

## ภาคผนวก F

# การทดลองที่ 6 การพัฒนาโปรแกรมภาษาแอสเซมบลี

การทดลองนี้คาดว่าผู้อ่านผ่านการเขียนหรือพัฒนาโปรแกรมด้วยภาษา C ใน การทดลองที่ 5 ภาคผนวก E แล้ว และมีความคุ้นเคยกับ IDE จากการพัฒนาโปรแกรมและการติดตั้งโปรแกรมด้วยภาษา C/C++ ด้วย CodeBlocks ดังนั้น การทดลองมีวัตถุประสงค์เหล่านี้

- เพื่อให้เข้าใจการพัฒนาและติดตั้ง (Debug) โปรแกรมภาษาแอสเซมบลีด้วย IDE ชื่อ CodeBlocks บนระบบปฏิบัติการตระกูลยูนิกซ์
- เพื่อให้เข้าใจความแตกต่างระหว่างการพัฒนาโปรแกรมภาษาแอสเซมบลีด้วย IDE และ Makefile

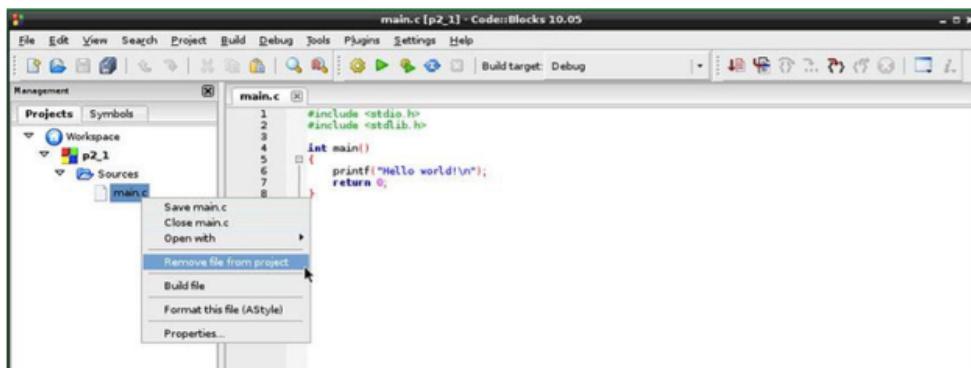
### F.1 การพัฒนาโดยใช้ IDE

- พิมพ์คำสั่งนี้ในโปรแกรม Terminal เพื่อเริ่มต้นใช้งาน CodeBlocks

```
$ codeblocks
```

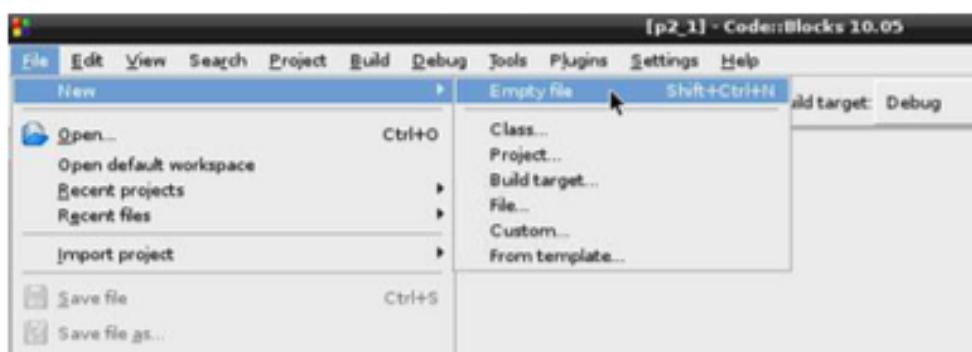
- หน้าต่างหลักจะปรากฏขึ้น หลังจากนั้น ผู้อ่านสามารถสร้างโปรเจกต์ใหม่โดยเลือก "Create a new project" ในช่องด้านซ้าย แล้วเลือก "Console application" ในช่องด้านขวาเพื่อสร้างโปรแกรม
- กรอกชื่อโปรเจ็คใหม่ชื่อ Lab6 ในช่อง Project title: และกรอกชื่อไดเรกทอรี /home/pi/asm ในช่อง Folder to create project in: โปรดสังเกตข้อความในช่อง Project filename: ว่าตรงกับ Lab6.cbp ใช่หรือไม่ แล้วจึงกด Next > **ต่อไป**
- โปรแกรม CodeBlocks จะสร้างไดเรกทอรีต่าง ๆ ภายใต้ไดเรกทอรีชื่อ /home/pi/asm/Lab6/
- กดปุ่ม "Next >" เพื่อดำเนินการต่อ และสุดท้ายจะเป็นขั้นตอนการเลือกค่า configuration สำหรับคอมไฟเลอร์ เลือก option Debug หมายสำหรับการเริ่มต้นและแก้ไขข้อผิดพลาด แล้วจึงกดปุ่ม "Finish" เมื่อเสร็จสิ้น

6. คลิก ชื่อ Workspace ในหน้าต่าง ด้านซ้าย เพื่อขยายโครงสร้างโปรเจกต์แล้ว ค้นหาไฟล์ชื่อ "main.c" คลิกขวาบนชื่อไฟล์แล้วเลือกเมนู "Remove file from project" ตามรูปที่ F.1



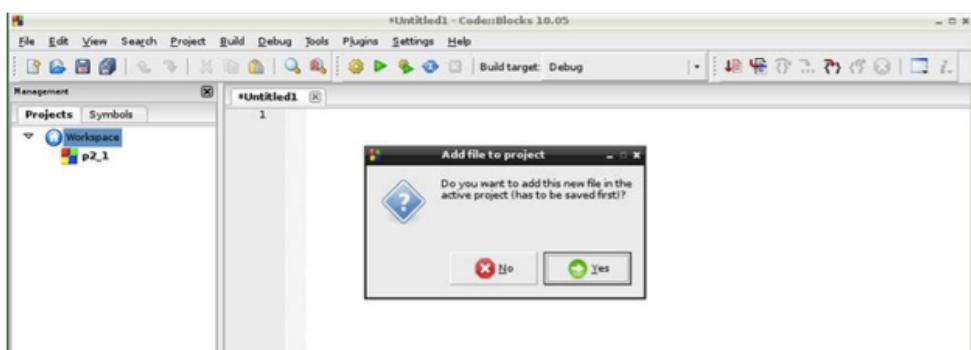
รูปที่ F.1: การรื้อไฟล์ main.c ออกจากโปรเจกต์

7. เพิ่มไฟล์ใหม่ลงในโปรเจกต์โดยกดเมนู File->New->Empty file ตามรูปที่ F.2



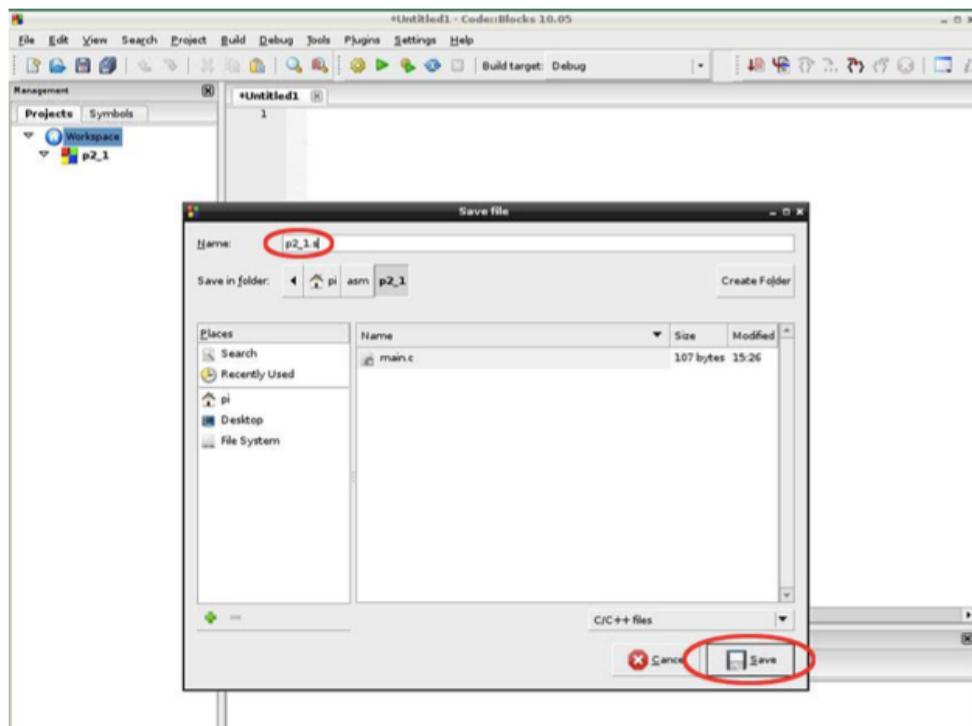
รูปที่ F.2: การเพิ่มไฟล์ใหม่ลงในโปรเจกต์

8. คลิกปุ่ม "Yes" เพื่อยืนยันในรูปที่ F.3



รูปที่ F.3: หน้าต่างกดปุ่ม "Yes" เพื่อยืนยัน

9. หน้าต่าง "Save file" จะปรากฏขึ้น กรอกชื่อไฟล์ว่า main.s และจึงกดปุ่ม "Save" ดังรูปที่ F.4



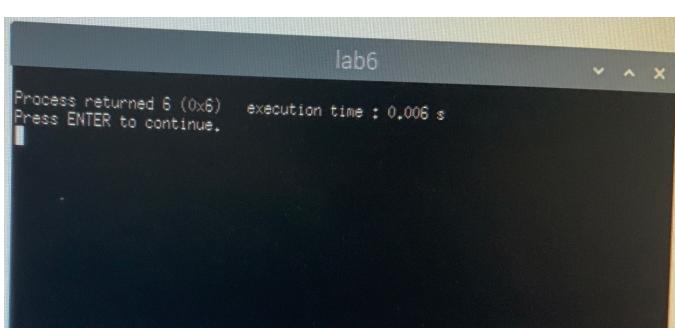
รูปที่ F.4: หน้าต่าง Save File ชื่อไฟล์ว่า main.s

10. คลิกชื่อ Workspace ในหน้าต่างด้านซ้าย และคลิกขวาเพิ่ม (Add) ไฟล์ main.s เข้าไปในโปรเจกต์
11. ป้อนคำสั่งเหล่านี้ในไฟล์ main.s

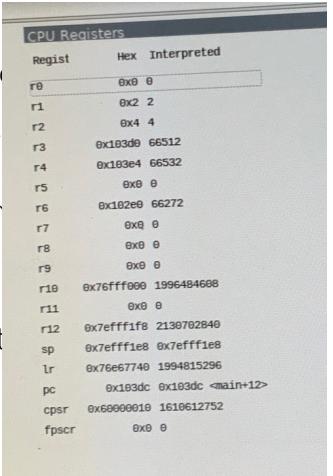
```
.global main

main:
    MOV R0, #0
    MOV R1, #2
    MOV R2, #4
    ORR R0, R1, R2
    BX LR
```

12. เลือกเมนู Build->Build เพื่อแปล (Assemble) โปรแกรมที่เขียนให้เป็นโปรแกรมภาษาเครื่อง
13. เลือกเมนู Build->Run เพื่อรันโปรแกรม
14. อ่านและบันทึกประโยคที่เกิดขึ้นในหน้าต่าง Terminal ที่ปรากฏขึ้นมา



## F.2 การดีบักโปรแกรมโดยใช้ IDE

1. ในไฟล์ main.s เลื่อนเครื่องเซอร์ไพร์สเบอร์ที่มีคำสั่ง ORR R0, R1, R2 คลิกเมนู Debug หรือกดปุ่ม F5 ผู้อ่านจะสังเกตว่ากลมสีแดงปรากฏขึ้นด้านซ้ายของคำสั่ง ORR
2. กด เมนู Debug->Debugging Windows->CPU Registers เพื่อแสดงค่าของหน้าต่างที่ปรากฏขึ้นมาเพิ่มเติม
3. เมื่อพร้อมแล้ว ผู้อ่านสามารถเริ่มต้นการดีบักโดยกดเมนู Debug->Start/Cont โปรแกรมจะเริ่มต้นทำงานตั้งแต่ประযุกแรกจนหยุดที่คำสั่ง ORR R0, R1, R2
4. อ่านและบันทึกค่าของ R0 และ PC ในหน้าต่าง CPU Registers 
5. ประมวลผลคำสั่งถัดไปโดยกดเมนู Debug->Next Instruction หรือกดปุ่ม Alt+F7 พร้อมกัน
6. อ่านและบันทึกค่าของ R0 และ PC ในหน้าต่าง CPU Registers และสังเกตการเปลี่ยนแปลงที่เกิดขึ้น โดยเปรียบเทียบกับค่า R0 และ PC ในข้อ 4 กับข้อนี้
7. อธิบายว่าเกิดอะไรขึ้นกับค่าของรีจิสเตอร์ R0 และ PC

## F.3 การพัฒนาโดยใช้ประยุคคำสั่ง (Command Line)

ผู้อ่านควรเข้าใจคำสั่งพื้นฐานในการแปลงโปรแกรมภาษาแอสเซมบลีที่สร้างขึ้นใน CodeBlocks ก่อนหน้านี้ ตามขั้นตอนต่อไปนี้

1. ใช้โปรแกรมไฟล์เมเนจเจอร์เพื่อเบรน์ไฟล์ในไดเรกทอรี /home/pi/asm/Lab6
2. ดับเบิล คลิก บน ชื่อ ไฟล์ main.s เพื่อ เปิด อ่าน ไฟล์ และ เปรียบ เทียบ กับ ไฟล์ ที่ เขียน ใน โปรแกรม CodeBlocks
3. เปิดโปรแกรม Terminal หน้าต่างใหม่ แล้วป้อน ไดเรกทอรีปัจจุบัน (cd: change directory) ไปยัง /home/pi/asm/Lab6 โดยใช้คำสั่ง

```
$ cd /home/pi/asm/Lab6
```

4. แปลไฟล์ซอร์สโค๊ดให้เป็นไฟล์อ๊อบเจกต์ โดยเรียกใช้คำสั่ง as (assembler) ดังนี้

```
$ as -o main.o main.s
```

5. ใช้คำสั่ง ls -la ใน Terminal เพื่อค้นหาไฟล์อ๊อบเจกต์ชื่อ main.o ว่ามีจริงหรือไม่

6. อ่านและบันทึกค่าของ R0 และ PC ในหน้าต่าง CPU Registers และสังเกตการเปลี่ยนแปลงที่เกิดขึ้น โดยเปรียบเทียบกับค่า R0 และ PC ในข้อ 4 กับข้อนี้

CPU Registers		
Regist	Hex	Interpreted
r0	0x0 0	
r1	0x2 2	
r2	0x4 4	
r3	0x103d0 66512	
r4	0x103e4 66532	
r5	0x0 0	
r6	0x102e0 66272	
r7	0x0 0	
r8	0x0 0	
r9	0x0 0	
r10	0x76ffff000 1996484608	
r11	0x0 0	
r12	0x7efff1f8 2130702840	
sp	0x7efff1e8 0x7efff1e8	
lr	0x76e67740 1994815296	
pc	0x103dc 0x103dc <main+12>	
cpsr	0x60000010 1610612752	
fpscr	0x0 0	

Before

CPU Registers		
Regist	Hex	Interpreted
r0	0x6 6	
r1	0x2 2	
r2	0x4 4	
r3	0x103d0 66512	
r4	0x103e4 66532	
r5	0x0 0	
r6	0x102e0 66272	
r7	0x0 0	
r8	0x0 0	
r9	0x0 0	
r10	0x76ffff000 1996484608	
r11	0x0 0	
r12	0x7efff1f8 2130702840	
sp	0x7efff1e8 0x7efff1e8	
lr	0x76e67740 1994815296	
pc	0x103e0 0x103e0 <main+16>	
cpsr	0x60000010 1610612752	
fpscr	0x0 0	

After

7. อธิบายว่าเกิดอะไรขึ้นกับค่าของรีจิสเตอร์ R0 และ PC

R0 เปลี่ยนจาก 6 เป็น 0

PC เปลี่ยนจาก 0x103dc 0x103e0 <main+16>

เป็น 0x103dc 0x103dc <main+12>

2. ดับเบิลคลิกบนชื่อไฟล์ main.s เพื่อเปิดอ่านไฟล์และเปรียบเทียบกับไฟล์ที่เขียนในโปรแกรม CodeBlocks

```

main.s - /home/pi/asm/lab6 - Geany
File Edit Search View Document Project Build Tools Help
Symbols main.c main.s
Labels main [2]
1 main: _global main
2     main:
3         MOV R0, #0
4         MOV R1, #2
5         MOV R2, #4
6         ORR R0, R1, R2
7         BX LR
8

1 main: _global main
2     main:
3         MOV R0, #0
4         MOV R1, #2
5         MOV R2, #4
6         ORR R0, R1, R2
7         BX LR
8

Status 21:40:08: This is Geany 1.37.1.
21:40:08: Setting Spaces indentation mode for /home/pi/asm/Lab5/main.s
21:40:08: Setting Spaces indentation mode for /home/pi/asm/Lab5/main.c
21:40:08: File /home/pi/asm/Lab5/main.c opened (1).
21:40:08: Setting Spaces indentation mode for /home/pi/asm/lab6/main.s
21:40:08: Setting Spaces indentation mode for /home/pi/asm/lab6/main.c
21:40:08: Setting Spaces indentation mode for /home/pi/asm/lab6/main.s
21:40:08: Setting Spaces indentation mode for /home/pi/asm/lab6/main.c

Logs & others
CodeBlocks Search results
done Setting breakpoints
raspberrian 10.1-1.7) 10.1.90.20210103-git

```

5. ใช้คำสั่ง ls -la ใน Terminal เพื่อค้นหาไฟล์อ้อมเจตซื้อ main.o ว่ามีจริงหรือไม่

29

```

pi@raspberrypi: ~ cd /home/pi/asm/Lab6
bash: cd: /home/pi/asm/Lab6: No such file or directory
pi@raspberrypi: ~ /home/pi/asm/Lab6
bash: /home/pi/asm/Lab6: Is a directory
pi@raspberrypi: ~ cd /home/pi/asm/lab6
pi@raspberrypi: ~/asm/lab6 $ as -o main.o main.s
pi@raspberrypi: ~/asm/lab6 $ ls -la
total 36
drwxr-xr-x 4 pi pi 4096 Jan 30 21:59 .
drwxr-xr-x 4 pi pi 4096 Jan 30 21:31 ..
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 bin
-rw-r--r-- 1 pi pi 998 Jan 30 21:31 lab6.cbp
-rw-r--r-- 1 pi pi 75 Jan 30 21:43 lab6.depend
-rw-r--r-- 1 pi pi 99 Jan 30 21:31 main.c
-rw-r--r-- 1 pi pi 652 Jan 30 21:59 main.o
-rw-r--r-- 1 pi pi 97 Jan 30 21:37 main.s
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 obj
pi@raspberrypi: ~/asm/lab6 $ 

```

7. ใช้คำสั่ง ls -la ใน Terminal เพื่อค้นหาไฟล์โปรแกรมซื้อ Lab6 ว่ามีจริงหรือไม่

29

```

pi@raspberrypi: ~/asm/lab6
File Edit Tabs Help
total 36
drwxr-xr-x 4 pi pi 4096 Jan 30 21:59 .
drwxr-xr-x 4 pi pi 4096 Jan 30 21:31 ..
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 bin
-rw-r--r-- 1 pi pi 998 Jan 30 21:31 lab6.cbp
-rw-r--r-- 1 pi pi 75 Jan 30 21:43 lab6.depend
-rw-r--r-- 1 pi pi 99 Jan 30 21:31 main.c
-rw-r--r-- 1 pi pi 652 Jan 30 21:59 main.o
-rw-r--r-- 1 pi pi 97 Jan 30 21:37 main.s
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 obj
pi@raspberrypi: ~/asm/lab6 $ gcc -o lab6 main.o
pi@raspberrypi: ~/asm/lab6 $ ls -la
total 44
drwxr-xr-x 4 pi pi 4096 Jan 30 22:01 .
drwxr-xr-x 4 pi pi 4096 Jan 30 21:31 ..
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 bin
-rw-r--r-- 1 pi pi 8000 Jan 30 22:01 lab6
-rw-r--r-- 3 pi pi 998 Jan 30 21:31 lab6.cbp
-rw-r--r-- 1 pi pi 75 Jan 30 21:43 lab6.depend
-rw-r--r-- 1 pi pi 99 Jan 30 21:31 main.c
-rw-r--r-- 1 pi pi 652 Jan 30 21:59 main.o
-rw-r--r-- 1 pi pi 97 Jan 30 21:37 main.s
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 obj
pi@raspberrypi: ~/asm/lab6 $ 

```

### 6. ทำการลิงก์และแปลงไฟล์อ็อบเจกต์เป็นไฟล์โปรแกรมโดย

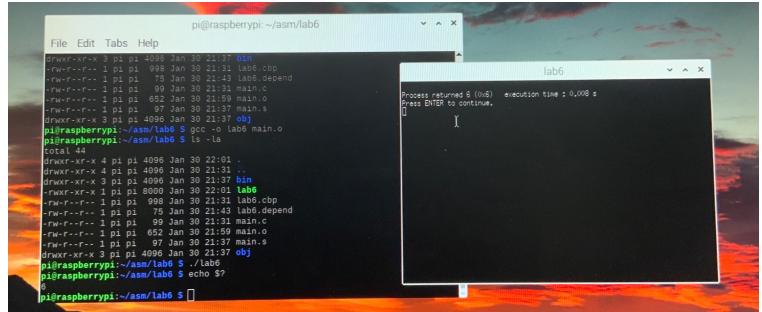
```
$ gcc -o Lab6 main.o
```

### 7. ใช้คำสั่ง ls -la ใน Terminal เพื่อค้นหาไฟล์โปรแกรมชื่อ Lab6 ว่ามีจริงหรือไม่

### 8. เรียกโปรแกรม Lab6 โดยพิมพ์

```
$ ./Lab6
```

```
$ echo $?
```



### 9. เปรียบเทียบหมายเลขที่ปรากฏขึ้นว่าตรงกับผลการรันใน IDE หรือไม่ อย่างไร

**ทราบ กัน**

## F.4 การพัฒนาโดยใช้ Makefile

การใช้ makefile สำหรับพัฒนาโปรแกรมภาษาแอสเซมบลีคล้ายกับการทดลองที่ 5 ในภาคผนวก E ก่อนหน้านี้

1. เปิดไดเรกทอรี /home/pi/asm/Lab6 ด้วยโปรแกรมไฟล์เมเนจero

2. กดปุ่มขวาบนเมาส์ในพื้นที่ไดเรกทอรีเพื่อสร้างไฟล์เปล่าใหม่ (New Empty File) โดยกำหนดชื่อ makefile

3. ป้อนข้อความเหล่านี้ลงในไฟล์ makefile:

```
Lab6: main.o
    gcc -o Lab6 main.o

main.o: main.s
    as -o main.o main.s

clean:
    rm *.o Lab6
```

4. บันทึกไฟล์แล้วปิดหน้าต่างบันทึก ผู้อ่านควรตรวจสอบรายชื่อไฟล์ที่อยู่ภายใต้ไดเรกทอรีนี้ว่ามีไฟล์อะไรบ้าง

5. ลบไฟล์อ็อบเจกต์ที่มีอยู่โดยใช้คำสั่ง

```
$ make clean
```

ในโปรแกรม Terminal เพื่อเปรียบเทียบหลังจากที่รันคำสั่ง make clean

```

File Edit Tabs Help
lab6
-rw-r--r-- 1 pi pi 998 Jan 30 21:31 lab6.cbp
-rw-r--r-- 1 pi pi 75 Jan 30 21:43 lab6.depend
-rw-r--r-- 1 pi pi 99 Jan 30 21:31 main.c
-rw-r--r-- 1 pi pi 652 Jan 30 21:59 main.o
-rw-r--r-- 1 pi pi 97 Jan 30 21:37 main.s
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 obj
pi@raspberrypi:~/asm/lab6 $ ./Lab6
pi@raspberrypi:~/asm/lab6 $ echo $?
6
pi@raspberrypi:~/asm/lab6 $ make clean
rm *.o lab6
pi@raspberrypi:~/asm/lab6 $ ls -la
total 36
drwxr-xr-x 4 pi pi 8000 Jan 30 22:16 .
drwxr-xr-x 4 pi pi 4096 Jan 30 21:31 ..
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 bin
-rw-r--r-- 1 pi pi 998 Jan 30 21:31 lab6.cbp
-rw-r--r-- 1 pi pi 75 Jan 30 21:43 lab6.depend
-rw-r--r-- 1 pi pi 99 Jan 30 21:31 main.c
-rw-r--r-- 1 pi pi 93 Jan 30 22:14 main.s
drwxr-xr-x 3 pi pi 4096 Jan 30 21:37 obj
pi@raspberrypi:~/asm/lab6 $ make

```

6. ใช้คำสั่ง ls -la ใน Terminal ค้นหาไฟล์อ้อมเจกต์ main.o และ Lab6 ว่าถูกลบหรือไม่ **ดูดู**
7. ทำการแอกซเซมเบิล main.s โดยใช้คำสั่ง make ในโปรแกรม Terminal และขอให้สังเกตวันเวลาของไฟล์ต่าง ๆ

```
$ make
```

8. ใช้คำสั่ง ls -la ใน Terminal เพื่อค้นหาไฟล์ชื่อ main.o และ Lab6 ว่ามีจริงหรือไม่ **มีจริง**
9. เรียกโปรแกรม Lab6 โดยพิมพ์

```
$ ./Lab6
```

```
$ echo $?
```

## F.5 กิจกรรมท้ายการทดลอง

1. จงปรับแก้คำสั่ง ORR เป็นคำสั่ง AND ในโปรแกรม main.s และตรวจสอบผลการเปลี่ยนแปลงแล้ว  
**จึงอธิบาย ลากเกิมที่ ใจดี ใจ หมวด เป็น O หาก ประทุม ORR (นิว่า เป็น AND (ดู)**
2. จงปรับแก้โปรแกรมใน main.s เป็นดังนี้ จดบันทึกผลการทดสอบและอธิบาย

```

.global main
main:
    MOV R5, #1
    จุด RS=1
    CMP R4, #0
    โปรตุปเปอร์บ
    BLE end
    - R4<0 แล้วก็จะ
    ถ้า R4>0 แล้วก็จะ
    ถ้าไม่เท่ากับ 0 แล้วก็จะ
else:
    MOV R5, #2
    จุด RS=2
end:
    MOV R0, R5
    กรณี RS=RS
    BX LR

```

3. จงปรับแก้โปรแกรมใน main.s เป็นดังนี้ จดบันทึกผลการทดสอบและอธิบาย

```
.data
.balign 4 — request 4byte space
var1: .word 1 — ค่า var1=1

.text
.global main

main:
    MOV R1, #2 — R1=2
    LDR R2, var1addr — R2 ค่า Var1Addr คือ R2 get value var1addr
    STR R1, [R2] — R2 ค่า R1
    LDR R0, [R2] — R0 get value R2
    BX LR
var1addr: .word var1 — var1addr = R1
```

