เทคนิคการตรวจสอบความถูกต้องของข้อมูล

Error Detection and Correction

กล่าวนำ

การรับส่งข้อมูลสิ่งที่เป็นหัวใจในการดำเนินการคือ ความถูกต้องตรงกันของข้อมูล ซึ่งในวิธีการรับส่งไม่ว่าจะเป็นแบบแอนะล็อกหรือแบบดิจิตอล ต่างก็ให้ความสำคัญต่อสิ่งนี้เช่นเดียวกัน แต่ในการรับส่งข้อมูลแบบดิจิตอลจะมีข้อดีในเรื่องของการตรวจสอบความถูกต้องของข้อมูล

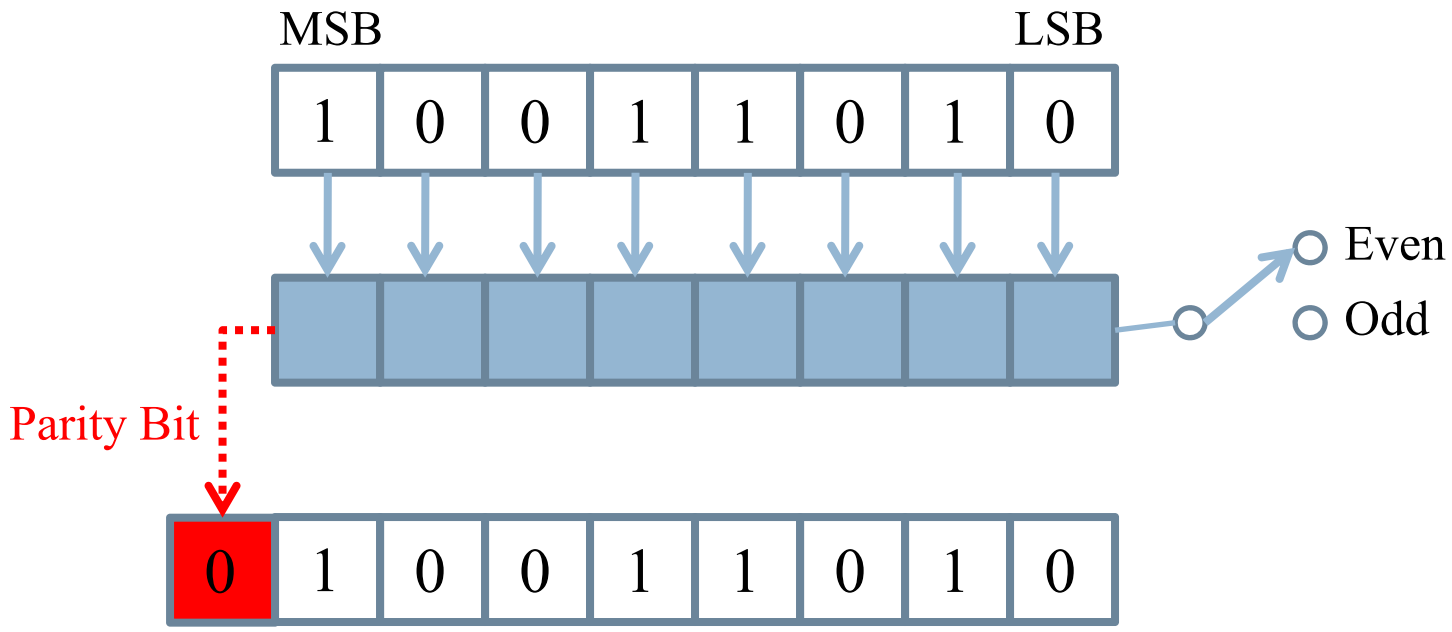
วิธีที่ใช้ในการตรวจสอบความถูกต้อง

- Parity Error Detection
- Data Correction Using LRC/VRC
- Cyclic Redundancy Check (CRC)
- Checksum Error Detection

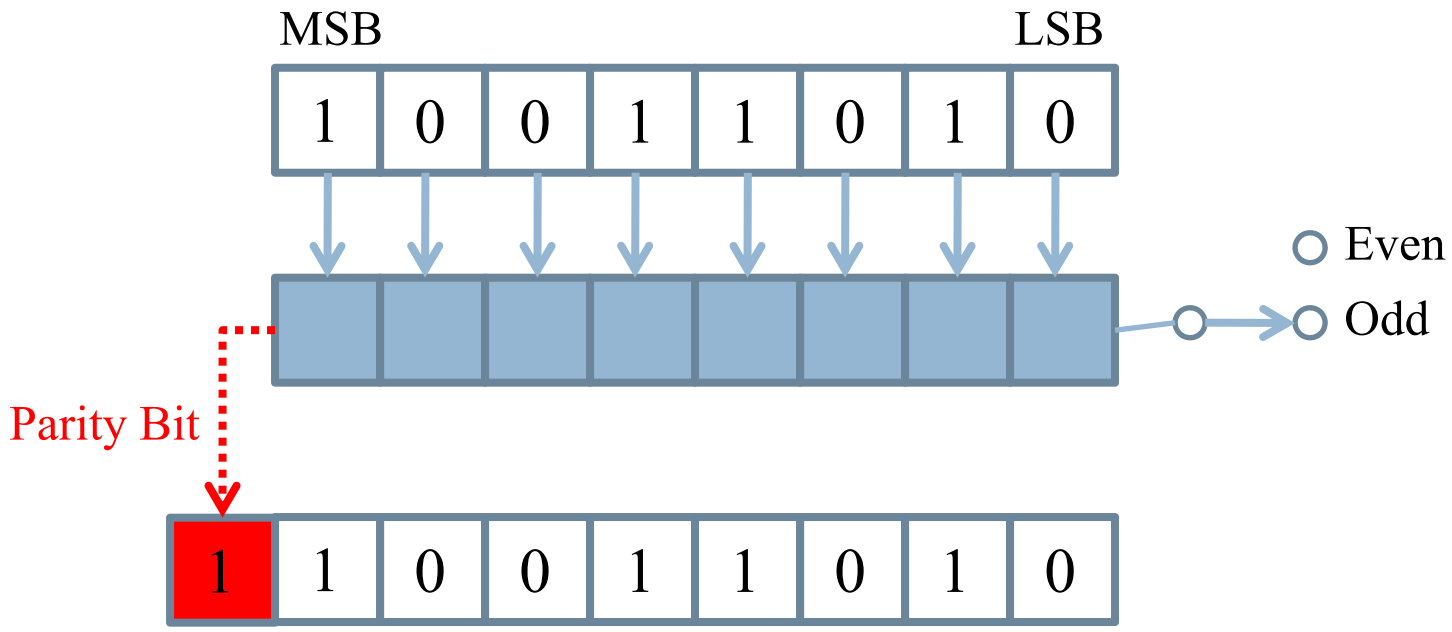
Parity Error Detection

เป็นกระบวนการตรวจสอบความถูกต้องที่ใช้ในวิธีการส่งแบบไม่เข้าจังหวะ(Asynchronous) โดยวิธีการทำงานคือจะมีบิตข้อมูลเพิ่มขึ้นมาอีก 1 บิต ซึ่งจะเรียกว่า พาริตี้บิต(Parity Bit) เป็นตัวตรวจสอบ โดยจำเป็นต้องมีการตกลองกันก่อนจะทำการให้จำนวนของลอจิก 1 เมื่อรวมกันกับพาริตี้บิตแล้วจะให้จำนวนลอจิก 1 เป็นจำนวนคู่(Even) หรือจำนวนคี่(Odd)

รูปแบบการทำงานของ Even Parity Error Detection



รูปแบบการทำงานของ Odd Parity Error Detection



แบบฝึกหัดประกอบความเข้าใจ

ให้นักศึกษาทำการคำนวณหาค่าพาริตี้บิตแบบลอจิก 0 แบบจำนวนคู่ (Even Parity Error Detection) พร้อมแสดงวิธีทำ

1. 0100 0011

2. 1100 1111

3. 0101 0101

4. 0110 1100

5. 0000 0000

คำถามชวนคิด

เมื่อมีการส่งข้อมูลออกไปคือ 0010 1100 แต่เมื่อผู้รับได้รับข้อมูล 1000 1100 และมีกระบวนการตรวจสอบโดยใช้วิธี Even Parity Error Detection นักศึกษาคิดว่าผลลัพธ์ที่ได้จะเป็นอย่างไร ระบบการตรวจสอบจะสามารถทำได้ถูกต้องหรือไม่ เพราะเหตุใดพร้อมให้เหตุผลประกอบ

Data Correction Using LRC/VRC

เป็นเทคนิคที่ใช้รูปแบบผสมผสานแบบ 2 มิติ โดยนำกระบวนการ Even Parity Check Detection ในการหาค่า Vertical Redundancy Check(VRC) เพื่อหาค่าความถูกต้องของแต่กลุ่มข้อมูล จากนั้นนำข้อมูลทั้งหมดมาทำการหาค่าความถูกต้องด้วยเทคนิค XOR เพื่อให้ได้ค่า Longitudinal Redundancy Check(LRC) เมื่อได้ค่าทั้งสองก็นำไป LRC ทำการ XOR กับค่าที่ทำการส่งออกไปถ้าได้คำตอบเป็น 0 ก็แสดงว่าข้อมูลที่ได้รับถูกต้อง

ตารางความจริงของลอจิก Excusive-OR (XOR)

INPUT A	INPUT B	Result
0	0	0
0	1	1
1	0	1
1	1	0

รูปแบบการทำงานของ LRC/VRC

ตัวอย่าง ให้ทำการคำนวณหาค่า LRC/VCR พร้อมทั้งตรวจสอบความถูกต้องของคำตอบ เมื่อทำการส่งข้อมูล Help! ออกไป โดยข้อมูลตัวอักษรจะตรงกับค่าในตารางแอสกี(ASCII Table)

Solution(1)

แปลงข้อมูลที่ได้แล้วทำการหาค่า VRC ด้วยวิธี Even Parity Check

ASCII Code	VRC	MSB						LSB
H	0	1	0	0	1	0	0	0
e	0	1	1	0	0	1	0	1
l	0	1	1	0	1	1	0	0
p	1	1	1	1	0	0	0	0
!	0	0	1	0	0	0	0	1

Solution(2)

ใช้เทคนิค XOR กับกลุ่มข้อมูลทั้งหมดรวมทั้ง VRC บิตด้วย

**เทคนิค ถ้าจำนวนลอจิก 1 ได้เป็นจำนวนคี่ ผลลัพธ์ที่ได้คือ 1 **

ASCII Code	VRC	MSB						LSB
H	0	1	0	0	1	0	0	0
e	0	1	1	0	0	1	0	1
l	0	1	1	0	1	1	0	0
p	1	1	1	1	0	0	0	0
!	0	0	1	0	0	0	0	1
LRC	1	0	0	1	0	0	0	0

วิธีการตรวจคำตอบ

ผลลัพธ์ที่นำข้อมูลที่ทำการส่งทั้งหมด มาทำการ XOR กับค่า LRC ผลลัพธ์ที่ได้จะต้องได้ลอจิก 0 ทั้งหมด

ASCII Code	VRC	MSB						LSB
H	0	1	0	0	1	0	0	0
e	0	1	1	0	0	1	0	1
l	0	1	1	0	1	1	0	0
p	1	1	1	1	0	0	0	0
!	0	0	1	0	0	0	0	1
LRC	1	0	0	1	0	0	0	0
Result	0	0	0	0	0	0	0	0

แบบฝึกหัดประกอบความเข้าใจ

ให้ทำการคำนวณหาค่า LRC/VRC พร้อมทั้งตรวจสอบความถูกต้องของคำตอบ เมื่อทำการส่งข้อมูลดังต่อไปนี้

1. Hello
2. Net
3. Work
4. Map
5. Pop

ค่า ASCII Code : H = 0x48 M = 0x4D N = 0x4E P = 0x50 W = 0x57 a = 0x41

e = 0x65 k = 0x6B l = 0x6C o = 0x6F p = 0x70 t = 0x74

CRC

วิธีการ Cyclic Redundancy Check(CRC) เป็นวิธีการตรวจสอบที่ได้รับความนิยมในการใช้งานสำหรับการรับส่งข้อมูลแบบเข้าจังหวะ(Synchronous) ซึ่งได้รับการพัฒนาโดย IBM ขั้นตอนการทำงานคือ จะทำการสร้างตัวหาร (divisor)ขึ้นมาตามจำนวนบิตที่ต้องการ ตัวอย่างเช่น CRC-16 จะใช้จำนวนบิตที่เป็นตัวหารจำนวน 17 บิต ส่วน CRC-4 จะใช้จำนวนบิตที่เป็นตัวหารจำนวน 5 บิต และใช้กระบวนการ XOR ในการหาผลลัพธ์ของคำตอบ

ขั้นตอนการทำงานของ CRC

- กำหนดค่าตัวหาร(divisor) ตามรูปแบบ CRC ที่ใช้ในการตรวจสอบโดยต้องมีจำนวนบิตเพิ่มขึ้นมาอีก 1 บิต
- ทำการเพิ่มลอจิก 0 ท้ายของลำดับข้อมูลตามจำนวนบิตที่ตกลงตามรูปแบบ CRC
- นำค่าข้อมูลที่เพิ่มลอจิก ทำการ XOR กับค่าตัวหาร(divisor) ผลลัพธ์ที่ได้เราจะเรียกว่า ค่า CRC character
- นำค่า CRC character ที่ได้แทนค่าลอจิก 0 ที่ใส่ในตอนแรก
- เมื่อนำค่าข้อมูลที่เพิ่มค่า CRC character แล้วกับการ XOR กับค่าตัวหาร ถ้าผลลัพธ์ที่ได้เป็นศูนย์แสดงว่าข้อมูลที่ส่งมีความถูกต้อง

รูปแบบการทำงานของ CRC

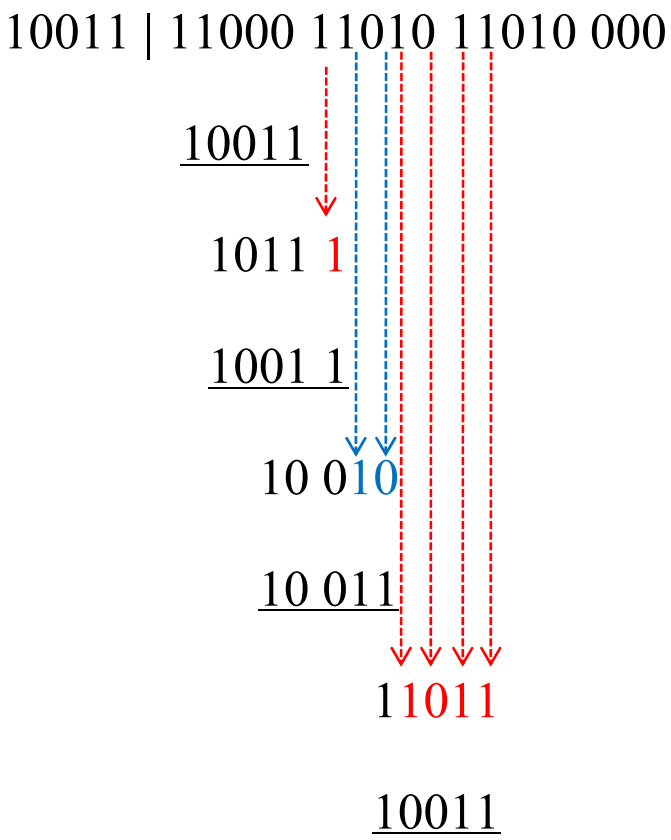
ตัวอย่าง ให้ทำการคำนวณค่า CRC โดยใช้กระบวนการ CRC-4 โดยมีค่าตัวหาร (divisor) เท่ากับ 10011 และมีค่าข้อมูลคือ 1100 0110 1011 01 จงแสดงวิธีการคำนวณ พร้อมทั้งตรวจสอบผลลัพธ์ที่ได้

<u>วิธีการคำนวณ</u>

ทำการเพิ่มลอจิก 0 ตามจำนวนบิตตามรูปแบบการตรวจสอบ(ในที่นี้คือ CRC-4) ค่าตั้งต้นของข้อมูล

1100 0110 1011 01 0000

นำค่าข้อมูลที่เพิ่มลอจิก 0 มาทำการ XOR กับ ค่าตัวหาร



เมื่อตามวิธีการขั้นต้น จนกระทั่งเสร็จสิ้นจะได้ CRC Character = 1001

ดังนั้นข้อมูลที่ทำการส่งจากเดิม

1100 0110 1011 01 0000

ข้อมูลใหม่

1100 0110 1011 01 1001

ค่าตัวหาร(divisor) จะเท่ากับ 10011

แบบฝึกหัดประกอบความเข้าใจ

ให้ทำการคำนวณหาค่า CRC-4 และมีค่าตัวหาร(divisor) 10011 พร้อมทั้งตรวจสอบความถูกต้องของคำตอบ เมื่อทำการส่งข้อมูลดังต่อไปนี้

1. 1100 0100
2. 0011 1100
3. 1100 1100
4. 0110 0011
5. 1010 1010

คำถามชวนคิด

ทำไมค่าตัวหาร(divisor) ต้องใช้จำนวนบิตเพิ่มขึ้น 1 บิต เมื่อกำหนดค่า CRC ที่กำหนด ตัวอย่างเช่น CRC-4 ทำไมถึงใช้ตัวหาร(divisor)จำนวน 5 บิต

Checksum Error Detection

เป็นวิธีการตรวจสอบข้อผิดพลาด ที่นิยมเพื่อตรวจสอบความตรงกันของข้อมูล 2 ชุด ซึ่งมีข้อดีในเรื่องความเร็วในการตรวจสอบแต่วิธีนี้ก็ไม่สามารถรับประกันความถูกต้องตรงกันได้ 100 เปอร์เซ็นต์ โดยมีวิธีการตรวจสอบได้ 2 แบบคือ

1.นำค่าข้อมูลทั้งหมดมาบวกกันแล้วทำการตัดบิตทด(Carry-Flag) ออกแล้วนำผลลัพธ์ที่ได้มาทำการทดสอบ

2.นำค่าข้อมูลทั้งหมดมาทำการ XOR กันเพื่อหาผลลัพธ์

ตารางสรุปการใช้งานรูปแบบการตรวจจับข้อผิดพลาด

Error Method Summary			
Error Method	Data Type	Detection Corrections	Number of Errors Detectable
Parity	Asynchronous	Detection	One per Character
LRC/VRC	Asynchronous	Correction	One per Message
CRC	Either	Detection	Unlimited
Checksum	Synchronous	Correction	Unlimited

ขนาดข้อมูลเพิ่มเติม(Overhead) ของแต่ละวิธี

Error Method	Overhead
Parity	One bit per Character
LRC/VCR	One bit per Character plus LRC Character
CRC	CRC Bytes at end of Message
Checksum	Checksum Character at end of Message