



ITQ1001

ดิจิทัลสำหรับเทคโนโลยีสารสนเทศ

Digital for Information Technology

การแทนข้อมูลในหน่วยความจำ



การแทนข้อมูลในหน่วยความจำคอมพิวเตอร์

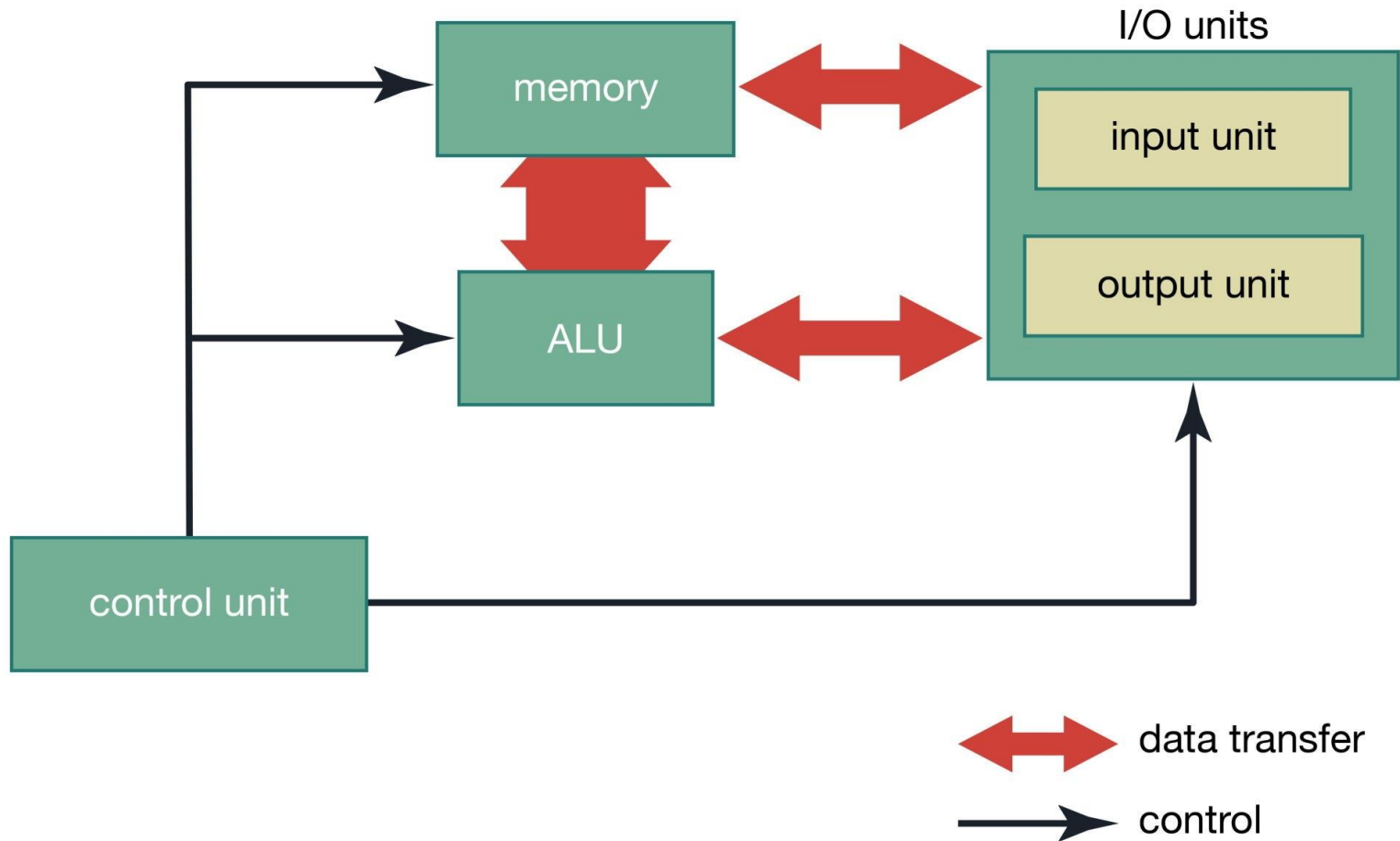
- การแทนข้อมูลในหน่วยความจำคอมพิวเตอร์ (**Data Representation in Memory**)
- คือกระบวนการแปลงข้อมูลรูปแบบต่างๆ ไม่ว่าจะเป็นข้อความ ตัวเลข ภาพ หรือเสียง ให้อยู่ในรูปของ ระบบเลขฐานสอง (**Binary System**)
- ซึ่งประกอบด้วยตัวเลขเพียงสองตัวคือ 0 และ 1 เนื่องจากอุปกรณ์อิเล็กทรอนิกส์ในคอมพิวเตอร์เข้าใจเพียงสถานะทางไฟฟ้าสองสถานะ (มีกระแสไฟฟ้า/ไม่มีกระแสไฟฟ้า)

หน่วยของข้อมูลในหน่วยความจำ

- บิต (**Bit - Binary Digit**): หน่วยที่เล็กที่สุด เก็บค่า 0 หรือ 1
- ไบต์ (**Byte**): กลุ่มของ 8 บิต
- 1 Byte = 8 Bits เป็นหน่วยพื้นฐานในการวัดขนาดข้อมูลและอ้างอิงตำแหน่งในหน่วยความจำ

(Memory Address)

แผนภาพการเชื่อมโยงข้อมูลในระบบคอมพิวเตอร์



ข้อมูลตัวเลข (Numeric Data)

- จำนวนเต็ม (Integers)

- **Unsigned Integer:** เก็บเฉพาะค่าบวก แปลงเป็นเลขฐานสองโดยตรง
- **Signed Integer:** ใช้บิตซ้ายสุด (Sign Bit) บอกเครื่องหมาย 0 = บวก, 1 = ลบ)
- นิยมเก็บในรูปแบบ **Two's Complement** เพื่อให้คำนวณทางคณิตศาสตร์ได้ง่าย

ข้อมูลตัวเลข (Numeric Data)

- จำนวนจริง/ทศนิยม (Floating-Point Numbers):

ใช้มาตรฐาน **IEEE 754** แยกเก็บออกเป็น 3 ส่วน:

บิตเครื่องหมาย (Sign),

ส่วนยกกำลัง (Exponent), และ

ส่วนน้อยสำคัญ (Mantissa/Fraction)

ข้อมูลตัวเลข (Numeric Data)

- ข้อมูลตัวอักษรและความ (Text Data)
- การเก็บตัวอักษรใช้วิธีจับคู่ตัวอักษรเข้ากับรหัสตัวเลข (Character Encoding):
- **ASCII:** ใช้ 7–8 บิต แทนตัวอักษรภาษาอังกฤษและสัญลักษณ์พื้นฐานได้ 128–256 ตัว
- **Unicode (เช่น UTF-8, UTF-16):** ใช้ 1 ถึง 4 ไบต์ รองรับตัวอักษรทุกภาษาทั่วโลก รวมถึงภาษาไทยและ **Emoji**

ข้อมูลตัวเลข (Numeric Data)

- ข้อมูลสื่อผสม (Multimedia Data)
- ภาพ (Images): แบ่งภาพเป็นพิกเซล (Pixel) แต่ละพิกเซลเก็บค่าสีในระบบ RGB (Red, Green, Blue) เช่น ภาพสี 24-bit ใช้ 1 ไบต์ต่อหนึ่งช่องสี
- เสียง (Audio): ใช้วิธี สุ่มสัญญาณ (Sampling) แปลงคลื่นเสียงต่อเนื่องให้อยู่ในรูปตัวเลขดิจิทัล
- วิดีโอ (Video): เก็บเป็นภาพนิ่งต่อเนื่องกันหลายๆ เฟรมร่วมกับไฟล์เสียง

สรุปตารางเปรียบเทียบขนาดการจัดเก็บข้อมูล

ประเภทข้อมูล	ขนาดตัวอย่าง	รูปแบบการแทนข้อมูล
ตัวอักษร (ASCII)	1 Byte (8 bits)	01000001 (แทนอักษร 'A')
จำนวนเต็ม (Int32)	4 Bytes (32 bits)	เลขฐานสองแบบ Two's Complement
ทศนิยม (Float32)	4 Bytes (32 bits)	มาตรฐาน IEEE 754
อักขระ ยูนิโค้ด (UTF-8)	1 - 4 Bytes	รหัสตามมาตรฐาน Unicode

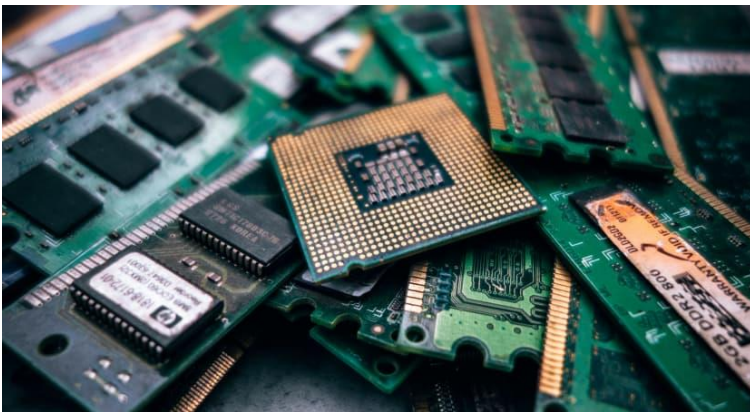
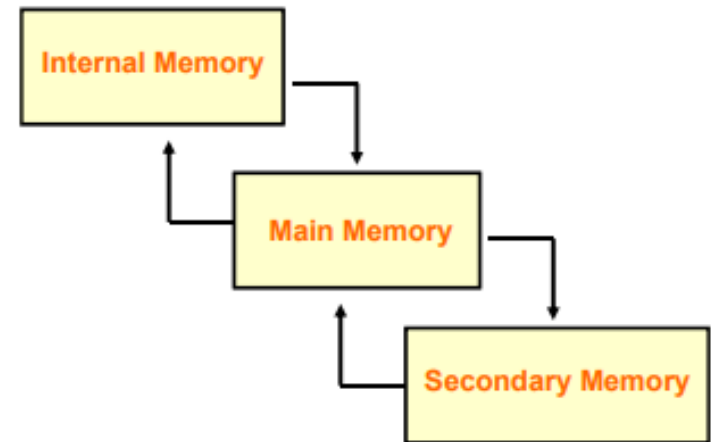
การแทนข้อมูลในหน่วยความจำ

หน่วยความจำ

การเคลื่อนย้ายข้อมูลของหน่วยความจำ
ประเภทต่างๆ

ประเภทของหน่วยความจำ

1. Internal Memory
2. Main Memory
3. Secondary Memory



หน่วยความจำ

การแทนข้อมูลในหน่วยความจำ

ประเภทของหน่วยความจำ

1. Internal Memory

หน่วยความจำภายใน (Internal Memory)

หน่วยความจำภายใน (Internal Memory) เป็นหน่วยความจำถาวรที่มีการเก็บและลบข้อมูลจากผู้ใช้งานโดยตรงซึ่งเมื่อไม่มีไฟฟ้าเลี้ยงข้อมูลที่อยู่ในหน่วยความจำภายใน (Internal Memory) ก็ยังคงอยู่ไม่สูญหายไปไหน ข้อมูลส่วนมากที่จะเก็บไว้ในหน่วยความจำภายใน (Internal Memory) ก็คือ ภาพถ่าย วิดีโอ เพลง เอกสารและข้อมูลแอปพลิเคชัน

กรณีหน่วยความจำภายในของสมาร์ทโฟนอาจเป็นประเภท

eMMC (embedded MultiMediaController) หรือ UFS (Universal Flash Storage)

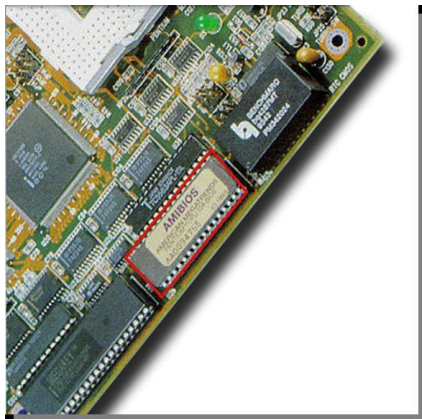
หน่วยความจำ

2. หน่วยความจำหลัก (Main Memory Unit)

เป็นอุปกรณ์ที่ใช้ในการจดจำข้อมูล และโปรแกรมต่าง ๆ ที่อยู่ระหว่างการประมวลผลของคอมพิวเตอร์ บางครั้งอาจเรียกว่า หน่วยเก็บข้อมูลหลัก (**Primary storage**) แบ่งออกได้เป็น 2 ประเภท คือ

2.1 หน่วยความจำหลักแบบอ่านได้อย่างเดียว (Read Only Memory - ROM)

เป็นหน่วยความจำแบบสารกึ่งตัวนำชั่วคราวชนิดอ่านได้อย่างเดียว ใช้เป็นสื่อบันทึกในคอมพิวเตอร์ เพราะไม่สามารถบันทึกซ้ำได้ หน่วยความจำประเภทนี้แม้ไม่มีไฟเลี้ยงต่ออยู่ ข้อมูลก็จะไม่หายไปจากหน่วยความจำ (**nonvolatile**)



หน่วยความจำ

2. หน่วยความจำหลัก (Main Memory Unit)

2.1 หน่วยความจำหลักแบบอ่านได้อย่างเดียว (Read Only Memory - ROM)

โดยทั่วไปจะใช้เก็บข้อมูลที่ไม่ต้องมีการแก้ไขอีกแล้วเช่น เก็บโปรแกรมไบออส

(Basic Input output System : BIOS) หรือเฟิร์มแวร์ที่ควบคุมการทำงานของคอมพิวเตอร์ใช้เก็บโปรแกรมการทำงานสำหรับเครื่องคิดเลขใช้เก็บโปรแกรมของคอมพิวเตอร์ที่ทำงานเฉพาะด้าน

เช่น ในรถยนต์ที่ใช้ระบบคอมพิวเตอร์ควบคุมวงจร ควบคุมในเครื่องซักผ้า เป็นต้น

หน่วยความจำ

การแทนข้อมูลในหน่วยความจำ

2. หน่วยความจำหลัก (Main Memory Unit)

2.2 หน่วยความจำหลักแบบแก้ไขได้ (Random Access Memory - RAM)

เป็นหน่วยความจำหลัก ที่ใช้ในระบบคอมพิวเตอร์ยุคปัจจุบัน หน่วยความจำชนิดนี้ อนุญาตให้เขียนและอ่านข้อมูลได้ในตำแหน่งต่างๆ อย่างอิสระข้อมูลในแรม อาจเป็นโปรแกรมที่กำลังทำงาน หรือข้อมูลที่ใช้ในการประมวลผล ของโปรแกรมที่กำลังทำงานอยู่ ข้อมูลในแรมจะหายไปทันที เมื่อระบบคอมพิวเตอร์ถูกปิดลง เนื่องจากหน่วยความจำชนิดนี้จะเก็บข้อมูลได้เฉพาะเวลาที่มีกระแสไฟฟ้าหล่อเลี้ยงเท่านั้น



หน่วยความจำ

การแทนข้อมูลในหน่วยความจำ

3.Secondary Memory

หรือ เป็นหน่วยเก็บข้อมูลสำรอง (**Secondary Storage Unit**)
อุปกรณ์เก็บข้อมูลสำรอง สามารถจำแนกได้เป็น 2 ประเภทหลัก ๆ ดังนี้

จานแม่เหล็ก (magnetic disk storage)



จานแม่เหล็กเป็นอุปกรณ์เก็บข้อมูลสำรองที่นิยมใช้กันอย่างแพร่หลายกับเครื่องคอมพิวเตอร์ทุกประเภท จานแม่เหล็กประกอบด้วยแผ่นพลาสติกหรือโลหะที่เคลือบด้วยสารแม่เหล็ก ข้อมูลสามารถบันทึกและอ่านจากผิวหน้าที่เคลือบด้วยสารแม่เหล็กนี้ จานแม่เหล็กเป็นอุปกรณ์จัดเก็บข้อมูลที่มีความจุสูง มีความเชื่อถือได้ และยังสามารถเข้าถึงข้อมูลได้อย่างรวดเร็ว ประเภทของจานแม่เหล็ก เช่น ฮาร์ดดิสก์ (hard disk)

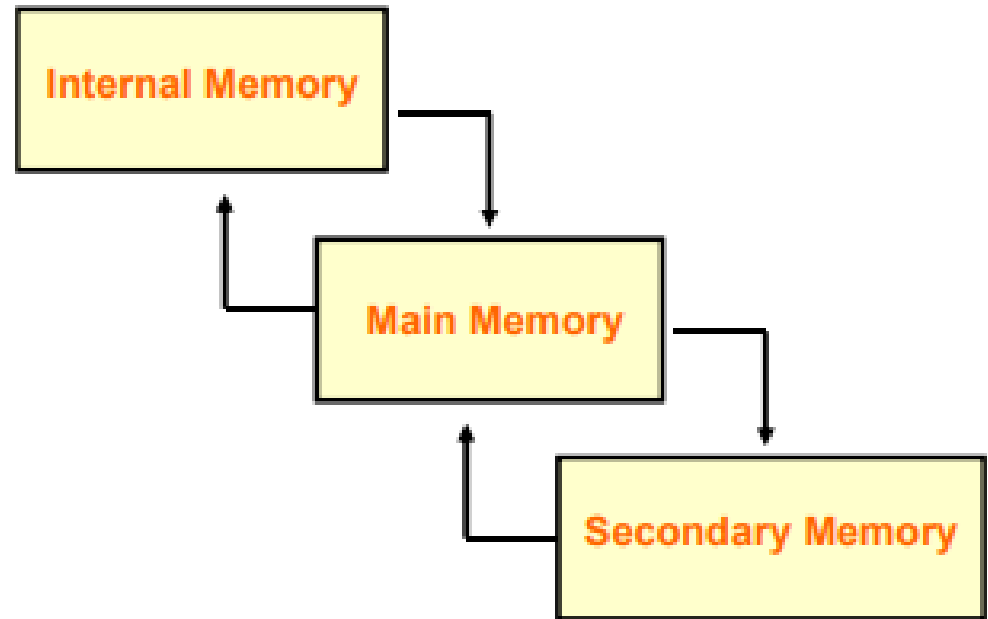
การแทนข้อมูลในหน่วยความจำ

การเคลื่อนย้ายข้อมูลของหน่วยความจำประเภทต่างๆ

หน่วยความจำ

ประเภทของหน่วยความจำ

1. Internal Memory
2. Main Memory
3. Secondary Memory



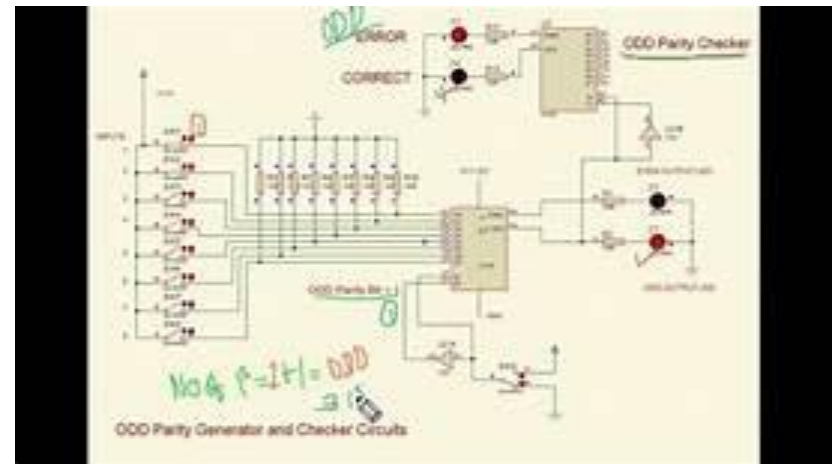
- โปรแกรมจะต้องอยู่ใน **memory** ก่อนที่จะเริ่มทำงานได้
- หลายๆโปรแกรมทำงานพร้อมกัน
- หลายๆโปรแกรมต้องอยู่ใน **memory**
- **memory** มีขนาดจำกัดทำอย่างไรจึงจะทำให้หลายๆโปรแกรมทำงานได้อย่างมีประสิทธิภาพมากที่สุดภายใต้ทรัพยากรที่มีจำกัด

การแทนข้อมูลในหน่วยความจำ/การตรวจสอบข้อมูล

หน่วยความจำของไมโครคอมพิวเตอร์ที่ใช้กันอยู่ในขณะนี้ มีขนาดความกว้าง 8 บิต และเก็บข้อมูลเรียงกันไป โดยมีการกำหนดตำแหน่งซึ่งเรียกว่า เลขที่อยู่ (address) เพื่อให้ข้อมูลที่เก็บมีความถูกต้อง การเขียนหรืออ่านทุกครั้งจึงต้องตรวจสอบความถูกต้องของข้อมูล

วิธีที่ง่ายและนิยมใช้กันคือการเพิ่มอีก 1 บิต เรียกว่า บิตพาริตี (parity bit)

บิตพาริตีที่เพิ่มเติมเข้าไปจะทำให้ข้อมูลทั้งหมดในส่วนนั้นมีเลข 1 เป็นจำนวนคู่ หรือจำนวนคี่



การตรวจสอบข้อมูลด้วย บิตตรวจสอบ Parity Bit

บิตตรวจสอบ (Parity Bit) คือวิธีการตรวจสอบความผิดพลาดของข้อมูล (Error Detection) ที่พื้นฐานที่สุดในการสื่อสารข้อมูลหรือจัดเก็บข้อมูลในคอมพิวเตอร์

- ทำหน้าที่ตรวจหาว่าข้อมูลเกิดการเปลี่ยนแปลงระหว่างส่ง (เช่น มีบิตเปลี่ยนจาก **0** เป็น **1** หรือ **1** เป็น **0** เนื่องจากสัญญาณรบกวน) หรือไม่

รูปแบบของ Parity Bit

Parity Bit ทำงานโดยการเพิ่มบิตพิเศษเข้าไป 1 บิต
ต่อท้ายชุดข้อมูล ซึ่งแบ่งออกเป็น 2 ระบบหลัก:

1. พาริตีคู่ (Even Parity)

กำหนดให้ จำนวนเลข 1 ทั้งหมด (รวมตัวข้อมูล + **Parity Bit**) ต้องเป็นจำนวนคู่

- ถ้าข้อมูลเดิมมีเลข 1 เป็นจำนวน คี่ $\rightarrow$ ใส่ Parity Bit เป็น **1**
(เพื่อให้รวมแล้วเป็นคู่)
- ถ้าข้อมูลเดิมมีเลข 1 เป็นจำนวน คู่ อยู่แล้ว $\rightarrow$ ใส่ Parity Bit เป็น **0**

2. พาริตีคี่ (Odd Parity)

พาริตีคี่ (Odd Parity)

กำหนดให้ จำนวนเลข 1 ทั้งหมด

(รวมตัวข้อมูล + **Parity Bit**) ต้องเป็นจำนวนคี่

- ถ้าข้อมูลเดิมมีเลข 1 เป็นจำนวน คู่ $\rightarrow$ ใส่ Parity Bit เป็น 1 (เพื่อให้รวมแล้วเป็นคี่)
- ถ้าข้อมูลเดิมมีเลข 1 เป็นจำนวน คี่ อยู่แล้ว $\rightarrow$ ใส่ Parity Bit เป็น 0

ตัวอย่างเปรียบเทียบการทำงาน

สมมติว่าชุดข้อมูลที่ต้องการส่งคือ **1011001**
(มีเลข 1 ทั้งหมด 4 ตัว ซึ่งเป็นจำนวนคู่):

ระบบที่ใช้	ข้อมูลเดิม (7 บิต)	จำนวนเลข 1 เดิม	Parity Bit ที่เติม	ข้อมูลที่ส่งออกไป (8 บิต)
Even Parity	1011001	4 (คู่)	0	10110010 (เลข 1 รวม = 4 ตัว)
Odd Parity	1011001	4 (คู่)	1	10110011 (เลข 1 รวม = 5 ตัว)

ขั้นตอนการตรวจสอบที่ฝั่งรับ (Receiver)

- ฝั่งรับจะนับจำนวนเลข 1 ในชุดข้อมูลทั้งหมดที่ได้รับ (รวม Parity Bit)
- ตรวจสอบตามเกณฑ์ที่ตกลงกันไว้:

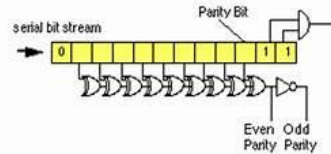
กรณี **Even Parity**: ผลรวมต้องได้จำนวน คู่

กรณี **Odd Parity**: ผลรวมต้องได้จำนวน คี่

หากพบความผิดปกติ: ฝั่งรับจะทราบทันทีที่เกิด Error และแจ้งขอส่งข้อมูลชุดนั้นใหม่อีกครั้ง (Retransmission)

การตรวจสอบข้อมูลด้วย บิตตรวจสอบ Parity Bit

บิตตรวจสอบ (Parity Bit) คือ บิตที่เพิ่มเข้าไปในข้อมูล โดยไม่จำเป็นว่า จะต้องนำไปต่อท้ายหรือขึ้นต้น เพื่อให้แน่ใจว่าบิตที่เป็นค่า 1 ในข้อมูลมีจำนวน เป็นเลขคู่หรือเลขคี่ การใช้พาริตีบิตเป็นวิธีที่ง่ายอย่างหนึ่งในการตรวจจับและ แก้ไขความผิดพลาด



พาริตีบิต มีไว้ตรวจสอบความผิดพลาดจากสื่อสารหรือส่งข้อมูล ซึ่งอาจมีสาเหตุ ต่างๆทำให้คอมพิวเตอร์ส่งข้อมูลผิดจาก 0 เป็น 1 ก็ได้

หากต้องการตรวจสอบว่าคอมพิวเตอร์ส่งข้อมูลผิดพลาดหรือไม่ก็สามารถดูได้จากพาริตีบิตนี้ โดยต้องทราบว่าคอมพิวเตอร์ที่ใช้นั้นเป็นระบบพาริตีเลขคู่หรือพาริตีเลขคี่ และคอมพิวเตอร์ จะนับจำนวนเลข 1 ของแต่ละไบต์ว่ามีจำนวนเท่าใด

การตรวจสอบข้อมูลด้วย บิตตรวจสอบ Parity Bit

บิตตรวจสอบ (Parity Bit)

พาริตีบิตมีสองชนิด คือ พาริตีบิตคู่ กับ พาริตีบิตคี่

พาริตีบิตคู่ มีค่าเป็น 1 เมื่อจำนวนของเลข 1 ในข้อมูลเป็นจำนวนคี่ (ซึ่งทำให้จำนวนเลข 1 ทั้งหมดเป็นจำนวนคู่เมื่อรวมกับบิตนี้)

พาริตีบิตคี่ มีค่าเป็น 1 เมื่อจำนวนของเลข 1 ในข้อมูลเป็นจำนวนคู่ (ซึ่งทำให้จำนวนเลข 1 ทั้งหมดเป็นจำนวนคี่เมื่อรวมกับบิตนี้)

การใช้บิตตรวจสอบ (Parity Checks)

เป็นวิธีการตรวจข้อผิดพลาดอย่างง่ายและเก่าแก่โดยจะใช้บิตพาริตีซึ่งประกอบด้วยเลขไบนารี 0 หรือ 1 ปะท้ายเพิ่มเข้ามาอีกหนึ่งบิตเพื่อใช้เป็นบิตตรวจสอบ

โดยวิธีนี้จะมีอยู่ 2 วิธีด้วยกันคือ การตรวจสอบบิตภาวะคู่ (Even Parity) และการตรวจสอบบิตภาวะคี่ (Odd Parity)

โดยการใช้ **Parity bit** โดยเพิ่มเลขฐานสองขึ้นมามาก 1 หลัก
ให้แก่ตัวเลขเป็นเลขคี่ (**Odd**) หรือ เลขคู่ (**Even**) ดังตารางทางด้านเครื่องรับก็ต้อง
ทำการตรวจสอบ **Parity Bit** ว่าตรงตามผลการบวกตัวเลขหรือไม่

การใช้บิตตรวจสอบ (Parity Checks)

ตัวเลข	Odd Parity	Even Parity
0000	1	0
0001	0	1
0010	0	1
0011	1	0
0100	0	1
0101	1	0
0110	1	0
0111	0	1
1000	0	1

ตารางแสดง 8 **Parity Bit** แบบ **Odd** และ **Even**

การใช้บิตตรวจสอบ (Parity Checks)

ตัวเลข	Odd Parity	Even Parity
1001	1	0
1010	1	0
1011	0	1
1100	1	0
1101	0	1
1110	0	1
1111	1	0

ตารางแสดง 8 Parity Bit แบบ Odd และ Even

เพิ่มเติมเกี่ยวกับ parity

พาริตีบิตเป็นบิตที่ขยายเพิ่มเข้าไปต่อท้ายจากบิต **LSB** ในรหัสของเลขไบนารีโดยเราจะใช้พาริตีบิตในการตรวจสอบรหัสไบนารีว่ามีความถูกต้องหรือไม่ในการส่งข้อมูลจากต้นทางไปยังปลายทาง ซึ่งพาริตีบิตนั้นมี 2 รูปแบบด้วยกันคือ

1. พาริตีคี่ (Odd parity) และ
2. พาริตีคู่ (Even parity)

เพิ่มเติมเกี่ยวกับ parity

พาริตีคี่ (Odd parity)

ในการเพิ่มพาริตีบิตให้เป็นพาริตีคี่ (Odd parity) สามารถทำได้โดยการนับจำนวนเลข “1” ในเลข รหัสไบนารีว่ามีจำนวนเป็นเลขคู่หรือเลขคี่

ถ้าจำนวนเลข “1” เป็นเลขคี่ (1 , 3 , 5 , 7 , 9 , ...) ให้ใส่เลข 0 ใน พาริตีบิต แต่

ถ้านับแล้วมีจำนวนจำนวนเลข “1” เป็นเลขคู่ (2 , 4 , 6 , 8 , 10 , ...)

ให้ใส่เลข 1 ในพาริตีบิตแทน

MSB							LSB	<u>Parity bit</u>
0	1	0	1	<u>1</u>	1	0	1	0
1	0	1	<u>1</u>	0	1	0	<u>0</u>	1

เพิ่มเติมเกี่ยวกับ parity

พาริตีคู่ (Even parity)

ส่วนในการเพิ่มพาริตีบิตให้เป็นพาริตีคู่ (Even parity) สามารถทำได้โดยการนับจำนวนเลข “1” เช่นเดียวกับการหาพาริตีคี่

โดยถ้าจำนวนเลข “1” เป็นเลข คู่ให้ใส่เลข 0 ในพาริตีบิต แต่ถ้านับแล้วมีจำนวนเลข “1” เป็นเลข คี่ให้ใส่เลข 1 ในพาริตีบิตแทน

MSB							LSB	<u>Parity bit</u>
1	0	0	1	0	1	0	1	0
0	0	1	1	0	1	0	0	1

การใช้บิตตรวจสอบ (Parity Checks)

เป็นวิธีการตรวจข้อผิดพลาดอย่างง่ายและเก่าแก่โดยจะใช้บิตพาริตีซึ่งประกอบด้วยเลขไบนารี 0 หรือ 1 ปะท้ายเพิ่มเข้ามาอีกหนึ่งบิตเพื่อใช้เป็นบิตตรวจสอบ โดยวิธีนี้จะมีอยู่ 2 วิธีด้วยกันคือ การตรวจสอบบิตภาวะคู่ (Even Parity) และ การตรวจสอบบิตภาวะคี่ (Odd Parity)

ตัวอย่างเช่น กรณีตรวจสอบบิตภาวะคู่หาบิตข้อมูลมีค่าเท่ากับ **0100110** ดังนั้นบิตพาริตีที่เพิ่มเข้าไปก็คือ บิต 1 เพื่อให้เป็นบิตคู่(ให้บิต 1 รวมกันเป็นเลขคู่) ก็จะได้ **01001101**

0 1 0 0 1 1 0

ข้อมูล

0 1 0 0 1 1 0 1

ภาวะคู่ (Even Parity)

การใช้บิตตรวจสอบ (Parity Checks)

อีกตัวอย่าง

แต่ถ้าข้อมูลมีค่าเท่ากับ **0110110** บิตพาริตีที่เพิ่มเข้าไปก็จะเป็นบิต 0 ก็
จะได้ **01101100**

0 1 1 0 1 1 0

0 1 1 0 1 1 0 0

การใช้บิตตรวจสอบ (Parity Checks)

พิจารณาจากรูป

ฝั่งส่งได้ส่งข้อมูล 0100110 ซึ่งเป็น ข้อมูลต้นทางโดยใช้วิธีการตรวจสอบบิตแบบภาวะคู่ ครั้นข้อมูลส่งไปยังปลายทาง ปลายทางก็จะตรวจสอบบิต ด้วย วิธีเดียวกันให้สังเกตบรรทัดที่สองของตารางจะพบว่าเมื่อข้อมูลส่งไปยังปลายทางปรากฏว่ามีค่าของบิต หนึ่งเปลี่ยน แปลงไป จึงตรวจจับข้อผิดพลาดได้

Original Data	Sender Parity Bit	Transmitted Information	Receiver Calculated Parity Bit	<u>Agree ?</u>
0100110	1	01001101	1	Yes
0100110	1	01001 <u>0</u> 01	0	No

การแทนข้อมูลในหน่วยความจำ

การแทนข้อมูลในหน่วยความจำ

หน่วยความจำหลักของคอมพิวเตอร์เป็นที่เก็บข้อมูลและคำสั่งในขณะประมวลผล การเก็บข้อมูลในหน่วยความจำเป็นการเก็บรหัสตัวเลขฐานสอง ข้อมูลที่ใช้ในการประมวลผลทั้งตัวเลขหรือตัวอักษรจะได้รับการแทนเป็นตัวเลขฐานสอง แล้วเก็บไว้ในหน่วยความจำ เช่น ข้อความว่า **BANGKOK** เก็บในคอมพิวเตอร์จะแทนเป็นรหัสเรียงกันไป ดังนี้

	หน่วยความจำ
B	01000010
A	01000001
N	01001110
G	01000111
K	01001011
O	01001111
K	01001011

การแทนข้อมูลในหน่วยความจำ

หน่วยความจำของไมโครคอมพิวเตอร์ที่ใช้กันอยู่ในขณะนี้ มีขนาดความกว้าง 8 บิต และเก็บข้อมูลเรียงกันไป โดยมีการกำหนดตำแหน่งซึ่งเรียกว่า เลขที่อยู่ (address)

เพื่อให้ข้อมูลที่เก็บมีความถูกต้อง การเขียนหรืออ่านทุกครั้งจึงต้องตรวจสอบความถูกต้องของข้อมูล วิธีที่ง่ายและนิยมใช้กันคือการเพิ่มอีก 1 บิต เรียกว่า บิตพาริตี (parity bit) บิตพาริตีที่เพิ่มเติมเข้าไปจะทำให้ข้อมูลทั้งหมดในส่วนนั้นมีเลข 1 เป็นจำนวนคู่ หรือจำนวนคี่ เช่น ในไมโครคอมพิวเตอร์เพิ่มอีก 1 บิต เพื่อให้เลขหนึ่งเป็นจำนวนคู่ เรียกว่าพาริตีคู่ (even parity) บิตพาริตีที่เติมสำหรับข้อมูลตัวอักษร **A** และ **E** เป็นดังนี้

A 01000001 0 <-- บิตพาริตี
E 01000101 1 <-- บิตพาริตี

ข้อมูล **A** มีเลข 1 สองตัว ซึ่งเป็นจำนวนคู่อยู่แล้ว ดังนั้น จึงใส่บิตพาริตีเป็นเลข 0
ข้อมูล **E** มีเลข 1 เป็นจำนวนคี่ จึงใส่บิตพาริตีเป็น 1 เพื่อให้มีเลข 1 เป็นจำนวนคู่

การแทนข้อมูลในหน่วยความจำ

ข้อความ **BANGKOK** เมื่อเก็บในหน่วยความจำหลักของไมโครคอมพิวเตอร์ที่มีบิตพาริตีด้วยจะเป็นดังรูป

	หน่วยความจำ	บิตพาริตี
B	01000010	0
A	01000001	0
N	01001110	0
G	01000111	0
K	01001011	0
O	01001111	1
K	01001011	0

แสดงตัวอย่างการแทนข้อความในหน่วยความจำแบบมีบิตพาริตี

การแทนข้อมูลในหน่วยความจำ

การแทนคำสั่งในหน่วยความจำ

หน่วยควบคุมของคอมพิวเตอร์ที่อยู่ในซีพียู ทำการอ่านคำสั่งจากหน่วยความจำมาแปลความหมายและกระทำตาม คำสั่งคอมพิวเตอร์พื้นฐานที่สุดเรียกว่า ภาษาเครื่อง (machine language)

ภาษาเครื่องมีลักษณะเป็นรหัสที่ใช้ตัวเลขฐานสอง ตัวเลขฐานสองเหล่านี้แทนชุดรหัสคำสั่ง คอมพิวเตอร์เครื่องหนึ่งมีคำสั่งที่ใช้ได้หลายร้อยคำสั่ง แต่ละคำสั่งจะมีความหมายเฉพาะ

การแทนข้อมูลในหน่วยความจำ

การแทนคำสั่งในหน่วยความจำ

เช่น คำสั่งนำข้อมูลที่มีค่าเป็น 3 จากหน่วยความจำตำแหน่งที่ 8000 มาบวกกับข้อมูลที่มีค่าเป็น 5 ในตำแหน่งที่ 8001 ผลลัพธ์ที่ได้ให้เก็บไว้ในหน่วยความจำตำแหน่งที่ 8002 เมื่อเขียนคำสั่งเป็นภาษาเครื่องจะมีลักษณะเป็นตัวเลขฐานสองเรียงต่อกันเป็นจำนวนมาก ซึ่งเข้าใจได้ยาก จึงมักใช้ตัวอักษรแทนรหัสภาษาเครื่องเหล่านี้ ตัวอย่าง

แอสเซมบลี	ภาษาเครื่อง
LD A,(8000)	00111010,00000000,10000000
LD B,A	01000111
LD A,(8001)	00111010,00000001,10000000
ADD A,B	10000000
LD (8002),A	00110010,00000010,10000000

แสดงตัวอย่างการแทนคำสั่งภาษาเครื่อง

การแทนข้อมูลในหน่วยความจำ

รหัสภาษาเครื่องเมื่อเก็บอยู่ในหน่วยความจำของคอมพิวเตอร์จะมีลักษณะเรียงต่อกันไป
สมมติให้ส่วนของโปรแกรมเก็บในหน่วยความจำตำแหน่งเริ่มจาก 1000 และข้อมูลเก็บไว้
ที่ตำแหน่งเริ่มจาก 8000 ดังรูป

การเก็บข้อมูลและ
คำสั่งลงใน
หน่วยความจำด้วย
รหัสเลขฐานสอง

ภาษาสั่งการพื้นฐานที่ใช้รหัสตัว
เลขฐานสองนี้เรียกว่า ภาษาเครื่อง
คอมพิวเตอร์ที่ใช้ซีพียูต่างตระกูลกันจะ
มีภาษาเครื่องที่ต่างกัน เช่น เครื่องที่ใช้
ซีพียูเพนเทียมกับซีพียูที่ใช้ในเครื่อง
แมกอินทอช มีรหัสคำสั่งต่างกัน

แอสเซมบลี	ตำแหน่ง	หน่วยความจำ	บิตพาริตี
LD A,(8000)	1000	00111010	0
	1001	00000000	0
	1002	10000000	1
LD B,A LD A,(8001)	1003	01000111	0
	1004	00111010	0
	1005	00000001	1
	1006	10000000	1
ADD A,B LD (8002),A	1007	10000000	1
	1008	00110010	1
	1009	00000010	1
	1010	10000000	1

ข้อมูล	8000	00000011	0
	8001	00000101	0
	8002	00001000	1
ผลลัพธ์			

ล่อจิกเกต เครื่องหมายที่ใช้ในสมการล่อจิก
ตารางความจริง นอตเกต แอนด์เกต แนนด์เกต
ออร์เกต
เอ็กซ์คลูซีฟออร์เกต เอ็กซ์คลูซีฟนอร์เกต