

องค์ประกอบของเครื่องคอมพิวเตอร์

และภาษาแอสเซมบลี:

ARM และ RaspberryPi3

Inode/File System  
R

## บทที่ 7 อุปกรณ์เก็บรักษาข้อมูลและระบบไฟล์

รศ.ดร.สุรินทร์ กิตติรกุล

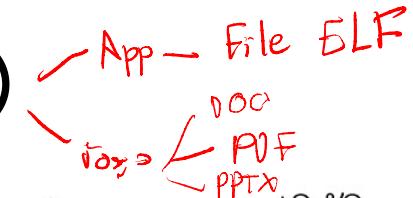
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

# สารบัญ

- 7.1 ระบบไฟล์ (File System) รีบ用
- 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip)
  - 7.3 การ์ดหน่วยความจำ SD (Secure Digital)
  - 7.4 โซลิดสเตทไดรฟ์ (Solid-State Drive: SSD)
  - 7.5 ฮาร์ดดิสก์ไดรฟ์ (Hard Disk Drive: HDD)
  - 7.6 สรุปท้ายบทดูว่าดูว่า

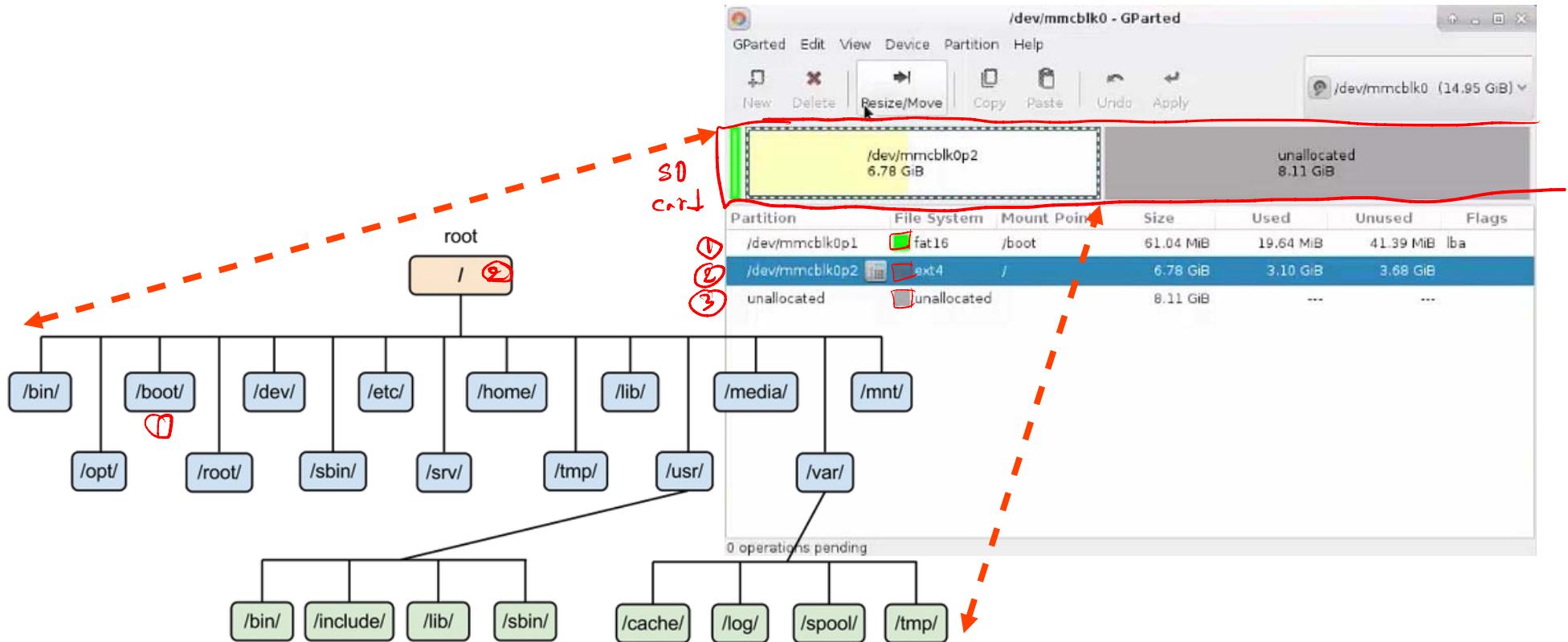
# 7.1 ระบบไฟล์ (File System)



ไฟล์จะเก็บอยู่ในอุปกรณ์เก็บรักษาข้อมูลตามโครงสร้างหรือระบบไฟล์ (File System) ของระบบปฏิบัติการ รนน์ๆ ผู้ใช้และนักพัฒนาสามารถใช้งานไฟล์โดยการ สร้างไฟล์ (Create) เขียน/บันทึก/อ่านไฟล์ เปลี่ยนชื่อ (Rename) ไฟล์ ลบ (Remove/Delete) ไฟล์ ทำสำเนา (Copy) ย้ายตำแหน่ง (Move) ไฟล์ และกู้คืน (Recover) ไฟล์เมื่อเกิดปัญหา โดยไม่จำเป็นต้องรับรู้ว่าระบบไฟล์เป็นชนิดใด เนื่องจากระบบปฏิบัติการได้ซ่อนรายละเอียดไว้ เพื่อให้ผู้ใช้และนักพัฒนาโปรแกรม สามารถพัฒนาโปรแกรมได้สะดวกมากขึ้น ตำราเล่มนี้ จะเน้นที่พื้นฐานของระบบไฟล์ดังเดิมของระบบยูนิกซ์ ซึ่งระบบอื่นๆ ได้พัฒนาต่อยอด ระบบบริหารจัดการไฟล์ ที่สำคัญและเป็นที่นิยม ได้แก่

- ระบบ NTFS (File System) สำหรับระบบวินโดวส์ รายละเอียดเพิ่มเติมที่ [ntfs.com](http://ntfs.com)
- ระบบ EXT4 สำหรับระบบลีนุกซ์ รายละเอียดเพิ่มเติมที่ [wikipedia](https://en.wikipedia.org/wiki/Ext4)
- ระบบ Apple File System (APFS) สำหรับระบบ MAC OS รายละเอียดเพิ่มเติมที่ [wikipedia](https://en.wikipedia.org/wiki/APFS)

# 7.1 ระบบไฟล์ (File System)



# 7.1 ระบบไฟล์ (File System)

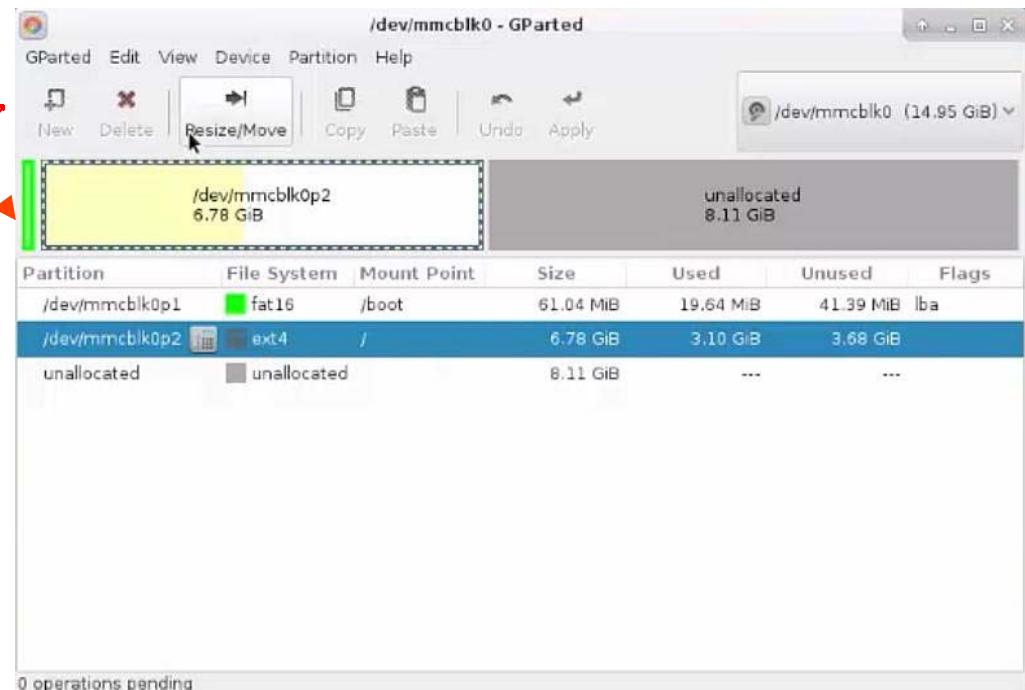
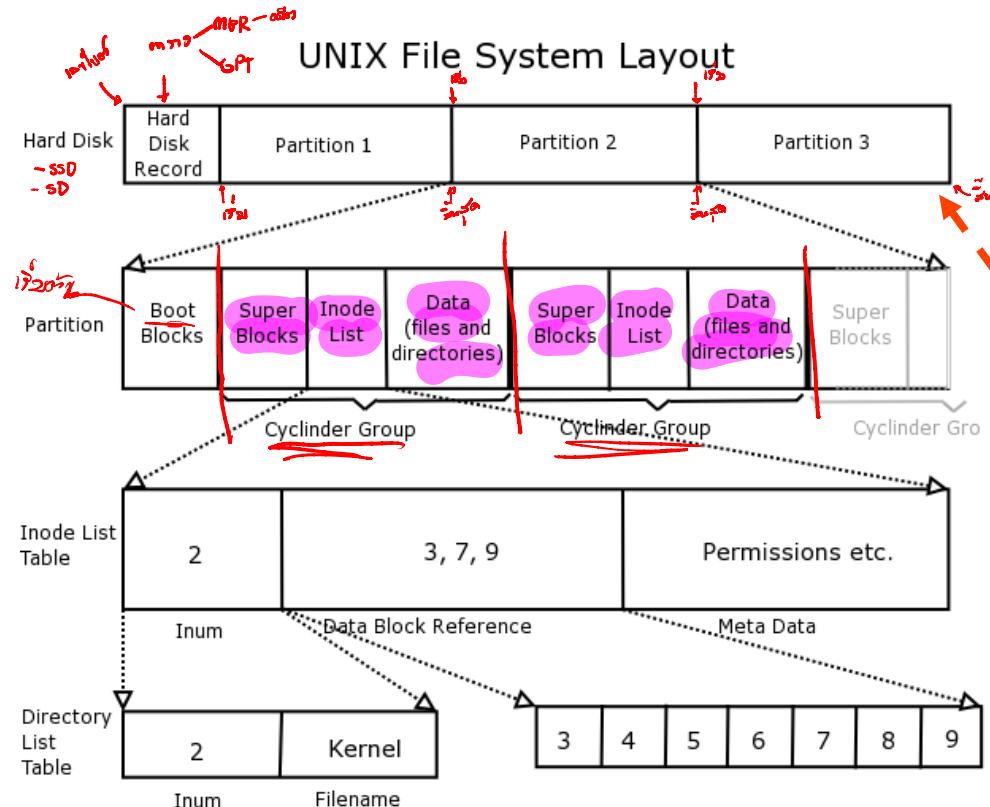
เราจะจินตนาการว่าอุปกรณ์เก็บรักษาข้อมูล เช่น การ์ดหน่วยความจำ SD โซลิดสเตทไดส์กและฮาร์ดดิส์กเหล่านี้ มีลักษณะเป็นแบบข้อมูลที่มีความยาวตามขนาดความจุ และแบ่งແນບออกเป็นพื้นที่ย่อยๆ เรียกว่า พาร์ทิชัน (Partition) ซึ่งอุปกรณ์รักษาข้อมูล 1 ตัวสามารถแบ่งเป็นหลายพาร์ทิชัน (Partition) แต่ละพาร์ทิชันมีระบบบริหารจัดการไฟล์ (File System) ของตนเอง ตามรูปที่ 7.1

การแบ่งพาร์ทิชัน คือ การแบ่งพื้นที่ແນບข้อมูลออกเป็นส่วนต่างๆ ตามที่ระบบไฟล์ออกแบบมา การแบ่งพาร์ทิชันสามารถตอบสนองความต้องการที่หลากหลาย ได้แก่ การบูตระบบปฏิบัติการได้หลากหลาย การจัดเก็บข้อมูลในบางพาร์ทิชัน เป็นต้น ผู้ใช้สามารถติดตั้งระบบปฏิบัติการในแต่ละพาร์ทิชันได้โดยอิสระจากกัน ที่มา [archlinux](#) และ [datadoctor.biz](#) โดยตำแหน่งเริ่มต้นจะเป็นพื้นที่สำหรับเก็บ ตารางพาร์ทิชัน (Partition Table) ประกอบด้วยรายชื่อและข้อมูลประกอบของแต่ละพาร์ทิชัน ตารางพาร์ทิชันที่สำคัญในทางปฏิบัติ ได้แก่

# 7.1 ระบบไฟล์ (File System)

- มาสเตอร์บูตเรคอร์ด (Master Boot Record: MBR) ทำหน้าที่เก็บรายชื่อพาร์ทิชัน ตำแหน่งเริ่มต้น หรือหมายเลข Logical Block Address (LBA) ในฮาร์ดดิสก์ไว้ในตารางพาร์ทิชัน (Partition Table) พื้นที่ส่วนหนึ่งของมาสเตอร์บูตเรคคอร์ดทำหน้าที่เก็บตารางพาร์ทิชัน ซึ่งต้องยูนิเซ็คเตอร์แรก หรือเซ็กเตอร์หมายเลข 0 ของฮาร์ดดิสก์ โดยใช้พื้นที่ 64 ไบต์ ซึ่งฮาร์ดดิสก์ 1 ตัวสามารถแบ่งพาร์ทิชันได้มากที่สุดเป็นจำนวน 4 พาร์ทิชัน รายละเอียดเพิ่มเติมที่ [wikipedia](#)
- ตาราง GPT ในปัจจุบันตารางพาร์ทิชันมีชื่อว่า Globally Unique Identification Partition Table หรือ GUIDPT หรือ GPT ทำหน้าที่คล้ายกับ MBR แต่รองรับจำนวนพาร์ทิชันในฮาร์ดดิสก์ได้มากกว่า และใบօօສหรือเฟิร์มแวร์ของเครื่องคอมพิวเตอร์นั้นๆ ตามมาตรฐาน เรียกว่า Unified Extensible Firmware Interface: UEFI รายละเอียดเพิ่มเติมเกี่ยวกับ GPT ที่ [wikipedia](#) เพื่อให้ตาราง GPT สามารถรองรับระบบปฏิบัติชนิด 64 บิต และฮาร์ดดิสก์ที่มีความจุเพิ่มสูงขึ้นระดับเพتل่า比บต์ (Peta Byte) หรือ  $10^{15}$  ไบต์ หรือประมาณ 1000 เทอร่าไบต์

# 7.1 ระบบไฟล์ (File System)



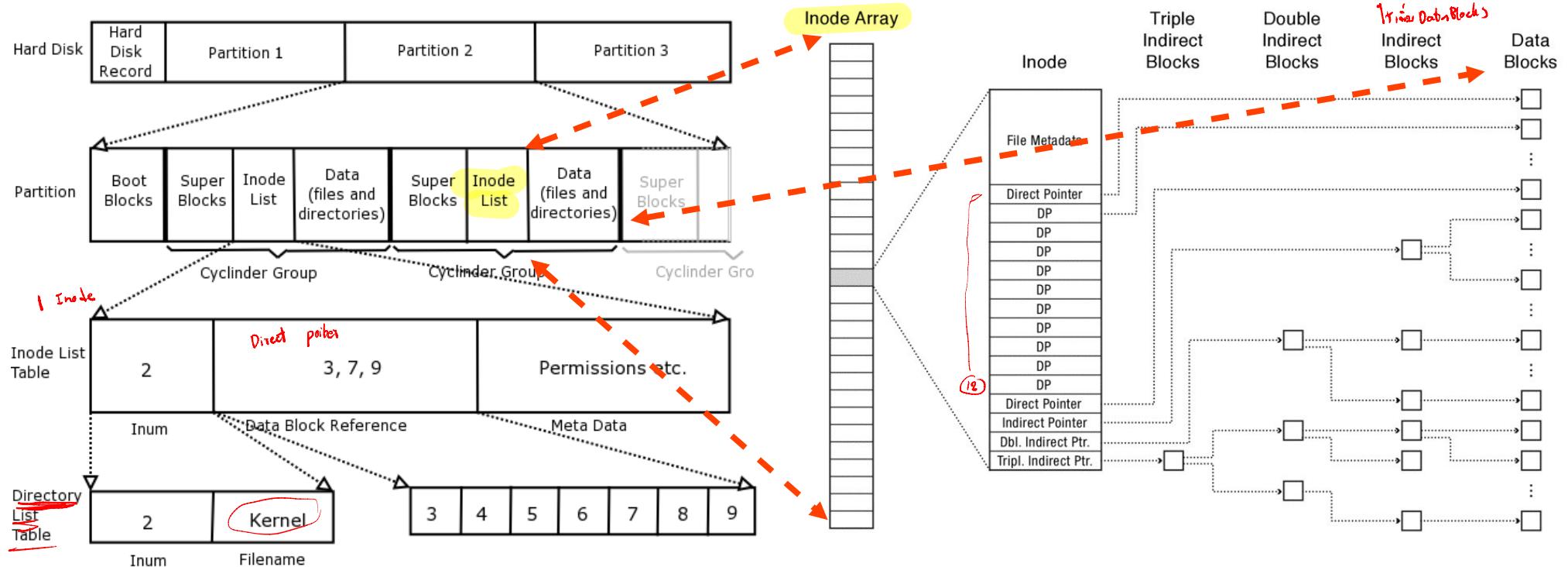
# 7.1 ระบบไฟล์ (File System)

โครงสร้างของระบบไฟล์ในระบบปฏิบัติการตระกูลยูนิกซ์มีโครงสร้าง และหน้าที่ของระบบไฟล์เป็นชั้นการจัดการ 3 ชั้น โดยเรียงลำดับ ดังนี้

- **ชั้นตรรกะ (Logical Layer)** เป็นชั้นบนสุด ทำหน้าที่เชื่อมต่อกับซอฟต์แวร์ประยุกต์ โดยใช้คำสั่งเปิดไฟล์ ปิดไฟล์ อ่านไฟล์ เขียนไฟล์ เป็นต้น ซึ่งไฟล์ที่ซอฟต์แวร์ประยุกต์ต้องการใช้งาน โดยเนื้อหาในบทนี้จะเน้นที่การทำงานในชั้นตรรกะ และชั้นกายภาพ
- **ชั้นเสมือน (Virtual Layer)** ทำหน้าที่เชื่อมต่อระหว่างชั้นตรรกะและชั้นกายภาพ รายละเอียดขึ้นอยู่กับวิธีการพัฒนาโปรแกรมของแต่ละระบบปฏิบัติการ
- **ชั้นกายภาพ (Physical Layer)** เป็นชั้นล่างสุด ทำหน้าที่จัดการบล็อกข้อมูลบนอุปกรณ์เก็บรักษาข้อมูลแต่ละชนิด เพื่อตอบสนองต่อการร้องขอจากชั้นเสมือน ซึ่งกลไกสำคัญคือ การบริหารบัฟเฟอร์ (Buffer) ในหน่วยความจำกายภาพ การเข้าถึงหน่วยความจำกายภาพโดยตรง (DMA) การจัดวางตำแหน่งบล็อกข้อมูลบนอุปกรณ์เหล่านั้น เพื่อประสิทธิภาพการอ่านหรือเขียนต่อเนื่อง และ การเชื่อมต่อกับไดไวซ์ไดเรเวอร์ของอุปกรณ์เก็บรักษาข้อมูล

# 7.1 ระบบไฟล์ (File System)

UNIX File System Layout



# 7.1 ระบบไฟล์ (File System)

- ชูเพอร์บล็อก (Superblock) ทำหน้าที่เก็บรายละเอียดต่างๆ ของระบบไฟล์ ขนาดของบล็อกข้อมูล รายละเอียดการใช้งานบล็อกข้อมูลต่างๆ เช่น สถานะว่างหรือใช้งานอยู่ เป็นต้น
- ตารางไอโหนด (Inode Table) หรืออาจเรียกว่า ไอโหนดอาร์เรย์ (Inode Array) หรือ ไอโหนดลิสต์ (Inode List) ทำหน้าที่เก็บโครงสร้างข้อมูล Inode จำนวนหนึ่งภายใต้ไลนเดอร์กรุปนี้ รายละเอียดเพิ่มเติมในหัวข้อถัดไป
- บล็อกข้อมูลจำนวนหนึ่ง (Data Blocks) ทำหน้าที่บรรจุข้อมูลของไฟล์หนึ่งไฟล์ให้เรียงต่อกันไปจนครบขนาดไฟล์ โดยทั่วไปขนาดของบล็อกข้อมูลแต่ละบล็อกมีความจุเท่ากับ 4096 ไบต์ หรือ 4 KiB (kibibyte) สำหรับระบบปฏิบัติการยูนิกซ์และอื่นๆ ทั้งนี้ขึ้นกับชนิดของเทคโนโลยีและระบบไฟล์ที่ใช้ เมื่อกำหนดขนาดความจุของแต่ละบล็อกแล้ว จำนวนบล็อกข้อมูลจะแบ่งผันตามขนาดความจุของไลนเดอร์กรุปนั้นๆ

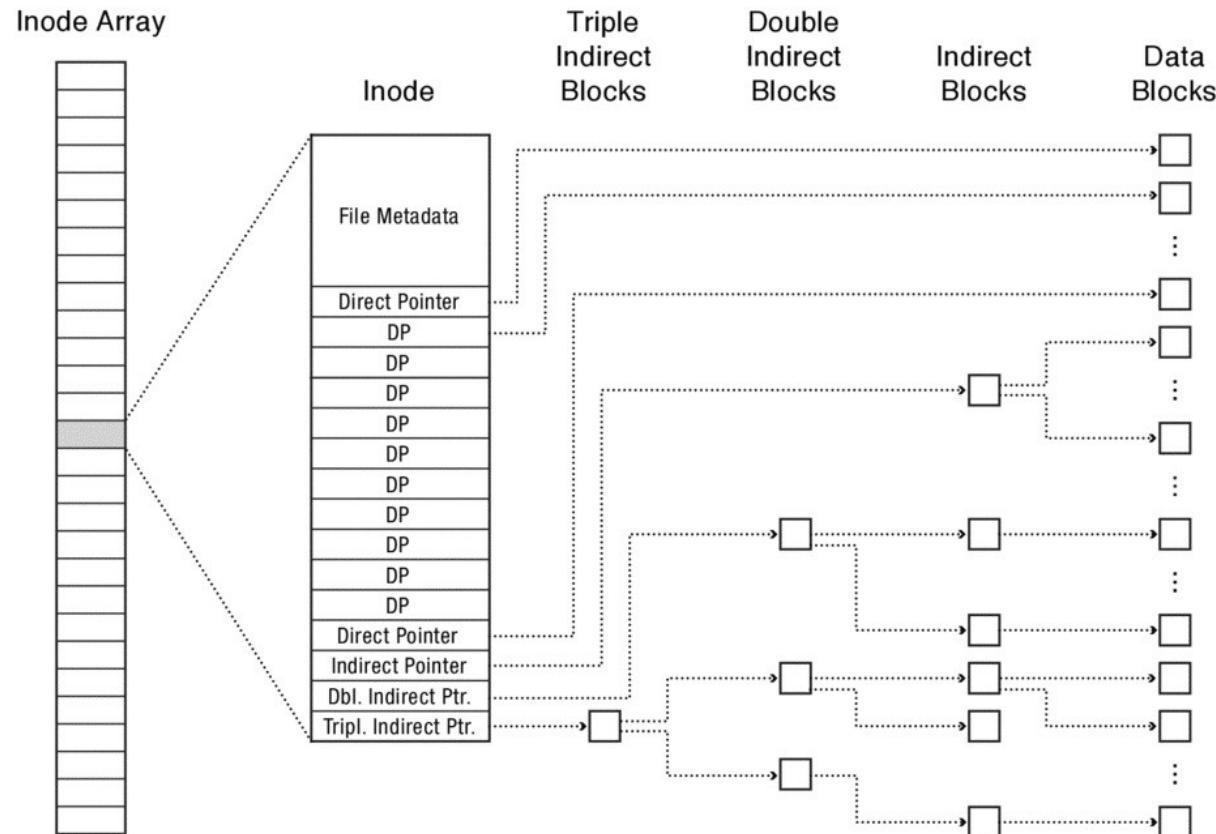
# 7.1 ระบบไฟล์ (File System): Inode

- หมายเลขไอโหนด (Inode Number: Inum): หมายเลขประจำตัวของไฟล์หรือไดเรกทอรีนั้นๆ เป็นเลขจำนวนเต็มชนิดไม่มีเครื่องหมายความกว้าง 32 บิต สำหรับระบบไฟล์ Ext4 โดยด 32 บิต ที่มา: [kernel.org](http://kernel.org)
- รายละเอียดของไฟล์ (File Metadata) แบ่งเป็น
  - ชนิดไฟล์: ไฟล์ (file), ไดเรกทอรี (directory), ไปป์ (pipe) เป็นต้น
    - \* ไฟล์ ตามที่เคยนิยามที่ 3.2.1 ในหัวข้อที่ 3.2.6 ว่าเป็นการเรียงตัวกันของตัวเลขฐานสองที่จะไปต่อๆ โดยอาศัยพื้นที่จัดเก็บในอุปกรณ์รักษาข้อมูล เรียกว่า บล็อกข้อมูล ไฟล์เกิดจาก การเรียงของบล็อกข้อมูลอย่างน้อย 1 บล็อกขึ้นไป เพื่อจัดเก็บข้อมูลหรือคำสั่งต่างๆ ลงใน อุปกรณ์เก็บรักษาข้อมูล
    - \* ไดเรกทอรี หรือ โฟลเดอร์ คือ ไฟล์ชนิดหนึ่งที่เก็บหมายเลขไอโหนดที่อยู่ภายใต้โครงสร้าง นี้โดยไดเรกทอรีสามารถซ่อนกันได้ เพื่อความสะดวกในการจัดหมวดหมู่ของไฟล์และ ไดเรกทอรี
    - \* ไปป์ (pipe) คำราเล่นนี้ไม่ครอบคลุม ผู้อ่านสามารถค้นคว้าเพิ่มเติมได้จาก [Wikipedia](https://en.wikipedia.org/wiki/Named_pipe)

# 7.1 ระบบไฟล์ (File System): Inode

- ลิงก์เชื่อมโยงไปยังบล็อกข้อมูล (Links to Data Blocks) ทำหน้าที่เก็บหมายเลขอุปกรณ์บล็อกข้อมูล (บล็อกสีเหลี่ยมจัตุรัส) ซึ่งทำหน้าที่เก็บข้อมูลจริงๆ การเชื่อมโยงบล็อกข้อมูลหลายๆ บล็อกเข้าด้วยกันเป็นไฟล์ 1 ไฟล์ หรือไดเรกทอรี 1 ไดเรกทอรี ลิงก์เชื่อมโยงแบ่งเป็น 2 ชนิดหลัก คือ
  - ลิงก์หรือพอยน์เตอร์บล็อกข้อมูลทางตรง (Direct Blocks หรือ Direct Pointers: DP) คือ หมายเลขอุปกรณ์บล็อกข้อมูลที่เก็บข้อมูลสำหรับไอโหนดนี้ สำหรับไฟล์ขนาดเล็ก ในรูปมีจำนวน DP เท่ากับ 12 ตำแหน่งๆ 8 ไบต์ ทำให้สามารถเก็บหมายเลขอุปกรณ์ได้สูงสุด 12 หมายเลขอุปกรณ์ รองรับไฟล์ขนาดเล็กที่มีขนาดไม่เกิน 48 KiB (kibibyte) โดยแต่ละบล็อกข้อมูลในระบบไฟล์มีขนาดเท่ากับ 4 KiB (kibibyte)
  - ลิงก์หรือพอยน์เตอร์บล็อกข้อมูลทางอ้อม ใช้สำหรับกรณีที่ไฟล์มีขนาดใหญ่กว่า 48 KiB (kibibyte) จนถึงหลายกิกะไบต์ (GiB) ระบบไฟล์สามารถจดจำจำนวนบล็อกข้อมูลมากขึ้นตามขนาดไฟล์ จนทำให้จำนวนพอยน์เตอร์บล็อกข้อมูลทางตรงไม่เพียงพอ ดังนั้น พอยน์เตอร์บล็อกข้อมูลทางอ้อมจะทำหน้าที่เก็บหมายเลขอุปกรณ์บล็อกข้อมูลจำนวนมากๆ เพื่อรับไฟล์ที่มีขนาดใหญ่ขึ้นดังกล่าว พอยน์เตอร์บล็อกข้อมูลทางอ้อมแบ่งเป็น 3 ระดับ ดังนี้

# 7.1 ระบบไฟล์ (File System): Inode



# 7.1 ระบบไฟล์ (File System): Inode

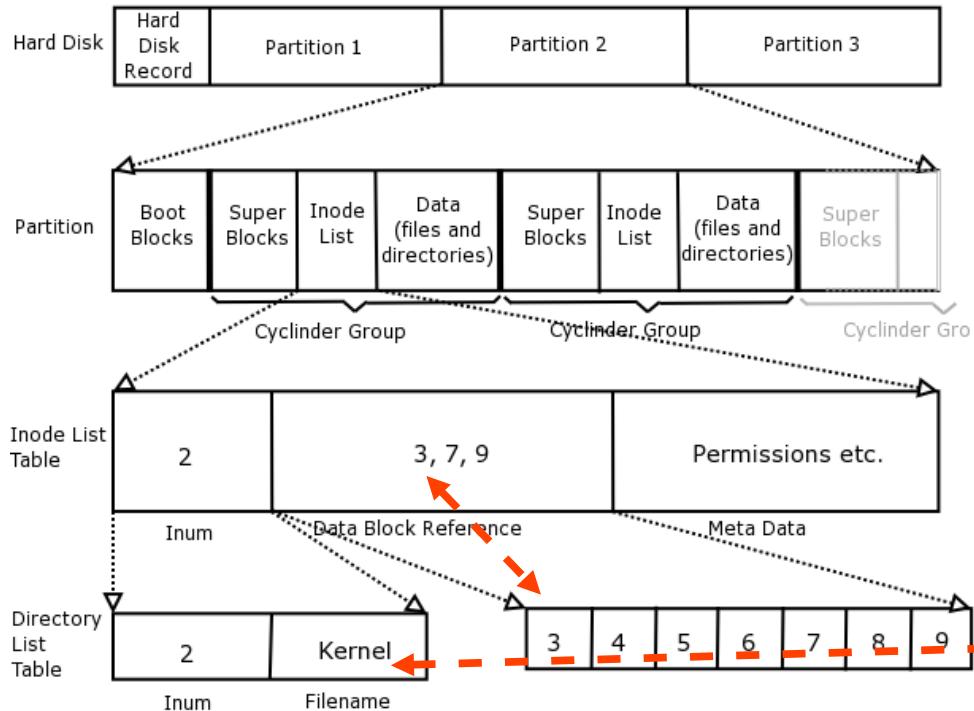
- \* พอยน์เตอร์หนึ่งชั้น (Indirect Pointer) เมื่อไฟล์มีขนาดใหญ่ขึ้นอีกจนเกิน 48 KiB ทำให้จำนวนพอยน์เตอร์ทางตรงไม่เพียงพอ ระบบไฟล์จะอาศัยพอยน์เตอร์หนึ่งชั้นในโครงสร้างไอโอนด เพื่อกับหมายเลขบล็อกพิเศษจำนวนหนึ่ง เรียกว่า บล็อกทางอ้อมหนึ่งชั้น (Indirect Blocks) ทำหน้าที่เก็บหมายเลขบล็อกข้อมูลเสริมจากพอยน์เตอร์บล็อกข้อมูลทางตรงที่เต็มแล้ว โปรดสังเกตบล็อกสีเหลี่ยมจัตุรัสที่ขึ้นระหว่างช่องพอยน์เตอร์หนึ่งชั้นและบล็อกข้อมูลในรูปที่ [7.2](#)
- \* พอยน์เตอร์สองชั้น (Double Indirect Pointer) เมื่อไฟล์มีขนาดใหญ่ขึ้นอีกจนพอยน์เตอร์หนึ่งชั้นไม่เพียงพอ ระบบไฟล์จะอาศัยพอยน์เตอร์สองชั้นในโครงสร้างไอโอนด เพื่อกับหมายเลขบล็อกพิเศษ เรียกว่า บล็อกทางอ้อมสองชั้น (Double Indirect Blocks) ทำหน้าที่เก็บหมายเลขบล็อกทางอ้อมหนึ่งชั้นเสริมจากบล็อกทางอ้อมหนึ่งชั้นที่เต็มแล้ว โปรดสังเกตบล็อกสีเหลี่ยมจัตุรัสที่ขึ้นระหว่างช่องพอยน์เตอร์สองชั้นและบล็อกข้อมูล ในรูปที่ [7.2](#)

# 7.1 ระบบไฟล์ (File System): Inode

- \* พอยน์เตอร์สามชั้น (Triple Indirect Pointer) เมื่อไฟล์มีขนาดใหญ่ขึ้นอีกจนพอยน์เตอร์สองชั้นไม่เพียงพอ ระบบไฟล์จะอาศัยพอยน์เตอร์สามชั้นในโครงสร้างไอโหนด เพื่อกีบหมายเลขบล็อกพิเศษจำนวนหนึ่ง เรียกว่า บล็อกทางอ้อมสามชั้น (Triple Indirect Blocks) ทำหน้าที่เก็บหมายเลขบล็อกทางอ้อมสองชั้นเสริมจากบล็อกทางอ้อมสองชั้นที่เต็มแล้ว โปรดสังเกตบล็อกสี่เหลี่ยมจัตุรัสที่ขึ้นระหว่างช่องพอยน์เตอร์สามชั้นและบล็อกข้อมูล ในรูปที่ [7.2](#)

# 7.1 ระบบไฟล์ (File System): Inode

UNIX File System Layout

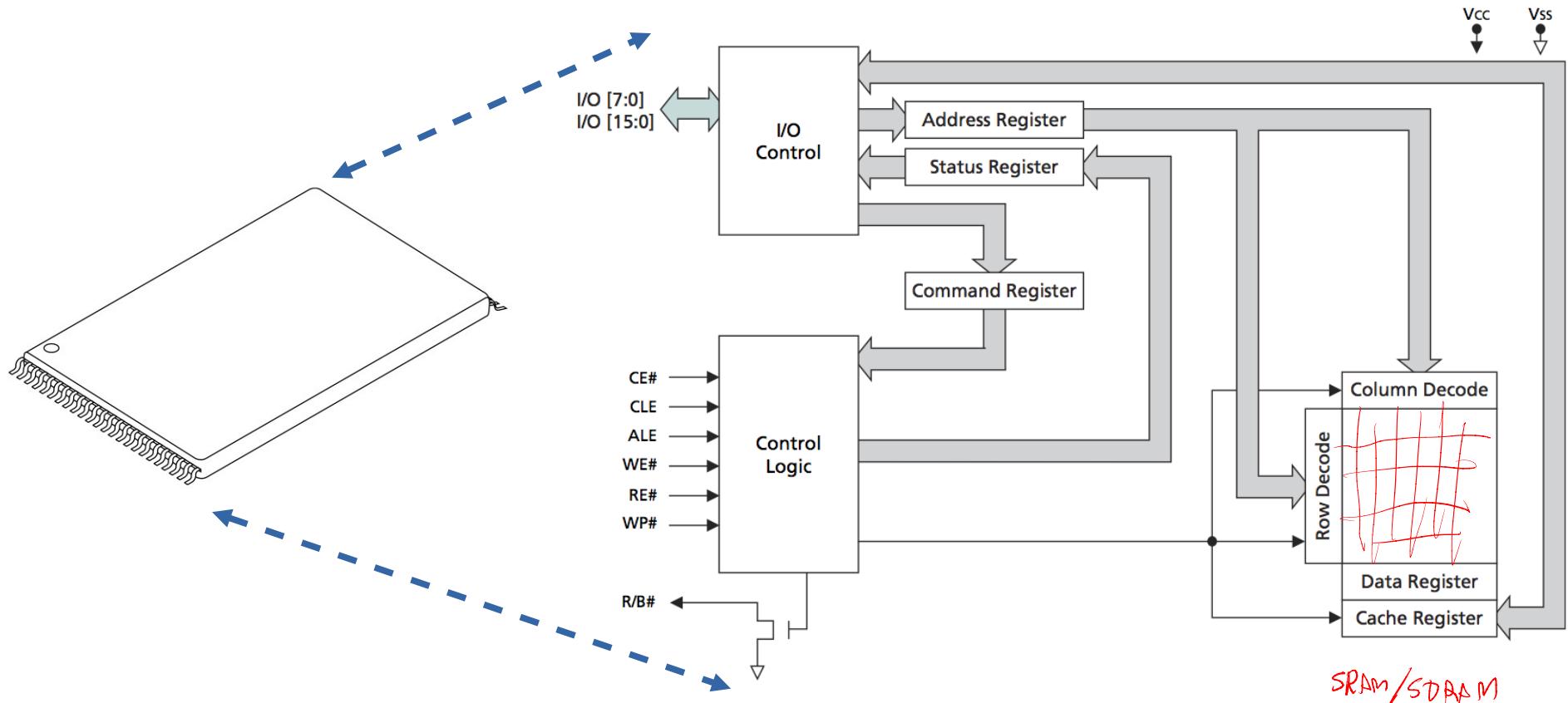


- ไฟล์ชื่อ Kernel ในไดเรคทอรีราก (/ หรือ root directory)
- มี inode หมายเลข (Inum = 2) ซึ่งข้อมูลจะบันทึกอยู่ในบล็อกข้อมูลหมายเลข 3, 7, 9
- มีรายละเอียดสิทธิ์ต่างๆ ตามมา เรียกรวมๆ ว่า MetaData
- ส่วนชื่อไฟล์ (Filename) Kernel จะเก็บบันทึกในตารางไดเรคทอรีอ้างอิงโดยใช้ Inum = 2 เป็นหมายเลขอ้างอิง

## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip)

- หน่วยความจำแฟลช เดิมเรียกว่า แฟลชROM (Flash ROM)
- คำว่า ROM ย่อมาจากคำว่า Read-Only Memory
- วัตถุประสงค์เดิมของแฟลชROM ทำหน้าที่เป็นหน่วยความจำที่สร้างจากเทคโนโลยีสารภึงตัวนำโดยมีคุณสมบัติใช้เก็บคำสั่งและข้อมูลเพื่อการอ่านเป็นหลัก
- นักพัฒนาประยุกต์ใช้เก็บคำสั่งประจำเครื่อง หรือ เฟิร์มแวร์ (Firmware) เป็นหลัก
- ด้วยความนิยมในเทคโนโลยีชนิดนี้ หน่วยความจำแฟลชจึงถูกพัฒนาอย่างต่อเนื่องจนสามารถเขียนหรือแก้ไขข้อมูลภายในแฟลชROMได้รวดเร็วขึ้นจนกลายเป็นอุปกรณ์เก็บรักษาข้อมูล

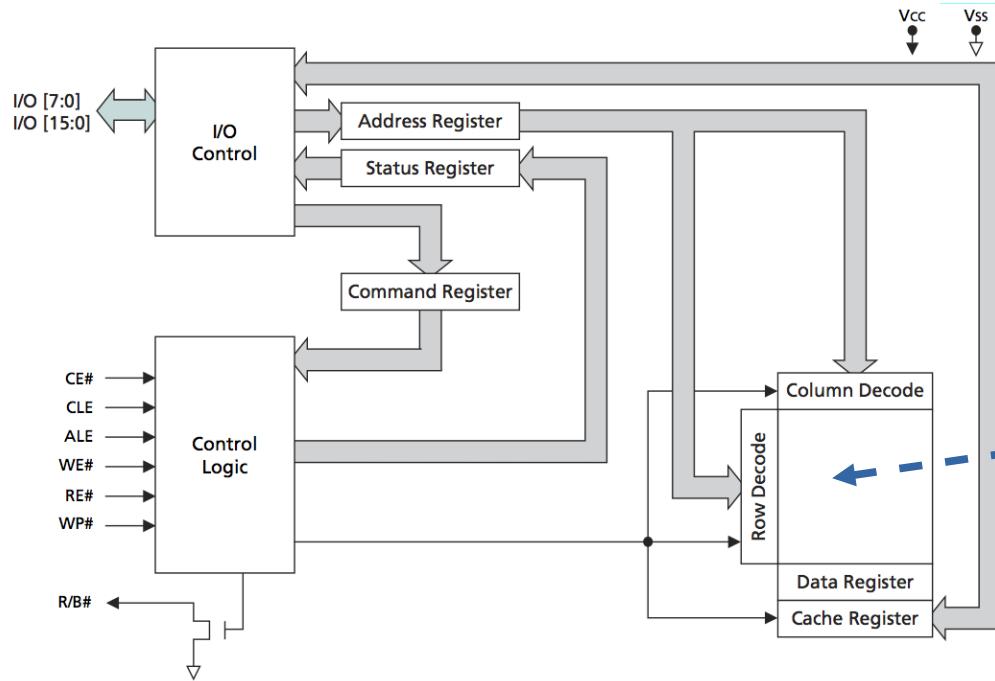
## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip)



## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip)

- ประสิทธิภาพการอ่านข้อมูลขึ้นอยู่กับตำแหน่งและรูปแบบการอ่านและรูปแบบการเรียกดูของข้อมูล เช่น การอ่านข้อมูลที่เรียงตัวต่อเนื่อง (Sequential Address) จะใช้เวลาเข้าถึง (Access Time) สั้นมากเพียง 30 นาโนวินาที การอ่านข้อมูลที่ไม่เรียงตัวต่อเนื่องและสุ่มตำแหน่ง (Random Address) จะใช้เวลาเข้าถึงนานขึ้นเป็น 25 ไมโครวินาที จะเห็นได้ว่าใช้เวลาเพิ่มขึ้นเกือบ 1000 เท่า เทียบกับการอ่านข้อมูลแบบเรียงตัวต่อเนื่อง
- ประสิทธิภาพการเขียนข้อมูลมีลักษณะเช่นเดียวกับการอ่านข้อมูล โดย การเขียนข้อมูลต้องเขียนครั้งละ เพจ (Page Program) หรือ 2048 ไบต์ ซึ่งจะใช้เวลาเข้าถึงยาวนานกว่าการอ่านข้อมูลเป็น 300 ไมโครวินาที ใช้เวลาเพิ่มขึ้นเป็น 10 เท่า เทียบกับการอ่านข้อมูลแบบสุ่ม
- ประสิทธิภาพการลบข้อมูล (Erase) จะใช้เวลาเพิ่มขึ้นสูงมากถึง 2 มิลลิวินาที จะเห็นได้ว่าใช้เวลาเพิ่มขึ้นเกือบ 100 เท่าเทียบกับการเขียนข้อมูลจำนวนหนึ่งเพจ

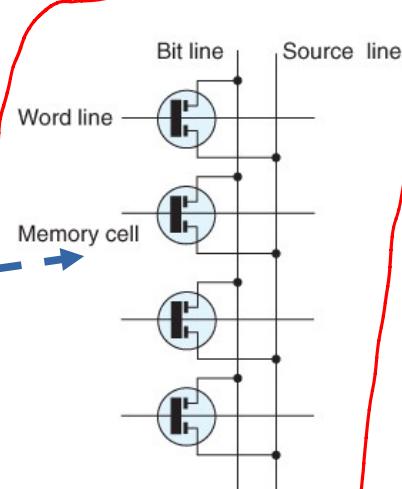
## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip)



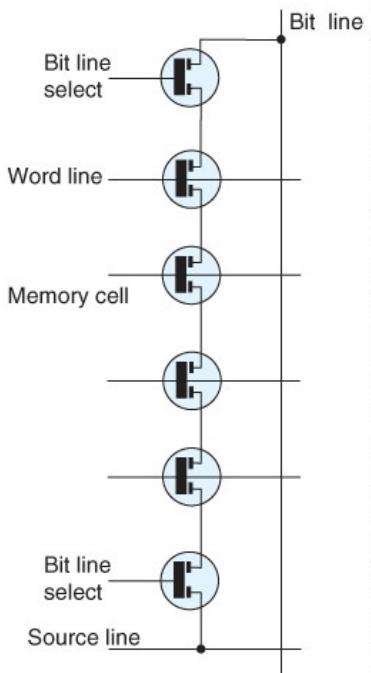
RE 10.33

NOR and NAND flash technologies

NOR flash array  
(parallel architecture)



NAND flash array  
(serial architecture)

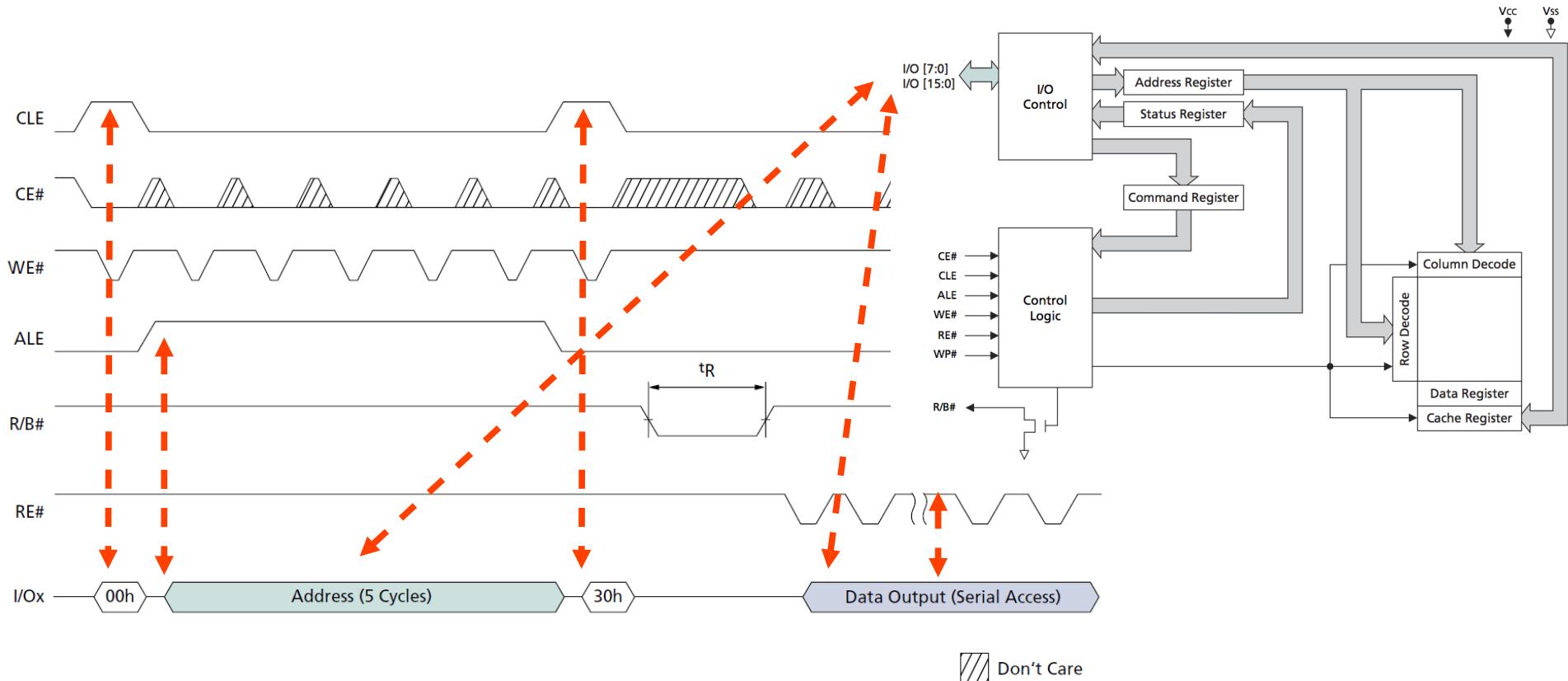


Adapted from M-Systems White Paper, "Two Technologies Compared: NOR vs. NAND," July 03 91-SR-012004-8L.  
Courtesy of SanDisk

## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip)

- โครงสร้างภายในชิปแฟลช NAND มีลักษณะคล้ายกับหน่วยความจำ SRAM และ DRAM คือมี อะเรย์ของเซลหน่วยความจำเป็นหลัก และ
- วงจรควบคุมการอ่านหรือเขียนข้อมูลลงไปในอะเรย์นี้ จะเข้าถึงโดยใช้ แอดเดรสແຕວ (เลขແຕວ) และ แอดเดรஸຄอลัมน์ (เลขຄอลัมน์)
- เพื่อป้องชีต์ตำแหน่งเซลนั้นๆ ในอะเรย์ แต่การทำงานของหน่วยความจำแฟลชนี้มีความซับซ้อนกว่า

## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip): Read



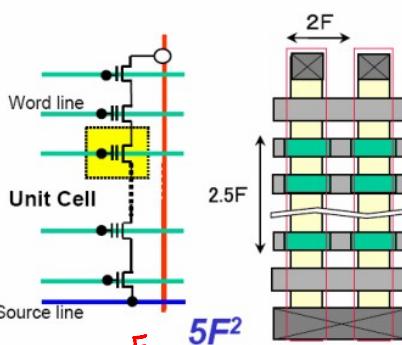
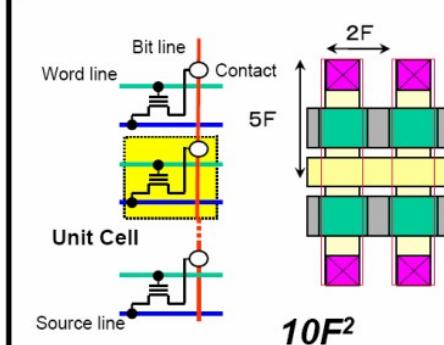
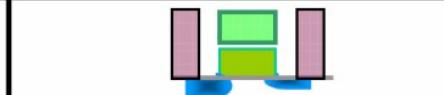
## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip): Read

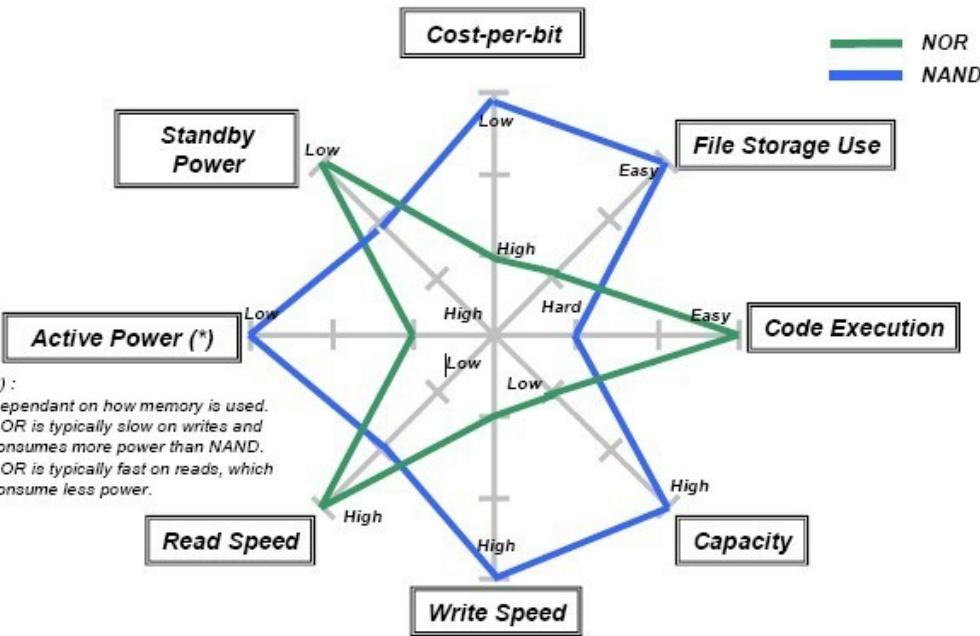
- สัญญาณ CLE เปลี่ยนจาก 0 เป็น 1 เพื่ออ่านค่าคำสั่งที่ปรากฏบนบัส I/O
- สัญญาณ CE# เปลี่ยนจาก 1 เป็น 0 เพื่อส่งชิปให้เริ่มต้นทำงาน
- สัญญาณ WE# จะมีการเปลี่ยนแปลงจาก 1 เป็น 0 สลับไปมา เพื่อเขียนคำสั่งและoadเดรสองขา I/Ox ไปเก็บพักในรีจิสเตอร์ต่างๆ
- สัญญาณ ALE จะเปลี่ยนแปลงจาก 0 เป็น 1 ในระหว่างที่บัส I/O รับoadเดรเป็นระยะเวลา 5 คากา (Cycles)
- สัญญาณ R/B# จะเปลี่ยนแปลงจาก 1 เป็น 0 ระยะเวลาหนึ่ง  $t_R$  เพื่อบ่งบอกว่าชิป มีสถานะยุ่ง (Busy) แล้วกลับไปเป็น 1 เพื่อบ่งบอกว่าชิปพร้อม

## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip): Read

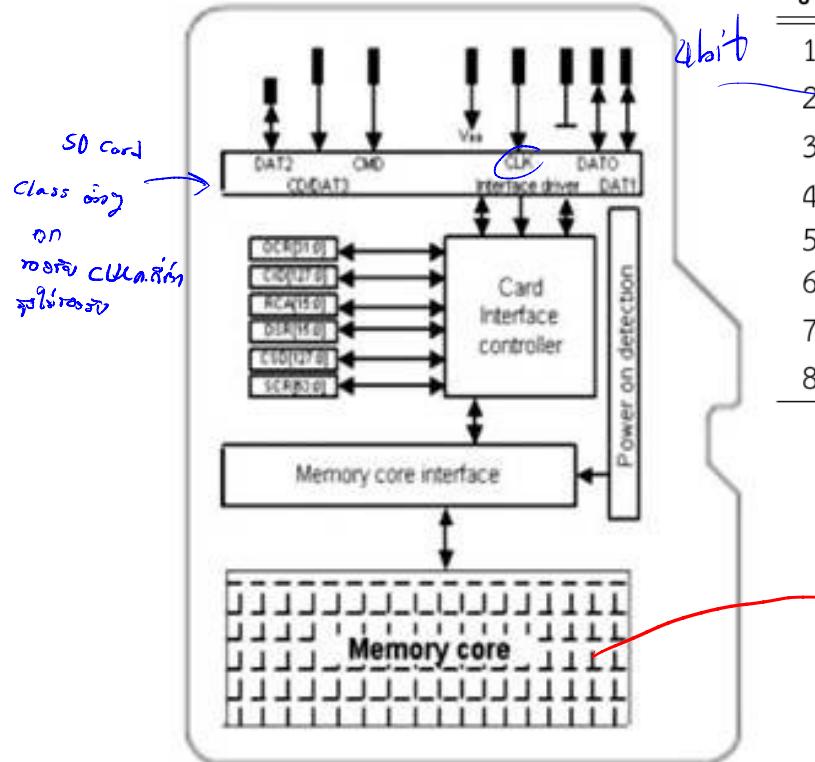
- สัญญาณ RE# จะเปลี่ยนแปลงจาก 1 เป็น 0 สลับไปมา เมื่อตรวจจับว่าสัญญาณ R/B# เปลี่ยนจาก 0 เป็น 1 เพื่ออ่านข้อมูลทางขา I/Ox โดยใช้ขอบขาขึ้นหรือขอบขาลงของสัญญาณ RE#
- สัญญาณ I/O[0:15] ทั้งหมด 16 ขา จะเปลี่ยนแปลงตามลำดับดังนี้
  - } คำสั่ง 00h จำนวน 1 คิบเวลา - แอดдресต์ จำนวน 5 คิบเวลา - คำสั่ง 30h จำนวน 1 คิบเวลา
- ข้อมูล จำนวนหลายคิบเวลาตามขนาดของเพจข้อมูลโดยอ่านข้อมูลเรียงตามลำดับหมายเลขแอดเดรส (Serial Access)

## 7.2 ชิปหน่วยความจำแฟลช (Flash Memory Chip)

	NAND	NOR
Cell Array & Size	 <p>Unit Cell Word line Source line <math>5F^2</math></p>	 <p>Unit Cell Word line Bit line Contact Source line <math>10F^2</math></p>
Cross-section	 <p>1 Transistor</p>	 <p>2 Transistors</p>
Features	<p>Small Cell Size, High Density Low Power &amp; Good Endurance → Mass Storage</p>	<p>Large Cell Current, Fast Random Access → Code Storage</p>



# 7.3 การ์ดหน่วยความจำ SD (Secure Digital)



ขา	ชื่อ	วัตถุประสงค์
1	DAT2	Data บิทที่ 2
2	DAT3/CD	Data บิทที่ 3 หรือ Card Detect
3	CMD	ขารับคำสั่ง (Command)
4	VDD	ขับวนแผลงจ่ายไฟ
5	CLK	สัญญาณคลื่อก ให้กับ CPU
6	VSS	ขาก拉ว์ด์แผลงจ่ายไฟ
7	DAT0	Data บิทที่ 0
8	DAT1	Data บิทที่ 1

Flash Memory  
NAND

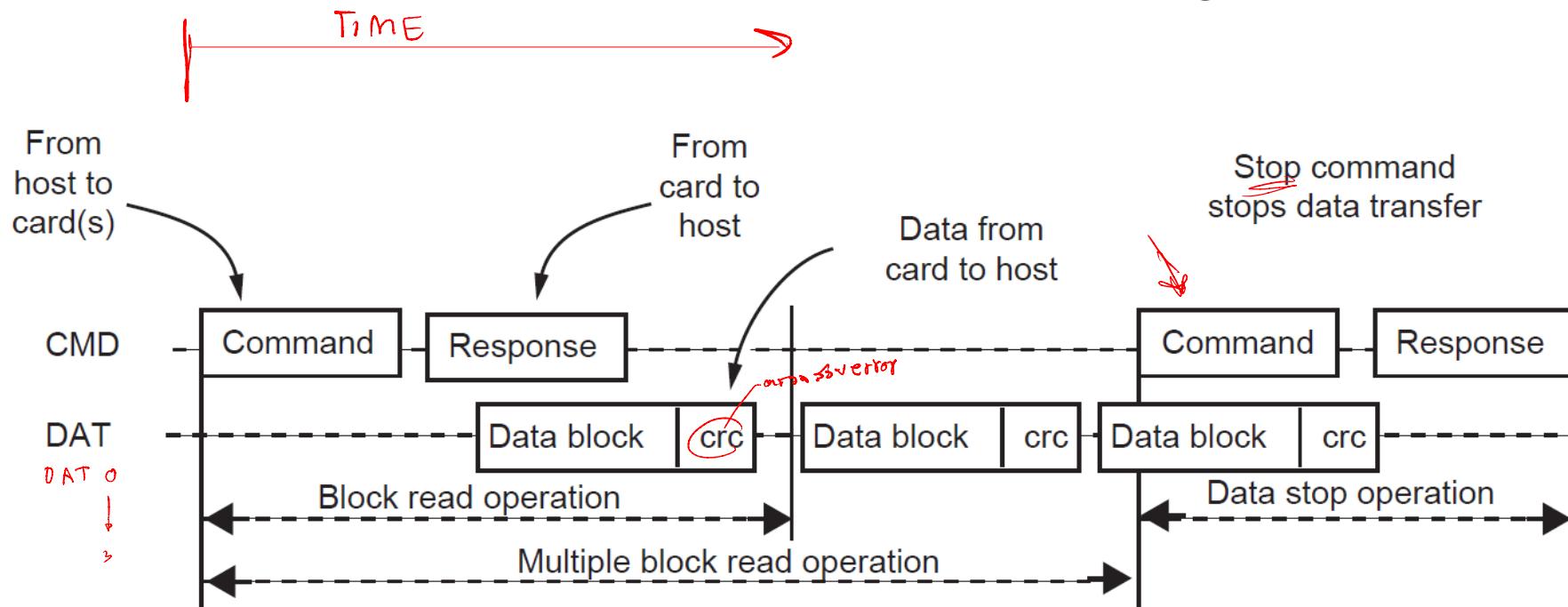
2 เท่านั้น  
SDIO 1 บิต  
4 จังหวะ

- CMD17: Read Single Block
- CMD18: Read Multiple Block
- CMD24: Write Single Block
- CMD25: Write Multiple Block
- CMD32: Erase Block Start  $\times 2048$   
block no Flash ROM
- CMD33: Erase Block End
- CMD38: Erase

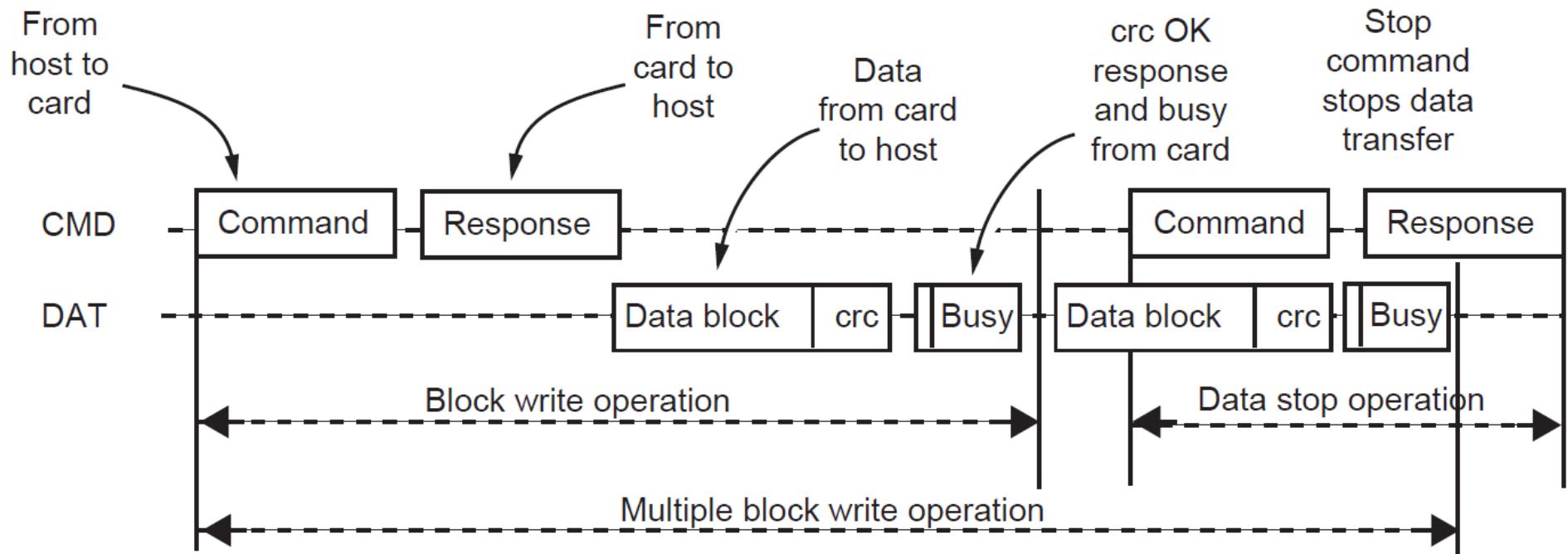
## 7.3 การ์ดหน่วยความจำ SD (Secure Digital)

- อะเรย์เซลหน่วยความจำชนิดแฟลช ตามขนาดของความจุที่ต้องการใช้
- Memory Core Interface เพื่อเชื่อมต่อกับหน่วยความจำ
- วงจรควบคุม Card Interface Controller เพื่อเชื่อมต่อกับบอร์ด ทำงานร่วมกับรีจิสเตอร์ต่างๆ เหล่านี้
  - รีจิสเตอร์ CID[27:0] (Card Identification) ขนาด 28 บิต เพื่อกีบหมายเลขประจำตัวการ์ด
  - รีจิสเตอร์ OCR[31:0] (Operation Condition Register) ขนาด 32 บิต เพื่อกีบสถานะการทำงานของการ์ด
  - รีจิสเตอร์ RCA[15:0] (Relative Card Address) ขนาด 32 บิต เพื่อกีบหมายเลขแอดเดรสของ การ์ด ซึ่งจะเปลี่ยนแปลงระหว่างที่ถอดเข้าออกจากระบบ
  - รีจิสเตอร์ CSD[27:0] (Card Specific Data) ขนาด 28 บิต เพื่อกีบสถานะจำเพาะของการ์ด
  - รีจิสเตอร์ SCR[63:0] (SD Configuration Register) ขนาด 64 บิต เพื่อกีบการตั้งค่าพิเศษประจำตัวการ์ด

## 7.3 การ์ดหน่วยความจำ SD (Secure Digital): Read



## 7.3 การ์ดหน่วยความจำ SD (Secure Digital): Write



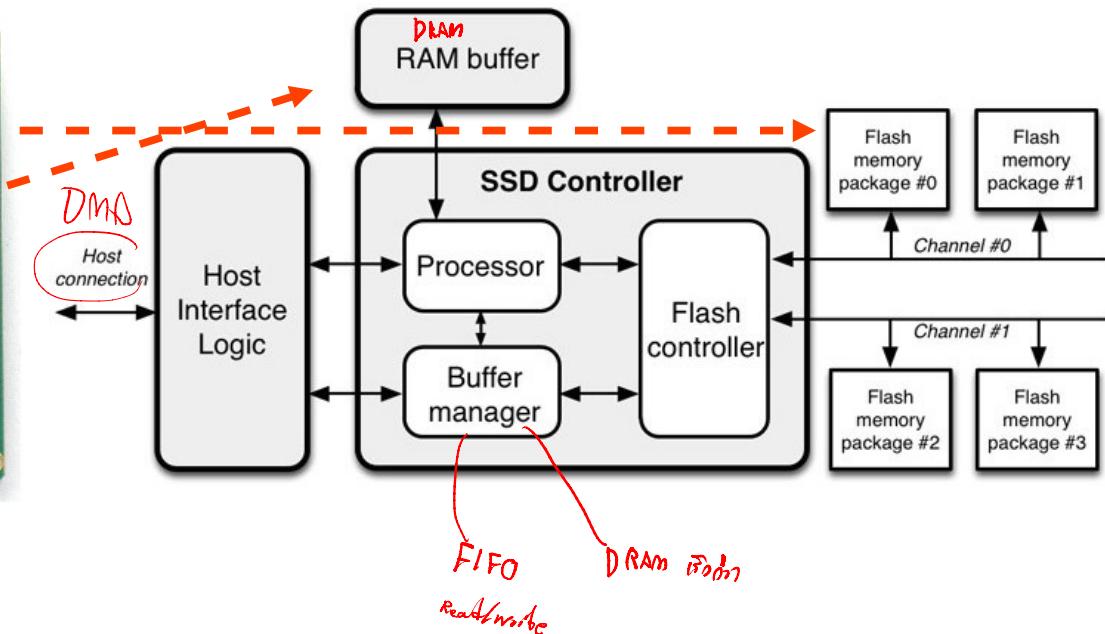
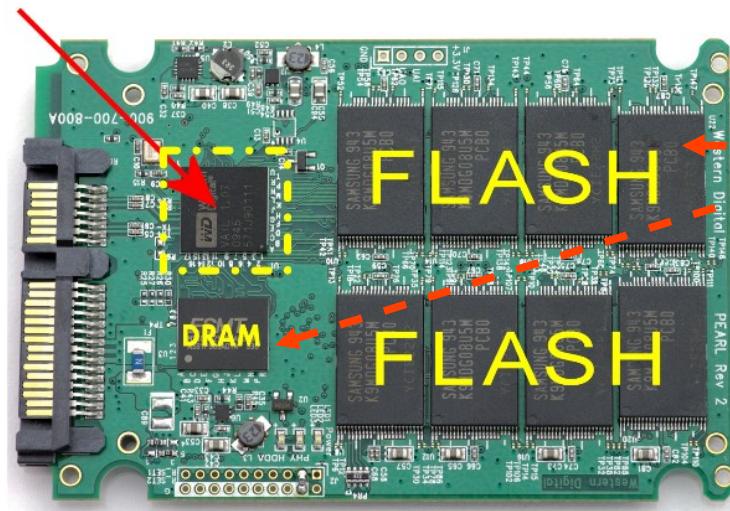
## 7.4 โซลิดสเตทไดรฟ์ (Solid-State Drive: SSD)

→ Flash

- ความจุของ SSD มีแนวโน้มเพิ่มสูงขึ้น ทำให้ SSD มีขนาดเริ่มต้นตั้งแต่ 120-128 กิกะไบต์ขึ้นไป
- แนวโน้มของ SSD ตันทุน/ความจุจะถูกลงเรื่อยๆ จนใกล้เคียงฮาร์ดดิสก์ในอนาคต โดยองค์ประกอบหลักที่สำคัญ คือ หน่วยความจำแฟลช NAND ที่มีความจุต่อชิปเพิ่มสูงขึ้นไปอีก
- ใช้หน่วยความจำ DRAM เพื่อทำหน้าที่เป็นแคช หรือ บัฟเฟอร์ เพื่อเพิ่มประสิทธิภาพการทำงานให้รวดเร็วขึ้น
- SSD ใช้ชิล์ดใกล้เคียงกับฮาร์ดดิสก์ไดร์ฟ เพื่อความหมายที่ใกล้เคียงกัน
- ชิปหน่วยความจำแฟลช NAND ความจุสูงเรียบตัวต่อเนื่องกันจนได้ความจุมากพอ
- การจัดเรียงทั้งด้านบน (Channel 0) และด้านล่าง (Channel 1) ของแผ่นวงจรพิมพ์หลักของหน่วยความจำแฟลช และ อาศัยข้อพหุนัยความจำไดนามิกแรมทำหน้าที่เป็นแคช

## 7.4 ໂໜ້າດສຕේທໄຣພື້ (Solid-State Drive: SSD)

SSD Controller



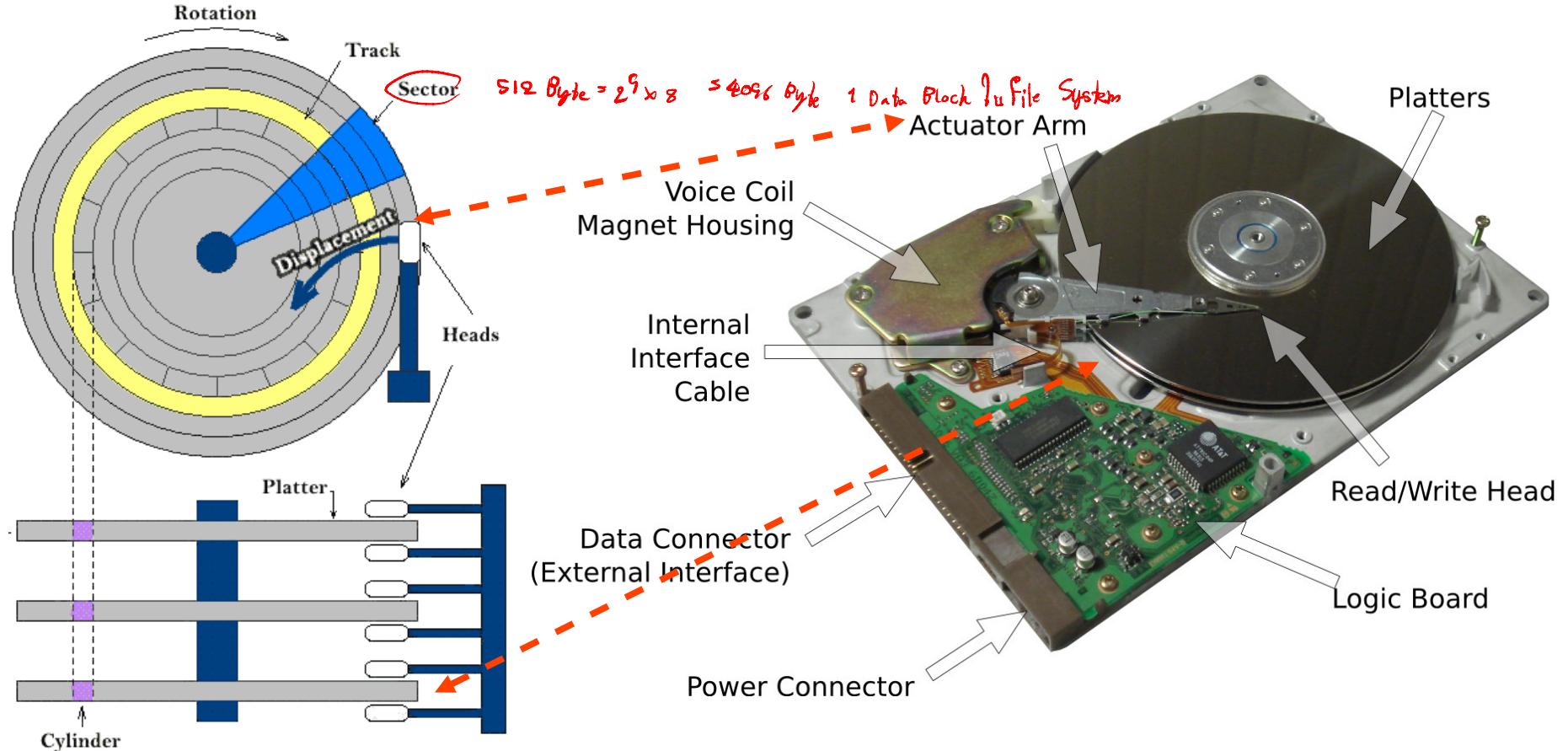
## 7.4 โซลิดสเตทไดรฟ์ (Solid-State Drive: SSD)

- โปรเซสเซอร์ (Processor) สำหรับรันคำสั่งในเฟิร์มแวร์สำหรับควบคุมการทำงานภายในและเชื่อมต่อ กับคอมพิวเตอร์ไฮสต์ ดังนั้น ผู้อ่านควรตรวจสอบผู้ผลิตเป็นระยะๆ ว่ามีการอัปเดตเฟิร์มแวร์ของ SSD รุ่นที่ใช้หรือไม่
- แฟลชคอนโทรลเลอร์ (Flash Controller) สำหรับควบคุมการอ่าน/เขียนข้อมูลหน่วยความจำแฟลช คล้ายกับการทำงานของหน่วยความจำ SD ซึ่งกล่าวในหัวข้อที่ 7.2
- บัฟเฟอร์ (Buffer) และ การบริหารจัดการบัฟเฟอร์ (Buffer Management) อาศัยหน่วยความจำ DRAM เป็นบัฟเฟอร์เพื่อพักเก็บข้อมูลชั่วคราวระหว่างที่เชื่อมต่อกับคอมพิวเตอร์ไฮสต์ ทำให้อ่าน/เขียนมีระยะเวลาเข้าถึงเฉลี่ย ดังนั้น การใช้หน่วยความจำ DRAM ให้เต็มประสิทธิภาพ และคุ้มค่า จึงต้อง มีการบริหารจัดการบัฟเฟอร์ ทำให้ต้นทุนการผลิตต่ำลง

## 7.5 ฮาร์ดดิสก์ไดรฟ์ (Hard Disk Drive: HDD)

- อุปกรณ์เก็บรักษาข้อมูลที่ได้รับความนิยมจากในอดีต และพัฒนามาเป็นเวลากว่าวนาน  
*Disco*  
*drive*
- แผ่นจานแม่เหล็กหมุน มีเส้นผ่าศูนย์กลางสองขนาดที่นิยมผลิต คือ 2.5 นิ้ว และ 3.5 นิ้ว  
หมุนด้วยความเร็วสูงตั้งแต่ 5,400 ถึง 10,000 รอบต่อนาที  
*ความเร็ว → ความไว*
- จุดเด่นของหน่วยความจำชนิดสารแม่เหล็ก คือ ความจุข้อมูลที่มากกว่า เริ่มต้นที่หลายร้อย กิกะไบต์จนถึงหลายเทอร่าไบต์ (Tera Byte) โดย 1 เทอร่าไบต์ ประมาณเท่ากับ 1000 กิกะไบต์
- ราคาย่อมเยา ต้นทุนโดยรวมจึงถูกลง มีอายุการใช้งานที่ยาวนานพอสมควร  
*cost/bit < 5\$*  
*ชีวิตก่อตัวต้องไม่ต้องซื้อใหม่*

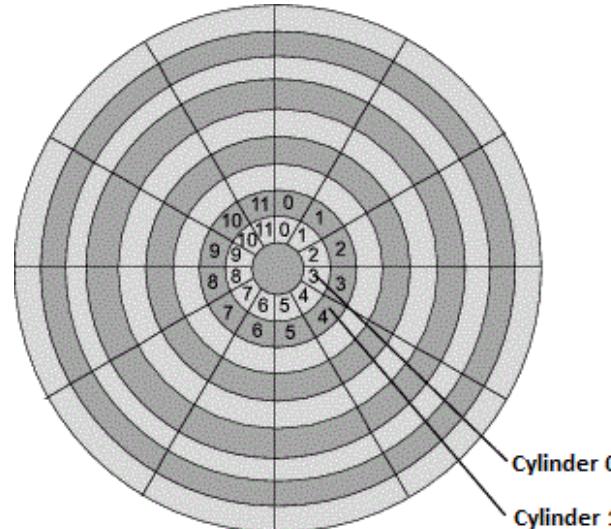
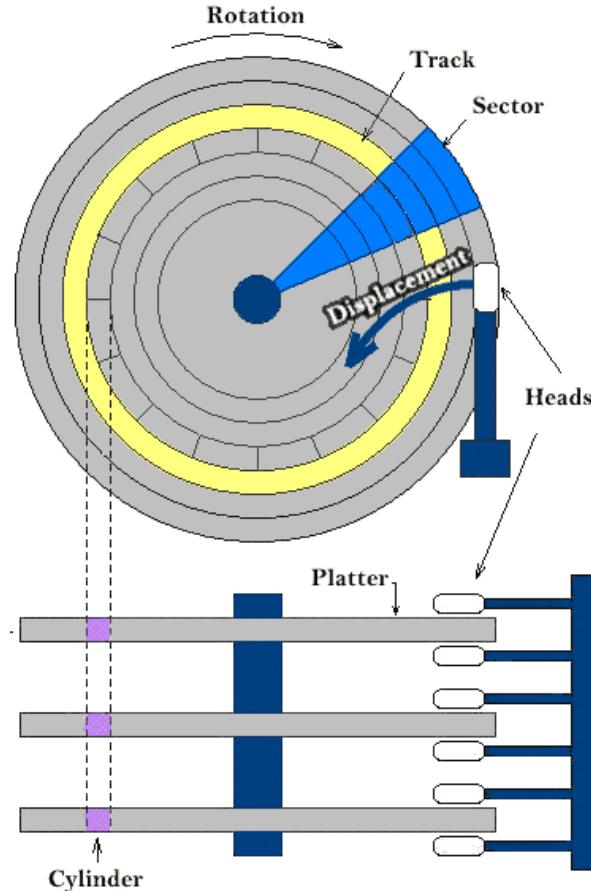
## 7.5 ฮาร์ดดิสก์ไดรฟ์ (Hard Disk Drive: HDD)



# 7.5 ฮาร์ดดิสก์ไดร์ฟ (Hard Disk Drive: HDD)

- แผ่นจานแม่เหล็กแบ่งเป็นหลายๆ แทร็ก หรือ วงรอบ แต่ละแทร็กจะมีจำนวนเซ็กเตอร์เท่าๆ กัน
- พื้นที่หนึ่งเซ็กเตอร์ มีความจุ 512 ไบต์เสมอ โดยจะแบ่งพื้นที่ในแนวรัศมีจากจุดศูนย์กลางmanyขอบจาน
- ไซลินเดอร์ (Cylinder) คือ แทร็กที่อยู่บนจานแม่เหล็กต่างๆ และอยู่ห่างจากจุดศูนย์กลางเท่ากัน เรียกว่ากันเป็นทรงกระบอก โดยจะนับจากไซลินเดอร์หรือแทร็กหมายเลข 0 ซึ่งอยู่ขอบนอกสุดของจานแม่เหล็ก โดยจะนับจากขอบนอกสุดเข้าสู่จุดศูนย์กลาง
- บล็อก หรือ คลัสเตอร์ประกอบด้วย เซ็กเตอร์ที่ต่อเนื่องกัน จำนวน  $2^n$  เซ็กเตอร์เสมอ เช่น 2 4 8 .. เซ็กเตอร์ ในแทร็กเดียวกัน โดยระบบปฏิบัติการจะกำหนดจำนวนเซ็กเตอร์ที่ต้องการ ยกตัวอย่างเช่น คลัสเตอร์ขนาด 4096 ไบต์ต้องการพื้นที่จำนวน 8 เซ็กเตอร์
- หมายเลขบล็อก LBA: Logical Block Addressing คือ การตั้งหมายเลขบล็อกที่กล่าวมาก่อนหน้านี้ให้เรียงตัวตามหมายเลขไซลินเดอร์ หมายเลขหัวอ่าน และหมายเลขแทร็ก ทำให้ง่ายต่อการบริหารจัดการ และเป็นพื้นฐานของระบบบริหารจัดการไฟล์

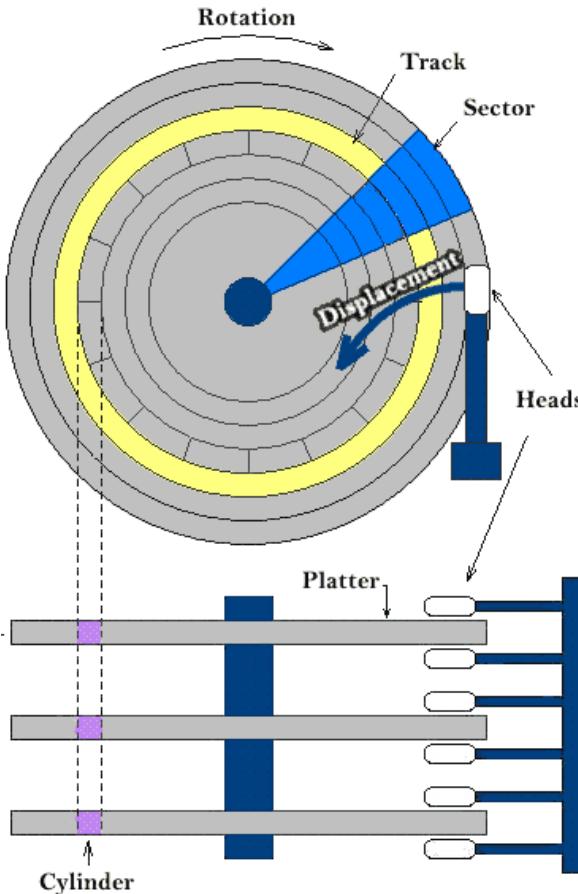
# 7.5 ฮาร์ดดิสก์ไดรฟ์ (Hard Disk Drive: HDD)



**CHS Addressing**  
Cylinder Head Sector

**LBA Addressing**  
Linear Block Address  
Logical

## 7.5 ฮาร์ดดิสก์ไดรฟ์ (Hard Disk Drive: HDD)



การวัดประสิทธิภาพของฮาร์ดดิสก์ อาศัยการวัดเวลาการเข้าถึง ( $T_{acc}$ , Access Time/Latency) หน่วยเป็น มิลลิวินาที ประกอบด้วย ช่วงเวลาการรอเฉลี่ยให้ตำแหน่งที่ต้องการอ่านหมุนมาอยู่หัวอ่าน เรียกว่า Rotation Latency  $T_{rotate}$  ช่วงเวลาการขยับหัวอ่านมาอยู่แทร็คที่ต้องการ  $T_{head}$  และช่วงเวลาการถ่ายโอนข้อมูล ( $T_{trans}$ , Transfer Time)

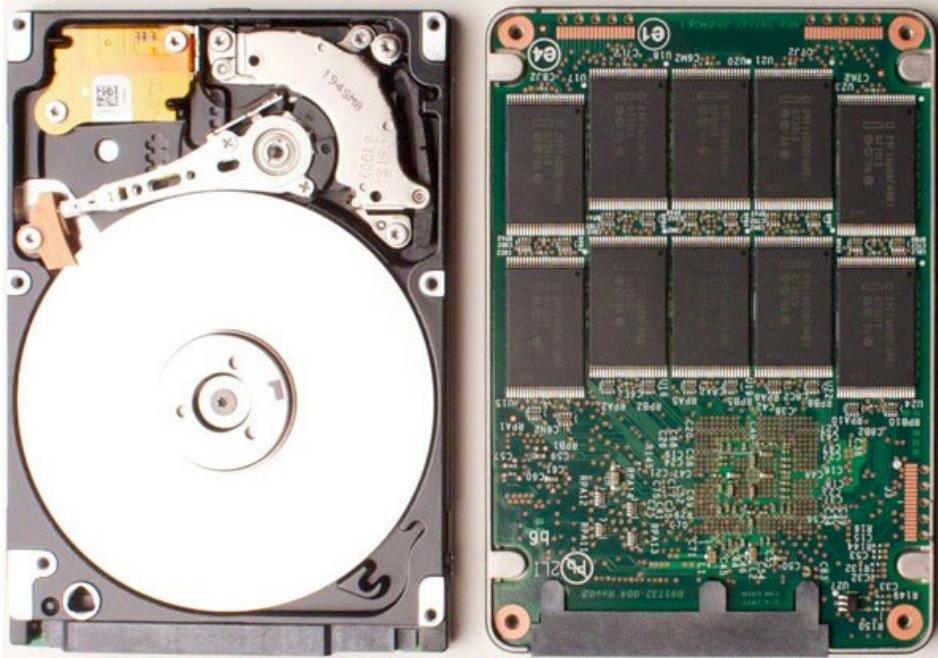
$$T_{acc} = T_{rotate} + T_{head} + T_{transf} \quad (7.1)$$

ดังนั้น เวลาการเข้าถึงจะมีความสัมพันธ์กับความเร็วรอบในการหมุนของจานแม่เหล็ก

$$T_{rotate} = \frac{0.5}{V_{rotate}} \quad (7.2)$$

โดยแปรผันกับ  $V_{rotate}$  หรือ ความเร็วรอบในการหมุนของจาน หน่วยเป็นรอบต่อวินาที (Round per Minute: RPM) และตำแหน่งของแทร็คที่ต้องขยับหัวอ่านไป ประสิทธิภาพการอ่านและการเขียน จะขึ้นกับตำแหน่งของข้อมูล เช่นเดียวกับการอ่านและเขียนข้อมูลของหน่วยความจำแฟลช ประสิทธิภาพการ

## 7.5 ฮาร์ดดิสก์ไดร์ฟ (Hard Disk Drive: HDD)



Usually 10 000 or 15 000 rpm SAS drives

**0.1 ms**

**Access times**

SSDs exhibit virtually no access time

**5.5 ~ 8.0 ms**

SSDs deliver at least  
**6000 io/s**

**Random I/O Performance**

SSDs are at least 15 times faster than HDDs

HDDs reach up to  
**400 io/s**

SSDs have a failure  
rate of less than  
**0.5 %**

**Reliability**

This makes SSDs 4 - 10 times more reliable

HDD's failure rate  
fluctuates between  
**2 ~ 5 %**

SSDs consume between  
**2 & 5 watts**

**Energy savings**

This means that on a large server like ours,  
approximately 100 watts are saved

HDDs consume between  
**6 & 15 watts**

SSDs have an average  
I/O wait of  
**1 %**

**CPU Power**

You will have an extra 6%  
of CPU power for other operations

HDDs' average I/O wait  
is about  
**7 %**

the average service time for  
an I/O request while running  
a backup remains below

**20 ms**

**Input/Output  
request times**

SSDs allow for much  
faster data access

the I/O request time with  
HDDs during backup rises up  
to

**400~500 ms**

SSD backups take about  
**6 hours**

**Backup Rates**

SSDs allows for 3 - 5 times faster  
backups for your data

HDD backups take up to  
**20~24 hours**

ຕະຫຼາດ  
volatile mem  
SRAM/DRAM

ມານີ້ກຳ  
non volatile mem  
ເກົ່າດັກຕະຫຼາດນີ້ກຳໄດ້  
ສະບຸປໍທ້າຍບທ

(file) System - UPS  
Storage - Direct Attach  
storage (DAS)  
Network Attach Storage (NAS)  
Storage Area Network (SAN)

ตารางที่ 7.3: การเปรียบเทียบประสิทธิภาพของอุปกรณ์เก็บรักษาข้อมูลชนิดต่างๆ

อุปกรณ์	แฟลช NAND	SSD	HDD
ผู้ผลิต หมายเลขโมเดล ที่มา: ปี ค.ศ. ความจุ	Micron MT29F2G08AABWP micron.com 2004 25 MiB	Micron MTFDDAK120MAV micron.com 2013 120 KiB	Western Digital 5K1000 hgst.com 2016 1000 กิกะไบต์ (GB)
<u>การอ่าน: เวลาเข้าถึง</u> - $T_{acc,avg}$ - $T_{acc,max}$	30 นาโนวินาที <u>25 ไมโครวินาที</u>	160 ไมโครวินาที 5 มิลลิวินาที	5.5 มิลลิวินาที -
<u>การเขียน: เวลาเข้าถึง</u> - $T_{acc,avg}$ - $T_{acc,max}$	300 ไมโครวินาที <u>2 มิลลิวินาที</u>	40 ไมโครวินาที 25 มิลลิวินาที	5.5 มิลลิวินาที -
ໄວລເຕຈສູງສຸດ ກຳລັງໄຟສູງສຸດ	4.6 ໂວລຕໍ 23 ມິລລິວັດຕໍ	5.0 ໂວລຕໍ 150 ມິລລິວັດຕໍ	5.0 ໂວລຕໍ 1.6 ວັດຕໍ

# สรุปท้ายบท

- ระบบไฟล์และอุปกรณ์เก็บรักษาข้อมูลจะต้องประสานงานกัน เพื่อให้เครื่องเนล ผู้ใช้งานและแอพพลิเคชันอื่นๆ สามารถบริหารจัดการไฟล์โปรแกรม ไฟล์ข้อมูลต่างๆ ได้อย่างถูกต้อง และมีประสิทธิภาพสอดคล้องกับการกิจของระบบคอมพิวเตอร์
- ขนาดหรือความจุของบล็อกข้อมูลในอุปกรณ์เก็บรักษาข้อมูล เช่น หน่วยความจำแฟลชรีอัม จะมีขนาดเท่ากับ  $2^{11}$  หรือ 2048 ไบท์ ไฮร์ดดิสก์ จะมีขนาดเท่ากับ  $2^9$  หรือ 512 ไบท์ สอดคล้องกับ ขนาดของบล็อกข้อมูลในระบบบริหารจัดการไฟล์ และ
- ขนาดของเพจข้อมูลในหน่วยความจำสมเมื่อน ซึ่งจะมีขนาดเป็นจำนวนเท่าของสองเสมอ และสำหรับลีนุกซ์มีขนาดเท่ากับ 4096 หรือ  $2^{12}$  ไบท์
- อุปกรณ์เก็บรักษาข้อมูลจะเชื่อมต่อกับหน่วยความจำผ่านวงจรด้านอินพุท/เอาท์พุท โดยใช้กลไก Direct Memory Access และ กลไกการทำ Interrupt ในชั้น Physical ร่วมกับกลไก Memory Mapped File ในระดับสูงขึ้น

# References

- Stewart Weiss, Chapter 3 File Systems and the File Hierarchy, UNIX Lecture Notes,  
[http://www.compsci.hunter.cuny.edu/~sweiss/course\\_materials/unix\\_lecture\\_notes/chapter\\_03.pdf](http://www.compsci.hunter.cuny.edu/~sweiss/course_materials/unix_lecture_notes/chapter_03.pdf)
- <https://www.acerspace.com/ssd-solid-state-drive/>
- [https://gabrieletolomei.wordpress.com/misCELLANEA/operating-systems/in-memory-layout/](https://gabrieletolomei.wordpress.com/miscellanea/operating-systems/in-memory-layout/)
- <https://freedompenguin.com/articles/how-to/learning-the-linux-file-system>
- <https://www.techpowerup.com/174709/arm-launches-cortex-a50-series-the-worlds-most-energy-efficient-64-bit-processors>
- Thomas Anderson and Michael Dahlin, Operating Systems: Principles and Practice 2nd Edition, Recursive Books, 2014
- <https://learn.adafruit.com/resizing-raspberry-pi-boot-partition/edit-partitions>

# References

- Micron Technology, I. (2004). Nand flash memory:  
<Mt29f2g08aabwp/mt29f2g16aabwp/mt29f4g08babwp/mt29f4g16babwp/mt29f8g08fabwp>.
- <https://www.raspberrypi.org/forums/viewtopic.php?t=63750>
- [https://www.researchgate.net/figure/Block-Diagram-of-Micro-SD-card\\_fig6\\_306236972](https://www.researchgate.net/figure/Block-Diagram-of-Micro-SD-card_fig6_306236972)
- <https://gabrieletolomei.wordpress.com/miscellanea/operating-systems/in-memory-layout/>
- <https://freedompenguin.com/articles/how-to/learning-the-linux-file-system>
- Harris, D. and S. Harris (2013). Digital Design and Computer Architecture (1st ed.). USA: Morgan Kauffman Publishing.
- <https://learn.adafruit.com/resizing-raspberry-pi-boot-partition/edit-partitions>

# References

- [https://www.researchgate.net/figure/Block-Diagram-of-Micro-SD-card\\_fig6\\_306236972](https://www.researchgate.net/figure/Block-Diagram-of-Micro-SD-card_fig6_306236972)
- <https://gabrieletolomei.wordpress.com/miscellanea/operating-systems/in-memory-layout/>
- <https://freedompenguin.com/articles/how-to/learning-the-linux-file-system>
- Harris, D. and S. Harris (2013). Digital Design and Computer Architecture (1st ed.). USA: Morgan Kauffman Publishing.
- <https://learn.adafruit.com/resizing-raspberry-pi-boot-partition/edit-partitions>