ASM Language	**ใบงานที่ 4** **Unix System call**
	สัปดาห์ที่ 1 **ระยะเวลา** ในชั้นเรียน **01418233 แอสแซมบลีและสถาปัตยกรรมคอมพิวเตอร์** ชื่อ ..... รหัสนิสิต .....

เวลา : 2 ชั่วโมง

จุดประสงค์การเรียนรู้

1. ทำการเรียนรู้การใช้งาน System call ด้วยการเขียนโปรแกรมภาษา Assembly
2. ทำการเรียนรู้การกำหนดชนิดการเรียก System call

เนื้อหาการเรียนรู้

1. System call

system calls คือชุดของคำสั่งที่ทำหน้าให้บริการกับยูสเซอร์โปรแกรมที่ต้องการใช้ทรัพยากรต่าง ๆ ของระบบ โดยที่ยูสเซอร์โปรแกรมนั้นจะไม่สามารถเข้าไปติดต่อกับทรัพยากรเหล่านั้นโดยตรงแต่ต้องสั่งผ่านทาง System calls เท่านั้น ในระบบปฏิบัติการ (Operating system) จะมีการป้องกันไม่ให้คำสั่งจากโปรแกรมของยูสเซอร์สามารถเข้าถึงทรัพยากรเหล่าได้โดยตรงเพื่อป้องกันไม่ให้คำสั่งบางคำสั่งในยูสเซอร์โปรแกรมซึ่งอาจจะมีความรู้เท่าไม่ถึงการหรือการขาดความชำนาญของโปรแกรมเมอร์ เข้าไปสร้างความเสียหายร้ายแรงให้แก่ทรัพยากรของระบบได้ โดยในระบบปฏิบัติการจะแบ่งโหมดการทำงานออกเป็นสองโหมด คือ kernel mode และ user mode โดยคำสั่งที่อยู่ในยูสเซอร์โปรแกรมจะสามารถทำงานได้เฉพาะใน user mode เท่านั้น

ระบบปฏิบัติการจะสามารถเข้าถึงและใช้งานทรัพยากรต่าง ๆ ของระบบได้ ก็ต่อเมื่ออยู่ใน kernel mode เท่านั้น แต่โดยปรกติแล้ว ระบบปฏิบัติจะทำงานใน user mode ดังนั้นเมื่อยูสเซอร์โปรแกรมต้องการใช้ทรัพยากรของระบบ ก็ต้องทำการเรียกใช้คำสั่ง system calls ให้ระบบปฏิบัติการทำการเปลี่ยนจาก user mode เป็น kernel mode แล้วเรียกใช้ทรัพยากรต่าง ๆของระบบต่อไปได้ ซึ่งเราเรียกเหตุการณ์นี้ว่าการ **interrupt** หรือการ **trap** ไปที่ CPU เพื่อให้ CPU ทำการเปลี่ยน mode การทำงานจาก user mode เป็น kernel mode

System calls จึงเป็นโปรแกรมตัวกลาง (application programming interface : API) ที่ทำหน้าที่เชื่อมโยงการทำงานระหว่าง user mode และ kernel mode ประโยชน์อีกประการหนึ่งของ system calls นอกจากการป้องกันไม่ให้ทรัพยากรต่าง ๆ เกิดความเสียหายจากความผิดพลาดของยูสเซอร์โปรแกรมแล้ว ก็คือการอำนวยความสะดวกในการเขียนคำสั่งในยูสเซอร์โปรแกรม ยูสเซอร์ไม่จำเป็นต้องรู้คำสั่งที่สลับซับซ้อนในการเรียกใช้งานทรัพยากรต่าง ๆ ยูสเซอร์เพียงแค่บอกว่าตนต้องการอะไรเท่านั้นในโปรแกรมแล้วทาง system calls ก็จะรับคำสั่งนั้นไปดำเนินการแล้วนำผลลัพธ์กลับมาส่งให้โดยที่ยูสเซอร์ไม่ต้องรู้เลยว่ามันมีกระบวนการทำงานอย่างไร

ในการเรียกใช้งาน system calls 32bit ในภาษา Assembly ต้องนำตัวเลขรหัสการใช้งานเอาไปเก็บไว้ใน Register EAX หากเป็นการเขียนโปรแกรมแบบ 32 bits หรือหากเป็น 64 bits เอาข้อมูลเก็บไว้ที่ Register RAX ก่อนแล้วดู Register ตัวต่อไปตามตาราง

ตารางที่ 1 system calls 32bit

EAX	Name	EBX(call code)	ECX(Address)	EDX
1	sys_exit	int	-	-
2	sys_fork	struct pt_regs	-	-
3	sys_read	unsigned int (0)	char *	size_t
4	sys_write	unsigned int (1)	const char *	size_t
5	sys_open	const char *	int	int
6	sys_close	unsigned int	-	-

ตารางที่ 2 system calls 64bit

RAX	Name	RDI(call code)	RSI(Address)	RDX
60	sys_exit	int	-	-
0	sys_read	unsigned int (0)	char *	size_t
1	sys_write	unsigned int (1)	const char *	size_t
2	sys_open	const char *	int	int
3	sys_close	unsigned int	-	-

ให้เขียนโปรแกรม SystemCall64.asm ด้านล่างนี้ เพื่อนำไปประกอบกับการฝึกปฏิบัติในหัวข้อทัดไป

```
        section .data ;Data segment
userMsg     db 'Please enter a number: ' ;Ask the user to enter a number
lenUserMsg  equ $-userMsg           ;The length of the message
dispMsg     db 'You have entered: '
lenDispMsg  equ $-dispMsg

        section .bss  ;Uninitialized data
num  resb 5
```

```
        section .text  ;Code Segment
                global _start
_start:  mov rax, 1
         mov rdi, 1
         mov rsi, userMsg
         mov rdx, lenUserMsg
         syscall

;Read and store the user input
        mov rax, 0
        mov rdi, 0
        mov rsi, num
        mov rdx, 5          ;5 bytes (numeric, 1 for sign) of that information
        syscall

;Output the message 'The entered number is: '
        mov rax, 1
        mov rdi, 0
        mov rsi, dispMsg
        mov rdx, lenDispMsg
        syscall

;Output the number entered
        mov rax, 1
        mov rdi, 0
        mov rsi, num
        mov rdx, 5
        syscall

; Exit code
        mov rax, 60
        mov rdi, 0
        syscall
```

2.1 ให้ทำการ Assemble และ Link SystemCall64.asm

2.2 ให้ทำการประมวลผลไฟล์ที่ได้จากทำการ Linker แล้วมี Terminal ด้วย ./SystemCall64

2.2.1 เขียนค่าของผลลัพธ์ที่ได้เมื่อป้อนเลข 1234 กด Enter

_____

_____

_____

2.2.2 ประมวลผลอีกครั้งแล้วเขียนค่าของผลลัพธ์ที่ได้เมื่อป้อนเลข 12345 กด Enter

_____

_____

_____

2.2.3 ประมวลผลอีกครั้งแล้วเขียนค่าของผลลัพธ์ที่ได้เมื่อป้อนเลข 123456 กด Enter

_____

_____

_____

2.2.4 ให้อธิบายทำไมผลลัพธ์ที่ได้จากข้อที่ 2.2.3 ไม่ได้เท่ากัน 123456 ลงในด้านล่างนี้

_____

_____

_____