Lab1 Hardware Overview and Basic Shell

HU Zhengdong 517370910249
September 9th, 2020

I. Hardware Overview

1. Locate the components

1.1 On the board

Figure 1. The motherboard

Figure 2. The power supply

Figure 3. The disk driver, optical driver and floppy disk driver together

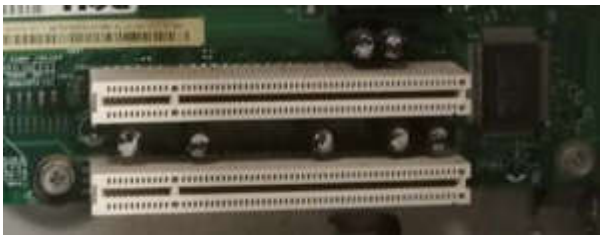

Figure 4. The PCI card

1.2 On the motherboard

Figure 5. The RAM

Figure 6. The North Bridge (down) and South Bridge (upper)

Figure 7. The PCI /PCI-e slot

Figure 8. The battery

Figure 9. The SATA socket

Figure 10. The CPU

Figure 11. The BIOS

2. Questions

- Where is the CPU hidden, and why?

CPU is under the fan, because CPU can generate much heat when running, requiring the fan to cool off.

- What are the North and South bridges?

The northbridge and South bridges are the two chips, or integrated circuits (ICs), within the chipset on the motherboard. The north bridge is also called by memory controller hub, handling the connections among CPU, RAM and south bridge, which requires the highest performance. The south bridge is also called by I/O controller hub, handling I/O communication with slow capabilities.

- How are the North and South bridges connected together?

The north bridge and south bridge are connected by a powerful bus, which sometimes is called a link channel:

- What is the BIOS?

BIOS stands for Basic Input Output System, is software stored on a small memory chip on the motherboard, to perform hardware initialization during the booting process, and to provide runtime services for operating systems and programs.

- Take out the CPU, rotate it and try to plug it back in a different position, is that working

The CPU will not work when it's inserted in a different position.

- Explain what overclocking is

Overclocking is the action of increasing a component's clock rate, running it at a higher speed than it was designed to run. This usually applies to the CPU or GPU, but other components can also be overclocked. Increasing a component's clock rate causes it to perform more operations per second, but it also produces additional heat, which requires additional cool and care.

- What are pins on a PCI/PCI-e card and what are they used for

The pins on the PCI/PCI-e card are Peripheral Component Interface Pinout, used to connect the PCI/PCI-e card to power supply and ground, and to transfer signals between PCI/PCI-e card and motherboard.

- Before PCI-e became a common standard, many graphics cards were using Accelerated Graphics Port (AGP), explain why

Accelerated Graphics Port refers to the actual expansion slot on the motherboard that accepts AGP video cards as well as to the types of video cards themselves. AGP replaces the slower Peripheral Component Interconnect (PCI) interfaces., because AGP provides a direct line of communication to the CPU and RAM, which in turns allows for quicker

rendering of graphics. Since PCI-e has better performance than AGP, AGP is replaced by PCI-e afterwards

II. Basic shell

- Use mkdir, touch, mv, cp, and ls commands

```
>touch test
>mkdir dir
>mv test dir/test.txt
>cp dir/test.txt dir/test_copy.txt
>ls dir -a
test_copy.txt test.txt
```

- Use the grep command

```
// -r read the directory recursively; -l only list the names of the files
>grep -rl "127.0.0.1" /etc
// -E extention
>grep -E "huzhengdong|root" /etc/passwd
```

- Use the find command

```
find /etc -atime -1
find /etc -name "*netw*"
```

- Use the redirection command

```
>: redirecting output
>>: appending redirected output
<<<: here strings, the string of the redirected input undergoes some operati
ons
%% &1 represents the standard output, &2 represents the standard error
>&1: redirect to standard output
2>&1 >: redirect the standdard error to the standard output, and then redire
ct the standard output
```

tee is the command to read from the standard input and write to the standard output and file

- Use the xargs command and the | sign

xargs is the command to execute command lines from standard input. The | sign pipes the output of the command on its left to the command on its right as the standard input.

- Use head and tail commands

The head and tail commands output the first part and the last part of each file to the standard output respectively. We can use the “tail -f” command to “lively play” a file as new lines are appended.

- Use ps, top, free, vmstat commands

ps is to report a snapshot of current process
top is to dynamically show the process in real-time.
free is to show the free and used memory in the system.
vmstat is to show virtual memory statistics, such as paging, block IO and cpu activity.

- What are the main differences between sh, bash, csh, and zsh?

sh (Bourne Shell) is the traditional Unix shells from the very early days, and its syntax is what is used in system initialization routines. The newer shells add some new functionality to the shell.

bash is souped-up version of sh with automated command completion and plenty of additional functionality.

csh adds a lot of interactive elements with which users could control their systems, like aliases, job management abilities, command history and more. It was modeled off the C programming language.

zsh is a Bourne-style shell that contains the features you'll find in bash, plus even more functionality, such as spell-checking, the ability to watch for logins/logouts and more features.

- What is the meaning of $0, $1,⋯, $?, $!

$0 is the name of the running shell script.
$1 is the first input.
⋯ is the line continuation operator
$? means exit the previous command
$! specifies the location of the shell

- What is the use of the PS3 variable? Provide a short code example

PS3: environment variable which contains the value of the prompt for the select operator inside the shell script.

```
1. #!/bin/bash
2. echo "please select a value to display a month on the list below"
3. select i in jan feb mar apr exit
4. do
```

```
5.      case $i in
6.         jan) echo "January";;
7.         feb) echo "February";;
8.         mar) echo "March";;
9.         exit) exit;;
10.     esac
11.done
```

The output for PS3 is “#!”.

- What is the purpose of the iconv command, and why is it useful?

iconv command is used to convert the text from one character encoding to another.
It’s useful because it can control the encoding format of the file.

- Given a variable $temp what is the effect of ${#temp}, ${temp%%word}, ${temp/pattern/string}.

${#temp} gives the length of the variable file
${temp%%word} extract partial contents from the character
${temp/pattern/string} extract the content by the pattern, and concatenate with the string.

- Search online (not on the man pages), how files are organized on a Unix like system. In particular. explain what are the following directories used for:

/: root directory which contains other files and directories
/bin: the directory where executable files are located, available to all users
/boot: the directory containing the boot system
/etc: the directory containing supervisor directory commands, configuration files, disk configuration files, valid user lists, groups, ethernet, hosts, where to send critical messages
/lib: the directory containing shared library files and sometimes other kernel-related files
/media: the directory where external storage will be automatically mounted when you plug it in and try to access it
/mnt: the directory used to mount other temporary file systems such as cdrom and floppy
/usr/bin: the directory containing the executable files that users would install and run
/usr/share: the directory containing the shared files
/usr/lib: the directory containing the library files for frequently used programs
/usr/src: the 'linux' sub-directory holds the Linux kernel sources, header-files and documentation.
/proc: the directory containing all processes marked as a file by process number
/sys: another virtual directory like /proc and /dev and also contains information about devices connected to your computer
/srv: the directory containing data for servers/
/opt: the directory where software you compile

- Write a game of guessing number

```
#! /bin/bash
i=$(($RANDOM % 100))
echo "Please guess the number"

while true
do
    read guess
    if [ $guess -gt $i ]
    then
        echo "Larger"
    elif [ $guess -lt $i ]
    then
        echo "Smaller"
    else
        break
    fi
done
```