technology Letters, vol. 21, issue 18, pp. 1302-1304, 2009.

[45] U. Tuntoolavest, Vector Symbol Decoding for Wireless Fading Channels, VDM publishing, 184 pages, 2009.

[46] นิธิ หลิ่มโตประเสริฐ และ นภทีป์ ไชยยงค์, ชิ้นงานต้นแบบเพื่อทดสอบตัวถอดรหัสเวกเตอร์ซิมโบลที่ใช้เทคนิคสเกลลิง, โครงงานวิศวกรรมไฟฟ้า, มหาวิทยาลัยเกษตรศาสตร์, 2557

[47] ธนากิต มั่นคง, โครงการการปรับปรุงโปรแกรมถอดรหัส เวกเตอร์ซิมโบลที่ใช้เทคนิคสเกลลิงให้เหมาะสมกับการทดสอบบนบอร์ดอิเล็กทรอนิกส์, โครงงานวิศวกรรมไฟฟ้า, มหาวิทยาลัยเกษตรศาสตร์, 2557