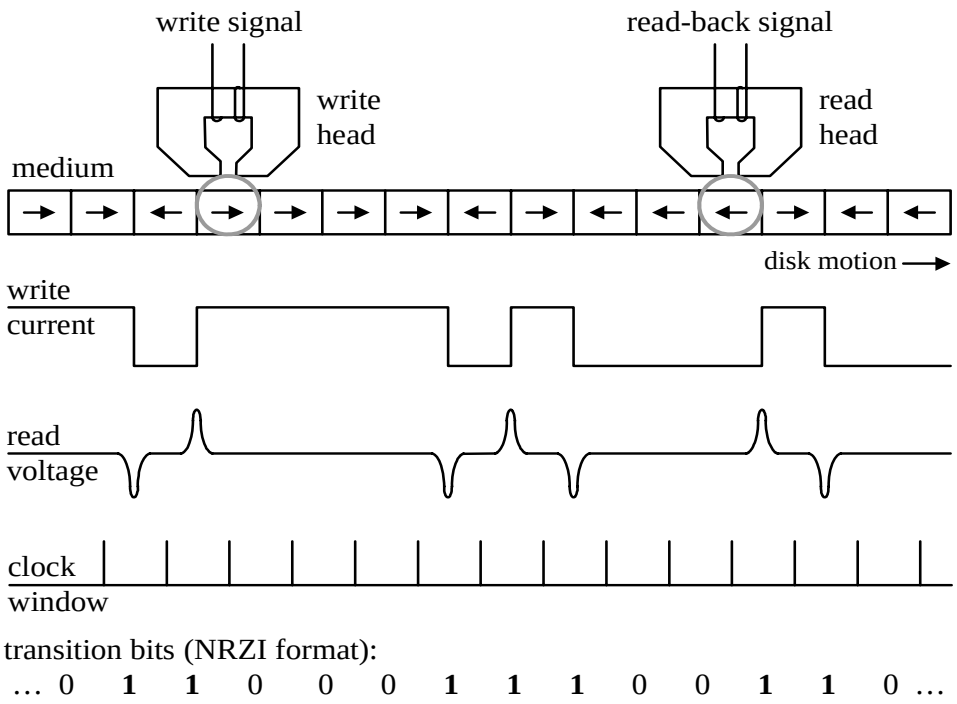
บทที่ 6

กระบวนการเขียนและการอ่านข้อมูลในฮาร์ดดิสก์ไดรฟ์

ในบทนี้จะอธิบายถึงหลักการพื้นฐานของกระบวนการเขียนและการอ่านข้อมูลในฮาร์ดดิสก์ไดรฟ์ เพื่อทำให้ผู้อ่านเข้าใจถึงภาพรวมของการทำงานของฮาร์ดดิสก์ไดรฟ์ว่า แต่ละกระบวนการประกอบไปด้วยขั้นตอนอะไรบ้าง นอกจากนี้ ยังกล่าวถึงสถาปัตยกรรมช่องสัญญาณอ่าน ซึ่งถือว่าเป็นหัวใจสำคัญของชิ้นส่วนอิเล็กทรอนิกส์ ที่ทำหน้าที่ในการประมวลผลสัญญาณของฮาร์ดดิสก์ไดรฟ์ เพราะฉะนั้นเนื้อหาในบทนี้จะช่วยทำให้ผู้อ่านสามารถทำการวิเคราะห์ระบบการประมวลผลสัญญาณของฮาร์ดดิสก์ไดรฟ์ได้ง่ายขึ้น

6.1 บทนำ

ในปัจจุบันนี้ อุปกรณ์อิเล็กทรอนิกส์ต่างๆ เช่น คอมพิวเตอร์, โทรศัพท์เคลื่อนที่, เครื่องเล่นเพลงแบบพกพา, และกล้องถ่ายรูปดิจิทัล เป็นต้น มีความต้องการเนื้อที่ในการจัดเก็บข้อมูลมากขึ้นเรื่อยๆ เทคโนโลยีการบันทึกระบบแม่เหล็กแบบดิจิทัล (digital magnetic recording) ถือได้ว่าเป็น วิธีการหลักที่ใช้ในการจัดเก็บข้อมูลของงานประยุกต์ (application) ต่างๆ รวมไปถึง ฮาร์ดดิสก์ไดรฟ์ (hard disk drive), แผ่นบันทึก (floppy disk), และแถบแม่เหล็ก (magnetic tape) อย่างไรก็ตาม ทุกงาน

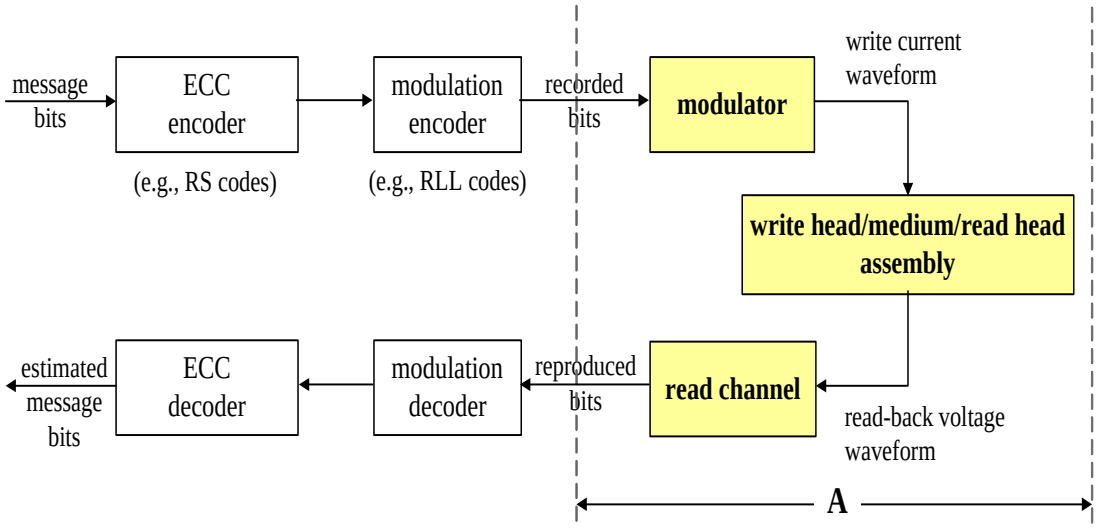


รูปที่ 6.1: หลักการพื้นฐานของการบันทึกระบบแม่เหล็กที่ใช้เทคโนโลยีการบันทึกแบบแนวนอน

ประยุกต์จะตั้งอยู่บนพื้นฐานของหลักการทำงานเดียวกันซึ่งเกี่ยวข้องกับ หัวอ่าน (read head), หัวเขียน (write head), และสื่อบันทึกแม่เหล็ก (magnetic media) ดังแสดงในรูปที่ 6.1 โดยที่ หัวอ่านและหัวเขียนแบบอินดักทีฟ (inductive head) จะทำมาจากสารแม่เหล็กรูปเกือกม้าที่มีค่าสภาพลบล้างแม่เหล็ก[1] (coercivity) ต่ำ และค่าสภาพให้ซึมผ่านได้ (permeability) สูง [1] โดยจะมีขดลวดพันอยู่รอบๆ ในขณะที่ สื่อบันทึกมักจะทำมาจากสารแม่เหล็กที่มีค่าสภาพลบล้างแม่เหล็กสูง

หนังสือเล่มนี้จะอธิบายเฉพาะเทคโนโลยีการบันทึกข้อมูล 2 แบบ คือ การบันทึกแบบแนวนอน (longitudinal recording) และการบันทึกแบบแนวตั้ง (perpendicular recording) โดยที่ เทคโนโลยีการบันทึกแบบแนวนอนเป็นเทคโนโลยีที่ใช้ในการบันทึกข้อมูลของฮาร์ดดิสก์ไดรฟ์ตั้งแต่อดีตจนถึง

[1]ค่าสภาพลบล้างแม่เหล็ก (coercivity) เป็นค่าที่บ่งบอกถึงปริมาณสนามแม่เหล็กที่ต้องการใช้ เพื่อจะเปลี่ยนทิศทางสภาพความเป็นแม่เหล็กของบริเวณที่จะบันทึกข้อมูลลงไปในสื่อบันทึก



รูปที่ 6.2: แบบจำลองทั่วไปของระบบการจัดเก็บข้อมูลดิจิทัลในฮาร์ดดิสก์ไดรฟ์

ปัจจุบัน นั่นคือ สภาพความเป็นแม่เหล็ก (magnetization) ของสื่อบันทึกจะขนานกับระนาบของจานบันทึกแม่เหล็ก (magnetic disk) ดังที่แสดงในรูปที่ 6.1 ในขณะที่ เทคโนโลยีการบันทึกแบบแนวตั้งได้เริ่มที่จะนำมาใช้สำหรับการบันทึกข้อมูลของฮาร์ดดิสก์ไดรฟ์ในปัจจุบัน[2] กล่าวคือ สภาพความเป็นแม่เหล็กของสื่อบันทึกจะตั้งฉากกับระนาบของจานบันทึกแม่เหล็ก ในปัจจุบันนี้งานวิจัยด้านเทคโนโลยีการบันทึกข้อมูลแบบแนวตั้งได้ดำเนินไปอย่างรวดเร็ว เพราะว่า สามารถช่วยเพิ่มขนาดความจุของฮาร์ดดิสก์ไดรฟ์ได้หลายสิบเท่า เมื่อเทียบกับการใช้เทคโนโลยีการบันทึกข้อมูลแบบแนวนอน [19]

ระบบการจัดเก็บข้อมูลดิจิทัล (digital data storage system) ในฮาร์ดดิสก์ไดรฟ์สามารถจำลองเป็นแผนภาพทั่วไปได้ ตามรูปที่ 6.2 เมื่อ บิตข่าวสาร (message bits) จะถูกทำการเข้ารหัสโดย “วงจรเข้ารหัสแก้ไขข้อผิดพลาด (error-correction code (ECC) encoder)” โดยที่ รหัส RS (Reed Solomon code) [42, 43] เป็นรหัสที่นิยมนำมาใช้ในการเข้ารหัสแก้ไขข้อผิดพลาดของฮาร์ดดิสก์ไดรฟ์ในปัจจุบัน จากนั้น ข้อมูลที่เข้ารหัสแล้วก็จะถูกทำการเข้ารหัสอีกครั้งหนึ่งด้วย “วงจรเข้ารหัสมอดูเลชัน (modu-

[2] บริษัทฮาร์ดดิสก์ไดรฟ์หลายๆ แห่งได้เริ่มวางจำหน่ายอุปกรณ์ฮาร์ดดิสก์ไดรฟ์ที่ใช้เทคโนโลยีการบันทึกแบบแนวตั้งสำหรับเครื่องคอมพิวเตอร์แบบพกพาในปี พ.ศ. 2549

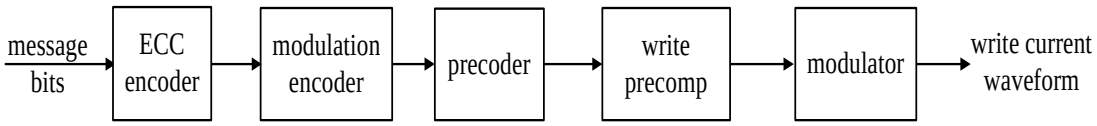
lation encoder)" เพื่อทำหน้าที่ในการปรับคุณสมบัติของข้อมูลให้เหมาะสมกับช่องสัญญาณของฮาร์ดดิสก์ไดรฟ์ เช่น ทำให้ลำดับข้อมูลมีรูปแบบตามที่ต้องการ หรือทำให้ลำดับข้อมูลไม่มีส่วนประกอบไฟฟ้ากระแสตรง (d.c. component) เป็นต้น รหัสที่นิยมใช้กันทั่วไปในวงจรเข้ารหัสมอดูเลชัน คือ รหัส RLL (run-length limited code) [44] ข้อมูลเอาต์พุตที่ได้จากวงจรเข้ารหัสมอดูเลชันจะถือว่าเป็นข้อมูลที่จะถูกเขียนเข้าไปในสื่อบันทึก ซึ่งจะเรียกกันว่า "บิตที่จะถูกบันทึก (recorded bit)" หลังจากนั้น บิตที่จะถูกบันทึกก็จะถูกส่งไปยัง "วงจรมอดูเลเตอร์ (modulator)" เพื่อแปลงข้อมูลบิตให้อยู่ในรูปคลื่นกระแสไฟฟ้าเขียน (write current waveform) จากนั้น รูปคลื่นกระแสไฟฟ้าเขียนก็จะถูกป้อนไปยังหัวเขียน เพื่อทำการเขียนข้อมูลลงไปในสื่อบันทึก

สำหรับขั้นตอนในการอ่านข้อมูล หัวอ่านจะทำการอ่านข้อมูลจากสื่อบันทึก เมื่อหัวอ่านเคลื่อนที่มาถึงบริเวณที่มีการเปลี่ยนแปลงสภาพความเป็นแม่เหล็ก[3] (ดูรูปที่ 6.1) จะได้ผลลัพธ์ออกมาเป็นสัญญาณรูปคลื่นแรงดันไฟฟ้า ที่เรียกว่า "สัญญาณ read-back" สำหรับรายละเอียดของกระบวนการเขียนและการอ่านข้อมูลจะอธิบายในหัวข้อที่ 6.2 และ 6.3 ตามลำดับ จากนั้น สัญญาณ read-back ก็จะถูกส่งเข้าไปทำการประมวลผลในช่องสัญญาณอ่าน (read channel) ซึ่งประกอบไปด้วยส่วนประกอบต่างๆ ได้แก่ วงจรกรองผ่านต่ำ (LPF: low-pass filter), วงจรชักตัวอย่าง (sampler หรือ analog-to-digital converter), อีควอไลเซอร์ (equalizer), และวงจรตรวจหา (detector) เป็นต้น โดยข้อมูลเอาต์พุตที่ได้ก็จะถูกทำการถอดรหัสด้วย วงจรถอดรหัสมอดูเลชัน (modulation decoder) และวงจรถอดรหัสแก้ไขข้อผิดพลาด (ECC decoder) เพื่อหาค่าประมาณของบิตข่าวสารที่ต้องการจะนำมาใช้งาน

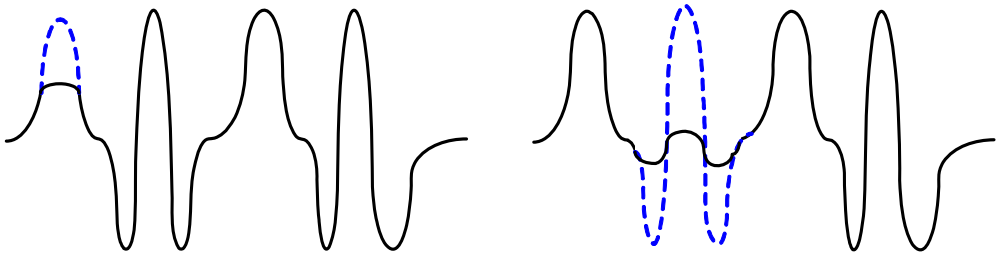
6.2 กระบวนการเขียนข้อมูล

รายละเอียดของกระบวนการเขียนข้อมูลในฮาร์ดดิสก์ไดรฟ์สามารถแสดงเป็นแผนภาพได้ตามรูปที่ 6.3 โดยที่ แต่ละบล็อกมีหลักการทำงาน ดังต่อไปนี้

[3]ในทางฮาร์ดดิสก์ไดรฟ์ บริเวณที่มีการเปลี่ยนแปลงสภาพความเป็นแม่เหล็ก จะถูกแทนด้วยข้อมูลบิต "1" และบริเวณที่มีไม่การเปลี่ยนแปลงสภาพความเป็นแม่เหล็ก จะถูกแทนด้วยข้อมูลบิต "0" โดยที่ รูปแบบข้อมูลลักษณะนี้จะเรียกกันว่า "รูปแบบ NRZI (non-return-to-zero interleaved)" เมื่อ ข้อมูลบิต "1" หมายถึง มีการเปลี่ยนสถานะ (transition) และข้อมูลบิต "0" หมายถึง ไม่มีการเปลี่ยนสถานะ



รูปที่ 6.3: แผนภาพกระบวนการเขียนข้อมูล

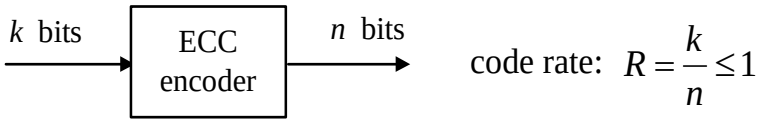


(a) single bit error (b) burst of errors

รูปที่ 6.4: ประเภทของข้อผิดพลาด (a) ข้อผิดพลาดแบบบิตเดียว และ (b) ข้อผิดพลาดแบบหลายบิตติดกัน โดยที่ เส้นประแสดงรูปร่างของสัญญาณที่ควรจะเป็น เมื่อไม่มีข้อผิดพลาด

6.2.1 วงจรเข้ารหัสแก้ไขข้อผิดพลาด

ข้อผิดพลาด (error) ที่เกิดขึ้นบ่อยในระบบการประมวลผลสัญญาณของฮาร์ดดิสก์ไดรฟ์สามารถแบ่งออกได้เป็น 2 ประเภทหลัก คือ ข้อผิดพลาดแบบบิตเดียว (single bit error) และข้อผิดพลาดแบบหลายบิตติดกัน (burst of errors) ดังแสดงในรูปที่ 6.4 โดยทั่วไป ข้อผิดพลาดแบบบิตเดียวมักจะเกิดจากการแทรกสอด (interference) ของสัญญาณรบกวนที่มีค่าแอมพลิจูดมากในช่วงระยะเวลาสั้นๆ ซึ่งจะส่งผลกระทบทำให้ได้เป็นสัญญาณพัลส์ส่วนเกินหรือสัญญาณพัลส์ที่ถูกลดทอน ดังแสดงในรูปที่ 6.4(a) โดยที่ เส้นประ (dashed line) แสดงรูปร่างของสัญญาณที่ควรจะเป็น เมื่อไม่มีข้อผิดพลาด ในขณะที่ ข้อผิดพลาดแบบหลายบิตติดกัน มักจะเกิดจากกรณีที่สื่อบันทึกมีข้อบกพร่อง (defect) เช่น มีรอยขีดข่วน หรือมีสิ่งสกปรกบนสื่อบันทึก เป็นต้น ซึ่งจะส่งผลกระทบทำให้เกิดข้อผิดพลาดหลายบิตติดกันที่วงจรภาครับ (receiver) ตามรูปที่ 6.4(b) ในทางปฏิบัติ ข้อผิดพลาดแบบหลายบิตติดกันจะก่อให้เกิดผลเสียหายกับประสิทธิภาพรวมของระบบมากกว่าข้อผิดพลาดแบบบิตเดียว



รูปที่ 6.5: แผนภาพการเข้ารหัสแก้ไขข้อผิดพลาด

รหัสแก้ไขข้อผิดพลาด (ECC: error-correction code) มีทำหน้าที่ช่วยในการแก้ไขข้อผิดพลาดที่เกิดขึ้นในระบบ โดยที่ รหัส RS (Reed Solomon) [44] จะเป็น ECC ประเภทหนึ่งที่นิยมใช้มากในฮาร์ดดิสก์ไดรฟ์ตั้งแต่อดีตจนถึงปัจจุบัน ทั้งนี้เป็นเพราะว่า รหัส RS มีความสามารถในการแก้ไขข้อผิดพลาดแบบหลายบิตติดกันได้อย่างมีประสิทธิภาพ สำหรับขั้นตอนในการเข้ารหัสที่วงจรภาคส่ง (transmiter) นั้น วงจรเข้ารหัส ECC (ECC encoder) จะทำการเข้ารหัสข้อมูลทีละบล็อกๆ ละ k บิต และได้ข้อมูลเอาต์พุตออกมาทีละบล็อกๆ ละ n บิต ตามรูปที่ 6.5 โดยที่ “อัตรารหัส (code rate)” จะนิยามโดย

$$R = \frac{k}{n} \tag{6.1}$$

เนื่องจาก จำนวนบิตเอาต์พุตจะมีมากกว่าหรือเท่ากับจำนวนบิตอินพุตเสมอ ดังนั้น $R \le 1$ เสมอ จากนั้นที่วงจรภาครับ วงจรถอดรหัส ECC (ECC decoder) ก็จะทำหน้าที่ในการถอดรหัสข้อมูลทีละบล็อกๆ ละ n บิต และได้ข้อมูลเอาต์พุตออกมาทีละบล็อกๆ ละ k บิต โดยทั่วไปแล้ว ระบบการจัดเก็บข้อมูลดิจิทัลในฮาร์ดดิสก์ไดรฟ์จะยอมให้มีอัตราข้อผิดพลาดบิต (BER: bit-error rate) $\mathrm{BER} < 10^{-9}$ เมื่อวัด ณ ด้านขาเข้าของวงจรถอดรหัส ECC

สำหรับความสามารถของรหัส RS ที่จะแก้ไขข้อผิดพลาดจำนวนกี่บิตติดกันนั้น จะขึ้นอยู่กับพารามิเตอร์ที่ใช้ในรหัส RS ซึ่งถูกกำหนดโดยพารามิเตอร์ (n, k) นั่นคือ รหัส RS มีความสามารถที่จะ

- ตรวจหา[4] (detect) ข้อผิดพลาดแบบหลายบิตติดกันที่มีความยาวไม่เกิน $(n-k)$ บิต
- แก้ไข (correct) ข้อผิดพลาดแบบหลายบิตติดกันที่มีความยาวไม่เกิน $(n-k)/2$ บิต

ตัวอย่างเช่น รหัส RS แบบ (31, 15) จะมีความสามารถในการตรวจหาข้อผิดพลาดแบบหลายบิตติด

[4]สามารถที่จะตรวจหาได้ว่า มีข้อผิดพลาดเกิดขึ้น แต่ไม่สามารถที่จะแก้ไขข้อผิดพลาดเหล่านั้นให้ถูกต้องได้

กันที่มีความยาวไม่เกิน $31 - 15 = 16$ บิต และมีความสามารถในการแก้ไขข้อผิดพลาดแบบหลายบิตติดกันที่มีความยาวไม่เกิน $(31 - 15)/2 = 8$ บิต

โดยสรุปแล้ว ข้อดีของการเข้ารหัส ECC คือ ช่วยทำให้ข้อผิดพลาดในระบบน้อยลง ซึ่งจะส่งผลทำให้ความน่าเชื่อถือของการจัดเก็บข้อมูลในฮาร์ดดิสก์ไดรฟ์เพิ่มสูงขึ้น กล่าวคือวงจรถอดรหัส ECC สามารถที่จะแก้ไขข้อผิดพลาดที่เกิดขึ้นที่วงจรภาครับได้อย่างอัตโนมัติ ส่วนข้อเสียของการเข้ารหัส ECC ก็คือ จำนวนบิตเอาต์พุตจะมีมากกว่าจำนวนบิตอินพุต โดยที่ ข้อมูลบิตที่เพิ่มขึ้นมานี้ จะเรียกว่า "บิตส่วนเกิน (redundant bit)" ซึ่งไม่ได้เป็นข้อมูลที่ผู้ใช้ (user) ต้องการจะเก็บเข้าไปในสื่อบันทึก แต่จะเป็นข้อมูลที่มีไว้ช่วยในการแก้ไขข้อผิดพลาดที่วงจรถอดรหัส ECC ดังนั้น บิตส่วนเกินจะทำให้เกิดการสูญเสียเนื้อที่การจัดเก็บข้อมูลในสื่อบันทึก ตัวอย่างเช่น แทนที่ผู้ใช้จะเก็บข้อมูลข่าวสารได้ 100 บิต ก็อาจจะเก็บข้อมูลข่าวสารได้เพียง 90 บิต เพราะต้องเหลือเนื้อที่ไว้สำหรับเก็บบิตส่วนเกินอีก 10 บิต เพราะฉะนั้น เนื้อที่การจัดเก็บข้อมูลในสื่อบันทึกจะสูญเสียไป 10% เป็นต้น

ทำไมการใช้รหัส ECC จึงช่วยเพิ่มความจุของฮาร์ดดิสก์ไดรฟ์

ผู้อ่านอาจจะสงสัยว่า ทำไมในระบบการประมวลผลสัญญาณของฮาร์ดดิสก์ไดรฟ์จึงยังใช้การเข้ารหัส ECC ทั้งๆ ที่การเข้ารหัส ECC จะทำให้เกิดบิตส่วนเกินขึ้นมาในระบบ ซึ่งจะทำให้สูญเสียเนื้อที่การจัดเก็บข้อมูลที่ต้องการในสื่อบันทึก อย่างไรก็ตาม ในความเป็นจริงแล้ว การเข้ารหัส ECC จะช่วยเพิ่มความจุของฮาร์ดดิสก์ไดรฟ์ได้ ดังที่จะอธิบายต่อไปนี้

ให้พิจารณาระบบ 2 ระบบ คือ ระบบ A ซึ่ง**ใช้**งานรหัส ECC และระบบ B ซึ่ง**ไม่ใช้**งานรหัส ECC ถ้ากำหนดให้รหัส ECC ที่ระบบ A ใช้งาน มีความสามารถในการแก้ไขข้อผิดพลาดที่เกิดขึ้นจำนวน 1 บิต ต่อข้อมูลหนึ่งบล็อกที่มีขนาด 16 บิต ซึ่งภายในข้อมูลหนึ่งบล็อกจะประกอบไปด้วย ข้อมูลที่ต้องการจำนวน 8 บิต และบิตส่วนเกิน 8 บิต ดังนั้น ระบบ A จะเกิดข้อผิดพลาดขึ้น ก็ต่อเมื่อ มีข้อผิดพลาดเกิดขึ้นในระบบมากกว่าหรือเท่ากับ 2 บิต ถ้าสมมุติให้ ความน่าจะเป็นของข้อผิดพลาด (probability of error) ในระบบมีลักษณะการแจกแจงทวินาม (binomial distribution) และให้ q_i แทนความน่าจะเป็นที่จะเกิดข้อผิดพลาดจำนวน 1 บิต ของระบบ i (เมื่อ i เท่ากับ A หรือ B) เพราะฉะนั้น ความน่าจะเป็นที่ระบบ A จะเกิดข้อผิดพลาด สามารถหาได้จากสมการ (4.39) นั่นคือ

$$P_e \approx \binom{16}{2} q_A^2 (1 - q_A)^{16-2} = 120 q_A^2 (1 - q_A)^{14} \tag{6.2}$$

ถ้ากำหนดให้ อัตราข้อผิดพลาดบิต BER $= P_e = 10^{-9}$ เป็นระดับที่ฮาร์ดดิสก์ไดรฟ์สามารถทำงานได้อย่างมีประสิทธิภาพ เมื่อแก้สมการ (6.2) แล้ว จะได้ $q_A \approx 3 \times 10^{-6}$ ซึ่งหมายความว่า ระบบ A จะเกิดข้อผิดพลาด ก็ต่อเมื่อ ความน่าจะเป็นที่จะเกิดข้อผิดพลาดจำนวน 1 บิต มีค่าเท่ากับ 3×10^{-6} ในทำนองเดียวกัน เนื่องจาก ระบบ B เป็นระบบที่ไม่ใช้รหัส ECC ดังนั้น ระบบ B จะเกิดข้อผิดพลาดก็ต่อเมื่อ ความน่าจะเป็นที่จะเกิดข้อผิดพลาดจำนวน 1 บิต มีค่าเท่ากับ $q_B = 10^{-9}$

ถ้าสมมุติให้ BER ในระบบเกิดจากสัญญาณรบกวนแบบสุ่ม (random noise) เท่านั้น เพราะฉะนั้น อัตราส่วนค่ากำลังเฉลี่ยของสัญญาณที่ต้องการต่อค่ากำลังเฉลี่ยของสัญญาณรบกวน (SNR: signal-to-noise ratio) ที่ระบบต้องการเพื่อให้ได้ BER ตามที่กำหนด สามารถที่จะคำนวณหาได้ ตัวอย่างเช่น ถ้าต้องการให้ระบบ A และ B ยังคงสามารถทำงานได้อย่างมีประสิทธิภาพ เพราะฉะนั้น ระบบ B (ไม่ใช่รหัส ECC) จะต้องมี BER = $q_B = 10^{-9}$ ซึ่งจะเกิดขึ้นได้ ก็ต่อเมื่อ ระบบ B ใช้ SNR = 22 เดซิเบล (dB) ในทำนองเดียวกัน ระบบ A (ใช้รหัส ECC) จะมี $q_A = 3 \times 10^{-6}$ เพื่อทำให้ได้ BER = 10^{-9} ก็ต่อเมื่อระบบ A ใช้ SNR = 19 dB (ค่า SNR ที่ใช้จะขึ้นอยู่กับค่า q_i ไม่ได้ขึ้นกับ BER) ดังนั้น จะถือได้ว่า ระบบ A จะมีอัตราการขยาย (gain) เท่ากับ 3 dB เมื่อเปรียบเทียบกับระบบ B ซึ่งในทางปฏิบัติ ค่าอัตราการขยายที่ได้มาจากการใช้รหัส ECC นี้ จะเรียกว่า “อัตราการขยายการเข้ารหัส (coding gain)” [42, 43] โดยที่ ค่าอัตราการขยายการเข้ารหัสในรูปของ SNR นี้ จะช่วยเพิ่ม “ความหนาแน่นการบันทึก (recording density)” ข้อมูลในฮาร์ดดิสก์ไดรฟ์ได้

พิจารณากรณีที่สัญญาณรบกวนหลักที่เกิดขึ้นในระบบ คือ สัญญาณรบกวนสื่อบันทึก (media noise) ในหนังสือ [1] ได้กล่าวว่า “การลดความกว้างของแทร็ก (track width) ลงครึ่งหนึ่ง จะเป็นผลทำให้ระบบสูญเสีย SNR ไป 3 dB” ดังนั้น ถึงแม้ว่า ระบบ A (ใช้รหัส ECC) จะต้องสูญเสียเนื้อที่การจัดเก็บข้อมูลในฮาร์ดดิสก์ไดรฟ์ไป 50% สำหรับบิตส่วนเกิน แต่เนื่องจาก ระบบ A มีอัตราการขยายการเข้ารหัส 3 dB (เมื่อเทียบกับระบบ B) ทำให้สามารถลดความกว้างของแทร็กในฮาร์ดดิสก์ไดรฟ์ของระบบ A ลงครึ่งหนึ่งได้ ซึ่งก็หมายความว่า ระบบ A จะมีเนื้อที่การจัดเก็บข้อมูลเพิ่มขึ้น 100% เพราะฉะนั้น อัตราการขยายสุทธิที่เกิดจากการใช้รหัส ECC คือ ความจุ (capacity) ของการจัดเก็บข้อมูลในฮาร์ดดิสก์ไดรฟ์จะเพิ่มขึ้นประมาณ $100\% - 50\% = 50\%$ ดังนั้น จึงสามารถที่จะสรุปได้ว่า การใช้รหัส ECC จะช่วยเพิ่มความจุของการจัดเก็บข้อมูลได้

แนวโน้มการใช้งานรหัส ECC ในอนาคต

การใช้งานฮาร์ดดิสก์ไดรฟ์ในปัจจุบันจะพบว่า วงจรถอดรหัส ECC จะทำงานเป็นอิสระ (independent) กับวงจรตรวจหาสัญลักษณ์ (symbol detector) อย่างไรก็ตาม จากการศึกษาค้นคว้าพบว่า [45] ถ้าออกแบบให้วงจรถอดรหัส ECC และวงจรตรวจหาสัญลักษณ์สามารถทำงานร่วมกันได้ ประสิทธิภาพรวมของระบบที่ได้ก็จะเพิ่มขึ้นอย่างมาก ดังนั้น แนวโน้มการใช้งานรหัส ECC ในอนาคต ระบบการประมวลผลสัญญาณของฮาร์ดดิสก์ไดรฟ์

อาจจะนำเทคนิคการใช้งานร่วมกันระหว่าง วงจรถอดรหัส ECC และวงจรตรวจหาสัญลักษณ์มาใช้งานจริงนั้น มีความเป็นไปได้สูง โดยรหัส ECC ที่จะนำมาใช้นอกเหนือไปจากรหัส RS ได้แก่ รหัส LDPC (low-density parity-check) [46] หรือรหัสเทอร์โบ (turbo code) [47] เป็นต้น

6.2.2 วงจรเข้ารหัสมอดูเลชัน

โดยทั่วไป รหัสมอดูเลชัน (modulation code) จะทำหน้าที่ในการจัดการกับความผิดเพี้ยน (distortion) ของช่องสัญญาณ และสัญญาณรบกวนต่างๆ ที่เกิดขึ้นระหว่างการส่งสัญญาณ สำหรับในระบบการบันทึกแม่เหล็ก (magnetic recording system) รหัสมอดูเลชันจะทำหน้าที่ได้หลายๆ แบบ ได้แก่ ทำให้ส่วนประกอบไฟฟ้ากระแสตรงหมดไป (หรือทำให้เหลือน้อยที่สุด), ช่วยปรับคุณสมบัติของสัญญาณให้เหมาะสมกับช่องสัญญาณ เพื่อทำให้สัญญาณที่ส่งไปมีความผิดเพี้ยนน้อยที่สุด, ช่วยเพิ่มระยะห่างของ “บิตเปลี่ยนสถานะ (transition bit)” ที่จะเขียนลงไปในสื่อบันทึก, และช่วยลดผลกระทบของการแทรกสอดระหว่างสัญลักษณ์ (ISI: intersymbol interference) เป็นต้น

รหัสมอดูเลชันที่นิยมใช้กันทั่วไปในฮาร์ดดิสก์ไดรฟ์ คือ รหัส RLL (run-length limited) [44] ซึ่งเป็นรหัสที่ทำหน้าที่ในการกำหนดจำนวนของบิตเปลี่ยนสถานะ บิต “0” และบิต “1” (ตามรูปแบบของ NRZI) ที่เรียงติดกันในลำดับข้อมูลที่ต้องการจะเขียนลงไปในสื่อบันทึก โดยทั่วไป รหัส RLL จะถูกกำหนดด้วยพารามิเตอร์ k/n (d, k) เมื่อ

- พารามิเตอร์ k คือ จำนวนอินพุตบิตที่จะทำการเข้ารหัส 1 ครั้ง
- พารามิเตอร์ n คือ จำนวนเอาต์พุตบิตที่ได้จากการเข้ารหัส 1 ครั้ง (อัตรารหัส = k/n)
- พารามิเตอร์ d คือ เลขจำนวนเต็ม ซึ่งกำหนดจำนวนของบิต 0 ที่น้อยที่สุด ที่อยู่ระหว่างบิต 1