System Service

Accessing system services is how the application requests that the operating system perform some specific operation i.e.,

- Display strings and numbers on screen
- Read keyboard input
- Read/write to file system
- Terminate execution of program

When calling system services, arguments are placed in the standard argument registers.

System services can be called using the following steps:

1. Determine the service function and number to use
2. Put service number in RAX register
3. Put all required parameters in specified registers, if any
4. Issue system call using ***syscall*** instruction

System Service

Console Output

The system service to output characters to the console is the system write (SYS_write).

Register	SYS_write
rax	Call code = SYS_write (1)
rdi	Output location, STDOUT (1)
rsi	Address of characters to output.
rdx	Number of characters to output.

```
STDOUT      equ 1          ; standard output
SYS_write   equ 1          ; call code for write
msg         db  "Hello World"
msgLen      dq 11

mov rax, SYS_write
mov rdi, STDOUT
mov rsi, msg                    ; msg address
mov rdx, qword [msgLen]         ; length value
syscall
```

System Service

Example: Console Output

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguageProgramming/code

File Edit View Search Terminal Help

```
global  _start

section .text                   ;code section
_start:                         ;start label
        mov     rax,1           ;call stdout
        mov     rdi,1
        mov     rsi,msg         ;pointer to starting address of message
        mov     rdx,msglen      ;length of message
        syscall                 ;execute system routine
        mov     rax,60          ;exit
        mov     rdi,0
        syscall

section .data
msg     db      "Hello, world!"
msglen  equ     $-msg
                                                        1,1           All
```

System Service

Example: Console Output

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguag...

File Edit View Search Terminal Help

```
global  _start
section .data
LF      equ     10
NULL    equ     0
EXIT_SUCCESS equ 0 ; success code
STDIN   equ     0
STDOUT  equ     1
STDERR  equ     2
SYS_write equ   1
SYS_exit equ    60
msg1    db      "Hello, World!",LF,NULL
msg2    db      "Enter answer: ",NULL
newLine db      LF, NULL

section .text
_start:
        mov     rdi, msg1
        call    printString
        mov     rdi, msg2
        call    printString
        mov     rdi, newLine
        call    printString

        mov     rax, SYS_exit           ;exit
        mov     rdi, EXIT_SUCCESS
        syscall
                                1,1           Top
```

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguag...

File Edit View Search Terminal Help

```
global printString
printString:
        push    rbx
        mov     rbx, rdi
        mov     rdx, 0
strCountLoop:
        cmp     byte [rbx], NULL        ;calcula
te string length
        je      strCountDone
        inc     rdx
        inc     rbx
        jmp     strCountLoop
strCountDone:
        cmp     rdx, 0
        je      prtDone
        mov     rax, SYS_write          ;write t
o console
        mov     rsi, rdi
        mov     rdi, STDOUT
        syscall
prtDone:
        pop     rbx
        ret
                                48,1-8        Bot
```

System Service

Console Input

The system service to read characters from the console is the system read (SYS_read).

Register	SYS_read
rax	Call code = SYS_read (0)
rdi	Input location, STDIN (0)
rsi	Address of where to store characters read.
rdx	Number of characters to read.

```
STDIN      equ 0   ; standard input
SYS_read   equ 0   ; call code for read
inChar     db  0

mov rax, SYS_read
mov rdi, STDIN
mov rsi, inChar    ; msg address
mov rdx, 1         ; read count
syscall
```

System Service

Example: Console Input

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguageProgramming/code

File Edit View Search Terminal Help

```
global  _start
section .data
LF      equ     10
NULL    equ     0
EXIT_SUCCESS equ 0 ; success code
STDIN   equ     0
STDOUT  equ     1
STDERR  equ     2
SYS_read  equ   0
SYS_write equ   1
SYS_exit equ    60
msg     db      "Enter name: ", NULL
newLine db      LF, NULL

section .bss
buffer  resb    255     ;input buffer

section .text
_start:
        mov     rdi, msg        ;display message
        call    printString
        mov     rdi, STDIN      ;read from keyboard
        mov     rsi, buffer     ;save keyboard input to buffer
        mov     rax, SYS_read   ;call system read routine
        mov     rdx, 10         ;read 10 characters
        syscall

        mov     rsi, buffer     ;display buffer
        mov     rax, SYS_write
        mov     rdi, STDOUT
        mov     rdx, 10         ;print 10 characters
        syscall

        mov     rdi, newLine
        call    printString

        mov     rax, SYS_exit   ;exit
        mov     rdi, EXIT_SUCCESS
        syscall

                                                27,1-8        Top
```

System Service

Example: Console Input

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguageProgramming/code

File Edit View Search Terminal Help

```
global  _start
section .data
LF      equ     10
NULL    equ     0
EXIT_SUCCESS equ 0 ; success code
STDIN   equ     0
STDOUT  equ     1
STDERR  equ     2
SYS_read  equ   0
SYS_write equ   1
SYS_exit equ    60
msg     db      "Enter name: ", NULL
newLine db      LF, NULL

section .bss
buffer  resb    255     ;input buffer

section .text
_start:
        mov     rdi, msg        ;display message
        call    printString
        mov     rdi, STDIN      ;read from keyboard
        mov     rsi, buffer     ;save keyboard input to buffer
        mov     rax, SYS_read   ;call system read routine
        mov     rdx, 10         ;read 10 characters
        syscall

        mov     rsi, buffer     ;display buffer
        mov     rax, SYS_write
        mov     rdi, STDOUT
        mov     rdx, 10         ;print 10 characters
        syscall

        mov     rdi, newLine
        call    printString

        mov     rax, SYS_exit   ;exit
        mov     rdi, EXIT_SUCCESS
        syscall

                                                27,1-8        Top
```

System Service

File Open Operations

- To use file, it must be opened
- There are two file operations: open and open/create
- If the file open operation fails, an error code will be returned.
- If the file open operation succeeds, a file descriptor is returned

File Open

Open the existing file. This operation requires the parameter to specify the access mode.

- Read-Only Access → O_RDONLY (0)
- Write-Only Access → O_WRONLY (1)
- Read/Write Access → O_RDWR (2)

Register	SYS_open
rax	Call code = SYS_open (2)
rdi	Address of NULL terminated file name string
rsi	File access mode flag

After system call, if there is any error, RAX will contain negative value

System Service

File Open/Create

A file open/create operation will create a file. If the file does not exist, a new file will be created. If the file already exists, it will be erased and a new file created

Register	SYS_creat
rax	Call code = SYS_creat (85)
rdi	Address of NULL terminated file name string
rsi	File access mode flag

```
SYS_creat       equ 85          ; file open
O_CREAT         equ 0x40        ; mode
O_TRUNC         equ 0x200       ; mode
O_APPEND        equ 0x400       ; mode
S_IRUSR         equ 00400q      ; owner, read permission
S_IWUSR         equ 00200q      ; owner, write permission
S_IXUSR         equ 00100q      ; owner, execute permission
```

System Service

File Read

Register	SYS_read
rax	Call code = SYS_read (0)
rdi	File descriptor (of open file)
rsi	Address of where to place characters read
rdx	Count of characters to read

File Write

Register	SYS_write
rax	Call code = SYS_write (1)
rdi	File descriptor (of open file)
rsi	Address of where to place characters to write
rdx	Count of characters to write

***** File descriptor is an abstract unique number assigned to a file to keep track of file. The number is assigned by operating system**

System Service

Close File

After operation with files are done, files must be closed (write buffer to physical storage)

Registers	SYS_close
rax	Call code = SYS_close (3)
rdi	File descripter

System Service

Example: Open file for writing

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguageProgram...

File Edit View Search Terminal Help

```
section .data
; -----
; Define standard constants.
LF      equ 10  ; line feed
NULL    equ 0   ; end of string
TRUE    equ 1
FALSE   equ 0
EXIT_SUCCESS equ 0 ; success code
STDIN   equ 0   ; standard input
STDOUT  equ 1   ; standard output
STDERR  equ 2   ; standard error
SYS_read equ 0  ; read
SYS_write equ 1 ; write
SYS_open equ 2  ; file open
SYS_close equ 3 ; file close
SYS_fork equ 57 ; fork
SYS_exit equ 60 ; terminate
SYS_creat equ 85 ; file open/create
SYS_time equ 201 ; get time
O_CREAT equ 0x40
O_TRUNC equ 0x200
O_APPEND equ 0x400
O_RDONLY equ 000000q ; read only
O_WRONLY equ 000001q ; write only
O_RDWR equ 000002q ; read and write
S_IRUSR equ 00400q
S_IWUSR equ 00200q
S_IXUSR equ 00100q
; -----
; Variables for main.
newLine db LF, NULL
header  db LF, "File Write Example."
        db LF, LF, NULL
fileName db "url.txt", NULL
url     db "http://www.google.com"
        db LF, NULL
len     dq $-url-1
writeDone       db "Write Completed.", LF, NULL
fileDescrip     dq 0
errMsgOpen      db "Error opening file.", LF, NULL
errMsgWrite     db "Error writing to file.", LF, NULL
                                        1,1           Top
```

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguageProgramming/code

File Edit View Search Terminal Help

```
section .text
global _start
_start:
; Display header line...
        mov rdi, header
        call printString
; Attempt to open file.
openInputFile:
        mov rax, SYS_creat ; file open/create
        mov rdi, fileName ; file name string
        mov rsi, S_IRUSR | S_IWUSR ; allow read/write
        syscall ; call the kernel
        cmp rax, 0 ; check for success
        jl errorOnOpen
        mov qword [fileDescrip], rax ; save descriptor
; Write to file.
; if error -> rax < 0
; if success -> rax = count of characters actually read
        mov rax, SYS_write
        mov rdi, qword [fileDescrip]
        mov rsi, url
        mov rdx, qword [len]
        syscall
        cmp rax, 0
        jl errorOnWrite
        mov rdi, writeDone
        call printString
        jmp exampleDone
; Close the file.
        mov rax, SYS_close
        mov rdi, qword [fileDescrip]
        syscall
        jmp exampleDone
; Error on open.
errorOnOpen:
        mov rdi, errMsgOpen
        call printString
        jmp exampleDone
; Error on write.
errorOnWrite:
        mov rdi, errMsgWrite
        call printString
        jmp exampleDone
; Example program done.
exampleDone:
        mov rax, SYS_exit
        mov rdi, EXIT_SUCCESS
        syscall
; ***********************************************************
                                        43,1          56%
```

System Service

Example: Open file for reading

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguageProgram...

File Edit View Search Terminal Help

```
section .data
; -----
; Define standard constants.
LF       equ 10  ; line feed
NULL     equ 0   ; end of string
TRUE     equ 1
FALSE    equ 0
EXIT_SUCCESS equ 0 ; success code
STDIN    equ 0   ; standard input
STDOUT   equ 1   ; standard output
STDERR   equ 2   ; standard error
SYS_read equ 0  ; read
SYS_write equ 1 ; write
SYS_open equ 2  ; file open
SYS_close equ 3 ; file close
SYS_fork equ 57 ; fork
SYS_exit equ 60 ; terminate
SYS_creat equ 85 ; file open/create
SYS_time equ 201 ; get time
O_CREAT equ 0x40
O_TRUNC equ 0x200
O_APPEND equ 0x400
O_RDONLY equ 000000q ; read only
O_WRONLY equ 000001q ; write only
O_RDWR equ 000002q ; read and write
S_IRUSR equ 00400q
S_IWUSR equ 00200q
S_IXUSR equ 00100q
; -----
; Variables for main.
newLine db LF, NULL
header  db LF, "File Write Example."
        db LF, LF, NULL
fileName db "url.txt", NULL
url     db "http://www.google.com"
        db LF, NULL
len     dq $-url-1
writeDone       db "Write Completed.", LF, NULL
fileDescrip     dq 0
errMsgOpen      db "Error opening file.", LF, NULL
errMsgWrite     db "Error writing to file.", LF, NULL
                                        1,1           Top
```

watis@ThinkPad-E570 ~/Desktop/EN812700AssemblyLanguageProgramming/code

File Edit View Search Terminal Help

```
global _start
_start:
; Display header line...
        mov rdi, header
        call printString
; Attempt to open file - Use system service for file open
openInputFile:
        mov rax, SYS_open ; file open
        mov rdi, fileName ; file name string
        mov rsi, O_RDONLY ; read only access
        syscall ; call the kernel
        cmp rax, 0 ; check for success
        jl errorOnOpen
        mov qword [fileDescrip], rax ; save descriptor
; Read from file.
        mov rax, SYS_read
        mov rdi, qword [fileDescrip]
        mov rsi, readBuffer
        mov rdx, BUFF_SIZE
        syscall
        cmp rax, 0
        jl errorOnRead
; Print the buffer, add the NULL for the print string
        mov rsi, readBuffer
        mov byte [rsi+rax], NULL
        mov rdi, readBuffer
        call printString
        printNewLine
; Close the file.
        mov rax, SYS_close
        mov rdi, qword [fileDescrip]
        syscall
        jmp exampleDone
; Error on open.
errorOnOpen:
        mov rdi, errMsgOpen
        call printString
        jmp exampleDone
; Error on read.
errorOnRead:
        mov rdi, errMsgRead
        call printString
        jmp exampleDone
; Example program done.
exampleDone:
        mov rax, SYS_exit
        mov rdi, EXIT_SUCCESS
        syscall
; ********************************************************
                                        44,1          51%
```