Appendix L

การทดลองที่ 12 การศึกษาอุปกรณ์เก็บรักษาข้อมูลและระบบไฟล์

การทดลองนี้อธิบายและเชื่อมโยงเนื้อหาความรู้ของทุกบทเข้าด้วยกัน แต่จะเน้นบทที่ 2 และบทที่ 7 เพื่อให้ผู้อ่านมองเห็นอุปกรณ์อินพุทและเอาท์พุทเหมือนไฟล์แต่ละไฟล์ โดยมีวัตถุประสงค์ดังนี้

- เพื่อให้เข้าใจขนาดของไฟล์และโฟลเดอร์ในระบบไฟล์
- เพื่อให้รู้จักโครงสร้างและระบบไฟล์ของหน่วยความจำการ์ด microSD ที่ใช้งานในปัจจุบัน
- เพื่อให้เข้าใจระบบไฟล์ (File System) ชนิดต่างๆ บนบอร์ด Pi3
- เพื่อให้สามารถเชื่อมโยงอุปกรณ์อินพุทเอาท์พุทชนิดต่างๆ กับระบบไฟล์

L.1 ขนาดของไฟล์และไดเรคทอรี

ผู้อ่านสามารถตรวจสอบขนาดของไฟล์ใดๆ ชื่อ filename ที่แท้จริง หน่วยเป็นไบท์ ด้วยคำสั่งต่อไปนี้ du (DIsk Usage)

- ย้ายโฟลเดอร์ปัจจุบันไปที่ /home/pi ซึ่งเป็นโฟลเดอร์หลักของผู้ใช้ชื่อ 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’ จะแสดงผลขนาดเป็นจำนวนไบท์นำหน้าชื่อไฟล์นั้น

```
$ du -b test.txt
4 test.txt
```

4 หมายถึง เลขจำนวนไบท์ที่คำสั่ง du แสดงผลมาตามพารามิเตอร์ b ที่ส่งไป เพื่อบอกค่าขนาดของไฟล์ test.txt เป็นจำนวน 4 ไบท์

- คำสั่ง ‘du -B1 filename’ ผู้อ่านสามารถตรวจสอบขนาดของไฟล์ใดๆ ชื่อ filenameที่จัดเก็บเป็นจำนวนเท่าของ 4096 ไบท์ ในอุปกรณ์เก็บรักษาข้อมูล SD ด้วยคำสั่งต่อไปนี้

```
$ du -B1 test.txt
4096 test.txt
```

4096 หมายถึง เลขจำนวนไบท์ที่คำสั่ง du แสดงผลมาตามพารามิเตอร์ B1 ที่ส่งไป โดยผู้อ่านจะสังเกตเห็นความแตกต่าง ถึงแม้ไฟล์มีข้อมูลจำนวนน้อยเพียงไม่กี่ไบท์ แต่การจองพื้นที่ในอุปกรณ์สำรองจะมีขนาดเป็นจำนวนเท่าของ 4096 ไบท์เสมอ

- คำสั่ง ‘du -h’ จะแสดงผลขนาดหรือจำนวนไบท์โดยใช้หน่วยเช่น K (Kilo) M (Mega) G (Giga) นำหน้าชื่อโฟลเดอร์หรือไดเรคทอรีที่อยู่ใต้โฟลเดอร์ปัจจุบัน และจดบันทึก 10 รายการสุดท้ายลงในตาราง

```
$ du -h
```

Size	Folder Name

L.2 ระบบไฟล์

ผู้ใช้หรือผู้ดูแลระบบลีนุกซ์ สามารถตรวจสอบการใช้งานอุปกรณ์เก็บรักษาข้อมูล เช่น ฮาร์ดดิสค์ โซลิดสเตทดิสค์ การ์ดหน่วยความจำ ได้โดยคำสั่ง

- คำสั่ง df (Disk File System) สามารถแสดงรายละเอียดของอุปกรณ์เก็บรักษาข้อมูลในเครื่อง
- คำสั่ง 'df -h' จะแสดงรายการ ดังต่อไปนี้ จดบันทึก 10 รายการแรกลงในตาราง

```
$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on

โดย Size จะแสดงผลขนาดหรือจำนวนไบท์โดยใช้หน่วยเช่น K (Kilo) M (Mega) G (Giga)

- คำสั่ง 'df -T' จะเพิ่มคอลัมน์ชนิด (Type) ของแต่ละรายการในการแสดงผล และขนาดเป็นจำนวนเท่าของ 1 กิโลไบท์ (1K) แทน จดบันทึก 5 รายการแรกลงในตาราง

```
$ df -T
```

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

- คำสั่ง 'df -i' จะแสดงรายการต่างๆ ดังนี้ จดบันทึก 10 รายการแรกลงในตาราง

```
$ df -i
```

Filesystem	Inodes	IUsed	IFree	IUse%	Mounted on

โดยคอลัมน์จะแสดงผลเป็นจำนวน inode แทน รายละเอียดเรื่อง inode ผู้อ่านสามารถค้นคว้าเพิ่มเติมได้ในบทที่ 7 และทาง wikipedia

- คำสั่ง stat แสดงรายละเอียดของไฟล์หรือไดเรคทอรี

```
$ stat asm

  File: asm
  Size: 4096       Blocks: 8          IO Block: 4096   directory
Device: b307h/45831d Inode: 521754      Links: 3
Access: (0755/drwxr-xr-x)  Uid: ( 1000/      pi)   Gid: ( 1000/      pi)
Access: 2019-03-19 19:43:05.449401732 +0700
Modify: 2019-03-19 19:43:05.449401732 +0700
Change: 2019-03-19 19:43:05.449401732 +0700
 Birth: -
```

ผู้เขียนอธิบายผลลัพธ์ที่ได้ตามลำดับดังนี้

- ชื่อ asm
- ขนาด 4096 ไบท์ ใช้พื้นที่จำนวน 8 Blocks เป็นไดเรคทอรี (directory)
- มีหมายเลข Device = b307h/45831d หรือเท่ากับ $b307_{16}/45831_{10}$
- มีหมายเลข Inode ที่ 521754 จำนวน 3 Links
- สิทธิ์เข้าถึง (Access) ด้วยรหัส 0644 หรือ $011_2{:}100_2{:}100_2$ โดยผู้ใช้หมายเลข Uid (User ID)=1000 ชื่อผู้ใช้ (Username)=pi ในกรุปหมายเลข Groupid=1000 ชื่อกรุป pi
- เข้าถึง (Access) ณ วันที่ 19 มีนาคม 2019 เวลา 19.43.05

- เปลี่ยนแปลง (Modify) ณ วันที่ 19 มีนาคม 2019 เวลา 19.43.05
- เวลาที่ Inode เปลี่ยนแปลง (Change) ณ วันที่ 19 มีนาคม 2019 เวลา 19.43.05

เบื้องต้นผู้เขียนขอให้ผู้อ่านสร้างไฟล์ผลลัพธ์จากคำสั่ง stat ไปเก็บในไฟล์ เพื่อมาใช้ประกอบการทดลองต่อไป โดย

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

หลังจากนั้น เราสามารถตรวจสอบสถานะของไฟล์ stat_asm.txt ได้ดังนี้

```
$ stat stat_asm.txt

  File: stat_asm.txt
  Size: 341        Blocks: 8          IO Block: 4096   regular file
Device: b307h/45831d Inode: 524766      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/      pi)   Gid: ( 1000/      pi)
Access: 2019-03-19 19:45:05.459401732 +0700
Modify: 2019-03-19 19:45:05.459401732 +0700
Change: 2019-03-19 19:45:05.459401732 +0700
 Birth: -
```

ผู้เขียนอธิบายผลลัพธ์ที่ได้ตามลำดับดังนี้

- ชื่อ stat_asm.txt
- ขนาด 341 ไบท์ ใช้พื้นที่จำนวน 8 Blocks เป็นไฟล์ธรรมดา (regular File)
- มีหมายเลข Device = b307h/45831d หรือเท่ากับ $b307_{16}/45831_{10}$
- มีหมายเลข Inode ที่ 524766 จำนวน 1 Links
- สิทธิ์เข้าถึง (Access) ด้วยรหัส 0644 หรือ $011_2{:}100_2{:}100_2$ โดยผู้ใช้หมายเลข Uid (User ID)=1000 ชื่อผู้ใช้ (Username)=pi ในกรุปหมายเลข Groupid=1000 ชื่อกรุป pi
- เข้าถึง (Access) ณ วันที่ 19 มีนาคม 2019 เวลา 19.45.05
- เปลี่ยนแปลง (Modify) ณ วันที่ 19 มีนาคม 2019 เวลา 19.45.05
- เวลาที่ Inode เปลี่ยนแปลง (Change) ณ วันที่ 19 มีนาคม 2019 เวลา 19.45.05

L.3 อุปกรณ์อินพุทและเอาท์พุทในระบบไฟล์

การทดลองในหัวข้อนี้จะเชื่อมต่อกับเนื้อหาในบทที่ 3 และ การทดลองที่ 4 ภาคผนวก D หลักการของระบบปฏิบัติการ Unix คือ การเมาท์ (Mount) อุปกรณ์กับโฟลเดอร์ด้วยระบบไฟล์ (File System) ที่แตกต่างกัน โดยใช้ชื่อโฟลเดอร์ที่แตกต่างกัน โดยมีรูทไดเรคทอรีหรือโฟลเดอร์ (Root Directory) เป็นตำแหน่งเริ่มต้น ผู้อ่านสามารถพิมพ์คำสั่งใน Terminal

```
$ mount
```

คำสั่งนี้จะแสดงรายชื่อการเมาท์ หรือ ผูกยึด อุปกรณ์อินพุทเอาท์พุท เข้ากับโฟลเดอร์ของระบบปฏิบัติการ ตัวอย่างผลลัพธ์และคำอธิบายต่อไปนี้

- /dev/mmcblk0p7 on / type ext4 (rw,noatime,data=ordered)
- devtmpfs on /dev type devtmpfs (rw,relatime,size=470116k,nr_inodes=117529,mode=755)
- sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
- proc on /proc type proc (rw,relatime)
- tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
- devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
- ...

โดยมีชนิด (type) หรือระบบไฟล์ที่แตกต่างกัน เช่น

- ชนิด ext4 ซึ่งเป็นระบบไฟล์หลักของลีนุกซ์ ย่อมาจากคำว่า Fourth Extended File System เป็นเวอร์ชันที่ 4 พัฒนาจากชนิด ext3 รายละเอียดเพิ่มเติมที่ wikipedia
- ชนิด sysfs เป็นระบบไฟล์เสมือน (Virtual File System) รายละเอียดเพิ่มเติมที่ wikipedia
- ชนิด devfs เป็นระบบไฟล์เสมือน (Virtual File System) สำหรับอุปกรณ์อินพุทเอาท์พุทต่างๆ รายละเอียดเพิ่มเติมที่ wikipedia
- ชนิด tempfs ย่อมาจากคำว่า Temporary File System รายละเอียดเพิ่มเติมที่ wikipedia
- ชนิด proc เป็นระบบไฟล์เสมือน (Virtual File System) สำหรับระบบสำคัญต่างๆ เช่น CPU, โดยจะสร้างขึ้นเมื่อบูทเครื่อง และลบทิ้งเมื่อชัทดาวน์ระบบ รายละเอียดเพิ่มเติมที่ 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 เพื่อตั้งค่าไม่ให้รันไฟล์ที่อยู่ในโฟลเดอร์นี้ได้ เช่น ไฟล์ที่เป็นไวรัสหรือมัลแวร์ (Malware) ที่แอบแฝงเข้ามา
- nodev: No Device หมายถึง การไม่อนุญาตให้สร้างหรืออ่านโหนด (Node) ซึ่งเป็นไฟล์ชนิดพิเศษ
- mode หมายถึง Group 3 บิท คือ บิทควบคุม Read Write Execute รวมทั้งหมด 9 บิท

ผู้อ่านสามารถแสดงรายชื่อไดเร็คอทรีหรือโฟลเดอร์หรือชื่ออุปกรณ์ภายใต้โฟลเดอร์ /dev โดยพิมพ์คำสั่งบนโปรแกรม Terminal

```
$ ls /dev
```

ผู้อ่านจะเห็นผลลัพธ์ที่ได้ทั้งหมดซึ่งมีจำนวนมากพอสมควร แต่ผู้เขียนได้พิมพ์ชื่ออุปกรณ์ที่สำคัญๆ ด้วยตัวหนา เพื่อให้ผู้อ่านมองเห็นชัดว่า **mmcblk0p7** มีอยู่จริงและระบบได้ทำการเมาท์เข้ากับโฟลเดอร์รูท (Root) นั่นคือ โฟลเดอร์ / ด้วยชนิด ext4 ตามที่ได้แสดงในคำสั่งก่อนหน้าแล้ว

autofs **block** btrfs-control bus cachefiles **char** console cpu_dma_latency cuse disk fb0 fd full fuse gpiochip0 gpiochip1 gpiochip2 **gpiomem** hidraw0 hidraw1 hwrng initctl input kmsg log loop0 loop1 loop2 loop3 loop4 loop5 loop6 loop7 loop-control mapper mem memory_bandwidth **mmcblk0** mmcblk0p1 mmcblk0p2 mmcblk0p5 mmcblk0p6 **mmcblk0p7 mqueue** net network_latency network_throughput null ppp ptmx **pts** ram0 ram1 ram10 ram11 ram12 ram13 ram14 ram15 ram2 ram3 ram4 ram5 ram6 ram7 ram8 ram9 random raw rfkill serial1 **shm** snd **stderr stdin stdout** tty tty0 tty1 tty10 tty11 tty12 tty13 tty14 tty15 tty16 tty17 tty18 tty19 tty2 tty20 tty21 tty22 tty23 tty24 tty25 tty26 tty27 tty28 tty29 tty3 tty30 tty31 tty32 tty33 tty34 tty35 tty36 tty37 tty38 tty39 tty4 tty40 tty41 tty42 tty43 tty44 tty45 tty46 tty47 tty48 tty49 tty5 tty50 tty51 tty52 tty53 tty54 tty55 tty56 tty57 tty58 tty59 tty6 tty60 tty61 tty62 tty63 tty7 tty8 tty9 ttyAMA0 ttyprintk uhid uinput urandom vchiq vcio vc-mem vcs vcs1 vcs2 vcs3 vcs4 vcs5 vcs6 vcs7 vcsa vcsa1 vcsa2 vcsa3 vcsa4 vcsa5 vcsa6 vcsa7 vcsm vhci 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. สร้างไฟล์ตัวอักษรจำนวน 1 ตัว ด้วยโปรแกรมเท็กซ์อีดิเตอร์ทั่วไป แล้วบันทึก (Save) ใช้คำสั่ง ls -l เพื่อแสดงรายละเอียดของโฟลเดอร์ที่บรรจุไฟล์นั้น เพื่อหาขนาดไฟล์ที่แท้จริง
3. โปรดสังเกตว่าใน test.txt เราได้พิมพ์ประโยค fdd คิดเป็นจำนวน 3 ตัวอักษรๆ ละ 1 ไบท์เท่านั้น จงหาว่าไบท์ที่ 4 คือตัวอักษรใด
4. เพิ่มจำนวนตัวอักษรไปเรื่อยๆ ใน test.txt จนทำให้ไฟล์มีขนาดมากกว่าเท่ากับ 4096 ไบท์ แล้วใช้คำสั่ง du -B1 test.txt ตรวจสอบขนาดไฟล์อีกรอบ บันทึกและอธิบายผลที่ได้ q
5. จงเปรียบเทียบผลลัพธ์ของคำสั่ง stat ระหว่าง ไดเรคทอรี และ ไฟล์
6. สิทธิ์การเข้าถึง (Permission) ของไดเรคทอรีหรือของไฟล์ประกอบด้วยบิทจำนวน 9 บิท แบ่งเป็น 3 ชุดๆ ละ 3 บิท จงเรียงลำดับชุดต่างๆ ว่าเป็นของสิทธิ์ของใครบ้าง
7. จงใช้คำสั่งต่อไปนี้ เพื่อแสดงรายชื่อไดเรคทอรีและไฟล์ และอธิบายผลว่าหมายเลขที่อยู่ด้านซ้ายสุดคืออะไร และเหตุใดจึงมีค่าซ้ำ

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

8. จงใช้คำสั่งต่อไปนี้ เพื่อแสดงรายละเอียดของชื่อไดเรคทอรีคู่ที่ซ้ำจากข้อที่แล้ว และอธิบายผลว่ามีอะไรที่แตกต่างกัน เพราะเหตุใด

   ```
   $ stat /proc
   $ stat /sys

   $ stat /dev
   $ stat /run
   ```

9. จงใช้คำสั่งต่อไปนี้ เพื่อแสดงรายละเอียดของอุปกรณ์ และอธิบายผลว่ามีอะไรที่แตกต่างกัน เพราะเหตุใด

   ```
   $ stat /dev/mmcblk0p7
   $ stat /
   ```

10. จงอธิบายว่าเหตุใดว่าโฟลเดอร์ asound จึงอยู่ใต้ /proc ในหัวข้อที่ I.2.1
11. จงอธิบายความเชื่อมโยงระหว่าง gpiomem ที่ได้จากคำสั่ง ls /dev กับการทดลองที่ 8

ข้อ 1) Total size of files คือ ขนาดจริงๆของไฟล์ ส่วน size on disk คือ ขนาดที่จองไว้เก็บไฟล์

ข้อ 2) ขนาดไฟล์ 2 bytes

ข้อ 3) ตัวอักษร EOF (End of file)

ข้อ 4) ผลลัพธ์คือ 12288 จองที่เพิ่ม 3 เท่าของ 4096 bytes

ข้อ 5) คำสั่งไม่สนใจไดเรคทอรี่ ใช้พื้นที่ไป 4096 bytes นับเป็น 8 blocks

ข้อ 6) เรียงดังนี้ Owner > Group > Other

ข้อ 7) index number และที่ซ้ำได้เพราะ data ในไดเรคทอรี่ย่อยๆ ก็จะเริ่มนับที่ indexแรกใหม่เหมือนกัน

ข้อ 8) Device ต่างกัน เพราะ IO block

ข้อ 9) มีสองแบบ link file (มีเยอะ) และ special file (มีอันเดียว) ทั้งสองมีสิ่งที่ต่างกันคือ index number, size, block, links, device, type

ข้อ 10) proc ถูกสร้างตอน boots จึงมีข้อมูลระบบต่างๆอยู่ภายใน

ข้อ 11) gpiomem จะ map virtual address ให้เป็น physical address ของ gpio