# Topic 1: Basic Concept

## Overview

### Guidance

Operating System Concepts 10th Edition.pdf chapter 1, chapter 20.1-20.3

Advanced Programming in the UNIX Environment, 3rd Edition.pdf 1.6 2.2 2.3

wiki:[History of Unix](https://en.wikipedia.org/wiki/History_of_Unix), [Linux distribution](https://en.wikipedia.org/wiki/Linux_distribution)

### Practice

1. Describe the following concepts

DMA, POSIX, Timer

1. How does the application program using the OS resources? What OS resources are there?
2. What types of operating systems exist? What are their characteristics?
3. What is the relationship between unix and linux? What is the relationship between redhat and linux? What is the defference between linux destribution and linux kernel?
4. What is the relationship between program and process?
5. Operating System Concepts 10th Edition.pdf chapter1 -> Practice Exercises 1.1
6. Operating System Concepts 10th Edition.pdf chapter1 -> Practice Exercises 1.3

## System Call

### Guidance

Operating System Concepts 10th Edition.pdf Chapter 2 System Calls

Advanced Programming in the UNIX Environment, 3rd Edition.pdf 1.11

ioctl dup

ulimit -a, What does these content mean? How to change it

### Practice

Write a program for each type of system call using the function in page 68.

If it can be done early by you, to explore more system calls and put the cases in your programs

## Process

### Guidance

Operating System Concepts 10th Edition.pdf 3.1-3.3

Operating Systems - Internals and Design Principles 7th.pdf 3.1-3.3

* What are similarity and difference between parent process and child process?
* ps/pstree
* Process state switch

### Practice

1. Run a program try to find it in pstree
2. What is the meaning of each filed of command ps -elf?
3. How many ways to create, suspense or terminate a process? Illustrate
4. Why do we need a PCB? What informations are stored in the PCB? What role does the PCB play when a process is created/scheduled/executed?
5. In what scenario switch between different states of a process？Illustrate

## Thread

### Guidance

Operating Systems - Internals and Design Principles 7th.pdf 4.1-4.2

Operating System Concepts 10th Edition.pdf 4.3-4.4

### Practice

1. For multiple threads in the same process, which resources are shared and which resources are unique?
2. What is the thread type created in Linux using pthread\_create? ULT or KLT?
3. What are the advantages and disadvantags for multi-process(only one thread in each process) and multi-thread do programming?
4. What is reentracy functions? What is non-reentrant functions? Illustration
5. Can multi thread share a routine function start\_routine? What should we pay attention to if we use it this way
6. How to make a thread never end?

# Topic 2: IPC

## Basic Concept

### Guidance

unix network programming Vol.2.pdf chapter 1-3

Advanced Programming in the UNIX Environment, 3rd Edition.pdf chapter 15

* The life cycle of the IPC object
* What is different between Posix and System V IPC?
* Can these ways be used for inter-thread communication

### Practice

1. According to your understanding, try to divide all IPCs into different categories, and the reason
2. To summarize, what are the characteristics of each IPC?
3. Which can be used for remote communication and which can be used for local communication. Can remote ways be used for local communication?

## Data Passing

### Guidance

unix network programming Vol.2.pdf part 2

Advanced Programming in the UNIX Environment, 3rd Edition.pdf 17

### Practice

1. Use fork/exec simulation executive "ping 127.0.0.1 -c 3" and get the result
2. Follow the client/server model, client sends a filename to server, response its content, implement by unix socket
3. Follow the producer/consumer model, producer sends message MSG1/MSG2/MSG3 to consumer, print the messages when the consumer received it. Give a sample for the 2 kinds of message queue separately

## Synchronization & Mutual Exclusion

### Guidance

unix network programming Vol.2.pdf 7 10 11

mutexes, condition variables, read-write locks, semaphores

### Practice

Use semaphores(system v or posix) to achieve synchronization and mutual exclusion respectively

1. Write a program synchronizes by semaphore
2. Write a program that is mutually exclusive by semaphore
3. Build several scenarios that generating deadlocks and implement them programmatically

## Remote Communication

### Guidance

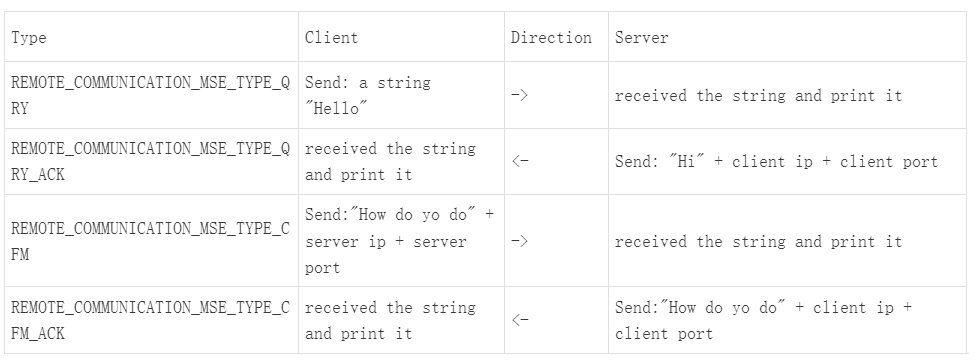
Advanced Programming in the UNIX Environment, 3rd Edition.pdf 16

### Practice

1. Implement a 1-to-1 client/server program based on the following requirements.

The python code has already provided as the client which run in windows

Implement the server program code by C which run in linux



1. Reconstruction question 2 on "Data Passing" make it can be usd for remote communication.

server in linux code by C

client in windows code by python

## Asynchronous Communication

### Guidance

Advanced Programming in the UNIX Environment, 3rd Edition.pdf 10

kill -l

### Practice

1. Summarize all signals

|  |  |  |  |
| --- | --- | --- | --- |
| Signal name | Signal number | Generation Mode | Generation scenario |
|  |  |  |  |

1. Explain the meaning of following conceptes

general signal, real-time signal, reliable signal, unreliable signal

1. Modify the default behavior of ctrl + c when ctrl + c is pressed, the output is Hello World! ; Use the kill command to achieve the same effect
2. Ignore the signal generated by ctrl + c and run while(1); How to terminate a process at this point; Provide another example of ignoring other signals
3. Which system calls can generate signals? For what scenarios? What signal is being generated? List at least 5
4. Implement sleep() and a timer using alarm() and pause() simulations. And wake up the thread in the sleep state in advance in the timer

# Topic 3: VxWorks Task

## Task

### Guidance

Vxworks\_programmers\_guide5.5.pdf 2.2

### Practice

Experiment

1. Create and run a task when switch boot
   1. Create and run a task named DEMO1 with stack size 1024 bytes and priority 128 in function os\_demo\_init. Print The task ID and name here
   2. The start routine of DEMO1 is os\_demo\_main\_process. There is a main loop in the function. It will be active interval 10s.
   3. There is a global variable named g\_int\_data. Which will be increase 1 and print the task ID and name every time when the main loop is actived.
2. Create and run a task when execute command
   1. Create and run a task named DEMO2 with stack size 1024 bytes and priority 128 in function os\_demo\_cmd\_impl\_test1 of file "os-demo-cmd-impl.c", when command "os-demo test1" be executed. Print The task ID and name here
   2. The start routine of DEMO2 is os\_demo\_cmd\_impl\_test1\_process. There is a loop in the function. It will be active interval 30s.
   3. print the task ID, name and the global variable g\_int\_data every time when the loop is actived.
3. Destory specify task
   1. Reconstruction function second\_os\_demo\_config\_test2 to accept a number
   2. Delete a specify task id which is prefixed with DEMO

Observation and Thinking

1. How many task names are there output? List out them
2. Is it possiable for DEMO1 and DEMO2 to access both the global variable g\_int\_data? Why?
3. What is the meaning of in following fields that display on "show task"?

NAME ENTRY TID PRI CPU invoked

1. Can the command "os-demo test1" be executed multi-times? using command "show task | include DEMO" to check
2. When deleting a task, what happens if the task does not exist? How to prevent this phenomenon from happening

## Inter Task Communication

### Guidance

Vxworks\_programmers\_guide5.5.pdf 2.3, 2.4

### Practice

If not specified, the default values for the tasks are as follows:

stack size is 1024 bytes, priority 128, No parameter

Experiment for message and event in different task

1. Messaage and Event receiving
   1. Create a message queue Which len is 20,as MSG\_Q\_FIFO mode
   2. Create and run a task named DMMSG,Which start routine is os\_demo\_msg\_recv and used for receiving message.
   3. Create and run a task named DMEVNT,Which start routine is os\_demo\_event\_recv and used for receiving event.
2. Messaage and Event sending
   1. Send a message by command line "os-demo msg "
   2. Send a event by command line "os-demo event "
3. Process demonstration

