Operating System **Hsu PO-CHUN**
10610 CS342302 Student ID: s105062803
Prof. Pai H. Chou Submitted: Oct. 01, 2017
TA: Due Date: Oct. 01, 2017

Assignment 2

1 Problem 1

[20 points] 2.7: What are the two models of interprocess communication? What are the strengths and weaknesses of the two approaches?

1.1 Answer

The two models of IPC are **message passing model** and **shared memory model**.

In message passing model, processes exchange data through a common mailbox. The main advantage of message passing model is message passing model no need to handle the conflicts and consistence. And message passing model is easier to implement than shared memory model as well. The disadvantage is message passing model can deal with only the small amount of data.

In shared memory model, processes use shared memory area to access for exchanging data. The advantage of shared memory model are shared memory model allowing maximum exchange data speed depending on memory access time, and can handle large amount of data. It is more convenience for IPC than message passing model as well. The disadvantage is shared memory model need additional mechanisms for protection and synchronization of the data between processes' IPC.

2 Problem 2

[20 points] 2.10: What is the main advantage of the microkernel approach to system design? How do user programs and system services interact in a microkernel architecture? What are the disadvantages of using the microkernel approach?

2.1 Answer

Advantage of the microkernel:

- With microkernel, it is easier to extend the OS because services are easier to be attached in the user space than the kernel space.
- With microkernel, it is easier to port from one architecture to another because the kernel is quite small.
- Services are running on the user space hence it provides relibility and protect the resources.
- Microkernel provides efficient IPC interface between user programs and services on the user space.

Interacting in microkernels:

User programs and system services are interacted with message passing.

For instance, a user application need to move a text file. It have to interact with the file server because hard drive cannot execute itself. Hence, user programs use message passing to communication with the file servers.

Disadvantage of the microkernel:
There are some overheads due to the communication with the microkernel between user programs and system services, so the performance would be decreased.

3 Section 2.3

[5 points] On line test/start.S:48, you have the MIPS assembly instruction
addiu $2, $0, SC_Halt
which is another way of saying
register2 = 0 + SC_Halt;
Explain the purpose of this assignment statement.

3.1 Section 2.3 Explain

To determine which system call to be executed in the exception handler
SC_Halt is the system call code defined in syscall.h. The ExceptionHandler routine read register 2 to execute the specific system calls in **exception.cc**:

```
int type = machine->ReadRegister(2);
```

Then the SyscallException use type to determine the cases:

```
case SyscallException:
        switch(type) {
            case SC_Halt:
```

4 Section 2.4.1

Answer the following questions for the Write() system call. Hint: the answers can be found in the source code of exception.cc, including the comments!

- How does the kernel get the pointer to the array of bytes to write? Give the description and C code.
- How does the kernel get the number of bytes to write? Give the description and C code.
- How does the kernel return the value to the caller? (for the number of bytes written?) Give the description and C code.

4.1 Section 2.4.1 Answer

```
int Write(char *buffer, int size, OpenFileId id);//defined in syscall.h
```

How does the kernel get the pointer to the array of bytes to write?
According to the standard C calling convention on the MIPS, arg1 is in r4, the kernel read the register 4 to obtain it:

```
kernel->machine->ReadRegister(4)//get the pointer of the array
```

How does the kernel get the number of bytes to write?
According to the standard C calling convention on the MIPS, arg2 is in r5, the kernel read the register 5 to obtain it:

```
kernel->machine->ReadRegister(5)//get the number of bytes to write
```

How does the kernel return the value to the caller?
According to the standard C calling convention on the MIPS, the return value is put in r2, the kernel write the register 2 to obey it:

```
//copy the size as its return value
kernel->machine->WriteRegister(2, kernel->machine->ReadRegister(5));
```