Detection and Correction Error

Introduction to Computer Science

(88612159)

หัวข้อ

- ชนิดของความผิดพลาด
- การตรวจสอบความผิดพลาดเบื้องต้น
- การแก้ไขความผิดพลาด

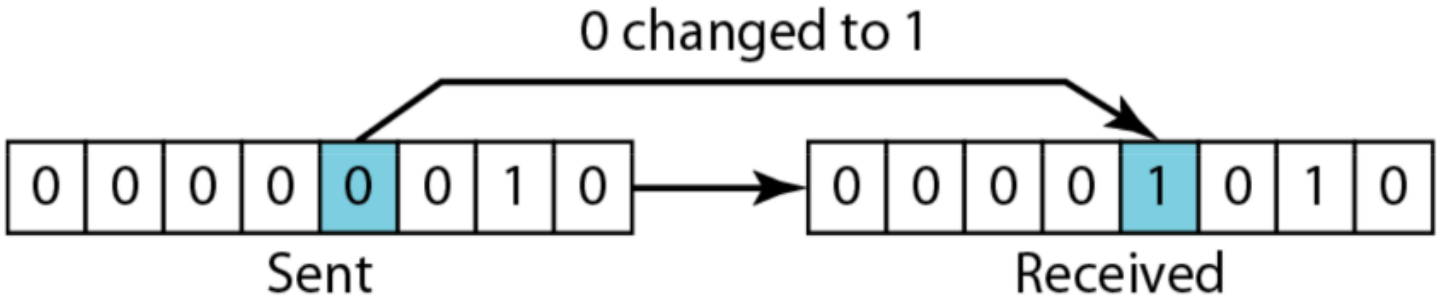
การตรวจสอบและแก้ไขความผิดพลาดของข้อมูล

- การส่งข้อมูล
- ระหว่างอุปกรณ์คอมพิวเตอร์
- ระหว่างเครื่องคอมพิวเตอร์ผ่านระบบเครือข่าย
- การจัดเก็บข้อมูลในสื่อบันทึกข้อมูล
- อาจเกิดผิดพลาด (Error) จาก สัญญาณรบกวน (Noise)
- วิธีการเข้ารหัสข้อมูล เพื่อให้สามารถตรวจสอบความผิดพลาด
- วิธีที่สามารถแก้ไขข้อมูลที่ผิดพลาดนั้นให้ถูกต้อง

ชนิดของความผิดพลาด

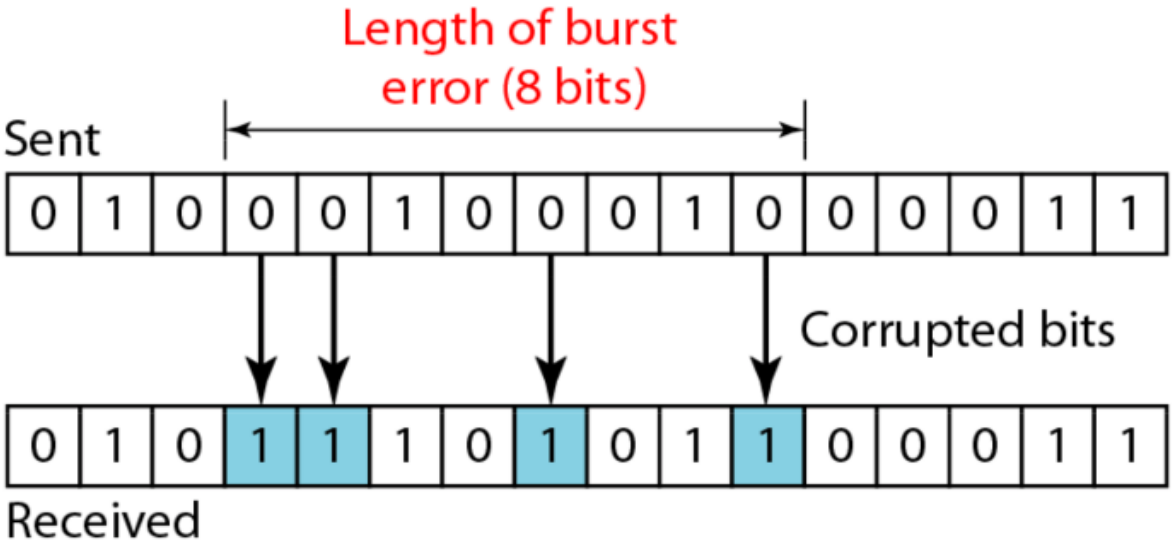
ชนิดของความผิดพลาด

1. ความผิดพลาดชนิดบิตเดียว (Single Bit Error)



ชนิดของความผิดพลาด

2. ความผิดพลาดชนิดหลายบิต (Burst Error)



การตรวจสอบความผิดพลาดเบื้องต้น

Redundancy

"บนขนนมีรถยนต์วิ่ง"

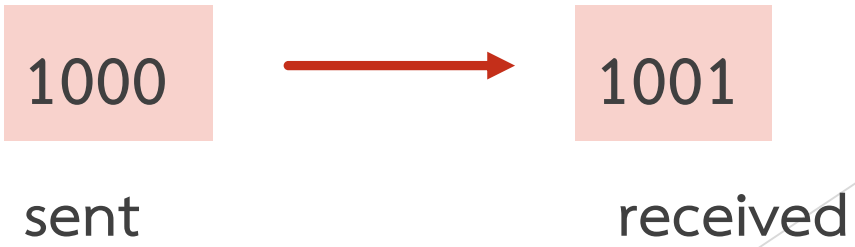
เราสามารถรู้ได้ว่าข้อความที่ถูกต้องคือ **มนุษย์มี Redundancy**

"บนถนนมีรถยนต์วิ่ง"

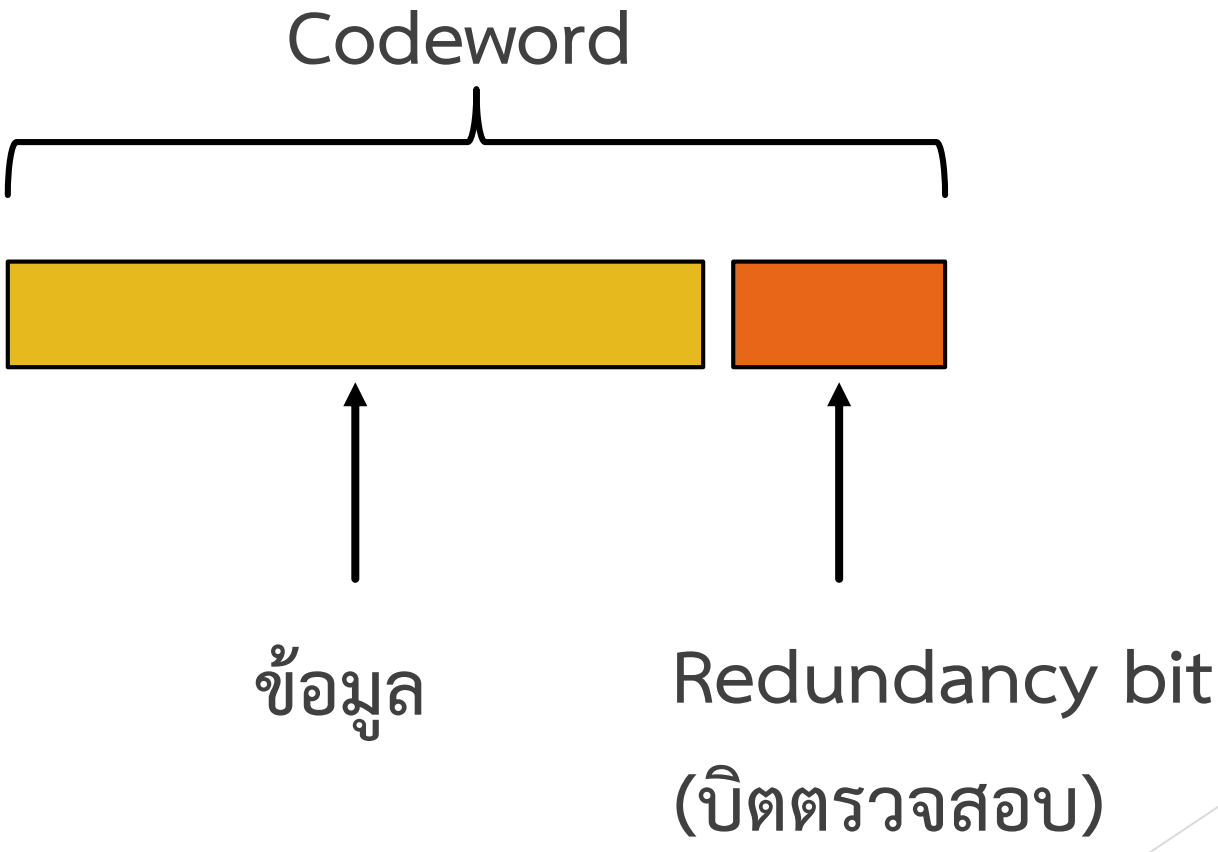
ในระบบคอมพิวเตอร์ ข้อมูลดิจิตอล

ก = 1000 และ ข = 1001

ระบบดิจิตอลนั้นไม่มี Redundancy



Codeword



วิธีการตรวจสอบความผิดพลาด

1. การใช้บิตตรวจสอบ (Parity Check)
   - แบบเชิงเส้น (Linear Parity Check)
   - แบบ 2 มิติ (2 Dimension Parity Check)
2. การใช้ผลรวมตรวจสอบ (Checksum)

1. การใช้บิตตรวจสอบ (Parity Check)

- ผู้ส่งจะมีระบบคำนวณหาบิตตรวจสอบ
- บิตตรวจสอบมีจำนวน 1 บิต และนำไปต่อท้ายข้อมูล
- รูปแบบของบิตแต่ละชุดมีจำนวนบิตที่มีค่า 1 เป็นจำนวนคี่หรือจำนวนคู่
  - Even Parity จำนวนบิตที่มีค่า 1 เป็นจำนวนคู่
  - Odd Parity จำนวนบิตที่มีค่า 1 เป็นจำนวนคี่
- ผู้ส่งและผู้รับจะตกลงกันก่อนว่าจะใช้รูปแบบบิตตรวจสอบแบบใด

Linear Parity Check

- การตรวจสอบข้อมูล 1 หน่วย ที่เป็นสายของบิต 1 สาย

<u>ตัวอย่างที่ 1</u> การเข้ารหัส ASCII ของอักษร 'A' เป็น Codeword โดยกำหนดให้ใช้บิตตรวจสอบชนิด Odd parity

รหัส ASCII ฐานสองของ 'A' คือ

0	1	0	0	0	0	0	1

ดังนั้น Codeword ของ 'A' คือ

Linear Parity Check

ตัวอย่างที่ 2 การเข้ารหัส ASCII ของอักษร 'A' เป็น Codeword โดยกำหนดให้ใช้บิตตรวจสอบชนิด Even parity

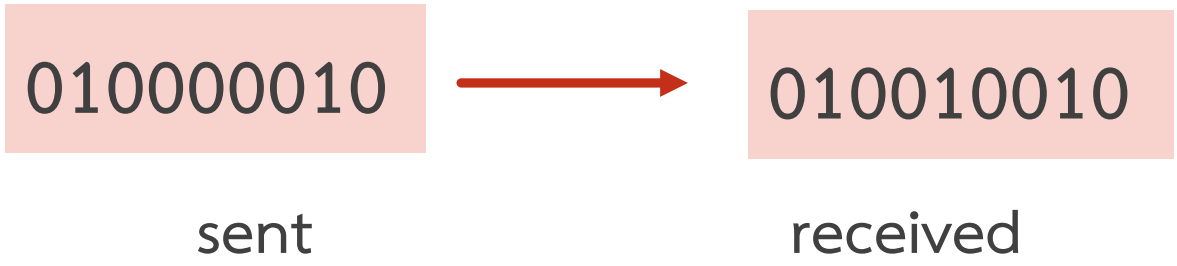
รหัส ASCII ฐานสองของ 'A' คือ

0	1	0	0	0	0	0	1

ดังนั้น Codeword ของ 'A' คือ

Linear Parity Check

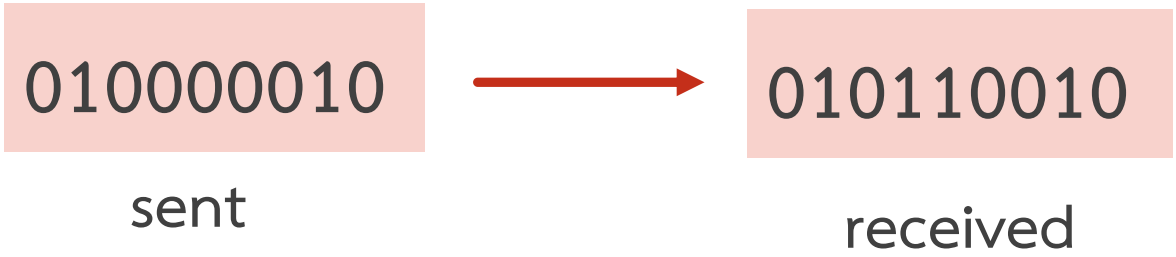
- หากผู้ส่งและผู้รับตกลงกันว่า บิตตรวจสอบเป็นชนิด Even parity



- ผู้รับพบว่าข้อมูลไม่เป็น Even parity จะทำการปฏิเสธ (Reject)

Linear Parity Check

- หากผู้ส่งและผู้รับตกลงกันว่า บิตตรวจสอบเป็นชนิด Even parity



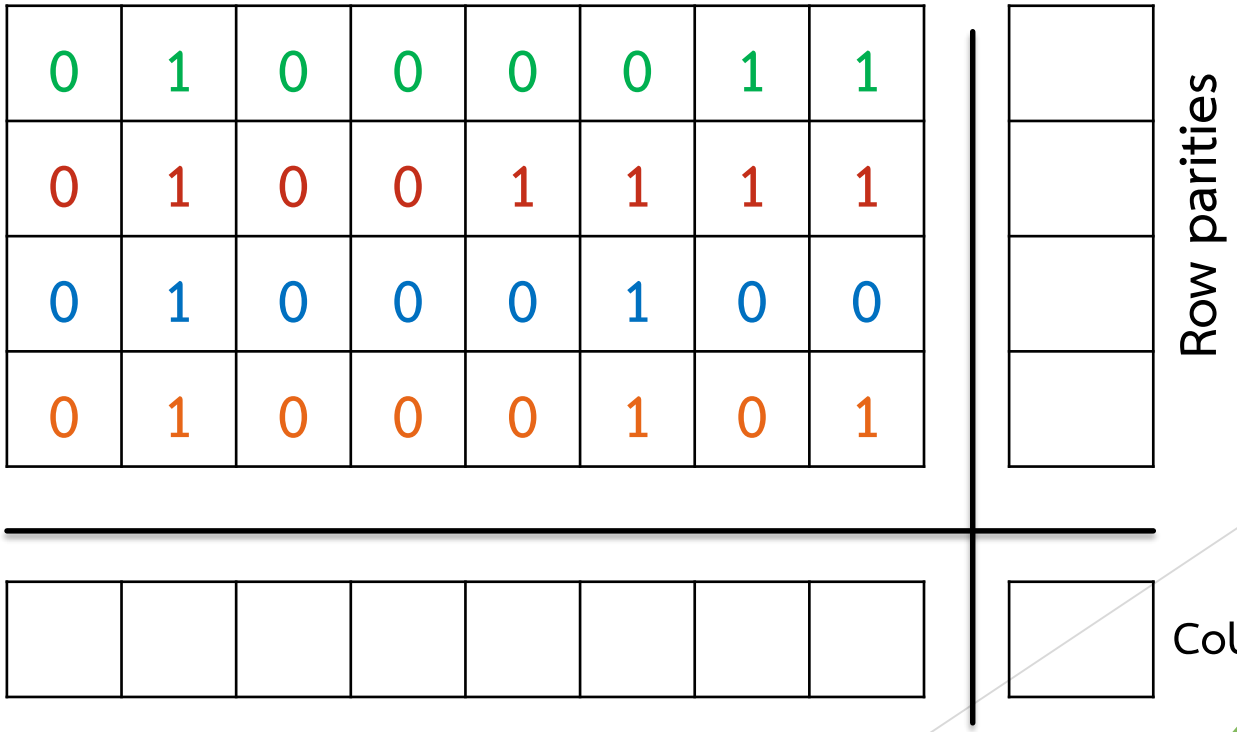
- ผู้รับพบว่าข้อมูลเป็น Even parity จะทำการยอมรับ (Accept)
- การใช้บิตตรวจสอบแบบเชิงเส้นตรวจสอบความผิดพลาดชนิดบิตเดียวเท่านั้น

2–Dimensional Parity

- จัดข้อมูลออกเป็น n หน่วยข้อมูล แล้วสร้าง Matrix
- ตัวอย่าง ต้องการส่งข้อมูล ASCII 4 ตัวอักขระ

รหัส ASCII ฐานสองของ “CODE” คือ

01000011 01001111 01000100 01000101



2–Dimensional Parity

- นำมาคำนวณบิต Parity ในแนว Row และ Column
- ผู้รับและผู้ส่งตกลงใช้บิตตรวจสอบชนิด Even parity

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

Row parities

Column parities

Codeword คือ

010000111 010011111 010001000 010001011 000011011

2–Dimensional Parity

010000111 010011111 010001000 010001011 000011011

- ผู้รับเมื่อได้รับ Codeword แล้ว จะตรวจสอบว่าเป็น Even parity ทุก Row ทุก Column หรือไม่

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

Row parities

Column parities

เป็น Even parity ทั้งหมด

Accept

2–Dimensional Parity

เมื่อนำข้อมูล Codeword มาตรวจสอบแบบ Even parity แล้วถูกต้องทั้งหมด

010000111 010011111 010001000 010001011 000011011

นำบิตตรวจสอบทั้งหมดออออก แล้วจะได้รหัส ASCII ฐานสองของ “CODE” คือ ตามที่ผู้ส่ง ส่งมา ดังนี้

01000011 01001111 01000100 01000101

2–Dimensional Parity

จากตัวอย่าง หากผู้รับได้รับข้อมูลที่ผิดพลาด 1 บิต (ตำแหน่งสีน้ำเงิน) เป็นดังนี้

010000111 010011111 010011000 010001011 000011011

ผู้รับตรวจสอบว่าเป็น Even parity ทุก Row ทุก Column หรือไม่

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

Row parities

Column parities

!!!ไม่เป็น Even parity ทั้งหมด

Reject

2–Dimensional Parity

จากตัวอย่าง หากผู้รับได้รับข้อมูลที่ผิดพลาดหลายบิต (ตำแหน่งสีนำเงิน) เป็นดังนี้

110000111 010011011 010011000 010001011 000011011

ผู้รับตรวจสอบว่าเป็น Even parity ทุก Row ทุก Column หรือไม่

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

Row parities

Column parities

!!!ไม่เป็น Even parity ทั้งหมด

Reject

2. การใช้ผลรวมตรวจสอบ (Checksum)

- ใช้ในกรณีที่มีข้อมูลเป็นกลุ่มใหญ่
- วิธีการ ดังนี้

1. แบ่งข้อมูลต้นฉบับออกเป็นหน่วย ซึ่งมีความยาว n bits
2. คำนวณหาผลรวม (sum) โดยนำข้อมูลในแต่ละหน่วยมาบวกกัน ผลลัพธ์ที่ได้จะมีขนาด n bits
3. คำนวณ Checksum โดยนำผลลัพธ์ที่ได้ในข้อ 2 ทำการกลับบิต (Complement)
4. นำ Checksum ต่อท้ายข้อมูลต้นฉบับ

2. การใช้ผลรวมตรวจสอบ (Checksum)

ตัวอย่าง ให้ส่งข้อมูลจำนวน 16 bits โดยแบ่งออกเป็น 2 หน่วย แต่ละหน่วยมีความยาว 8 bits ดังนี้

10101001 00111001

2. การใช้ผลรวมตรวจสอบ (Checksum)

นำจำนวนทั้ง 2 หน่วย มาบวกกันด้วยวิธีดังนี้

	1	0	1	0	1	0	0	1	+
	0	0	1	1	1	0	0	1	
Sum	1	1	1	0	0	0	1	0	
Checksum									

ดังนั้นข้อมูล Codeword คือ

2. การใช้ผลรวมตรวจสอบ (Checksum)

- การตรวจสอบความผิดพลาด
- เมื่อนำ Codeword ทั้งหมด แบ่งออกเป็นหน่วย แล้วบวกเข้าด้วยกัน จากนั้นทำ Complement
  - ผลลัพธ์เป็น 0 หมายถึง ไม่มีความผิดพลาดใดๆ
  - ผลลัพธ์ที่ไม่ใช่ 0 หมายถึง มีความผิดพลาด

2. การใช้ผลรวมตรวจสอบ (Checksum)

- จากตัวอย่าง หากต้องการตรวจสอบความผิดพลาด
- กรณีไม่มีความผิดพลาดเกิดขึ้นเลย ดังนี้

10101001 00111001 00011101

2. การใช้ผลรวมตรวจสอบ (Checksum)

นำจำนวนทั้ง 3 หน่วย มาบวกกันด้วยวิธีดังนี้

1	0	1	0	1	0	0	1	
0	0	1	1	1	0	0	1	+
0	0	0	1	1	1	0	1	

ผลลัพธ์ที่ได้เป็น 0 ซึ่งบ่งชี้ว่า**ไม่เกิดความผิดพลาดขึ้น**

Accept

2. การใช้ผลรวมตรวจสอบ (Checksum)

- จากตัวอย่าง หากต้องการตรวจสอบความผิดพลาด
- กรณีมีความผิดพลาดหลายบิต (ตำแหน่งสีนำเงิน) ดังนี้

10101001 00111001 00011101

2. การใช้ผลรวมตรวจสอบ (Checksum)

นำจำนวนทั้ง 3 หน่วย มาบวกกันด้วยวิธีดังนี้

1	0	1	0	1	1	1	1	
1	1	1	1	1	0	0	1	+
0	0	0	1	1	1	0	1	

ผลลัพธ์ที่ได้มีค่าไม่เป็น 0 ซึ่งบ่งชี้ว่าเกิดความผิดพลาดขึ้น

Reject

การแก้ไขความผิดพลาด

Hamming code

- การตรวจสอบและแก้ไขความผิดพลาดชนิดนี้ต้องสามารถระบุได้ 2 ประการ คือ

  (1) มีความผิดพลาดเกิดขึ้นหรือไม่ และ

  (2) ความผิดพลาดที่เกิดขึ้นนั้นอยู่ในตำแหน่งใด

- Richard Wesley Hamming (1915-1998) นักคณิตศาสตร์ชาวอเมริกันได้คิดวิธีการเข้ารหัสที่สามารถตรวจสอบและแก้ไขความผิดพลาดได้ในตัวเอง (Error-correcting code)

  เรียกว่า Hamming code

Redundancy Bit

- หากต้องการระบุตำแหน่งที่เกิดความผิดพลาดสำหรับข้อมูลขนาด 8 บิต ต้องสามารถระบุ สถานะของข้อมูล 9 สถานะ

0000 ไม่มี error	0001 Error บิตที่ 1	0010 Error บิตที่ 2
0011 Error บิตที่ 3	0100 Error บิตที่ 4	0101 Error บิตที่ 5
0110 Error บิตที่ 6	0111 Error บิตที่ 7	1000 Error บิตที่ 8

การคำนวณหาจำนวนบิตตรวจสอบ

- ความสัมพันธ์ระหว่างความยาวของข้อมูลและบิตตรวจสอบดังนี้

$$2^r \geq m + r + 1$$

ข้อมูลที่ต้องการส่ง m บิต

บิตตรวจสอบ r บิต

<u>ตัวอย่าง</u> หากต้องการส่งข้อมูลขนาด 8 บิต

ดังนั้น $2^r \geq 8 + r + 1$

จะได้ $r =$

ดังนั้นต้องส่งข้อมูลทั้งหมด $m + r =$ บิต

การคำนวณหาบิตตรวจสอบและสร้าง Codeword

- <u>ตัวอย่าง</u> ต้องการส่งข้อมูลขนาด 8 บิต ดังนี้

0	1	0	0	1	1	0	1

- เมื่อคำนวณหาจำนวนบิตตรวจสอบแล้ว จะได้ 4 บิต

การคำนวณหาบิตตรวจสอบและสร้าง Codeword

- สร้างรูปแบบบิต Codeword
- ให้บิตตรวจสอบอยู่ในตำแหน่งที่ 1, 2, 4 และ 8
- นำข้อมูลมาจัดวางในตำแหน่งที่เหลือ ดังนี้

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	?	?	0	?	1	0	0	?	1	1	0	1

การคำนวณหาบิตตรวจสอบและสร้าง Codeword

- คำนวณหาบิตตรวจสอบชนิด Even parity ในแต่ละตำแหน่ง ดังนี้

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	?	?	0	?	1	0	0	?	1	1	0	1

ตำแหน่งที่ 1: ? 0 1 0 1 0 ดังนั้น ? = 0

ตำแหน่งที่ 2: ? 0 0 0 1 0 ดังนั้น ? = 1

ตำแหน่งที่ 4: ? 1 0 0 1 ดังนั้น ? = 0

ตำแหน่งที่ 8: ? 1 1 0 1 ดังนั้น ? = 1

การคำนวณหาบิตตรวจสอบและสร้าง Codeword

- นำบิตตรวจสอบใส่ในแต่ละตำแหน่ง

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	0	1	0	0	1	0	0	1	1	1	0	1

- ดังนั้นข้อมูล Codeword ที่จะต้องส่งไปยังผู้รับคือ

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

การตรวจสอบความผิดพลาด Hamming Code

- การตรวจสอบตำแหน่งที่ผิดพลาดสามารถทำได้เหมือนเดิม
- คำนวณหาในแต่ละกลุ่มว่าเป็น Even parity หรือไม่
  - กลุ่มใดไม่เป็น Even parity แสดงว่า**เกิดความผิดพลาด** จะให้สถานะเป็น 1
  - กลุ่มใดเป็น Even parity แสดงว่า**ไม่เกิดความผิดพลาด** จะให้สถานะเป็น 0

การตรวจสอบความผิดพลาด Hamming Code

- การตรวจสอบว่าข้อมูลที่ส่งมาผดพลาดหรือไม่ หากผิดพลาด ผิดที่ตำแหน่งใด
- หากทำการส่งไปยังผู้รับแล้วไม่มีบิตที่ผิดพลาด

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	0	1	0	0	1	0	0	1	1	1	0	1

การตรวจสอบความผิดพลาด Hamming Code

ตรวจสอบความผิดพลาด

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	0	1	0	0	1	0	0	1	1	1	0	1

ตำแหน่งที่ 1: 0 0 1 0 1 0 เป็นพาริตี้คู่ สถานะ 0

ตำแหน่งที่ 2: 1 0 0 0 1 0 เป็นพาริตี้คู่ สถานะ 0

ตำแหน่งที่ 4: 0 1 0 0 1 เป็นพาริตี้คู่ สถานะ 0

ตำแหน่งที่ 8: 1 1 1 0 1 เป็นพาริตี้คู่ สถานะ 0

จากนั้นเรียงสถานะจากล่างขึ้นบนจะได้ 0000 ไม่มีความผิดพลาด

Accept

การตรวจสอบความผิดพลาด Hamming Code

- การตรวจสอบว่าข้อมูลที่ส่งมาผดพลาดหรือไม่ หากผิดพลาด ผิดที่ตำแหน่งใด
- หากทำการส่งไปยังผู้รับแล้วมีข้อมูลผิดพลาดตำแหน่งที่ 9

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	0	1	0	0	1	0	0	1	0	1	0	1

การตรวจสอบความผิดพลาด Hamming Code

ตรวจสอบความผิดพลาด

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	0	1	0	0	1	0	0	1	0	1	0	1

ตำแหน่งที่ 1: 0 0 1 0 0 0 เป็นพาริตี้คี่ สถานะ 1

ตำแหน่งที่ 2: 1 0 0 0 1 0 เป็นพาริตี้คู่ สถานะ 0

ตำแหน่งที่ 4: 0 1 0 0 1 เป็นพาริตี้คู่ สถานะ 0

ตำแหน่งที่ 8: 1 0 1 0 1 เป็นพาริตี้คี่ สถานะ 1

จากนั้นเรียงสถานะจากล่างขึ้นบนจะได้ 1001 มีความผิดพลาด ตำแหน่งที่ 9

แก้ไข

การตรวจสอบความผิดพลาด Hamming Code

- บิตในตำแหน่งที่ 9 มีความผิดพลาด
- ทำการแก้ไข โดยกลับบิตจาก 0 เป็น 1
- ผู้รับจะได้รับข้อมูลเดิมที่ถูกต้อง

ตำแหน่ง	1	2	3	4	5	6	7	8	9	10	11	12
Codeword	0	1	0	0	1	0	0	1	1	1	0	1

Accept