

บทที่ 5

องค์ประกอบของคอมพิวเตอร์

วัตถุประสงค์

หลังจากเรียนจบบทที่ 5 แล้ว นักศึกษาต้องสามารถ:

- อธิบายความแตกต่างระหว่าง 3 องค์ประกอบหลักของคอมพิวเตอร์ฮาร์ดแวร์
- อธิบายหน้าที่ของแต่ละองค์ประกอบของคอมพิวเตอร์ฮาร์ดแวร์
- เข้าใจการกำหนดตำแหน่งที่อยู่ในหน่วยความจำและสามารถคำนวณจำนวนของใบ์ตามวัตถุประสงค์ที่ระบุได้
- อธิบายความแตกต่างระหว่างหน่วยความจำประเภทต่างๆ
- เข้าใจและอธิบายการทำงานของอุปกรณ์ Input/Output แต่ละชนิดได้

ต่อหน้าถัดไป

วัตถุประสงค์ (ต่อ)

- เข้าใจระบบการเชื่อมต่อองค์ประกอบของคอมพิวเตอร์อาร์ดแวร์ที่ต่างกันเข้าด้วยกัน
- เข้าใจระบบการทำงานด้วยหน่วยที่อยู่ของอุปกรณ์ input/output
- เข้าใจลำดับขั้นของการ execute โปรแกรม และ machine cycles
- อธิบายความแตกต่างระหว่าง programmed I/O, interrupt-driven I/O และ direct memory access (DMA)
- เข้าใจสถาปัตยกรรมหลัก 2 แบบที่ใช้ในการกำหนดชุดคำสั่งของคอมพิวเตอร์คือ: CISC และ RISC

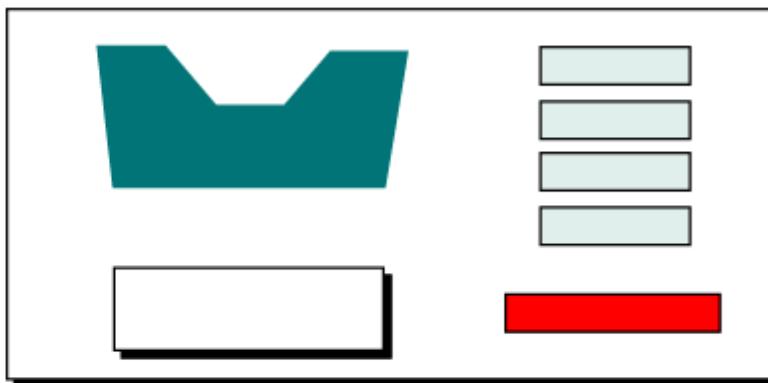
โครงสร้างของคอมพิวเตอร์อาร์ดแวร์

- เราสามารถแบ่งโครงสร้างในส่วนของคอมพิวเตอร์อาร์ดแวร์ได้ออกเป็น 3 ส่วน แต่ละส่วนต่างทำหน้าที่ที่แตกต่างกันไป แต่ก็มีความสัมพันธ์ เกี่ยวข้องกัน ส่วนประกอบทั้งสามได้แก่
 1. ส่วนประมวลผลกลาง (Central Processing Unit: CPU)
 2. หน่วยความจำหลัก (Main Memory Unit)
 3. หน่วยรับข้อมูลและหน่วยแสดงผลข้อมูล (Input/Output Unit)
- ส่วนประกอบแสดงดังรูปคือไป

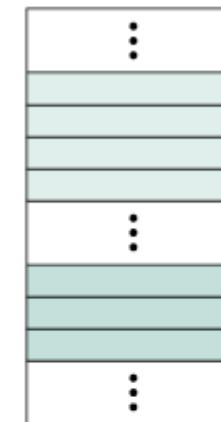
รูปที่ 5-1

คอมพิวเตอร์อาร์ดแวร์ (ระบบย่อ)

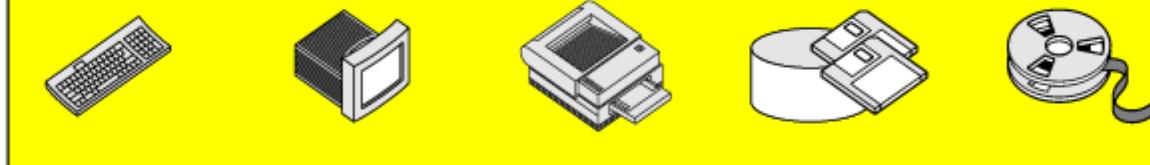
CPU



Memory



Input/Output

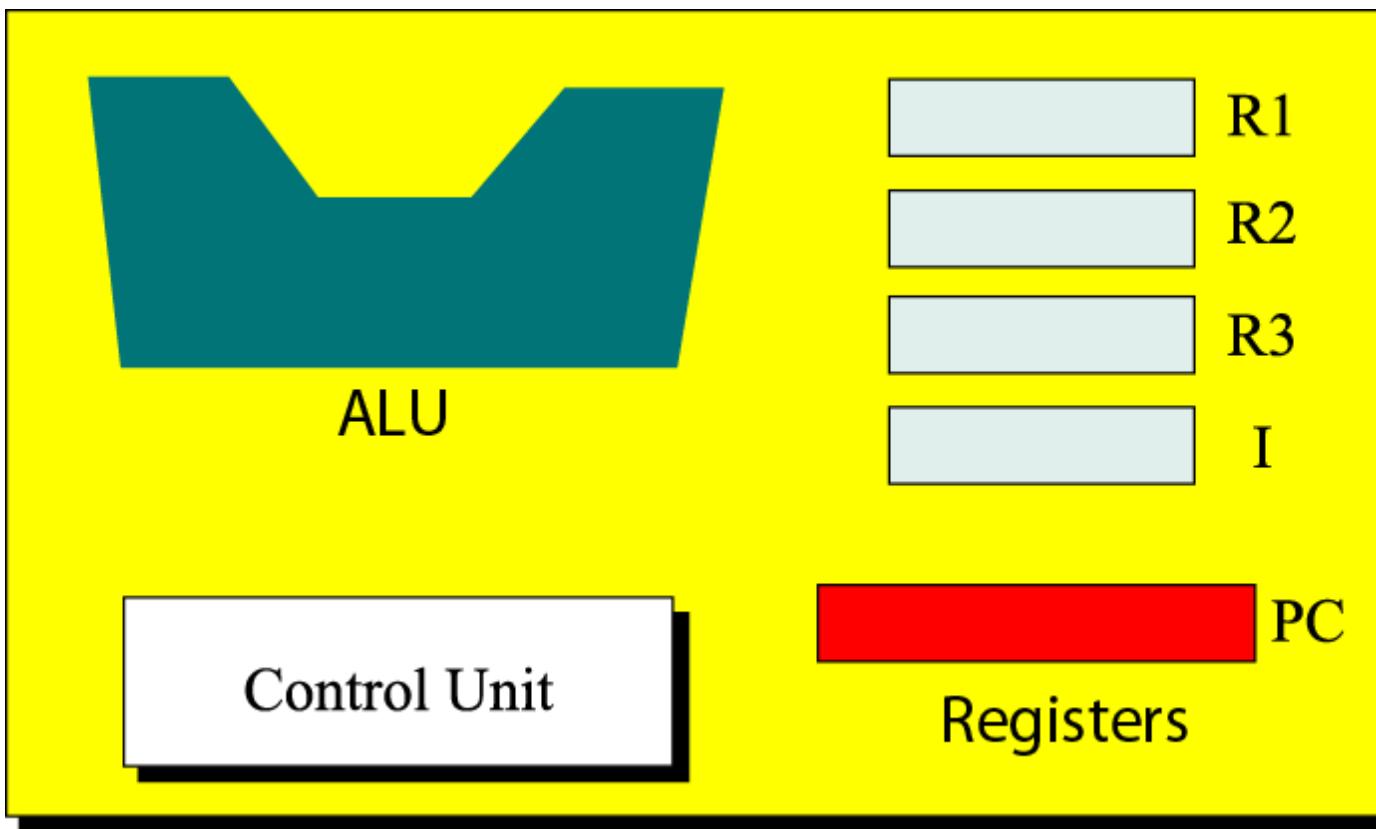


Computer Hardware

5.1

หน่วยประมวลผลกลาง (CPU)

CPU



องค์ประกอบของ CPU

1. **ARITHMETIC LOGIC UNIT: ALU..** ทำหน้าที่โดยตรงเกี่ยวกับ

1.1 **Arithmetic Operations** เช่น increment (บวก 1), decrement

(ลบ 1), บวก, ลบ, คูณ, และ หาร ALU มีหน้าที่เลือก operation
เหล่านี้เพื่อทำการ execute

1.2 **Logical Operations** เช่น NOT, AND, OR, XOR ALU มีหน้าที่
เลือก operation เหล่านี้เพื่อทำการ execute เช่นเดียวกัน

2. **REGISTERS:** เป็นหน่วยความจำชั่วคราวประเภท stand-alone ที่มีความเร็ว
สูง โดยปกติใน CPU จะมี registers ออยู่หลายตัวเพื่อช่วยในการทำงานของ CPU
ที่สำคัญๆ มีดังนี้

องค์ประกอบของ CPU (ต่อ): Registers

2.1 Data Registers: ในอดีต คอมพิวเตอร์มีเพียง 1 register เท่านั้นโดยใช้สำหรับเก็บข้อมูลนำเข้า 1 ตัว (อีก 1 ตัวรับตรงจากหน่วยความจำหลัก) ลับกับการใช้เก็บผลลัพธ์ แต่คอมพิวเตอร์ในปัจจุบัน ใน CPU มี register จำนวนมากตามที่ผู้ออกแบบต้องการเพื่อช่วยให้ในการคำนวณให้เร็วขึ้น การคำนวณที่ลับซับซ้อนจะใช้ชาร์ดแวร์มากกว่าซอฟท์แวร์ เช่นในรูปที่ 5.2 R1 และ R2 เป็น register ที่ใช้เก็บ input data ส่วน R3 เป็น register ที่ใช้เก็บ output data

2.2 Instruction Register: คอมพิวเตอร์ตาม Von Neumann Model คำสั่ง/โปรแกรมต้องอยู่ในหน่วยความจำ CPU มีหน้าที่ในการดึงคำสั่ง

องค์ประกอบของ CPU: Registers

(fetch) ที่จะคำสั่งจากหน่วยความจำหลักไปเก็บใน instruction register และทำการตีความหมาย (interpret) และ execute

2.3 **Program Counter:** PC.. เป็น register ที่เก็บค่า address ของคำสั่งที่กำลังถูก execute เมื่อคำสั่งถูก execute เสร็จ ค่าที่เก็บใน PC จะเพิ่มขึ้นโดยจะเก็บค่า address ของคำสั่งที่จะ execute ต่อไป

3. **CONTROL UNIT:** หน่วยควบคุมเปรียบได้กับส่วนหนึ่งของสมองมนุษย์ที่ทำการควบคุมการเคลื่อนไหวและการกระทำของส่วนต่างๆ ของร่างกายเรา การควบคุมกระทำโดยใช้สายไฟ (wires) จากหน่วยควบคุม

องค์ประกอบของ CPU: Control Units

ไปยังอุปกรณ์ที่เกี่ยวข้องแล้วส่งสัญญาณที่อาจเป็น on (ร้อน) หรือ off (เย็น) เช่น สมมติว่า ALU ต้องการทำ 10 operations ในการระบุ operations เหล่านี้ เราจะต้องใช้สายไฟ 4 เส้นจากหน่วยควบคุมไปยัง ALU สายไฟทั้งสี่เส้นสามารถระบุหรือกำหนด operation ได้ 16 รูปแบบ แต่เราต้องการเพียง 10 operations เท่านั้น ส่วนที่เหลืออีก 6 รูปแบบอาจนำไปใช้เพื่อวัตถุประสงค์อื่น เราสามารถกำหนดสถานะของสายไฟให้เป็น 0 หมายถึง off และ 1 หมายถึง on ดังนั้นสถานะของสายไฟทั้งสี่เส้นจึงกำหนดได้เป็น 0000, 0001, 0010,...,1111 เราอาจให้ 0000 แทน สภาวะไม่มีการกระทำ 0001 แทน increment 0010 แทน decrement เป็นต้น

5.2

หน่วยความจำหลัก

Main Memory: หน่วยความจำหลัก

- หน่วยความจำหลัก ประกอบด้วยกลุ่มของเซลล์หรือพื้นที่ที่เก็บข้อมูลที่สามารถอ้างถึงได้โดยใช้**เลขที่อยู่** (address) หรือ**เลขตำแหน่ง** (location) โดยที่แต่ละตำแหน่งจะมี**ตัวบ่งบอกที่อยู่** (identifier) ที่ไม่ซ้ำกัน
- ข้อมูลถูกส่งผ่านเข้าและออก (transfer) จากหน่วยความจำเป็นกลุ่มของบิตที่เรียกว่า “words” แต่ละ word อาจจะประกอบไปด้วยกลุ่มของบิตจำนวน 8 บิต 16 บิต 32 บิต หรือ 64 บิต
- ถ้า word ประกอบด้วย 8 บิต จะเรียกว่า byte ถ้า word ประกอบด้วย 16 บิตจะเรียกว่า 2-byte word และถ้า word ประกอบด้วย 32 บิตก็จะเรียกว่า 4-byte word เป็นต้น

เลขตำแหน่งที่อยู่ทั้งหมดในเครื่องคอมพิวเตอร์ (Address Space)

- การเข้าถึง (access) ข้อมูลหรือ word ในหน่วยความจำต้องใช้ **ตัวบ่งบอกที่อยู่** หรือ **identifier** ตามที่ผู้เขียนโปรแกรมตั้งขึ้น แต่ในระดับอาร์ดแวร์แล้วแต่ละ word จะถูกอ้างถึงโดยใช้เลขตำแหน่งที่อยู่
- จำนวน **เลขตำแหน่งที่อยู่ที่ไม่ซ้ำกันทั้งหมด** (total number of uniquely identifiable locations) ภายในหน่วยความจำเรียกว่า **address space** ตัวอย่างเช่น หน่วยความจำขนาด 64 กิโลไบท์และมีขนาดของ word เป็น 1 ไบท์จะมี address space จาก 0 ถึง 65,535 เป็นต้น
- ตารางที่ 5.1 แสดงหน่วยที่ใช้อ้างอิงในหน่วยความจำ ขอให้สังเกตว่าคำศัพท์ที่ใช้อาจทำให้เข้าใจผิดได้ นั่นคือค่าโดยประมาณจำนวนไบท์ใช้ 10 ยกกำลัง ส่วนค่าที่แท้จริงของจำนวนไบท์ใช้ 2 ยกกำลัง

ตารางที่ 5.1 หน่วยอ้างอิงของหน่วยความจำ

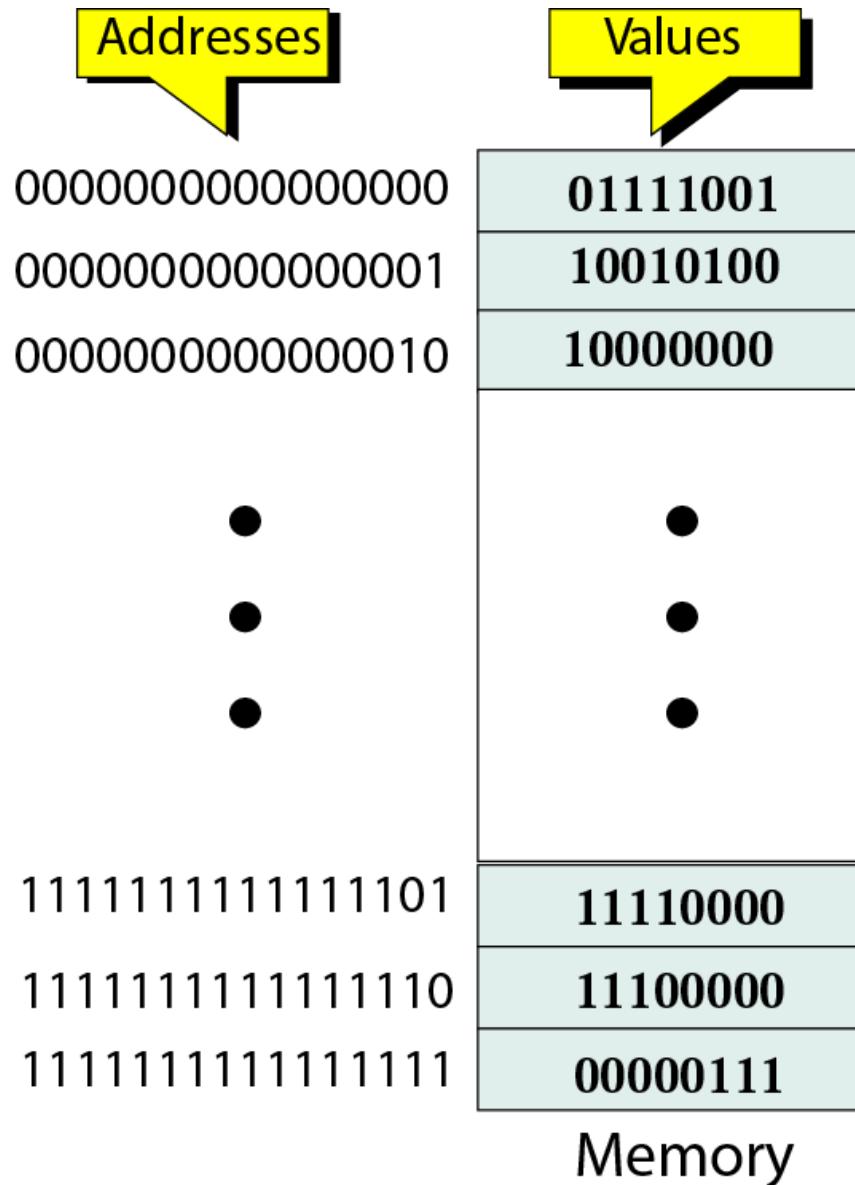
หน่วยอ้างอิง	จำนวนไบท์แท้จริง	จำนวนไบท์โดยประมาณ
kilobyte	2^{10} bytes	10^3 bytes
megabyte	2^{20} bytes	10^6 bytes
gigabyte	2^{30} bytes	10^9 bytes
terabyte	2^{40} bytes	10^{12} bytes
petabyte	2^{50} bytes	10^{15} bytes
exabyte	2^{60} bytes	10^{18} bytes

เลขตำแหน่งที่อยู่ในลักษณะที่เป็น Bit Pattern

- เนื่องจากคอมพิวเตอร์ทำงานด้วยการแทนและเก็บตัวเลขเป็น bit pattern และตัวตำแหน่งที่อยู่เองก็เป็นตัวเลข จึงแทนและเก็บด้วย bit pattern เช่นเดียวกัน
- ถ้าหน่วยความจำมีขนาด 64 กิโลไบท์ (2^{16}) ใช้ word ขนาด 1 ไบท์ แล้ว การกำหนดแทนเลขตำแหน่งที่อยู่จะต้องใช้ bit pattern ขนาด 16 บิตจึงจะพอ โดยเลขตัวแรกคือ 0000000000000000 (address 0) และตัวเลขสุดท้ายคือ 1111111111111111 (address 65,535) ดูรูปที่ 5.3
- กรณีทั่วไป ถ้าหน่วยความจำมีขนาด N words ต้องใช้ bit pattern ขนาด $\log_2 N$ บิต แทนหรือเก็บเลขตำแหน่งที่อยู่

รูปที่ 5-3

หน่วยความจำหลัก





หมายเหตุ:

เลขที่ตำแหน่งของหน่วยความจำหลักกำหนดโดยใช้
เลขจำนวนเต็มแบบ unsigned binary

ตัวอย่างที่ 1

คอมพิวเตอร์เครื่องหนึ่งมีหน่วยความจำหลักขนาด 32 MB (megabytes) จงหาจำนวนบิตที่จะใช้กำหนดเลขตัวแหน่งที่อยู่สำหรับ 1 ไบท์ในหน่วยความจำ

วิธีทำ

เนื่องจากขนาดของหน่วยความจำทั้งหมดคือ 32 MB หรือ 2^{25} หมายความว่าเราต้องใช้ $\log_2 2^{25}$ หรือ 25 บิต เพื่อกำหนดเลขที่ตัวแหน่งที่อยู่ 1 ไบท์ในหน่วยความจำ

ตัวอย่างที่ 2

คอมพิวเตอร์เครื่องหนึ่งมีหน่วยความจำขนาด 128 MB แต่ละ word มีขนาด 8 ไบท์ จงหาว่าจะต้องใช้กี่บิตในการกำหนดตำแหน่งที่อยู่ของ 1 word ในหน่วยความจำ

วิธีทำ

เนื่องจากขนาดของหน่วยความจำหลักเท่ากับ 128 MB หรือเท่ากับ 2^{27} อย่างไรก็ตามแต่ละ word มีขนาด 8 ไบท์ หรือ 2^3 ไบท์ ซึ่งหมายความว่า หน่วยความจำมีทั้งหมด 2^{24} words นั่นคือเราต้องการ $\log_2 2^{24}$ หรือ 24 บิต เพื่อที่จะกำหนดตำแหน่งที่อยู่ 1 word ในหน่วยความจำ

ประเภทของหน่วยความจำ

- หน่วยความจำแบ่งออกเป็น 2 ประเภทคือ: RAM และ ROM
- **RAM = Random Access Memory** เป็นหน่วยความจำส่วนใหญ่ของหน่วยความจำหลักของคอมพิวเตอร์ ผู้ใช้สามารถอ่านและเขียนข้อมูลกับ RAM ได้
- RAM มีคุณสมบัติที่เรียกว่า **Volatile** คือข้อมูลที่เก็บไว้จะหายไปหรือถูกลบทั้งหมดเมื่อปิดเครื่องหรือไฟดับ
- เทคโนโลยีที่ใช้สร้าง RAM มี 2 ประเภทคือ
 1. **Static RAM (SRAM) technology:** ใช้ flip-flop gates (gate ที่มี 2

ประเภทของหน่วยความจำ : RAM

สถานะคือ 0 กับ 1) สำหรับเก็บข้อมูล เมื่อคอมพิวเตอร์เปิดอยู่ gate แต่ละ gate จะมีสถานะเป็น 0 หรือ 1 เพื่อเก็บข้อมูลจนกว่าจะปิดเครื่อง
SRAM มีความเร็วสูง แต่ราคาแพง

2. **Dynamic RAM (DRAM) technology:** ใช้ capacitors เมื่อ capacitor ถูกชาร์จ สถานะจะเป็น 1 ถ้าไม่ชาร์จ สถานะจะเป็น 0 เนื่องจาก capacitor มีการสูญเสียพลจากการชาร์จ เมื่อเวลาผ่านไป เซลล์ของหน่วยความจำจะต้องมีการ **refresh** เป็นระยะๆ DRAM มีราคาถูก แต่การทำงานช้า

ประเภทของหน่วยความจำ (ต่อ)

- **ROM = Read-Only-Memory:** เป็นหน่วยความจำที่อ่านได้อย่างเดียว ข้อมูลเกิดจากผู้ผลิตบันทึกลงไป ข้อดีคือเป็นหน่วยความจำประเภทที่เรียกว่า **nonvolatile** คือข้อมูลที่เก็บไว้จะไม่หายไปเมื่อปิดเครื่องหรือไฟดับ
- คอมพิวเตอร์บางยี่ห้อ ผู้ผลิตจะเก็บ **โปรแกรมสำหรับเปิดเครื่อง** (booting program) ซึ่งจะทำงานทุกครั้งที่เปิดเครื่อง ROM มีหลายแบบ เช่น
 1. **PROM = programmable read-only-memory:** เมื่อซื้อคอมพิวเตอร์มา หน่วยความจำนี้จะว่าง ผู้ใช้สามารถใช้เครื่องมือพิเศษเขียนโปรแกรมลงในหน่วยความจำนี้ได้ เมื่อเขียนลงแล้ว PROM จะมีคุณสมบัติเหมือน ROM จึงสามารถใช้เก็บโปรแกรมที่ไม่ต้องการลบได้

ประเภทของหน่วยความจำ (ต่อ)

2. **EPROM = Erasable Programmable Read-Only-Memory:** เป็นหน่วยความจำที่ผู้ใช้สามารถเขียนโปรแกรมบันทึกข้อมูลหรือเก็บโปรแกรมได้และสามารถลบออกได้โดยใช้แสงอุลตร้าไวโอลเล็ต การลบข้อมูลจะต้องถอด EPROM ออกจากเครื่องคอมพิวเตอร์ เมื่อลบเสร็จก็นำกลับเข้าไปติดตั้งใหม่
3. **EEPROM = Electronically Erasable Programmable Read-Only Memory:** สามารถเขียนโปรแกรมบันทึกและลบข้อมูลได้โดยไม่ต้องถอดออกจากตัวเครื่องคอมพิวเตอร์

ลำดับขั้นของหน่วยความจำ

- ประสิทธิภาพของหน่วยความจำคอมพิวเตอร์มีหลายระดับ หน่วยความจำที่มีความเร็วสูงราคาก็จะแพง ผู้เขียนโปรแกรมควรใช้ให้เหมาะสมคือ
 - ถ้าจำเป็นต้องใช้หน่วยความจำไม่มาก แต่ความเร็วเป็นเรื่องจำเป็น ควรจะใช้หน่วยความจำประเภท **registers** ที่อยู่ใน CPU
 - ถ้าต้องใช้หน่วยความจำพอประมาณและความเร็วปานกลาง เพื่อเก็บข้อมูลที่มีการเข้าถึงบ่อย ควรจะใช้หน่วยความจำ **Cache**
 - ถ้าใช้หน่วยความจำมาก ความเร็วต่ำ เพื่อเก็บข้อมูลที่เข้าถึงไม่บ่อย ควรจะใช้หน่วยความจำหลัก

ลำดับขั้นของหน่วยความจำ

Fastest Speed
(Registers)

Faster Speed
(Cache Memory)

Fast Speed
(Main Memory)

หน่วยความจำ Cache

- หน่วยความจำ cache มีความเร็วช้ากว่า register แต่เร็วกว่าหน่วยความจำหลัก หน่วยความจำ cache ปกติจะมีขนาดเล็ก ติดตั้งอยู่ระหว่าง CPU กับหน่วยความจำหลัก (ดังรูปที่ 5.5)
- ณ เวลาใดเวลาหนึ่ง หน่วยความจำ cache จะเก็บข้อมูลที่เป็นส่วนหนึ่งของหน่วยความจำหลัก เมื่อ CPU ต้องการเข้าถึงข้อมูลในหน่วยความจำหลัก จะมีขั้นตอนดังนี้
 1. CPU ตรวจสอบข้อมูลใน cache
 2. ถ้าพบข้อมูลใน cache ก็จะคัดลอกแล้วนำไปใช้คำนวณใน CPU แต่ถ้าไม่พบ ก็จะเข้าไปอ่านข้อมูลจากหน่วยความจำหลัก 1 บล็อก แล้วนำมา

หน่วยความจำ Cache (ต่อ)

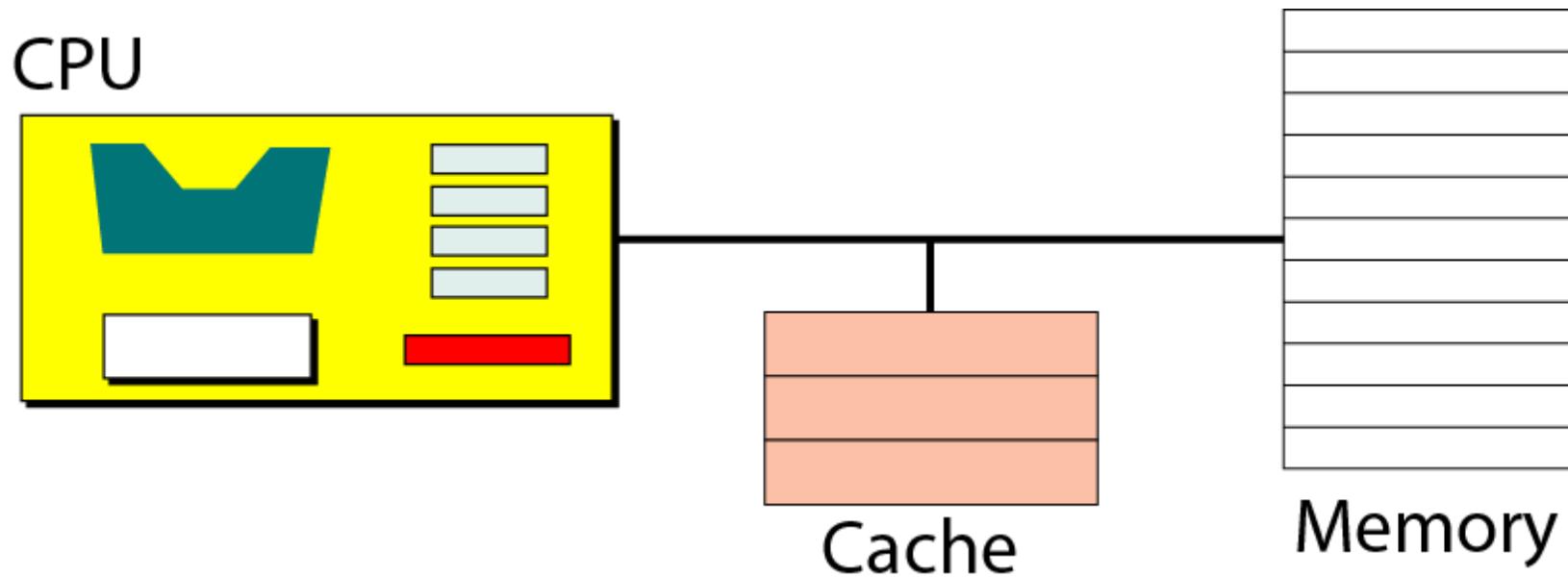
เก็บแทนข้อมูลเดิมในหน่วยความจำ cache

3. CPU เข้าถึงข้อมูลที่ต้องการ จาก cache แล้วคัดลอก word เข้าสู่ CPU เพื่อใช้ต่อไป

- กระบวนการนี้จะช่วยให้คอมพิวเตอร์ทำงานเร็วขึ้นโดยรวม เพราะ cache จะช่วยย่นเวลาการเข้าถึงข้อมูลในหน่วยความจำหลักของ CPU
- ถึงแม้ว่าหน่วยความจำ cache จะมีขนาดเล็ก แต่ก็มีประสิทธิภาพสูง ทั้งนี้因为 cache ใช้หลักการ **80-20 rule**

รูปที่ 5-5

หน่วยความจำ Cache



5.3

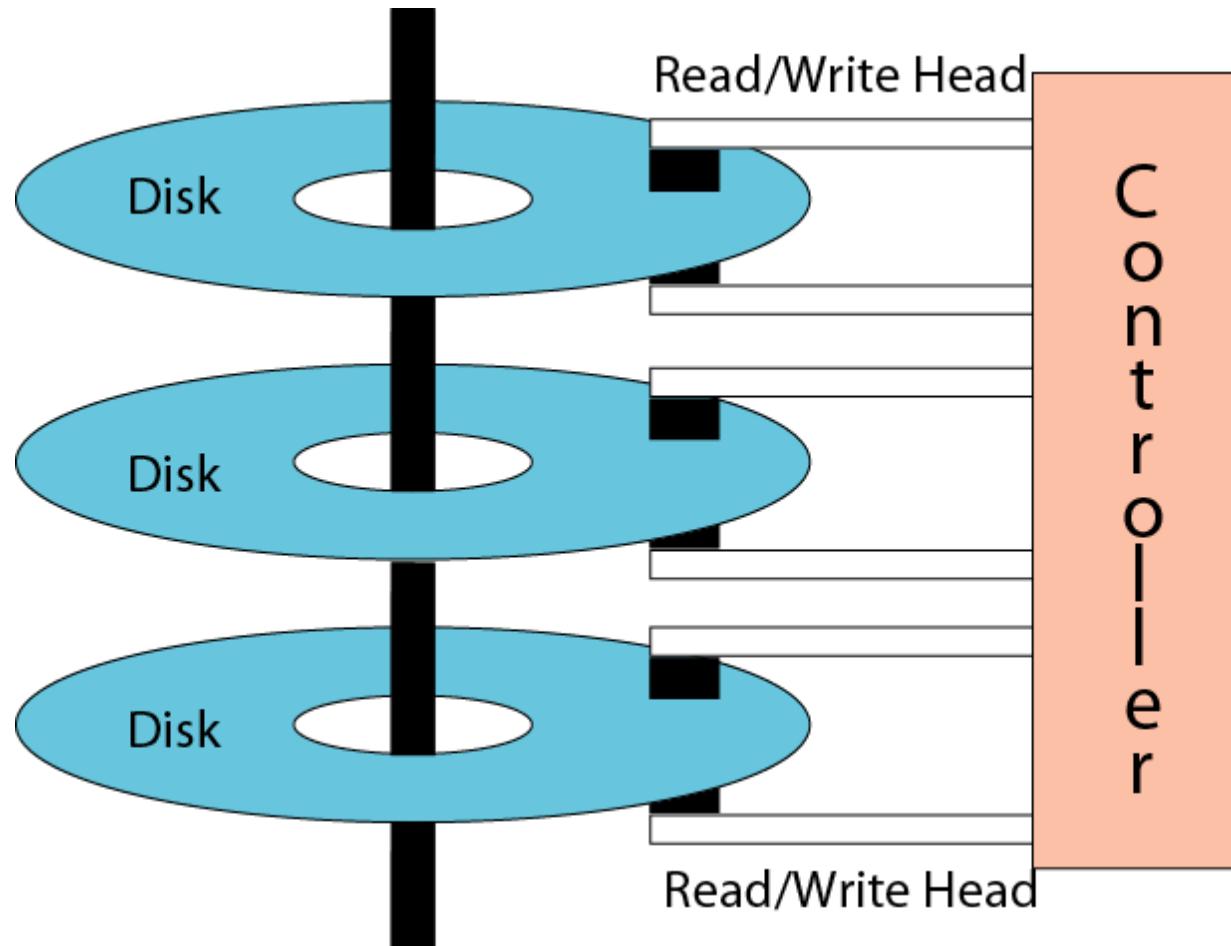
หน่วยรับข้อมูลและหน่วยแสดงผลลัพธ์ **INPUT / OUTPUT**

I/O Units

- I/O units มีไว้เพื่อให้คอมพิวเตอร์สามารถสื่อสารกับโลกภายนอกได้
- I/O devices สามารถแบ่งออกได้เป็น 2 กลุ่มใหญ่ๆ คือ อุปกรณ์ที่เก็บข้อมูลได้ กับ อุปกรณ์ที่เก็บข้อมูลไม่ได้
 1. **Nonstorage devices:** เป็นอุปกรณ์ที่ทำให้ CPU/memory สื่อสารกับโลกภายนอกได้ แต่อุปกรณ์เหล่านี้ไม่สามารถเก็บข้อมูลได้ อุปกรณ์เหล่านี้ได้แก่ แป้นพิมพ์, จอภาพ, และ เครื่องพิมพ์ เป็นต้น
 2. **Storage devices:** เป็นอุปกรณ์ I/O ที่สามารถเก็บข้อมูลได้จำนวนมาก เมื่อปิดเครื่องข้อมูลไม่หาย บางครั้งก็เรียกว่า หน่วยความจำสำรอง (**auxiliary** หรือ **secondary storage devices**) แบ่งเป็น 2 ประเภทคือ ใช้คลื่นแม่เหล็ก (**magnetic**) และ ใช้แสง (**optical**)

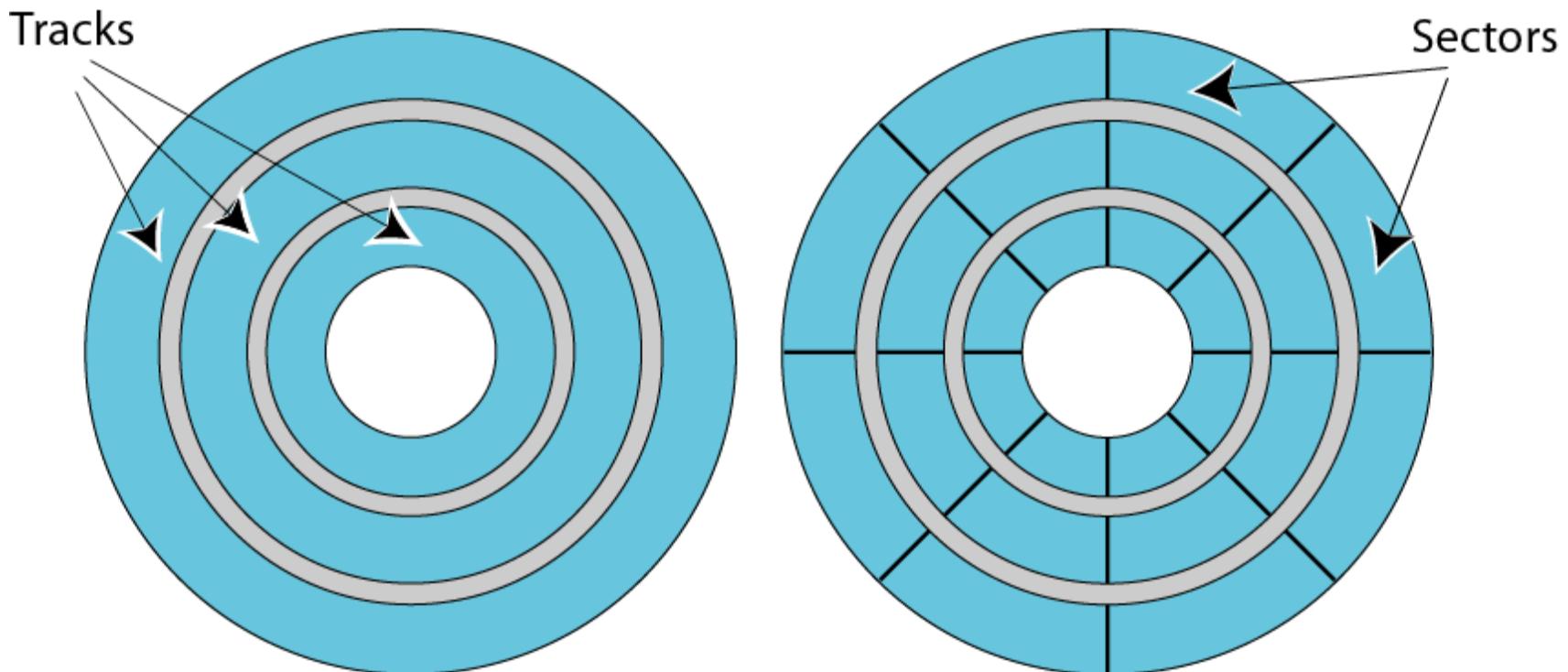
รูปที่ 5-6

ภาพแสดงการอ่านข้อมูลแบบ sequential



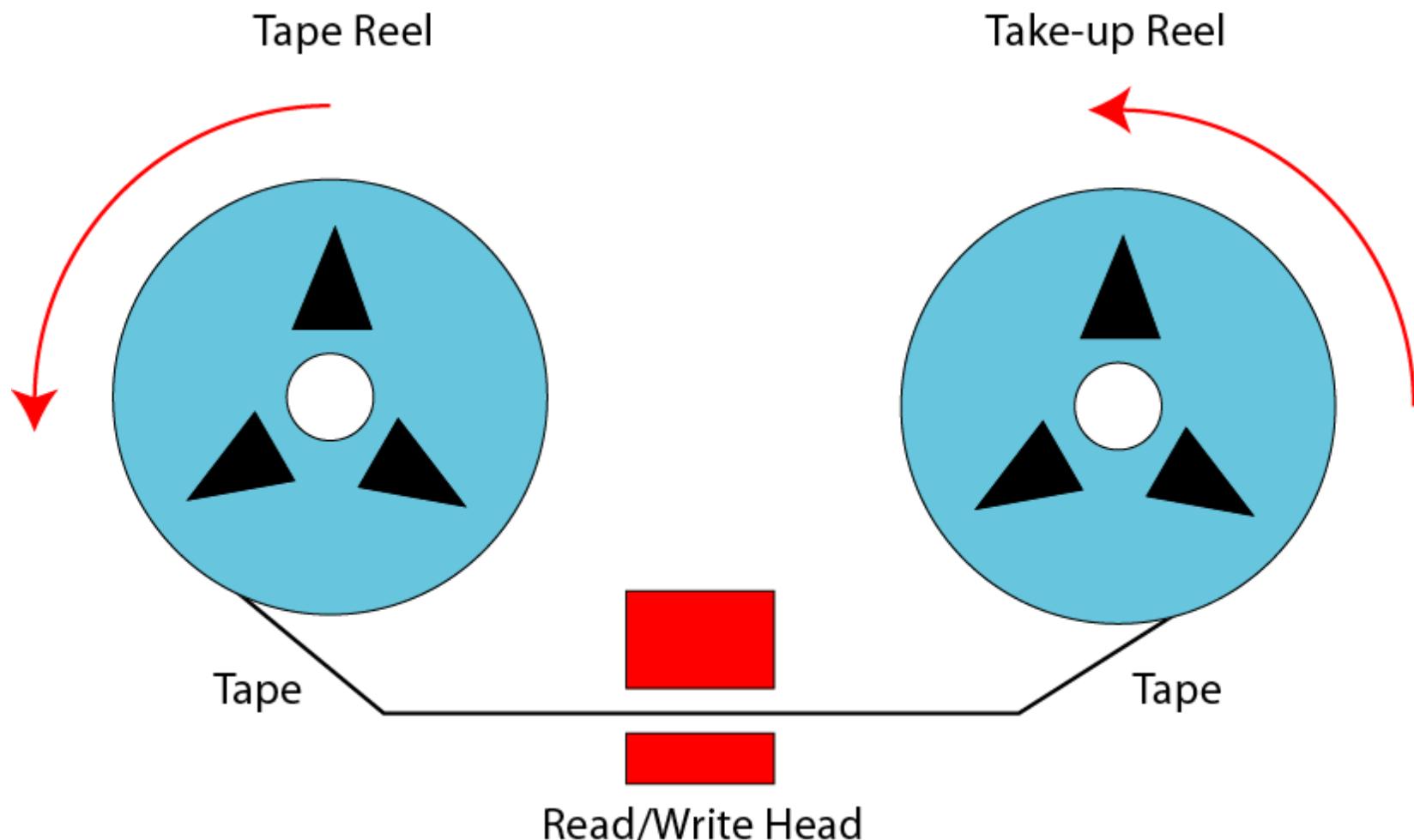
รูปที่ 5-7

โครงสร้างพื้นผิวของจานแม่เหล็ก



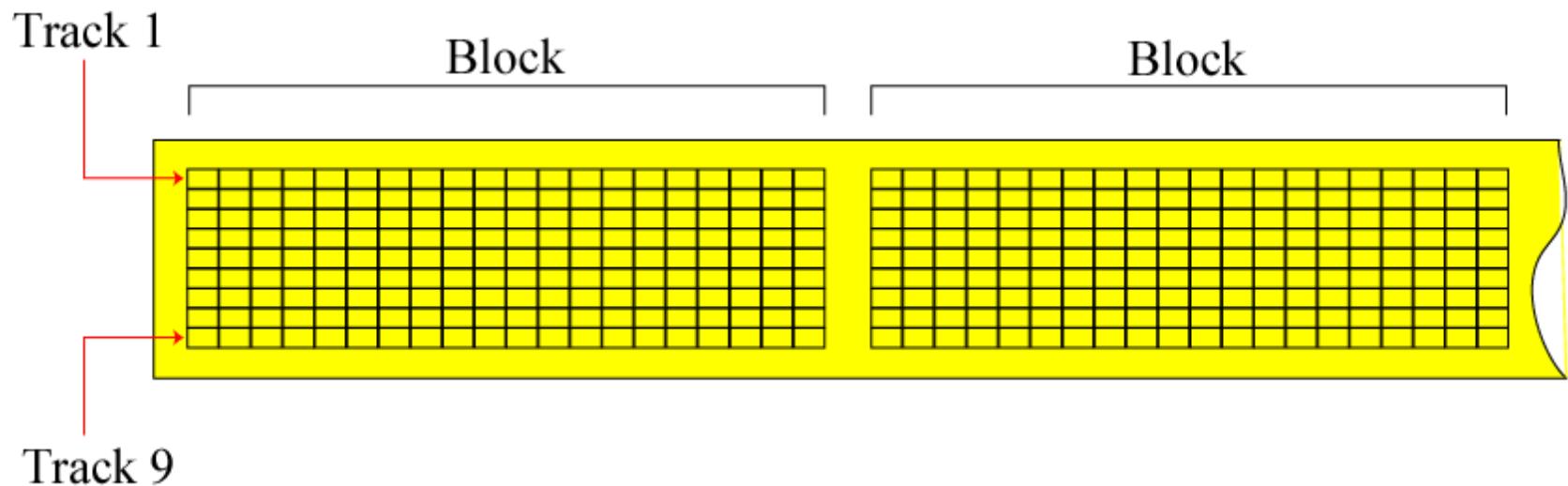
รูปที่ 5-8

ภาพแสดงกลไกของเทปแม่เหล็ก



รูปที่ 5-9

ภาพแสดงโครงสร้างพื้นผิวของเทปแม่เหล็ก



อุปกรณ์เก็บข้อมูลที่ใช้แสง : CDs

- ใช้เทคโนโลยีแสงเลเซอร์สำหรับบันทึกและอ่านข้อมูลซึ่งเป็นผลมาจากการคิดค้นแผ่น CD (Compact Disc) ที่ใช้บันทึกเสียงเพลง
- อุปกรณ์ที่ใช้เทคโนโลยีแสงเลเซอร์สำหรับบันทึกข้อมูลที่ประยุกต์ทางคอมพิวเตอร์ที่ใช้กันอยู่ปัจจุบัน เช่น CD-ROM, CD-R, CD-RW, และ DVD เป็นต้น
- CD-ROM = Compact Disc Read-Only-Memory ใช้เทคโนโลยีเหมือน CD แต่ CD-ROM drive สามารถตรวจสอบความผิดพลาดและใช้งานได้คงทนกว่า เทคโนโลยี CD เริ่มแรกคิดค้นและพัฒนาโดยบริษัทฟิลลิปและบริษัทโซนีจำกัดเพื่อใช้เป็นสื่อบันทึกเสียงดนตรี

อุปกรณ์เก็บข้อมูลที่ใช้แสง: CD-R

- CD-R = Compact Disk Recordable เป็น CD ที่ผู้ใช้สามารถเขียนได้จากเครื่องคอมพิวเตอร์ ไม่ต้องผ่านกระบวนการที่ยุ่งยาก
- ข้อมูลเป็นการบันทึกครั้งเดียวแต่อ่านได้หลายครั้ง บางครั้งเรียกว่า Write Once, Read Many = WORM
- เหมาะสมสำหรับการจัดเก็บข้อมูลที่ต้องการ back up และ ข้อมูลที่ต้องการจัดเก็บเป็น archive

อุปกรณ์เก็บข้อมูลที่ใช้แสง : CD-RW

- **CD-RW = Compact Disk Rewritable เป็น CD ที่ผู้ใช้สามารถเขียน และลบได้หลายครั้งโดยไม่ต้องผ่านกระบวนการที่ยุ่งยาก**
- บางครั้งเรียกว่า Erasable Optical Disk
- เหมาะสำหรับการจัดเก็บข้อมูลที่ต้องการให้มีการเปลี่ยนแปลงในอนาคต

ឧបករណ៍កែប្រិច្ឆេទសាស្ត្រ: DVD

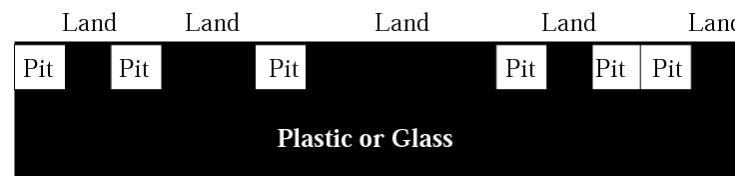
- DVD = Digital Versatlie Disc เป็นหน่วยความจำสำรองที่ใช้แสง
เพาะสำหรับใช้เก็บข้อมูลวิดีโอศน์ เพราะมีความจุสูง
 - CD-ROM ปกติจะมีความจุประมาณ 650 MB แต่ DVD มีความจุตั้งแต่ 4.7 GB ถึง 17 GB
 - DVD ใช้เทคโนโลยีคล้ายกับ CD-ROM แต่มีข้อแตกต่าง บางประการ
ดังนี้
 - (1) ขนาดของ pits เล็กกว่าคือของ DVD มีขนาดเส้นผ่าศูนย์กลาง 0.4 microns ในขณะที่ของ CD มีขนาดเส้นผ่าศูนย์กลาง 0.8 microns

อุปกรณ์เก็บข้อมูลที่ใช้แสง: DVD (ต่อ)

- (2) Tracks ของ DVD อยู่ชิดกันมากกว่า
- (3) ลำแสงที่ใช้กับ DVD เป็นแสงเลเซอร์สีแดง ล้ำกว่าของ CD ใช้แสงอินฟราเรด (infrared)
- (4) DVD อาจมี single-side หรือ double-side

รูปที่ 5-10

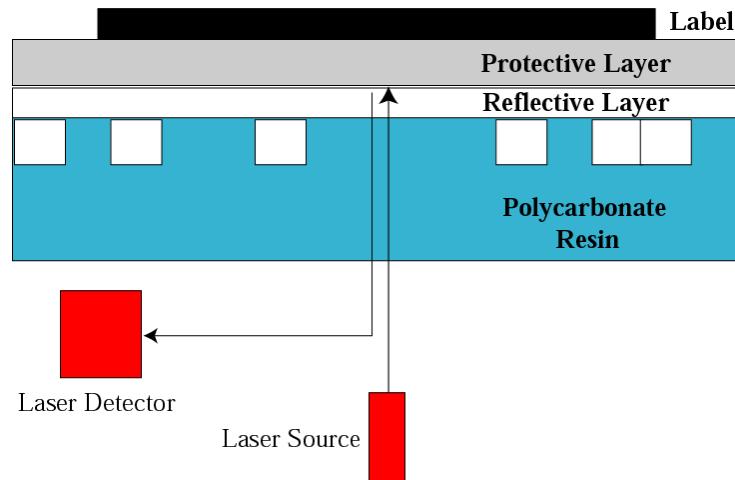
แสดงการสร้างและการใช้ CD-ROM



a. Master Disc



b. Mold



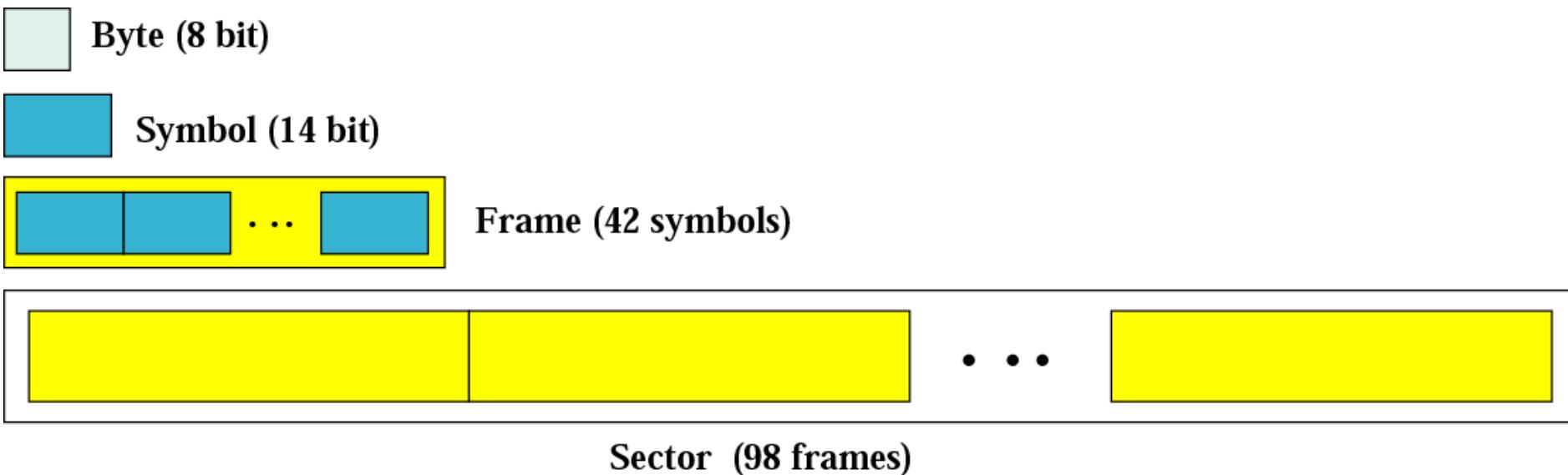
c. CD-ROM

ตารางที่ 5.2 แสดงความเร็วของ CD-ROM

ความเร็ว	อัตราความเร็วของข้อมูล	ขนาด(โดยประมาณ)
1x	153,600 bytes per second	150 KB/s
2x	307,200 bytes per second	300 KB/s
4x	614,400 bytes per second	600 KB/s
6x	921,600 bytes per second	900 KB/s
8x	1,228,800 bytes per second	1.2 MB/s
12x	1,843,200 bytes per second	1.8 MB/s
16x	2,457,600 bytes per second	2.4 MB/s
24x	3,688,400 bytes per second	3.6 MB/s
32x	4,915,200 bytes per second	4.8 MB/s
40x	6,144,000 bytes per second	6 MB/s

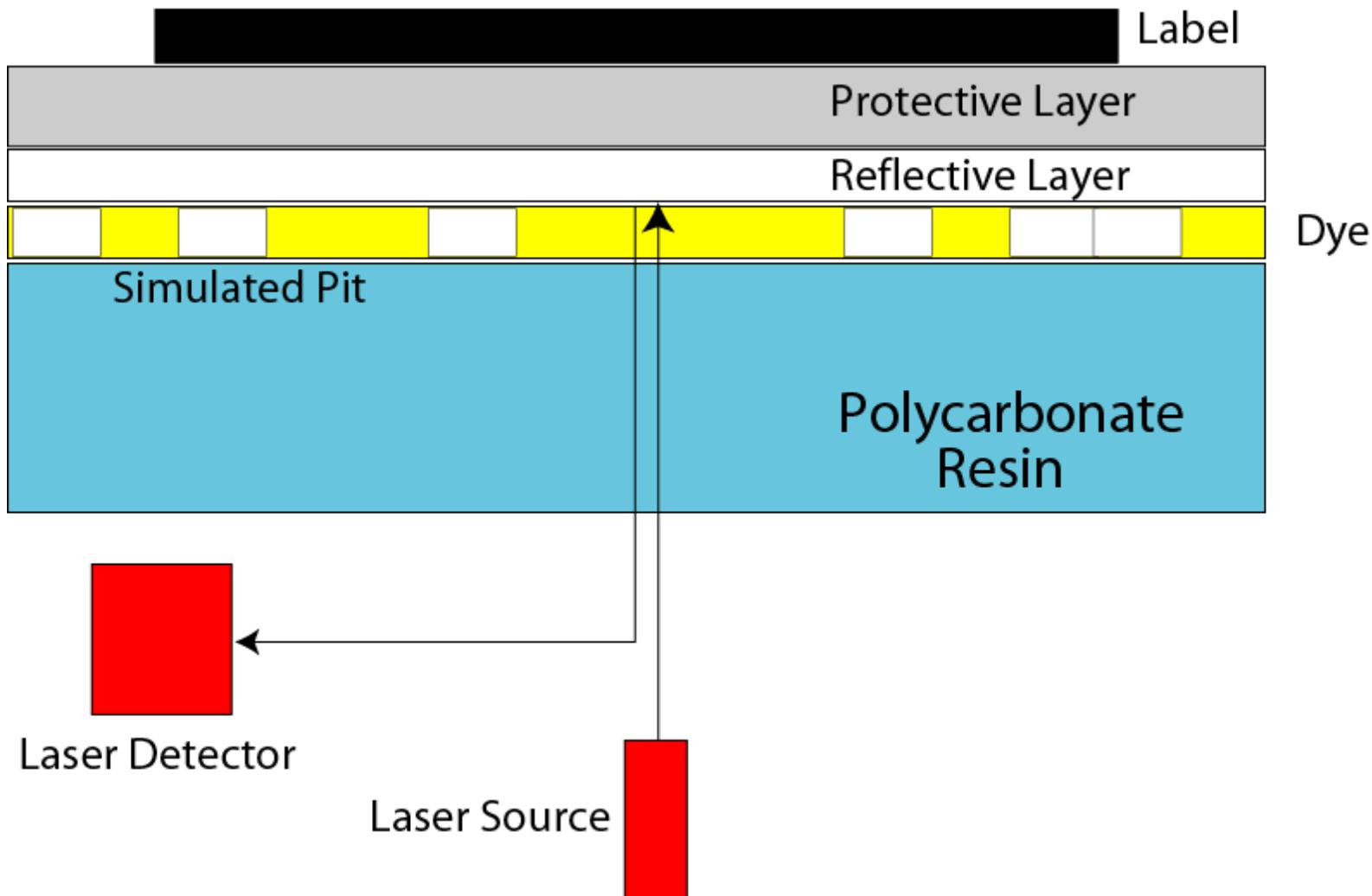
រូបភាព 5-11

រូបແບບទំនាក់ CD-ROM



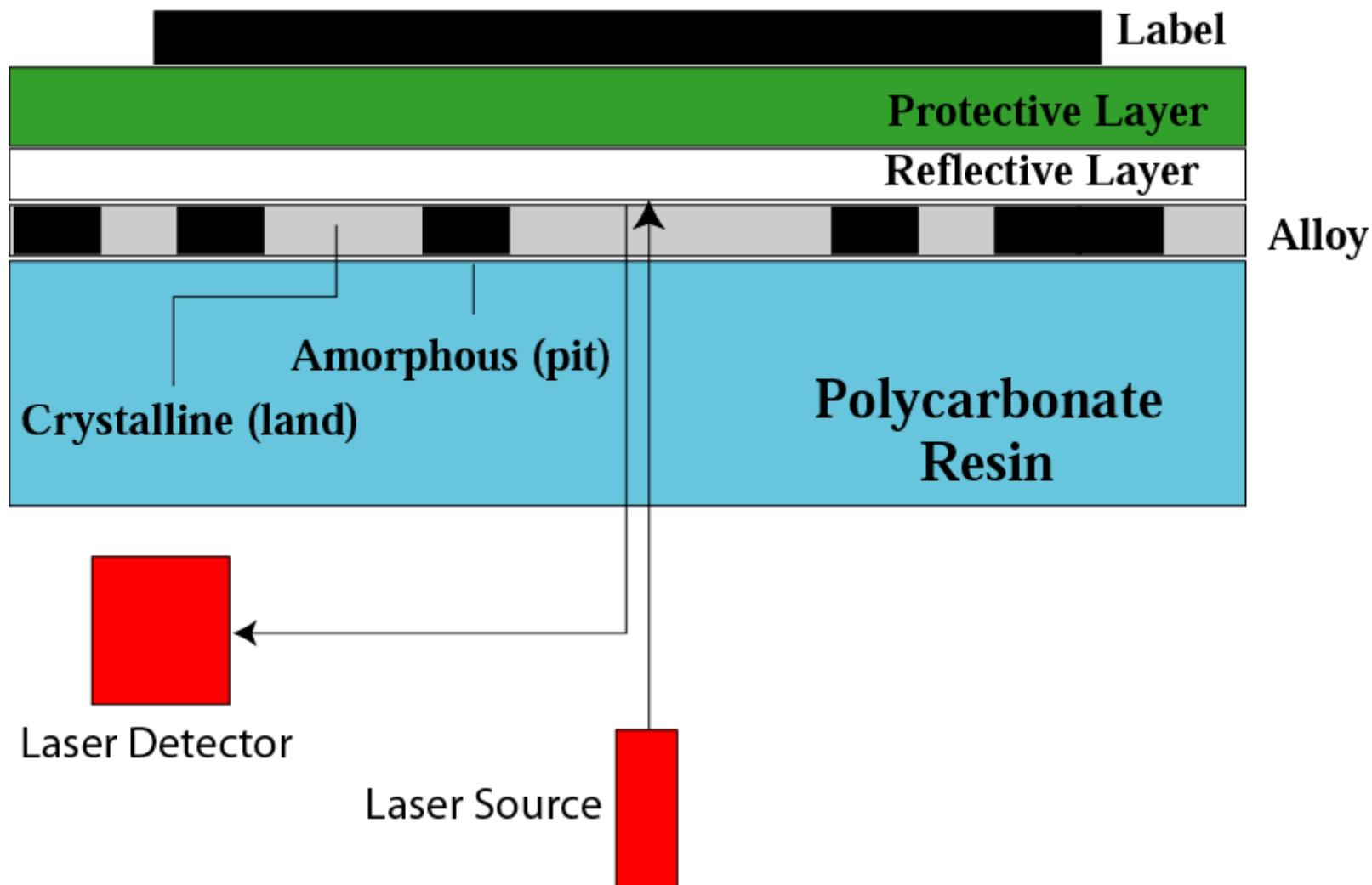
รูปที่ 5-12

แสดงการสร้าง CD-R



รูปที่ 5-13

แสดงการสร้าง CD-RW



ตารางที่ 5.3 ขนาดความจุของ DVD

คุณลักษณะ	ความจุ
single-sided, single-layer	4.7 GB
single-sided, dual-layer	8.5 GB
double-sided, single-layer	9.4 GB
double-sided, dual-layer	17 GB

การบีบอัดข้อมูลใน DVD

- DVD ใช้วิธีการบีบอัดข้อมูลที่เรียกว่า MPEG = Motion Picture Experts Group
- ทำให้ single-side, single-layer DVD สามารถจัดเก็บข้อมูลวิดีโอศักน์ที่มีความละเอียดสูงได้นานถึง 133 นาที
- ปัจจุบัน DVD ได้นำมาใช้อย่างกว้างขวางทั่วไปในการบันทึก การศึกษา การแพทย์ และการท่องเที่ยว ฯลฯ

5.4

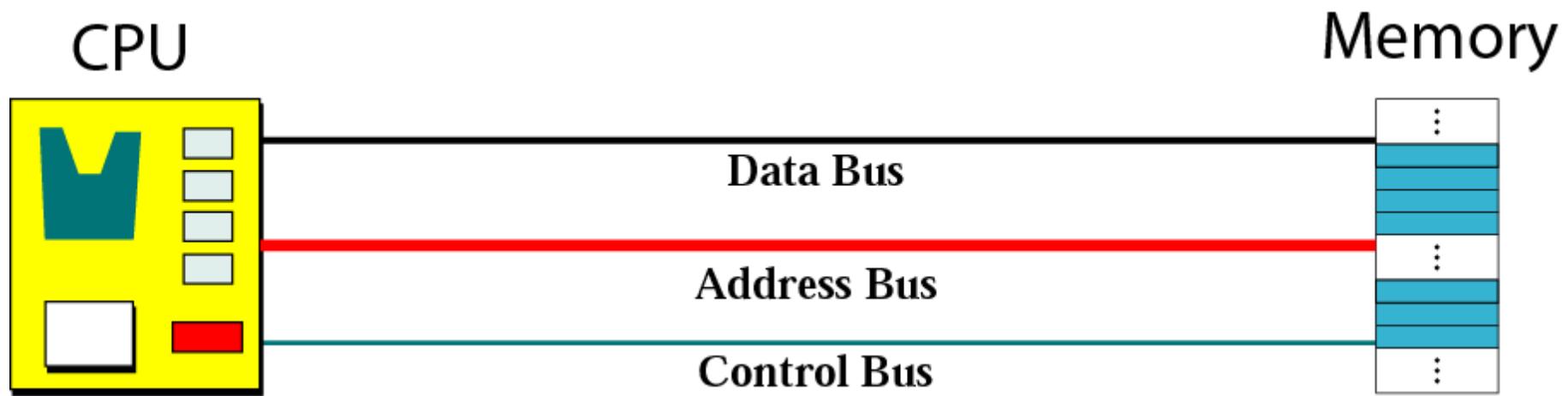
การซื้อมาต่อภัยในระหว่างองค์ประกอบอยู่ ของเครื่องคอมพิวเตอร์

การเชื่อมต่อระหว่างหน่วยประมวลผลกลางกับหน่วยความจำหลัก

- ปกติจะเชื่อมต่อด้วยด้วย wire 3 ประเภทคือ (ดูรูปที่ 5.14)
 - Data bus:** เป็นเส้นทางส่งผ่านข้อมูลระหว่าง memory กับ CPU ถ้าคอมพิวเตอร์มีขนาดของ word เป็น 32 บิต data bus จะประกอบด้วย wire จำนวน 32 เส้น
 - Address bus:** เป็นเส้นทางที่ทำให้ CPU สามารถเข้าถึงเลขที่ตำแหน่งที่อยู่ของหน่วยความจำหลักได้ โดยจำนวน wire ใน address bus ขึ้นอยู่กับขนาดของหน่วยความจำทั้งหมด (address space) ของคอมพิวเตอร์ ถ้าหน่วยความจำมีขนาด 2^n words และจำนวน wire จะต้องมี n เส้น
 - Control bus:** ทำให้ CPU สื่อสารกับ memory ได้ ถ้าคอมพิวเตอร์มีการควบคุม 2^m แบบ จะต้องใช้ wire จำนวน m เส้น เพราะว่าสัญญาณ m บิตสามารถส่งกระแสควบคุมได้ทั้งหมด 2^m การกระทำที่แตกต่างกัน

รูปที่ 5-14

แสดงการเชื่อมต่อระหว่าง CPU กับ memory โดยใช้ bus 3 ชนิด

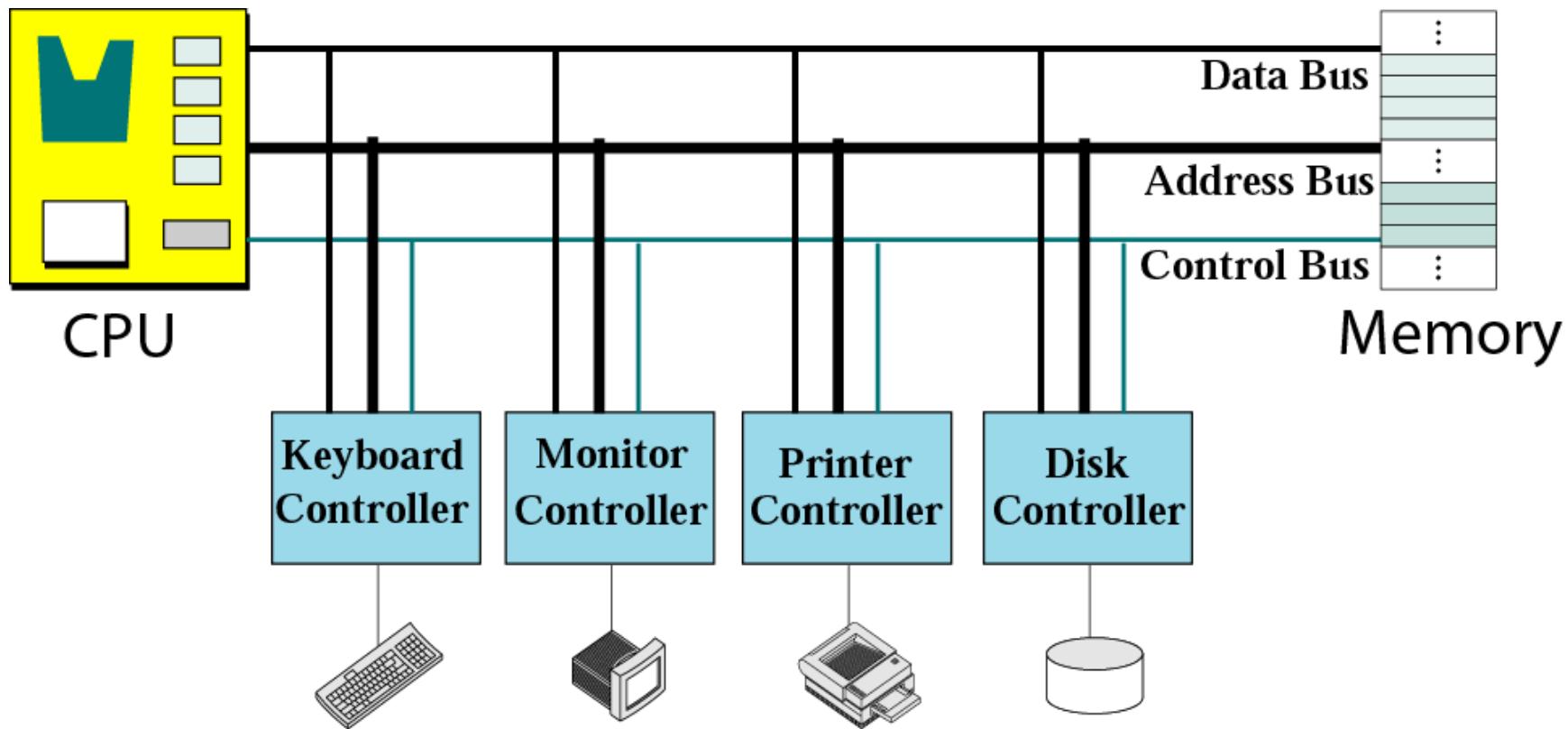


การเชื่อมต่ออุปกรณ์ I/O

- อุปกรณ์ I/O ไม่สามารถเชื่อมต่อโดยตรงกับ buses ที่เชื่อมต่อกับ CPU หรือ Memory ได้ เพราะมีคุณสมบัติของอุปกรณ์ I/O ต่างจากคุณสมบัติของ CPU และ memory
- อุปกรณ์ I/O เป็นอุปกรณ์ประเภท electromechanical, magnetic, หรือ optical ในขณะที่ CPU และ memory เป็นอุปกรณ์ประเภท electronic
- อุปกรณ์ I/O ทำงานช้ากว่า CPU/Memory มา ก จำเป็นที่จะต้องมีตัวกลางมาจัดการความแตกต่างอันนี้
- อุปกรณ์ตัวกลางที่ว่าคือ I/O controller หรือเรียกอีกอย่างหนึ่งว่า interface แต่ละอุปกรณ์ก็จะมี I/O controller แยกกันดังรูปที่ 5.1

รูปที่ 5-15

การเชื่อมต่ออุปกรณ์ I/O กับ Buses



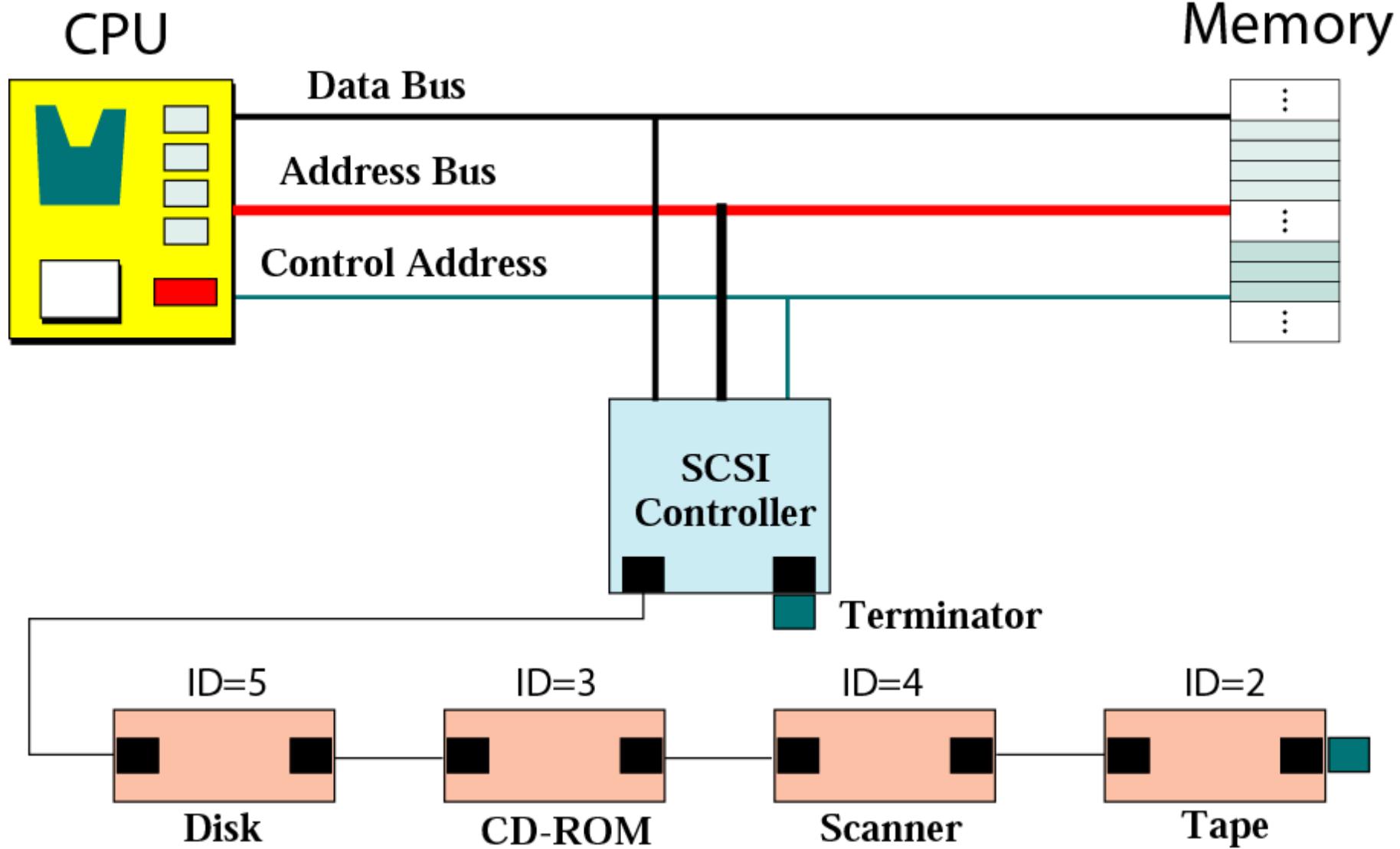
Controllers

- Controller หรือ Interface เป็นอุปกรณ์ที่แก้ปัญหาความแตกต่างระหว่างอุปกรณ์ I/O กับ CPU และ memory
- Controller อาจเป็น serial controller หรือ parallel controller
- Serial controller มี wire เพียงเส้นเดียวที่เชื่อมต่อกับอุปกรณ์ ส่วน parallel controller มี wire หลายเส้นที่เชื่อมต่อ ทำให้สามารถส่งผ่านข้อมูลได้ครั้งละหลายบิต
- Controller ที่ใช้อยู่ในปัจจุบันมีหลายประเภท แต่ที่ใช้กันเป็นที่แพร่หลายมีดังนี้ SCSI, FireWire, USB

Controllers (ต่อ)

(1) SCSI ย่อมาจาก Small Computer System Interface เป็น อุปกรณ์ที่พัฒนาขึ้นเพื่อใช้กับเครื่องคอมพิวเตอร์ Macintosh ในปี ค.ศ. 1984 ปัจจุบัน SCSI มีการใช้ในระบบคอมพิวเตอร์เป็นจำนวนมาก SCSI เป็น interface แบบขนานที่อาจประกอบด้วย wire จำนวน 8, 16, หรือ 32 เส้น Interface แบบ SCSI มีการเชื่อมต่อในลักษณะที่เรียกว่า daisy chain (ดังรูปที่ 5.16) ทั้งหัวและท้ายของการเชื่อมต่อลูกโซ่ต้องมีอุปกรณ์ที่เรียกว่า terminator เพื่อเป็นตัวบ่งบอกจุดสิ้นสุดการเชื่อมต่อ นอกจากนี้ แต่ละอุปกรณ์จะต้องมีเลขตำแหน่งที่อยู่ที่ไม่ซ้ำกันซึ่งเรียกว่า target ID

SCSI controller

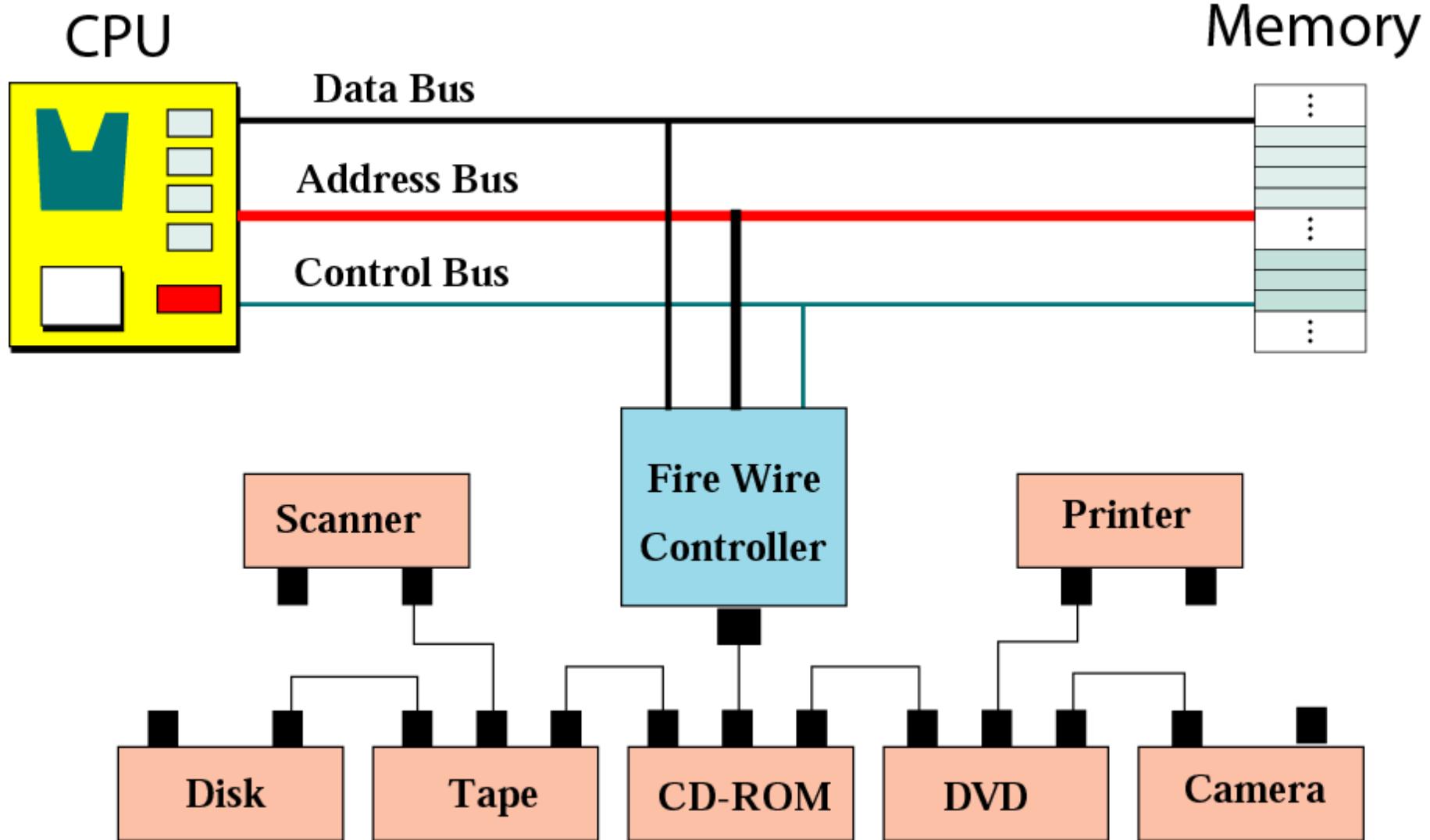


Controller (ต่อ)

(2) FireWire เป็น serial interface ประเภทความเร็วสูง (high-speed) สามารถส่งผ่านข้อมูลเป็นกลุ่มที่เรียกว่า packet ด้วยความเร็วสูงถึง 50 MB/วินาที สามารถเชื่อมต่อกับอุปกรณ์ได้ถึง 63 ชิ้นในลักษณะ daisy chain หรืออาจเชื่อมต่อในลักษณะต้นไม้ (tree) ซึ่งใช้ wire เพียงเส้นเดียว รูปที่ 5.17 แสดงการเชื่อมต่ออุปกรณ์ I/O กับ FireWire Controller ขอให้สังเกตว่า FireWire Controller ไม่ต้องมี Terminator เมื่อนับกับ SCSI Controller

Figure 5-17

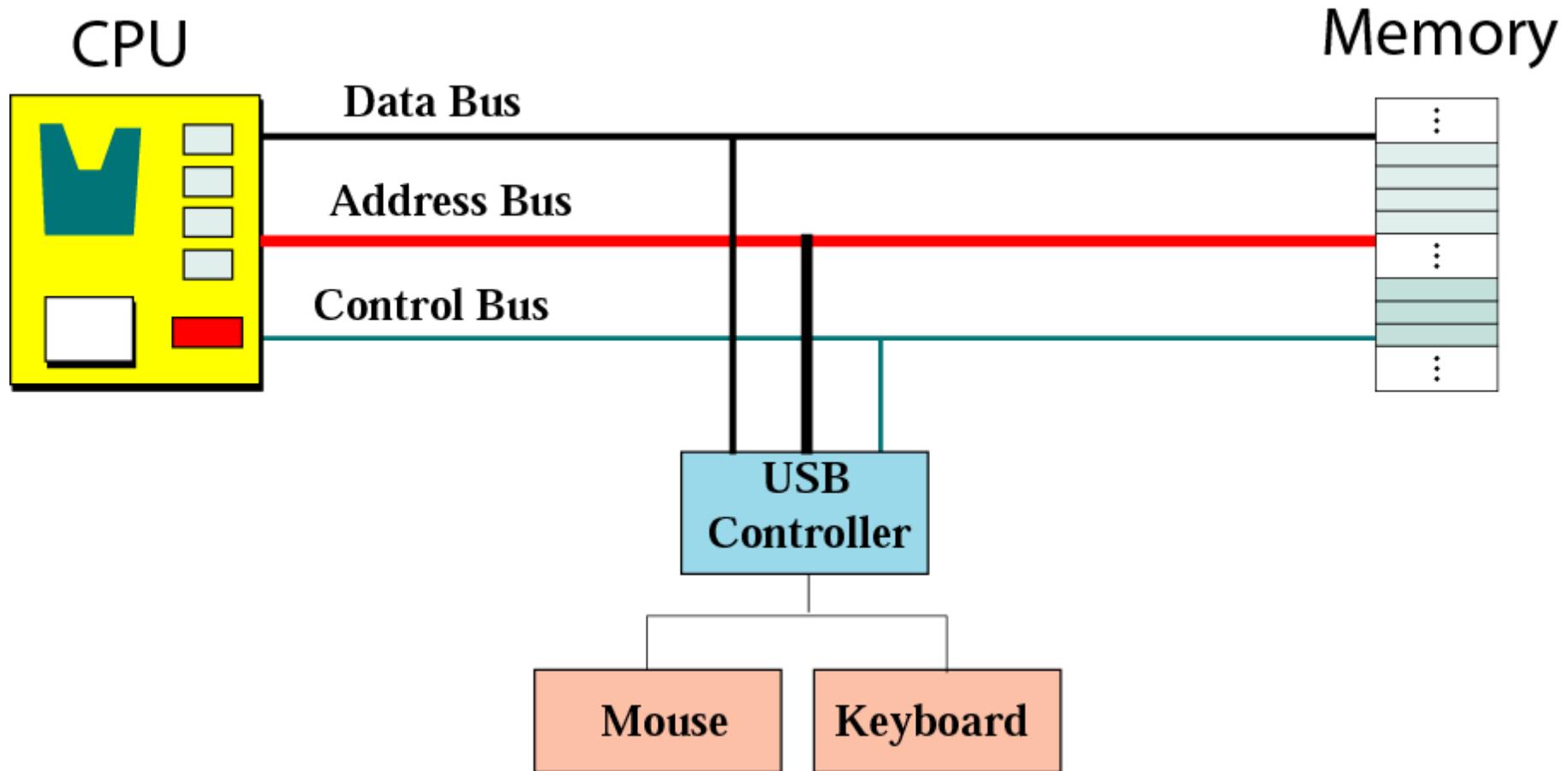
FireWire controller



Controller (ต่อ)

(3) USB ย่อมาจาก Universal Serial Bus ถือเป็น controller ที่เป็นคู่แข่งของ FireWire controller รุปที่ 5.18 แสดงการเชื่อมต่อ USB กับ Bus พ่วงด้วยอุปกรณ์ I/O โดยทั่วไป USB เป็น serial controller ที่ใช้เชื่อมต่อ กับ อุปกรณ์ ที่ ส่งผ่านข้อมูล ค่อนข้างช้า เช่น คีย์บอร์ด และ เม้าส์ กับ เครื่องคอมพิวเตอร์ USB controller สามารถ ส่งผ่านข้อมูล ด้วย ความเร็ว ถึง 1.5 MB/วินาที USB มี bus แบบ 4-wire ที่ใช้ เชื่อมต่อ กับ อุปกรณ์ I/O

USB controller



การกำหนดตำแหน่งที่อยู่ของอุปกรณ์ I/O

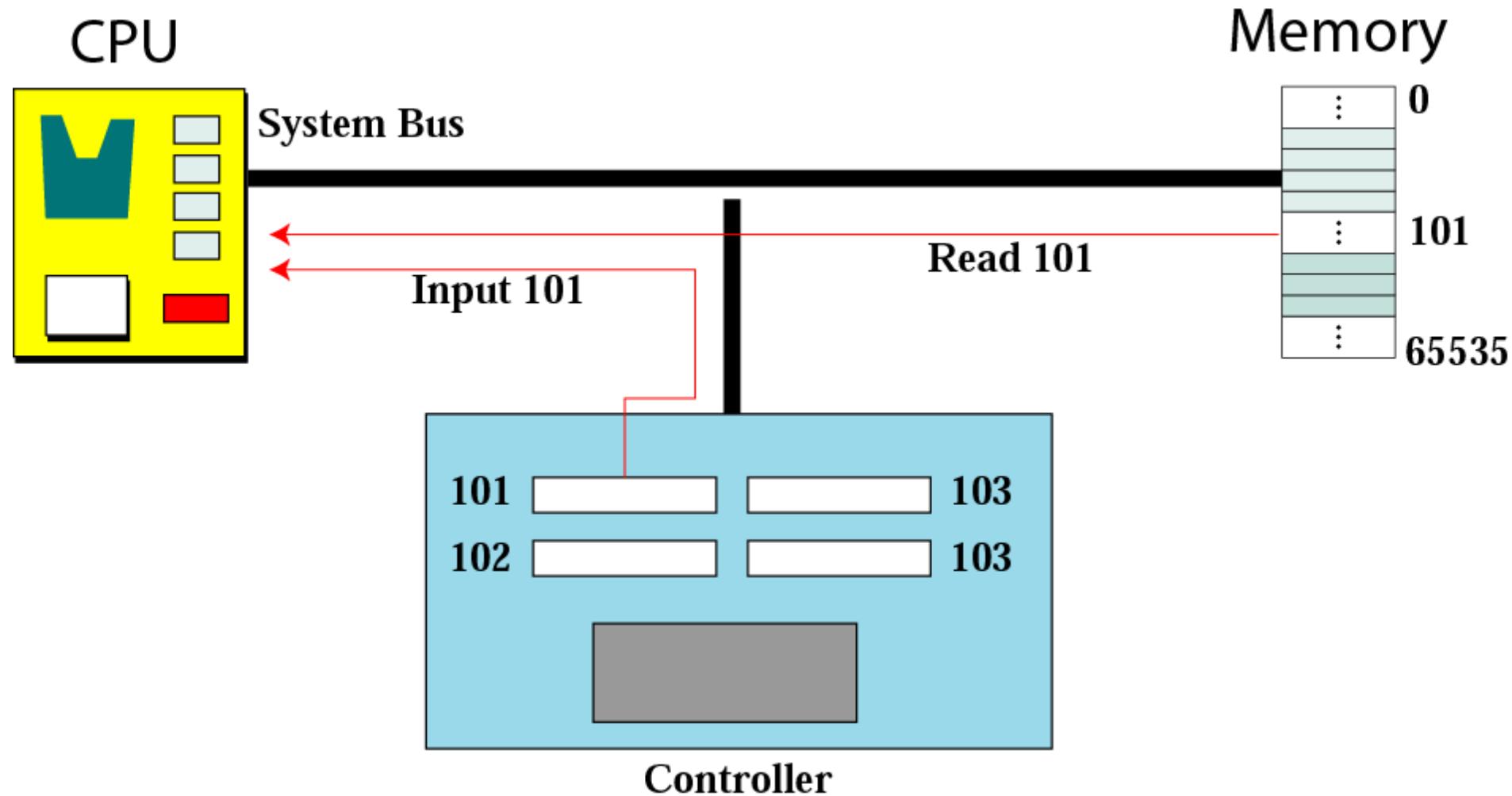
- ปกติ CPU จะใช้ bus เดียวกันสำหรับการรับหรือส่งข้อมูลจากหน่วยความจำและจากอุปกรณ์ I/O ส่วนที่ทำให้เกิดความแตกต่างและคอมพิวเตอร์สามารถทำงานได้โดยไม่ผิดพลาดคือคำสั่ง (instruction)
 - * ถ้าคำสั่งอ้างถึง word ในหน่วยความจำ การส่งผ่านข้อมูลระหว่าง memory กับ CPU ก็เกิดขึ้น ถ้าคำสั่งระบุ I/O device การส่งผ่านข้อมูลระหว่าง I/O device กับ CPU ก็จะเกิดขึ้น

การกำหนดตำแหน่งที่อยู่ของอุปกรณ์ I/O (ต่อ)

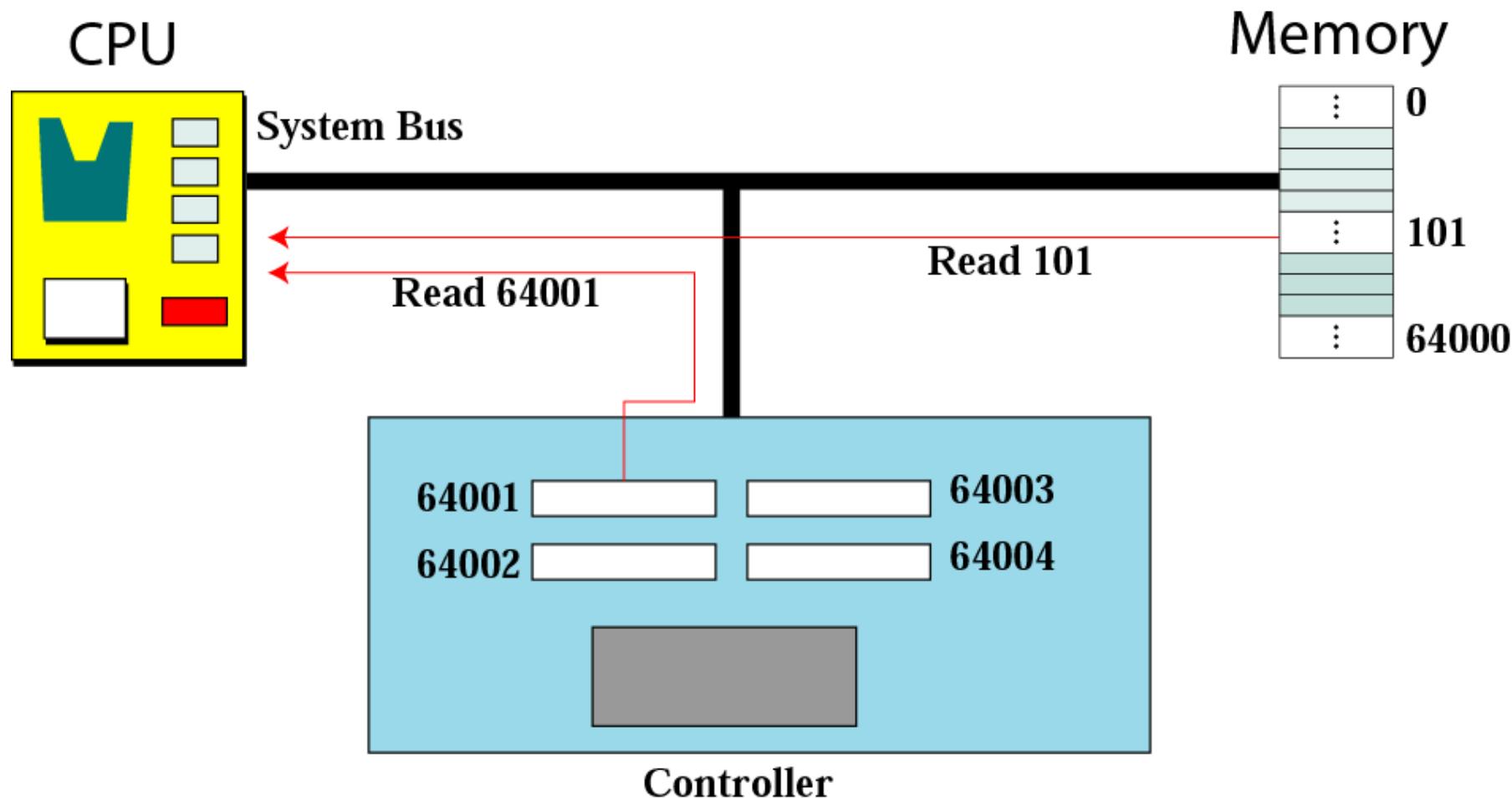
- วิธีการกำหนดตำแหน่งที่อยู่ของอุปกรณ์ I/O มี 2 แบบคือ
 - วิธี **Isolated I/O**: คำสั่งที่ใช้ read/write จาก memory จะแตกต่างจากคำสั่งที่ใช้ read/write จาก I/O device อย่างสิ้นเชิง สำหรับ I/O device จะมีการทดสอบความถูก การอ่านและการเขียนเสมอ แต่ละ I/O device จะมี address ของมันเอง ตัวอย่างคำสั่ง เช่น Read 05, Input 05 (รูปที่ 5.19)
 - วิธี **Memory-mapped I/O**: CPU จะมอง register แต่ละตัวใน I/O controller เหมือนกับ word ใน memory นั่นคือ CPU ไม่มีคำสั่งเฉพาะสำหรับการส่งผ่านข้อมูลจาก memory หรือ I/O device เช่น มีคำสั่ง Read (หรือ Write) เพียงคำสั่งเดียว รูปที่ 5.20 แสดงการกำหนดตำแหน่งที่อยู่แบบ Memory-mapped I/O

รูปที่ 5-19

การกำหนดตำแหน่งที่อยู่แบบ Isolated I/O



การกำหนดตำแหน่งที่อยู่แบบ Memory-mapped I/O



5.5

การເອົກຊື່ຄວ່າໂປຣແກຣມ

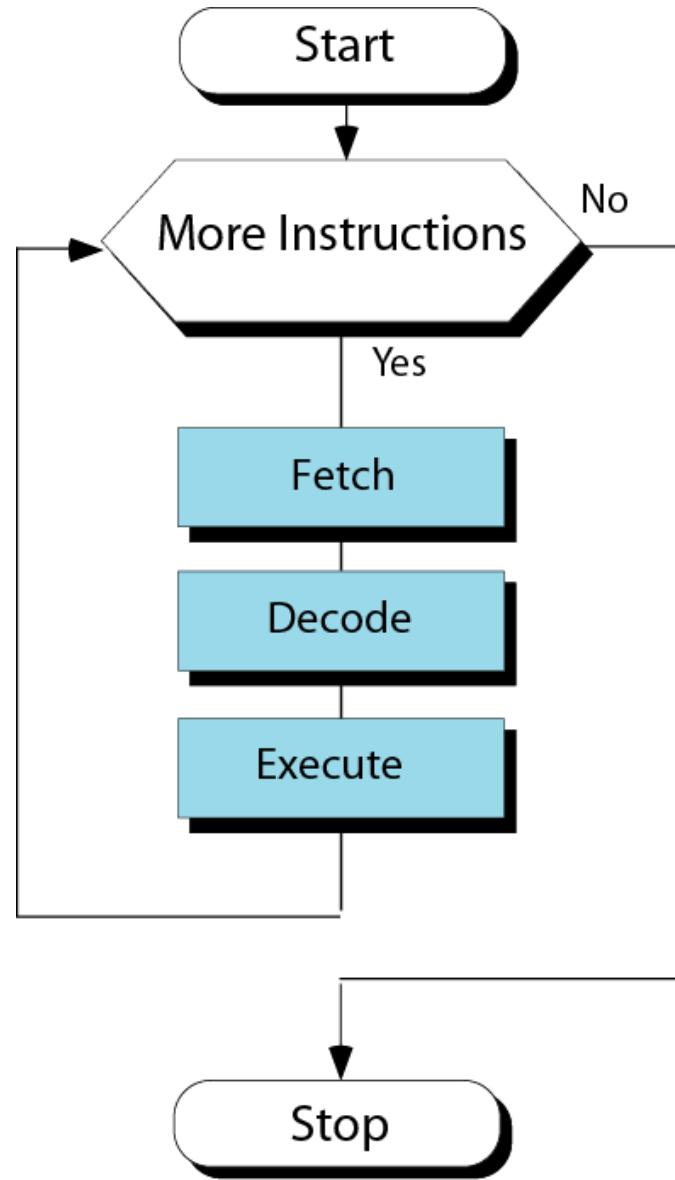
PROGRAM EXECUTION

Machine Cycle

- ในปัจจุบัน คอมพิวเตอร์ออนไลน์จะประมวลผลข้อมูลโดยใช้กลุ่มของคำสั่งที่เรียกว่าโปรแกรม คอมพิวเตอร์ทำการอ่านชีคิวท์โปรแกรมเพื่อสร้างผลลัพธ์จากข้อมูลนำเข้า ทั้งโปรแกรมและข้อมูลต้องอยู่ในหน่วยความจำหลักตามแบบจำลอง von Neumann
- CPU ใช้วิธีการแบบการทำงานที่เรียกว่า machine cycle เพื่อ execute คำสั่งในโปรแกรมตั้งแต่ต้นจนจบโดยทำทีละ 1 คำสั่ง
- Machine cycle ประกอบด้วย 3 ขั้นตอนคือ Fetch, Decode, และ Execute ดังรูปที่ 5.21

รูปที่ 5-21

ขั้นตอนการกระทำใน 1 วงรอบ



Machine Cycle (ต่อ)

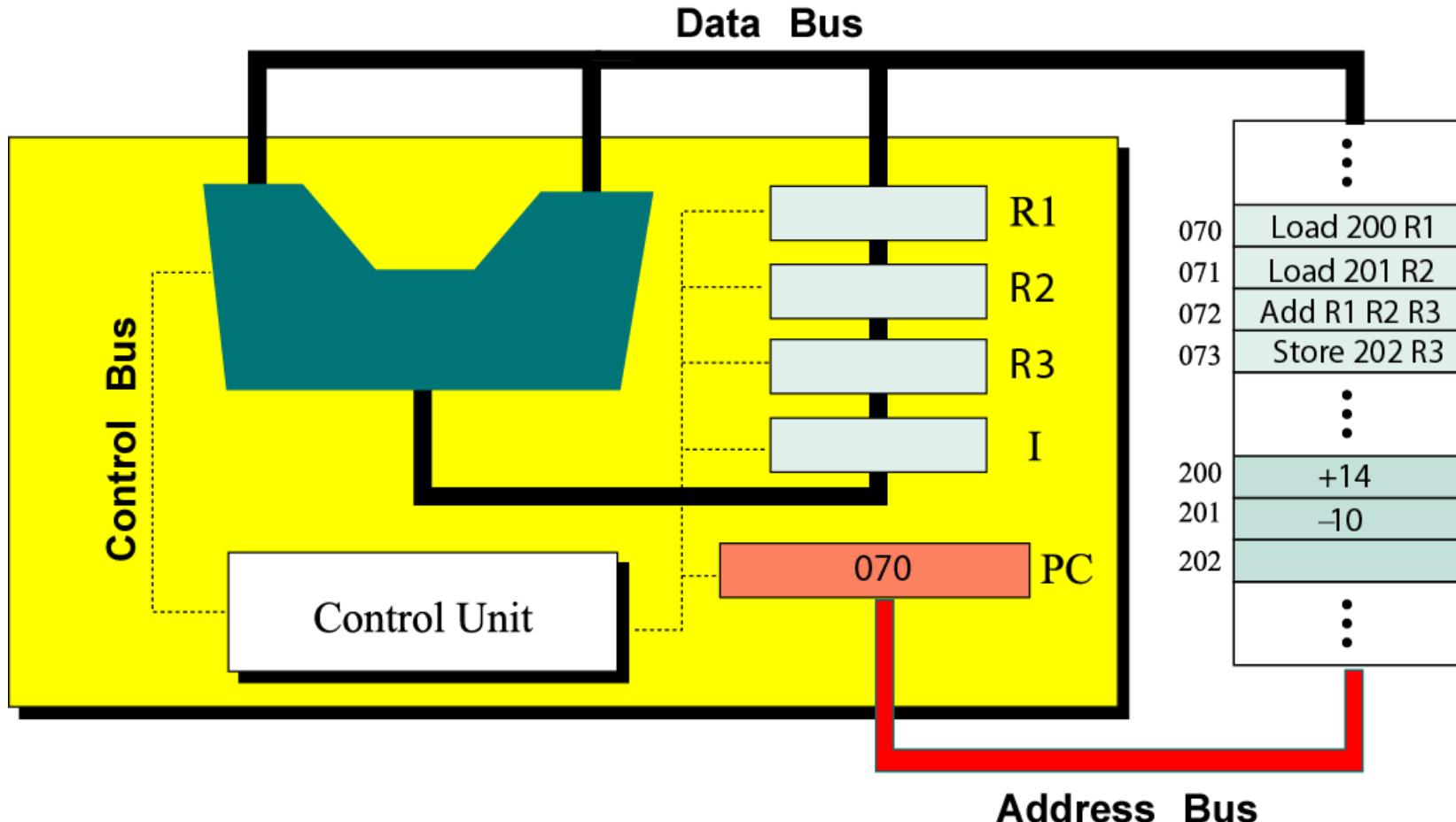
- **Fetch** เป็นขั้นตอนที่หน่วยควบคุมสั่งให้ระบบทำการคัดลอกคำสั่งถัดไป และนำไปเก็บไว้ใน instruction register (ภายใน CPU) ตำแหน่งที่อยู่ของคำสั่งดังกล่าวจะเก็บไว้ใน program counter register หลังจากการคัดลอกคำสั่งแล้ว ค่าของ program counter จะเพิ่มขึ้นอีก 1 เพื่ออ้างอิงถึงคำสั่งถัดไปในหน่วยความจำ
- **Decode** เป็นขั้นตอนที่หน่วยควบคุมตีความหมายของคำสั่งที่อยู่ใน instruction register ผลลัพธ์ที่ได้จะเป็นรหัสไบนาเรีย (binary code) ที่แทนการกระทำที่คอมพิวเตอร์จะต้องทำ

Machine Cycle (ต่อ)

- **Execute** เป็นขั้นตอนที่หน่วยควบคุมส่งสัญญาณไปยังองค์ประกอบใน CPU เพื่อกระทำการตามรหัสใน Nar ที่ได้รับจากขั้นตอน decode เช่นหน่วยควบคุมบอกให้หน่วยรับข้อมูลทำการอ่านข้อมูลจากหน่วยความจำ หรือหน่วยควบคุมบอกให้ ALU ทำการบวกเลข 2 จำนวนที่เก็บอยู่ใน input registers และเก็บผลลัพธ์ไว้ใน output register การกระทำเหล่านี้เกิดขึ้นในขั้นตอน Execute
- รูปที่ 5.22 – รูปที่ 5.23(a,b,c,d) แสดงการบวกเลข 2 จำนวนคือ +14 กับ -10 ผลลัพธ์ที่ได้เก็บไว้ที่ address 202

รูปที่ 5-22

ค่าในหน่วยความจำและรีจิสเตอร์ก่อนการทำงาน



คำอธิบายรูปที่ 5.23

รูปที่ 5.23a: คำสั่งโหลด (Load 200 R1) เครื่องจะทำงานตามว่างรอใน 3 ขั้นตอนคือ fetch, decode, และ execute ผลของคำสั่งนี้คือค่าในหน่วยความจำ ณ ตำแหน่งที่อยู่ 200 จะถูก load ไปเก็บไว้ที่ register R1

รูปที่ 5.23b: คำสั่งที่สอง (Load 201 R2) เครื่องจะทำงานตามว่างรอใน 3 ขั้นตอนคือ fetch, decode, และ execute ผลของคำสั่งนี้คือค่าในหน่วยความจำ ณ ตำแหน่งที่อยู่ 201 จะถูก load ไปเก็บไว้ที่ register R2

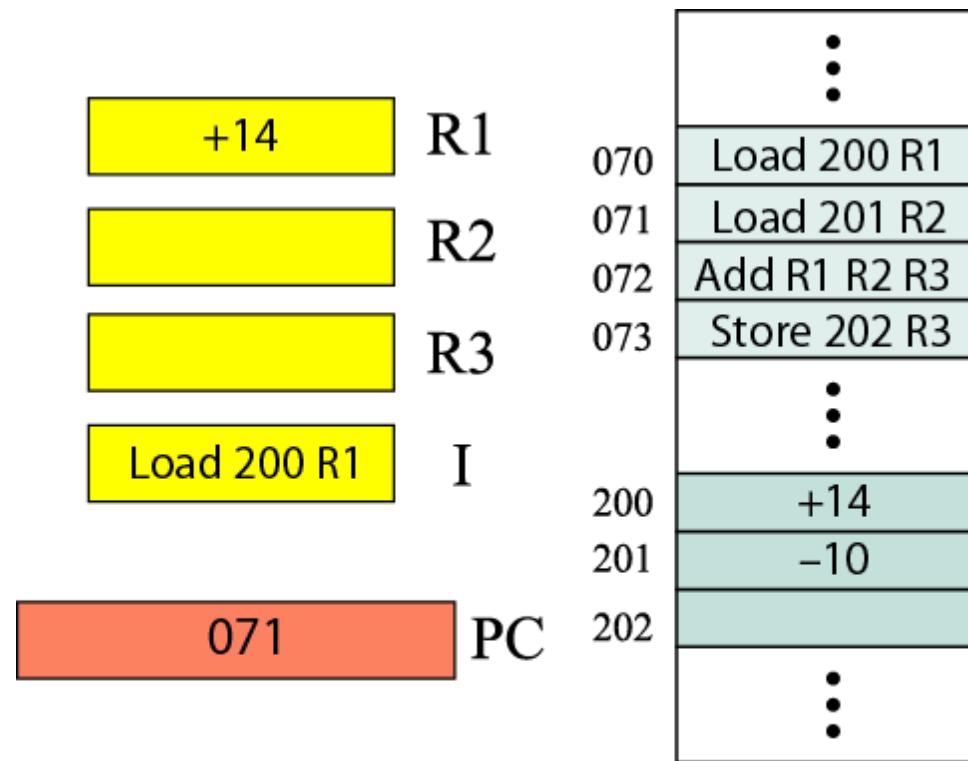
รูปที่ 5.23c: คำสั่งที่สาม (Add R1 R2 R3) เครื่องจะทำงานตามว่างรอใน 3 ขั้นตอนคือ fetch, decode, และ execute เพื่อบวกค่าที่อยู่ใน R1 กับค่าที่อยู่ใน R2 และเก็บผลลัพธ์ไว้ใน R3

รูปที่ 5.23d: คำสั่งที่สี่ (Store 202 R3) เครื่องจะทำงานตามว่างรอใน 3 ขั้นตอนคือ fetch, decode, และ execute เพื่อเก็บผลลัพธ์ไว้ที่ตำแหน่ง 203

รูปที่ 5-23.a

ค่าในหน่วยความจำและรีจิสเตอร์

หลังการทำงานในแต่ละวงรอบ

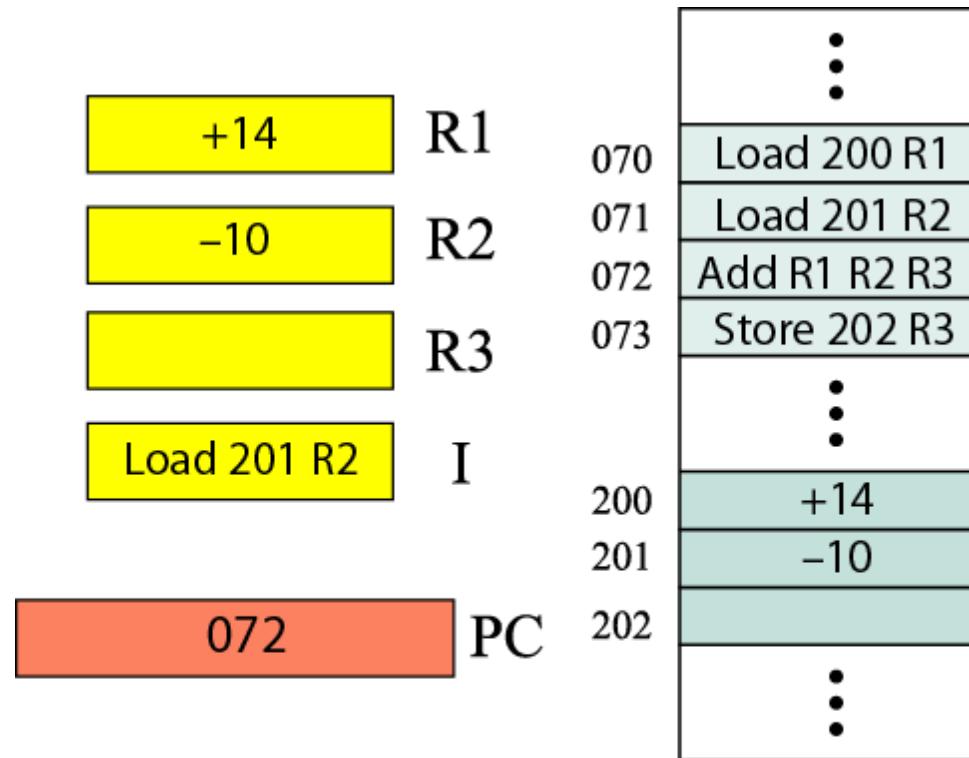


a. After first instruction

รูปที่ 5-23.b

ค่าในหน่วยความจำและรีจิสเตอร์

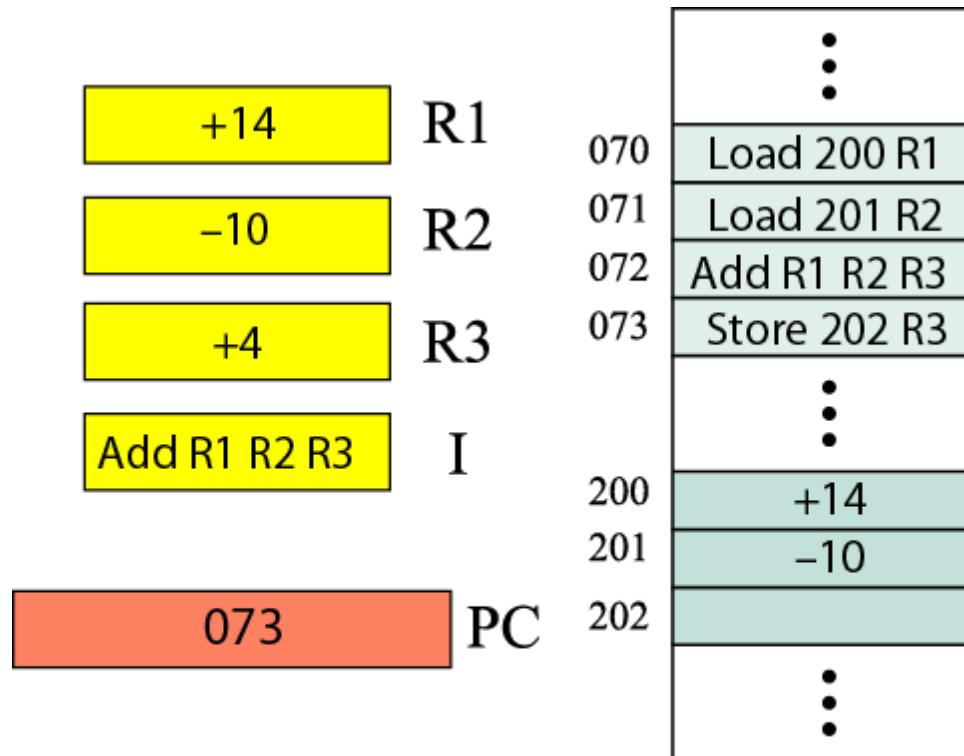
หลังการทำงานในแต่ละวงรอบ



b. After second instruction

ค่าในหน่วยความจำและรีจิสเตอร์

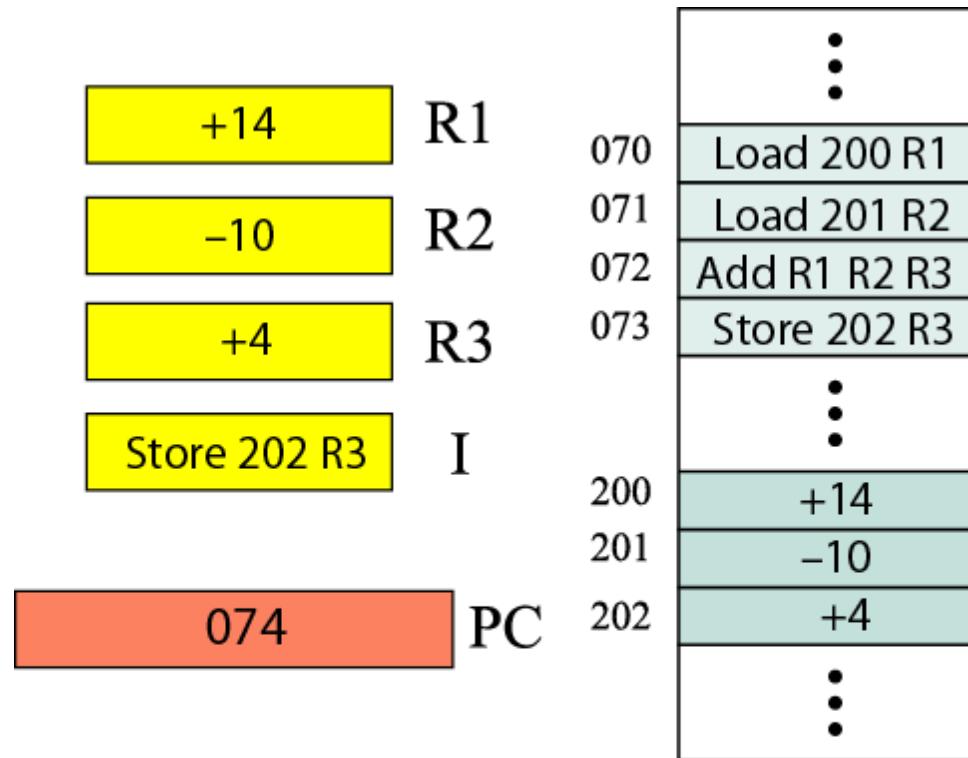
หลังการทำงานในแต่ละวงรอบ



c. After third instruction

รูปที่ 5-23.d

ค่าในหน่วยความจำและรีจิสเตอร์
หลังการทำงานในแต่ละวงรอบ



d. After fourth instruction

การกระทำ Input/Output

ในการทำงานของคอมพิวเตอร์นั้นจะต้องมีคำสั่งในการถ่ายโอนข้อมูลระหว่างอุปกรณ์ I/O กับ CPU และ หน่วยความจำ เนื่องจาก อุปกรณ์ I/O ทำงานช้ากว่า CPU มาดังนั้นการกระทำการของ CPU จะต้อง ทำให้เกิดขึ้นในเวลาเดียวกัน (synchronize) กับอุปกรณ์ I/O วิธีการทำให้เกิดขึ้นในเวลาเดียวกันที่ใช้กันอยู่มี 3 แบบคือ

(1) Programmed I/O วิธีนี้ CPU จะคอยอุปกรณ์ I/O การถ่ายโอนข้อมูลระหว่างอุปกรณ์ I/O กับ CPU กระทำโดยคำสั่งภายในโปรแกรม เมื่อ CPU พบรคำสั่ง I/O มันจะไม่ทำอะไรมากกว่าการถ่ายโอนข้อมูลจะถึงสุด CPU จะตรวจสอบสถานะของ I/O drive อยู่ตลอดเวลา

การกระทำ Input/Output (ต่อ)

ถ้าอุปกรณ์ I/O ว่าง ข้อมูลจะถูกถ่ายโอนเข้าสู่ CPU (ดังรูปที่ 5.24) ปัญหาหลักของวิธีนี้คือเป็นวิธีที่เสียเวลาของ CPU ที่ต้องตรวจสอบสถานะของอุปกรณ์ I/O ตลอดเวลา

(2) Interrupt-Driven I/O วิธีนี้ CPU จะส่งสัญญาณบอกอุปกรณ์ I/O ก่อนว่าการถ่ายโอนข้อมูลกำลังจะเกิดขึ้นโดยไม่มีการตรวจสอบสถานะของอุปกรณ์ I/O เมื่ออุปกรณ์ I/O ว่างก็จะส่งสัญญาณบอก (interrupt) CPU ในระหว่างนี้ CPU สามารถทำงานอื่นได้ เช่น execute โปรแกรมอื่น ถ่ายโอนข้อมูลจาก/ไปยังอุปกรณ์ I/O อื่นๆ เป็นต้น (รูปที่ 5.25) วิธีนี้ข้อมูลจะถูกส่งผ่านเข้าสู่หน่วยความจำหลังจาก การรับข้อมูลเข้า และส่งผ่านข้อมูลออกจากหน่วยความจำ ก่อนการกระทำแสดงผล

การกระทำ Input/Output (ต่อ)

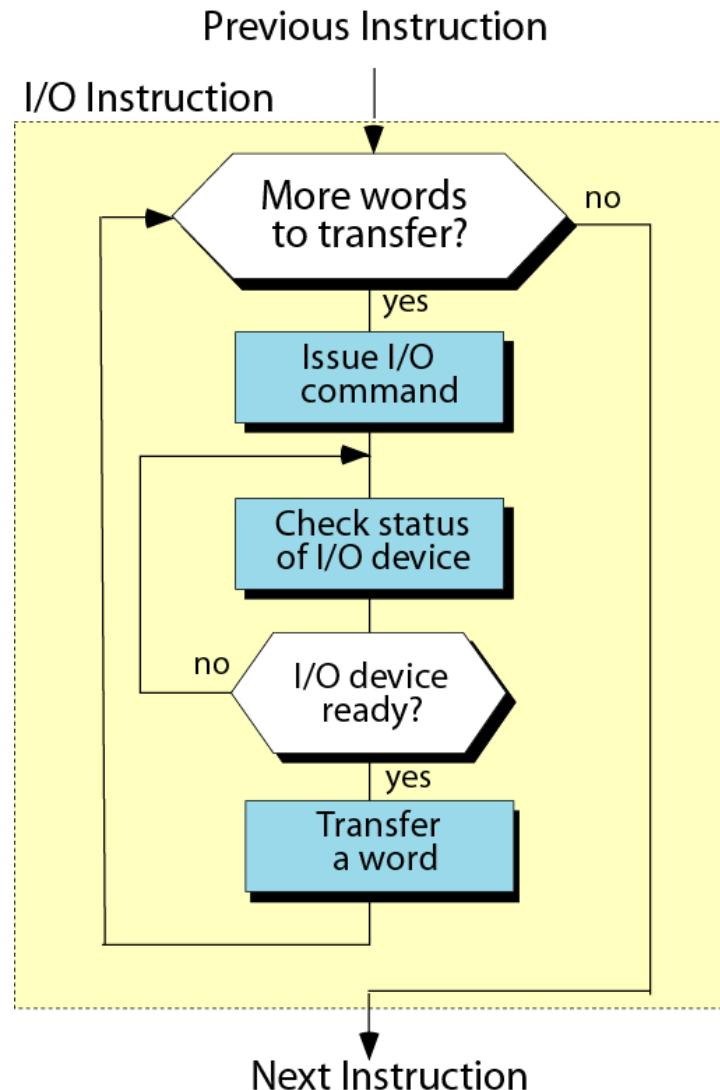
(3) Direct Memory Access (DMA) วิธีนี้จะทำการส่งผ่านข้อมูล เป็นบล็อกขนาดใหญ่โดยตรงระหว่างอุปกรณ์ I/O ที่มีความเร็วสูง (เช่น งานแม่เหล็ก) กับ หน่วยความจำโดยไม่ผ่าน CPU วิธีนี้จะช่วยแบ่งเบาภาระของ CPU จากการที่จะต้องทำฟังก์ชันบางอย่าง DMA controller มีกลุ่มของรีจิสเตอร์ที่ใช้สำหรับเก็บบล็อกของข้อมูลก่อนและหลังการส่งผ่านข้อมูลกับหน่วยความจำ รูปที่ 5.26 แสดงการเชื่อมต่อ DMA controller กับ bus สำหรับการกระทำ I/O ตัว CPU จะส่งสัญญาณไปยัง DMA โดยสัญญาณประกอบด้วย (1) ประเภทของการส่งผ่าน (input หรือ output) (2) เลขที่ตำแหน่งที่อยู่เริ่มต้นในหน่วยความจำ และ (3) จำนวนไบท์ทั้งหมดที่จะส่งผ่าน ณ ตอนนี้ CPU จะว่างสำหรับงานอื่น

การกระทำ Input/Output (ต่อ)

เมื่อพร้อมที่จะส่งผ่านข้อมูล DMA controller จะบอกกับ CPU ว่าต้องการใช้ (ครอบครอง) bus จากนั้น CPU ก็จะหยุดการใช้ bus แล้วปล่อยให้ controller ใช้ หลังจากการส่งผ่านข้อมูลสิ้นสุดลง CPU ก็จะกลับมาทำงานตามปกติ (ดังรูปที่ 5.27) ขอให้สังเกตว่า วิธีนี้ CPU จะว่างในช่วงระยะเวลาอันสั้น เมื่อเปรียบเทียบกับวิธีอื่น และจะว่างเฉพาะช่วงเวลาที่มีการส่งผ่านข้อมูลระหว่าง DMA กับหน่วยความจำเท่านั้น ไม่ใช่ช่วงเวลาที่อุปกรณ์ I/O กำลังเตรียมการส่งผ่าน

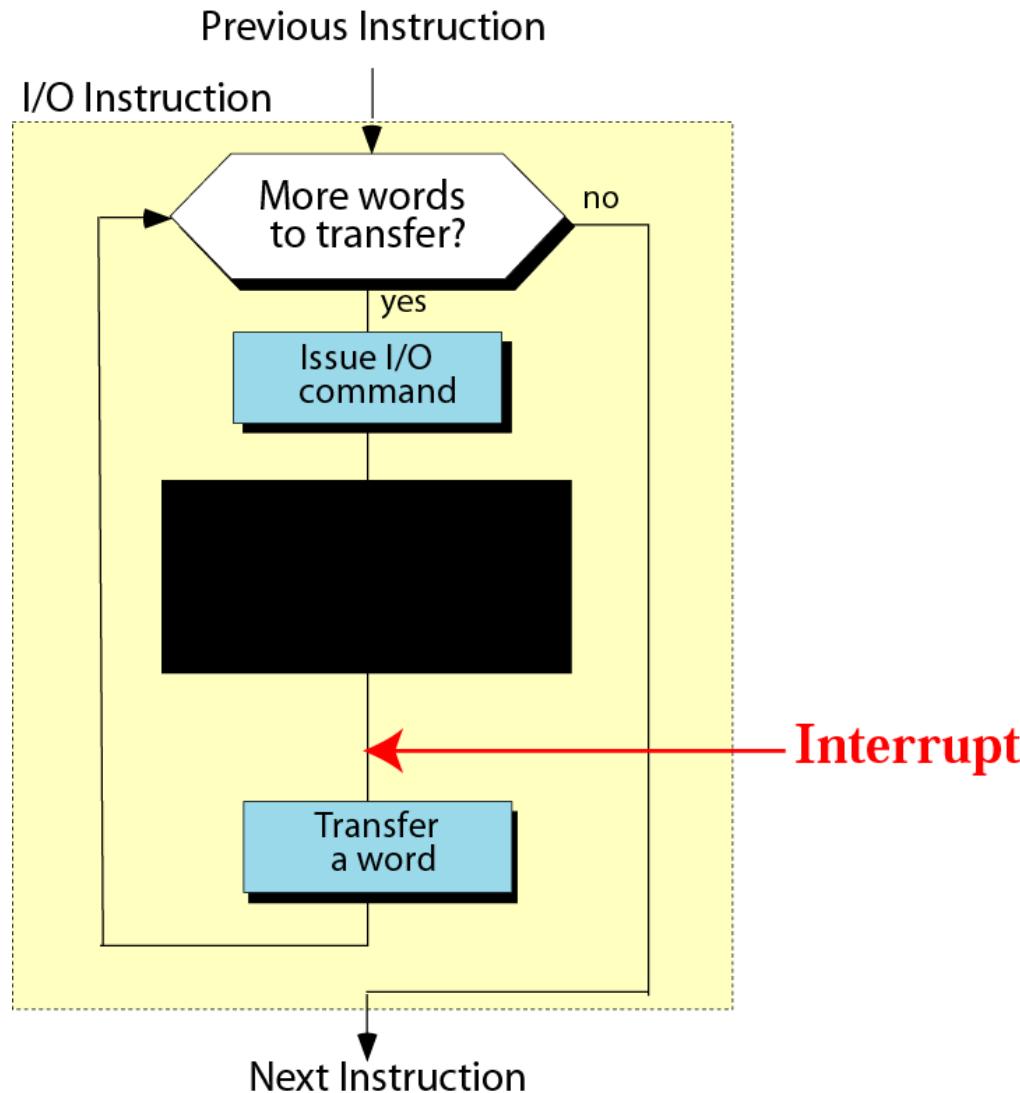
រូបទំនាក់ទំនង 5-24

Programmed I/O

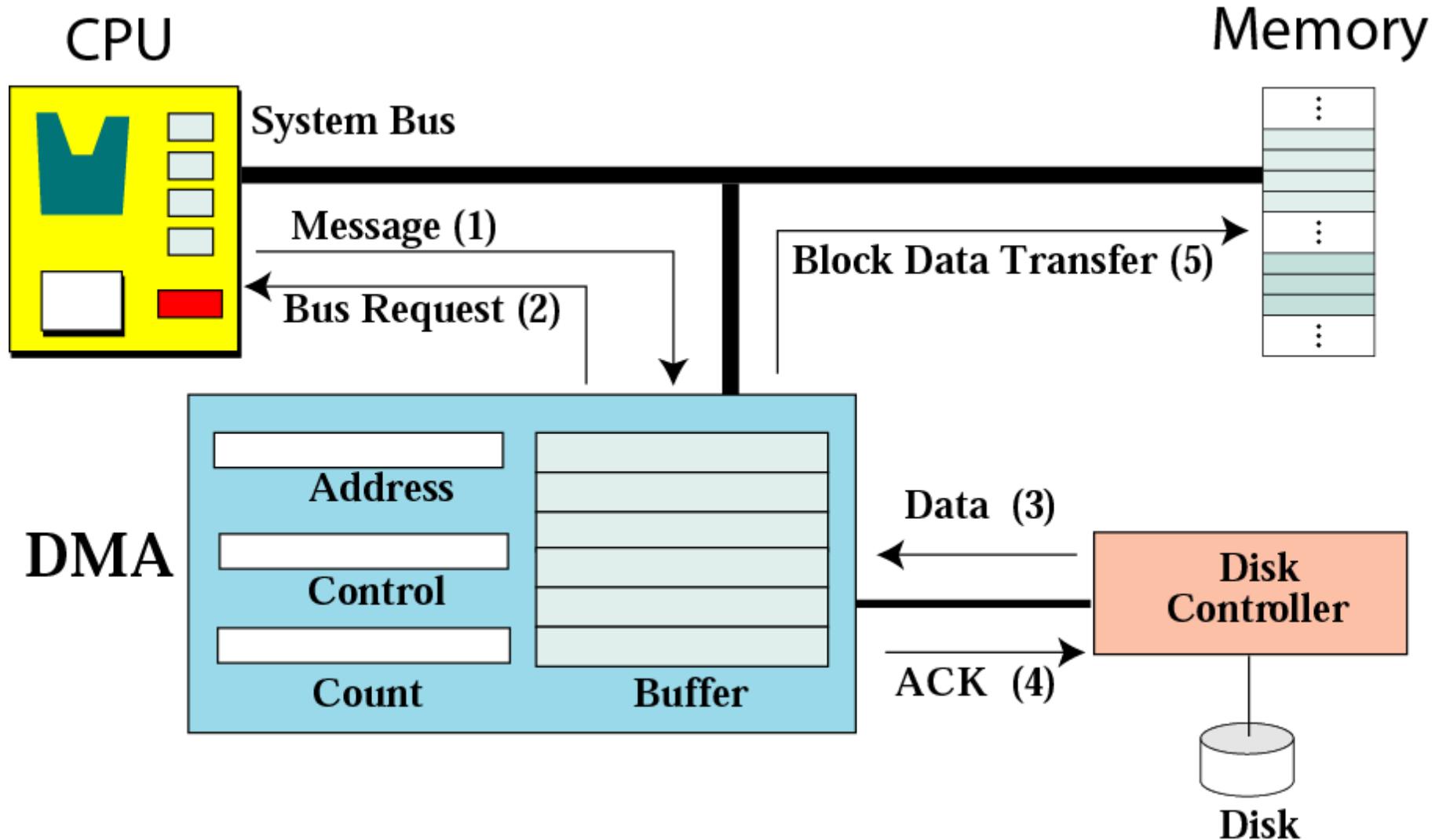


រូបភាព 5-25

Interrupt-driven I/O



ឧប្បី 5-26 DMA connection to the general bus



DMA input/output

Previous Instruction

I/O Instruction

Issue I/O
command

DMA Transfer

Transfer
a block

Transfer
a block

Transfer
a block

Next Instruction



Brooks/Cole
Thomson Learning™

©Brooks/Cole, 2003

5.6

สถาปัตยกรรม 2 รูปแบบ

สถาปัตยกรรม 2 รูปแบบ

การออกแบบเครื่องคอมพิวเตอร์ได้มีการเปลี่ยนแปลงอย่างมากมาย ในช่วงทศวรรษที่ผ่านมา แต่การออกแบบที่เป็นพื้นฐาน 2 รูปแบบยังคง คงอยู่จนถึงทุกวันนี้คือ CISC กับ RISC รายละเอียดพอสังเขป มีดังนี้

(1) **CICS = Complex Instruction Set Computer** มีชุดคำสั่ง จำนวนมาก รวมถึงคำสั่งที่ слับซับซ้อน การเขียนโปรแกรมไม่ยุ่งยาก เพราะมีคำสั่งทั้งที่ทำงานง่ายๆ และคำสั่งที่ทำงานซับซ้อนให้ใช้ความ เหนมาะสมแต่ใช้คำสั่งเดียว ผู้เขียนโปรแกรมไม่ต้องเขียนชุดคำสั่งที่ยุ่ง ยากเพื่อทำงานที่ слับซับซ้อน

สถาปัตยกรรม 2 รูปแบบ (ต่อ)

(2) RISC = Reduced Instruction Set Computer กลยุทธ์เบื้องหลังของสถาปัตยกรรมนี้คือการให้มีจำนวนคำสั่งน้อยๆ แต่ละคำสั่งทำงานพื้นฐานง่ายๆ งานที่ยุ่งยากจะใช้คำสั่งง่ายๆ หลายคำสั่งรวมกัน การเขียนโปรแกรมจะยุ่งยากและยาวกว่าแบบแรก เพราะงานที่ยุ่งยากซับซ้อน จะทำการเลียนแบบโดยการใช้คำสั่งพื้นฐานจำนวนมาก ตัวอย่างสถาปัตยกรรม RISC คือ PowerPC series ของ processor ที่ใช้ในเครื่องคอมพิวเตอร์ Apple

คำสำคัญ

address bus	execute	polycarbonate resin
address space	fetch	printer
arithmetic logic unit (ALU)	FierWire	program counter
arithmetic operation	frame	PROM
Bit pattern		