ภาคผนวก L

การทดลองที่ 12 การศึกษาอุปกรณ์เก็บรักษาข้อมูลและระบบไฟล์

การทดลองนี้อธิบายและเชื่อมโยงเนื้อหาความรู้ของทุกบทเข้าด้วยกัน แต่จะเน้นบทที่ 6 และบทที่ 7 เพื่อให้ผู้อ่านมองเห็นอุปกรณ์อินพุตและเอาต์พุตเหมือนไฟล์แต่ละไฟล์ โดยมีวัตถุประสงค์ดังนี้

- เพื่อให้เข้าใจการวัดขนาดของไฟล์และไดเรกทอรีในระบบไฟล์
- เพื่อให้รู้จักโครงสร้างและระบบไฟล์ของการ์ดหน่วยความจำไมโคร SD ที่ใช้งานในปัจจุบัน
- เพื่อให้เข้าใจระบบไฟล์ (File System) ชนิดต่างๆ บนบอร์ด Pi
- เพื่อให้สามารถเชื่อมโยงอุปกรณ์อินพุต/เอาต์พุตชนิดต่างๆ กับระบบไฟล์

L.1 ขนาดของไฟล์และไดเรกทอรี

ผู้อ่านสามารถ**ตรวจสอบ**ขนาดของไฟล์ใดๆ ชื่อ filename ที่แท้จริง หน่วยเป็นไบต์ ด้วยคำสั่ง **du** (DIsk Usage) โดยทำตามขั้นตอนต่อไปนี้

- ย้ายไดเรกทอรีปัจจุบันไปที่ **/home/pi** ซึ่งเป็นไดเรกทอรีหลักของผู้ใช้ชื่อ pi

```
$ cd /home/pi
```

- สร้างไฟล์ข้อความ test.txt ด้วยโปรแกรม nano ด้วยคำสั่งต่อไปนี้

```
$ nano test.txt
```

พิมพ์ข้อความ fdd ลงในไฟล์ ทำการ Write โดยกดปุ่ม Ctrl แช่ตามด้วยปุ่ม o ออกจากโปรแกรมโดยกดปุ่ม Ctrl แช่ตามด้วยปุ่ม x

- คำสั่ง ‘du -b filename’ จะแสดงผลขนาดเป็นจำนวนไบต์นำหน้าชื่อไฟล์นั้น

```
$ du –b test.txt
_ test.txt
```

ตัวเลข _ หมายถึง เลขจำนวนไบต์ที่คำสั่ง du แสดงผลมาตามพารามิเตอร์ b ที่ส่งไป เพื่อบอกค่าขนาดของไฟล์ test.txt เป็นจำนวน _ ไบต์

- คำสั่ง ‘du -B1 filename’ ผู้อ่านสามารถ**ตรวจสอบ**ขนาดของไฟล์ใดๆ ชื่อ filenameที่จัดเก็บเป็นจำนวนเท่าของ ____ ไบต์ ในอุปกรณ์เก็บรักษาข้อมูล SD ด้วยคำสั่งต่อไปนี้

```
$ du –B1 test.txt
____ test.txt
```

ตัวเลข ____ หมายถึง เลขจำนวนไบต์ที่คำสั่ง du แสดงผลมาตามพารามิเตอร์ B1 ที่ส่งไป โดยผู้อ่านจะสังเกตเห็นความแตกต่าง ถึงแม้ไฟล์มีข้อมูลจำนวนน้อยเพียงไม่กี่ไบต์ แต่การจองพื้นที่ในอุปกรณ์สำรองจะมีขนาดเป็นจำนวนเท่าของ ____ ไบต์เสมอ เช่น 8192, 16384 เป็นต้น

- คำสั่ง ‘du -h’ จะแสดงผลขนาดหรือจำนวนไบต์โดยใช้หน่วยเช่น K (Kibi: 1024) M (Mebi: 1048576) G (Gibi: 1073741824) นำหน้าชื่อไดเรกทอรีหรือโฟลเดอร์ที่อยู่ใต้ไดเรกทอรีปัจจุบัน และจดบันทึก 5 รายการแรกในตาราง

```
$ du –h
```

Size	Folder Name

L.2 ระบบไฟล์

ผู้ใช้หรือผู้ดูแลระบบลินุกซ์ สามารถ**ตรวจสอบ**การใช้งานอุปกรณ์เก็บรักษาข้อมูล เช่น ฮาร์ดดิสก์ไดรฟ์ โซลิดสเตทไดรฟ์ การ์ดหน่วยความจำ SD ได้โดยคำสั่ง

- คำสั่ง **df** (Disk File System) สามารถแสดงรายละเอียดของอุปกรณ์เก็บรักษาข้อมูลในเครื่อง
- คำสั่ง 'df -h' จะแสดงรายการ ดังต่อไปนี้ จดบันทึก 5 รายการแรกลงในตารางเพื่อเปรียบเทียบกับตารางที่แล้ว

```
$ df –h
```

Filesystem	Size	Used	Available	Use%	Mounted on

โดย Size จะแสดงผลขนาดหรือจำนวนไบต์โดยใช้ตัวคูณที่แตกต่างกัน เช่น K (Kibi: 1024) M (Mebi: 1048576) G (Gibi: 1073741824)

- คำสั่ง 'df -T' จะเพิ่มคอลัมน์ชนิด (Type) ของแต่ละรายการในการแสดงผล และขนาดเป็นจำนวนเท่าของ 1 KiB (KibiByte) (1K) แทน จดบันทึก 5 รายการที่ตรงกับตารางที่แล้ว

```
$ df –T
```

Filesystem	Type	1K-blocks Used	Available	Use%	Mounted on

- คำสั่ง ‘df -i‘ จะแสดงรายการต่างๆ ดังนี้ จดบันทึก 5 รายการที่ตรงกับตารางที่แล้ว

```
$ df -i
```

Filesystem	Inodes	IUsed	IFree	IUse%	Mounted on

โดยคอลัมน์ที่ 2 จากทางซ้ายจะแสดงผลเป็นจำนวน **ไอโหนด** แทน รายละเอียดเรื่องไอโหนด ผู้อ่านสามารถค้นคว้าเพิ่มเติมได้ในบทที่ 7 และทาง wikipedia

- คำสั่ง **stat** แสดงรายละเอียดของไฟล์หรือไดเรกทอรี การทดลองนี้จะใช้ไดเรกทอรี asm ที่มีอยู่ และเติมตัวเลขในช่องว่าง

```
$ cd /home/pi
$ stat asm

  File: asm
  Size: _____       Blocks: _          IO Block: _____   _____________
Device: ______h/______d Inode: _______     Links: 3
Access: (_____/drwxr-xr-x)  Uid: ( _____/____)   Gid: ( _____/____)
Access: ...
Modify: ...
Change: ...
 Birth: -
```

ผู้อ่านจะต้องกรอกผลลัพธ์ในช่องว่าง ดังต่อไปนี้

- ชื่อ asm
- ขนาด _____ ไบต์ ใช้พื้นที่จำนวน _ Blocks ซึ่งหมายถึง 8 **เซ็กเตอร์**ๆ ละ 512 ไบต์ เป็น _____
- มีหมายเลข Device = _____h/_____d หรือเท่ากับ $\_\_\_\_\_{}_{16}/\_\_\_\_\_{}_{10}$
- มีหมายเลข Inode = $\_\_\_\_\_{}_{10}$ จำนวน 3 Links
- สิทธิ์เข้าถึง (Access) ด้วยรหัส $\_\_\_\_{}_{16}$ หรือ $\_\_\_{}_{2}$:$\_\_\_{}_{2}$:$\_\_\_{}_{2}$ โดยผู้ใช้หมายเลข Uid (User ID)=_____ ชื่อผู้ใช้ (Username)=____ ในกรุปหมายเลข Groupid=_____ ชื่อกรุป ____

- เข้าถึง (Access) ...
- เปลี่ยนแปลง (Modify) ...
- เวลาที่ Inode เปลี่ยนแปลง (Change) ...

เบื้องต้นผู้เขียนขอให้ผู้อ่านสร้างไฟล์ผลลัพธ์จากคำสั่ง stat ไปเก็บในไฟล์ เพื่อมาใช้ประกอบการทดลองต่อไป โดย

```
$ stat asm > stat_asm.txt
```

หลังจากนั้น เราสามารถ**ตรวจสอบ**สถานะของไฟล์ stat_asm.txt ได้ดังนี้

```
$ stat stat_asm.txt

  File: stat_asm.txt
  Size: ____        Blocks: _          IO Block: 4096   ____________
Device: ____h/______d Inode: _______      Links: 1
Access: (____/-rw-r--r--)  Uid: ( ____/___)   Gid: ( ____/__)
Access: ...
Modify: ...
Change: ...
 Birth: -
```

ผู้อ่านจะต้องกรอกผลลัพธ์ในช่องว่าง ดังต่อไปนี้

- ชื่อ stat_asm.txt
- ขนาด ___ ไบต์ ใช้พื้นที่จำนวน _ Blocks ซึ่งหมายถึง 8 **เซ็กเตอร์**ๆ ละ 512 ไบต์ เป็น _____
- มีหมายเลข Device = ___h/_____d หรือเท่ากับ $\_\_\_\_{16}/\_\_\_\_\_{10}$
- มีหมายเลข Inode = $\_\_\_\_\_{10}$ จำนวน 1 Links
- สิทธิ์เข้าถึง (Access Permission) ด้วยรหัส $\_\_\_\_{16}$ หรือ $\_\_\_{2}:\_\_\_{2}:\_\_\_{2}$ โดยผู้ใช้หมายเลข Uid (User ID)=____ ชื่อผู้ใช้ (Username)=__ ในกรุปหมายเลข Groupid=____ ชื่อกรุป __
- เข้าถึง (Access) ...
- เปลี่ยนแปลง (Modify) ...
- เวลาที่ Inode เปลี่ยนแปลง (Change) ...
- หมายเลข Inode ของ asm กับ หมายเลข Inode ของ stat_asm.txt ตรงกันหรือไม่ เพราะเหตุใด

- asm เป็น **ไดเรกทอรี** ในขณะที่ stat_asm.txt เป็น _____

- สิทธิ์เข้าถึง (Access Permission) รหัส 0765_{16} มีความหมายดังต่อไปนี้
  - 111_2: เป็นของใคร ____
  - 110_2: เป็นของใคร ____
  - 101_2: เป็นของใคร ____

L.3 อุปกรณ์อินพุต/เอาต์พุตในระบบไฟล์

การทดลองในหัวข้อนี้จะเชื่อมต่อกับเนื้อหาในบทที่ 3 และ การทดลองที่ 4 ภาคผนวก D หลักการของระบบปฏิบัติการยูนิกซ์ คือ การ**เมาท์** (Mount) อุปกรณ์กับไดเรกทอรีด้วยระบบไฟล์ (File System) ที่แตกต่างกัน โดยใช้ชื่อไดเรกทอรีที่แตกต่างกัน โดยมีไดเรกทอรีรูท (Root Directory) หรือโฟลเดอร์รูท เป็นตำแหน่งเริ่มต้น ผู้อ่านสามารถพิมพ์คำสั่งใน Terminal

```
$ mount
```

คำสั่งนี้จะแสดงรายชื่อการเมาท์ หรือ ผูกยึด อุปกรณ์อินพุต/เอาต์พุต เข้ากับระบบไฟล์ที่เหมาะกับอุปกรณ์นั้นๆ ด้วยชื่อไดเรกทอรีหรือชื่อไฟล์ของระบบปฏิบัติการ ผู้อ่านจะต้องกรอกผลลัพธ์ที่สำคัญในช่องว่าง และศึกษาคำอธิบายต่อไปนี้

- **/dev/mmcblk0p__** on / type ext4 (rw,noatime) เป็นระบบไฟล์ **ext4** ซึ่งเป็นระบบไฟล์หลักของลินุกซ์ ย่อมาจากคำว่า Fourth Extended File System เป็นเวอร์ชันที่ 4 พัฒนาจากชนิด ext3 ซึ่งพัฒนาจากระบบยูนิกซ์ตามรายละเอียดในหัวข้อที่ 7.1 และ wikipedia
- devtmpfs on /dev type devtmpfs (rw, relatime, size=3834564k, nr_inodes=958641, mode=755)
- proc on /proc type proc (rw, relatime) เป็นระบบไฟล์เสมือน (Virtual File System) สำหรับระบบสำคัญต่างๆ เช่น CPU, โดยจะสร้างขึ้นเมื่อบูตเครื่อง และลบทิ้งเมื่อชัตดาวน์ระบบ รายละเอียดเพิ่มเติมที่ wikipedia
- sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime) เป็นระบบไฟล์เสมือน (Virtual File System) รายละเอียดเพิ่มเติมที่ wikipedia
- securityfs on /sys/kernel/security type securityfs (rw, nosuid, nodev, noexec, relatime)
- tmpfs on /dev/shm type tmpfs (rw, nosuid, nodev) ย่อมาจากคำว่า Temporary File System รายละเอียดเพิ่มเติมที่ wikipedia
- devpts on /dev/pts type devpts (rw, nosuid, noexec, relatime, gid=5, mode=620, ptmxmode=000) เป็นระบบไฟล์เสมือน (Virtual File System) สำหรับอุปกรณ์อินพุตเอาต์พุตต่างๆ รายละเอียดเพิ่มเติมที่ wikipedia
- ...

- **/dev/mmcblk0p__** on /boot type vfat ระบบไฟล์ **vfat** เป็นส่วนต่อขยายของระบบไฟล์ FAT ซึ่งย่อมาจากคำว่า File Allocation Table เพื่อรองรับชื่อไฟล์ที่ยาวกว่า FAT ที่มา: wikipedia
- ...

รายชื่อต่อไปนี้ คือ ตัวเลือกคุณสมบัติ (Atttribute) ที่สำคัญของระบบไฟล์ เช่น

- rw : read/write สามารถอ่านและเขียนได้
- noatime และ atime: No/ Access Time หมายถึง ไม่มี/มีการบันทึกเวลาในการสร้าง อ่านหรือเขียนไฟล์ทุกครั้ง
- relatime หมายถึง มีการบันทึกเวลาในการสร้าง อ่านหรือเขียนไฟล์ เมื่อเกิดการแก้ไขไฟล์ หรือ การอ่านหรือเข้าถึงไฟล์มากกว่าเวลาที่บันทึกไว้ก่อนหน้าอย่างน้อย 24 ชั่วโมง
- nosuid: No SuperUser ID เป็นการป้องกันไม่ให้ผู้ดูแลระบบ (SuperUser) กระทำการใดๆ ได้ เพื่อความมั่นคงปลอดภัย
- noexec: No Execution เพื่อตั้งค่าไม่ให้รันไฟล์ที่อยู่ในไดเรกทอรีนี้ได้ เช่น ไฟล์ที่เป็นไวรัสหรือมัลแวร์ (Malware) ที่แอบแฝงเข้ามา
- nodev: No Device หมายถึง การไม่อนุญาตให้สร้างหรืออ่านโหนด (Node) ซึ่งเป็นไฟล์ชนิดพิเศษ
- mode หมายถึง สิทธิ์การเข้าถึงไฟล์หรือไดเรกทอรี ประกอบด้วย บิตควบคุม Read Write Execute 3 ชุด รวมทั้งหมด 9 บิต ซึ่งได้อธิบายแล้วในหัวข้อที่ 7.1.4

ผู้อ่าน สามารถ แสดง รายชื่อ ไฟล์ หรือ ไดเรกทอรี หรือ ชื่อ อุปกรณ์ ภายใต้ ไดเรกทอรี /dev โดย พิมพ์ คำสั่ง บน โปรแกรม Terminal

```
$ ls /dev
```

ผู้อ่านต้องเปรียบเทียบกับชื่ออุปกรณ์ที่ผู้เขียนตัวหนาไว้ว่าตรงกันหรือไม่ อย่างไร เพื่อให้ผู้อ่านมองเห็นชัดว่า **mmcblk0p2** มีอยู่จริงและระบบได้ทำการเมาท์เข้ากับไดเรกทอรีรูท (Root) นั่นคือ ไดเรกทอรี / ด้วยชนิด ext4 ตามที่ได้แสดงในคำสั่งก่อนหน้าแล้ว

ashem autofs **block** btrfs-control bus cachefiles cec0 cec1 **char** console cpu_dma_latency cuse disk dma_heap dri fb0 fd full fuse gpiochip0 gpiochip1 gpiochip2 **gpiomem** hidraw0 hidraw1 hidraw2 hidraw3 hwrng i2c-20 i2c-21 initctl input kmsg kvm log loop0 loop1 loop2 loop3 loop4 loop5 loop6 loop7 loop-control mapper media0 media1 mem mmcblk0 **mmcblk0p__ mmcblk0p__ mqueue** net null port ppp ptmx **pts** ram0 ram1 ram10 ram11 ram12 ram13 ram14 ram15 ram2 ram3 ram4 ram5 ram6 ram7 ram8 ram9 random raw rfkill rpivid-h264mem rpivid-hevcmem rpivid-initc rpivid-vp9mem serial1 **shm** snd **stderr stdin stdout** tty tty0 ... ttyAMA0 ttyprintk uhid uinput urandom vchiq vcio vc-mem vcs ... watchdog watchdog0 zero

นอกจากนี้ อุปกรณ์สำคัญอื่นๆ เช่น stdin (standard input) stdout (standard output) และ stderr (standard error) นั้นเกี่ยวข้องกับโปรแกรม Terminal ซึ่งเชื่อมโยงกับประโยคในภาษา C ในการทดลองที่ 5 ภาคผนวก E

```
#include <stdio.h>
```

L.4 กิจกรรมท้ายการทดลอง

1. จงใช้โปรแกรม File Manager แล้วคลิกขวาบนชื่อไฟล์เพื่อแสดงคุณสมบัติ (Properties) ต่างๆ บนแท็บ General และอธิบายโดยเฉพาะหัวข้อ Total size of files และ Size on disk ว่าเหตุใดถึงแตกต่างกัน

2. สร้างไฟล์ (New) ด้วยโปรแกรม nano คีย์ข้อความด้วยตัวอักษรจำนวน 1 ตัวแล้วบันทึก (Save) ใช้คำสั่ง ls -l เพื่อแสดงรายละเอียดของไดเรกทอรีที่บรรจุไฟล์นั้น เพื่อหาขนาดไฟล์ที่แท้จริง

3. โปรดสังเกตว่าใน test.txt ที่สร้างด้วยโปรแกรม nano เราได้พิมพ์ประโยค fdd คิดเป็นจำนวน 3 ตัวอักษรๆ ละ 1 ไบต์เท่านั้น จงหาว่าไบต์ที่ 4 คือตัวอักษรใดในรูปที่ 2.12

4. เพิ่มจำนวนตัวอักษรไปเรื่อยๆ ใน test.txt จนทำให้ไฟล์มีขนาดมากกว่าเท่ากับ 4096 ไบต์ แล้วใช้คำสั่ง du -B1 test.txt **ตรวจสอบ**ขนาดไฟล์อีกรอบ บันทึกและอธิบายผลที่ได้โดยเฉพาะจำนวน Blocks ที่ได้จากคำสั่งว่าเท่ากับกี่**เซ็กเตอร์**

5. จงเปรียบเทียบผลลัพธ์ของคำสั่ง stat ระหว่าง ไดเรกทอรี และ ไฟล์

6. สิทธิ์การเข้าถึง (Permission) ของไดเรกทอรีหรือของไฟล์ประกอบด้วยบิตจำนวน 9 บิต แบ่งเป็น 3 ชุดๆ ละ 3 บิต จงเรียงลำดับชุดต่างๆ ว่าเป็นของสิทธิ์ของใครบ้าง

7. จงใช้คำสั่งต่อไปนี้ เพื่อแสดงรายชื่อไดเรกทอรีและไฟล์ และอธิบายผลว่าหมายเลขที่อยู่ด้านซ้ายสุดคืออะไร และเหตุใดจึงมีค่าซ้ำ

```
$ ls -i -l /
```

8. จงใช้คำสั่งต่อไปนี้ เพื่อแสดงรายละเอียดของชื่อไดเรกทอรีคู่ที่ซ้ำจากข้อที่แล้ว และอธิบายผลว่ามีอะไรที่แตกต่างกัน เพราะเหตุใด

```
$ stat /proc
$ stat /sys

$ stat /dev
$ stat /run
```

9. จงใช้คำสั่งต่อไปนี้ เพื่อแสดงรายละเอียดของอุปกรณ์ และอธิบายว่ามีผลลัพธ์ที่แตกต่างกันหรือไม่ เพราะเหตุใด

```
$ stat /dev/mmcblk0p2
$ stat /
```

10. จงอธิบายว่าเหตุใดไดเรกทอรี asound จึงอยู่ใต้ /proc ในหัวข้อที่ I.2.3 การทดลองที่ 9 ภาคผนวก I

11. จงอธิบายความเชื่อมโยงระหว่าง gpiomem ที่ได้จากคำสั่ง ls /dev กับกิจกรรมท้ายการทดลองที่ 10 ภาคผนวก J