

教育部与高通公司产学合作协同育人项目

燕山大学
机上作业报告

课程名称： 计算机操作系统

主题： 感受、体验和欣赏 OS 中的创新

题目： 多道 OS 安装、使用及应用分析和体验

| 姓名 | 学号 | 班级 | 自评成绩 |
|----|--------------|------|------|
| 乔翱 | 201811040809 | 软件六班 | A |

指导教师： 申利民

2020 年 09 月 13 日

www.wtosonline.com

目录

| | |
|---|----|
| 一、操作系统实战..... | 3 |
| 一、至少创建多种操作系统和硬件配置的虚拟机（虚拟出不同大小的内存，虚拟出单核和多核处理器，虚拟出不同大小的硬盘等） | 3 |
| 二、 多个程序在单 CPU（核）下顺序执行运行时间的实验，描述实验环境，记录时间，分析结果..... | 5 |
| 1、实验环境..... | 5 |
| 2、实验结果..... | 5 |
| 3、结果分析..... | 6 |
| 三、多个程序在单 CPU（核）下多道运行以及顺序执行的运行时间的实验，描述实验环境，记录时间，分析结果..... | 7 |
| 1、实验环境..... | 7 |
| 2、实验结果..... | 7 |
| 3、结果分析..... | 9 |
| 四、体验不同 OS 下的控制接口和操作方式：CUI、GUI、批处理，描述实验环境，记录操作内容和结果，进行对比..... | 10 |
| 1、实验环境..... | 10 |
| 2、实验内容..... | 10 |
| 3、实验结果..... | 13 |
| 五、完成和体验不同操作系统下，相同设备驱动程序安装方法，描述实验环境，记录操作内容和结果，进行对比..... | 14 |
| 1、实验环境..... | 14 |
| 2、实验内容..... | 14 |
| 3、实验结论..... | 16 |
| 六、 说明 VMware 的用途和作用，猜想一下 VMware 的实现方法..... | 17 |
| 1、用途和作用..... | 17 |
| 2、实现方法..... | 17 |
| 二、 操作系统问答题..... | 18 |
| 一、举例说明，OS 由需要转换为计算机系统中必需的软件..... | 18 |
| 二、举例说明，在现代计算机系统只有被 OS 管理和控制的资源才能被用户使用..... | 18 |
| 三、说明多用户分时系统是如何解决交互性与资源利用率之间的矛盾，说明其创新点 | 18 |
| 四、举例说明，OS 的生态环境对计算机系统 OS 生存与发展的影响..... | 18 |

一、操作系统实战

一、至少创建多种操作系统和硬件配置的虚拟机（虚拟出不同大小的内存，虚拟出单核和多核处理器，虚拟出不同大小的硬盘等）

根据要求，在 VMware 中共创建了 Linux (Ubuntu)、Windows (windows7)、MacOS 三种操作系统。

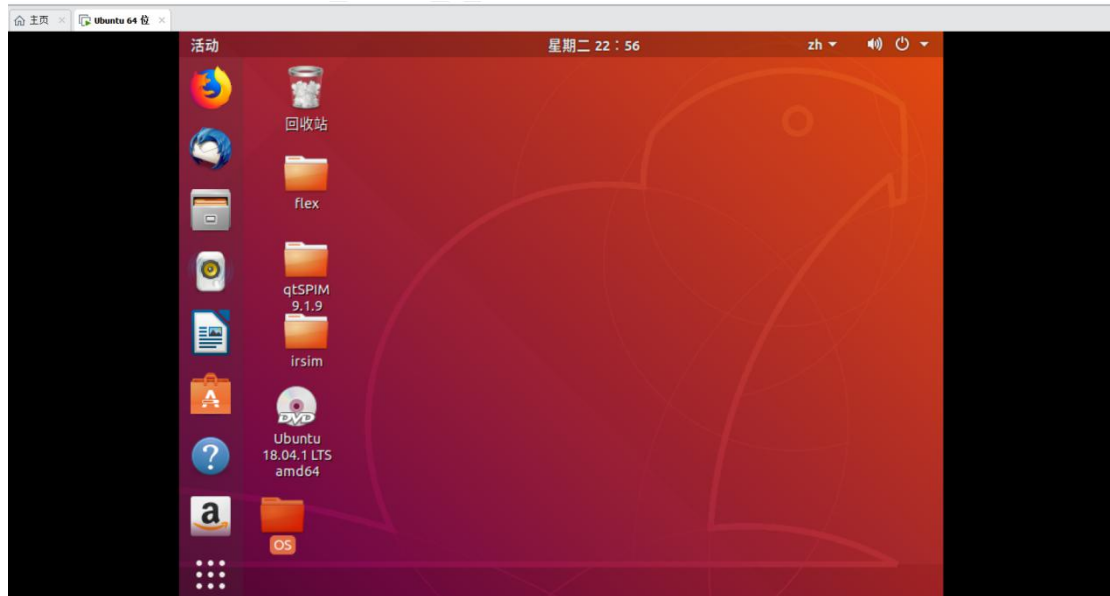


图 1.1.1 Ubuntu 操作系统

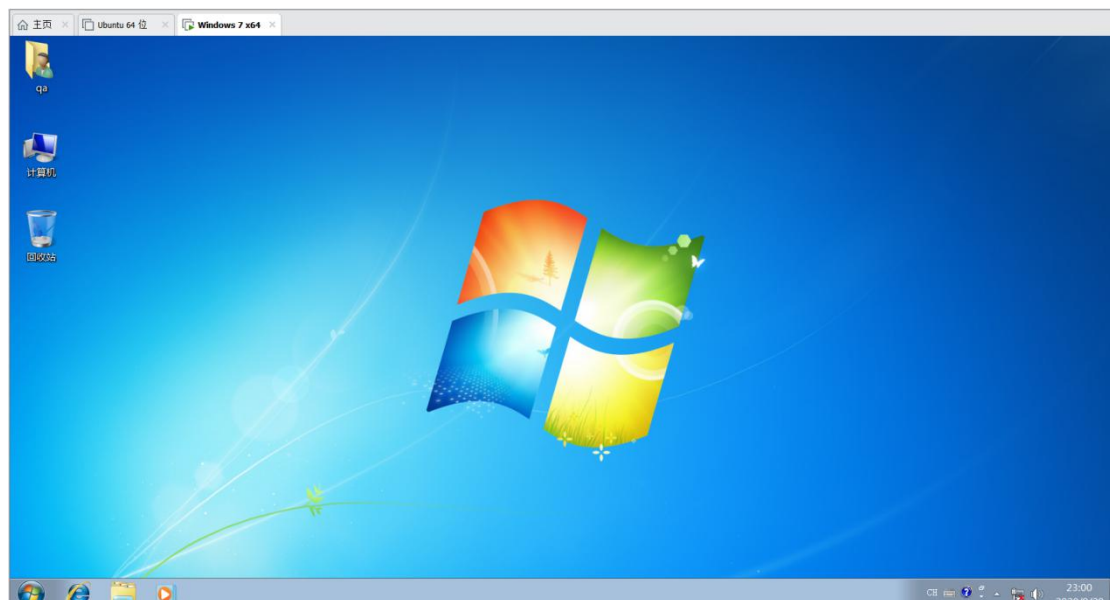


图 1.1.2 Windows7 操作系统



图 1.1.3 MacOS 操作系统

根据要求，在 VMware 中对三种不同的硬件系统进行了不同的硬件配置，配置了不同大小的内存，虚拟出单核和多核处理器，虚拟出不同大小的硬盘。

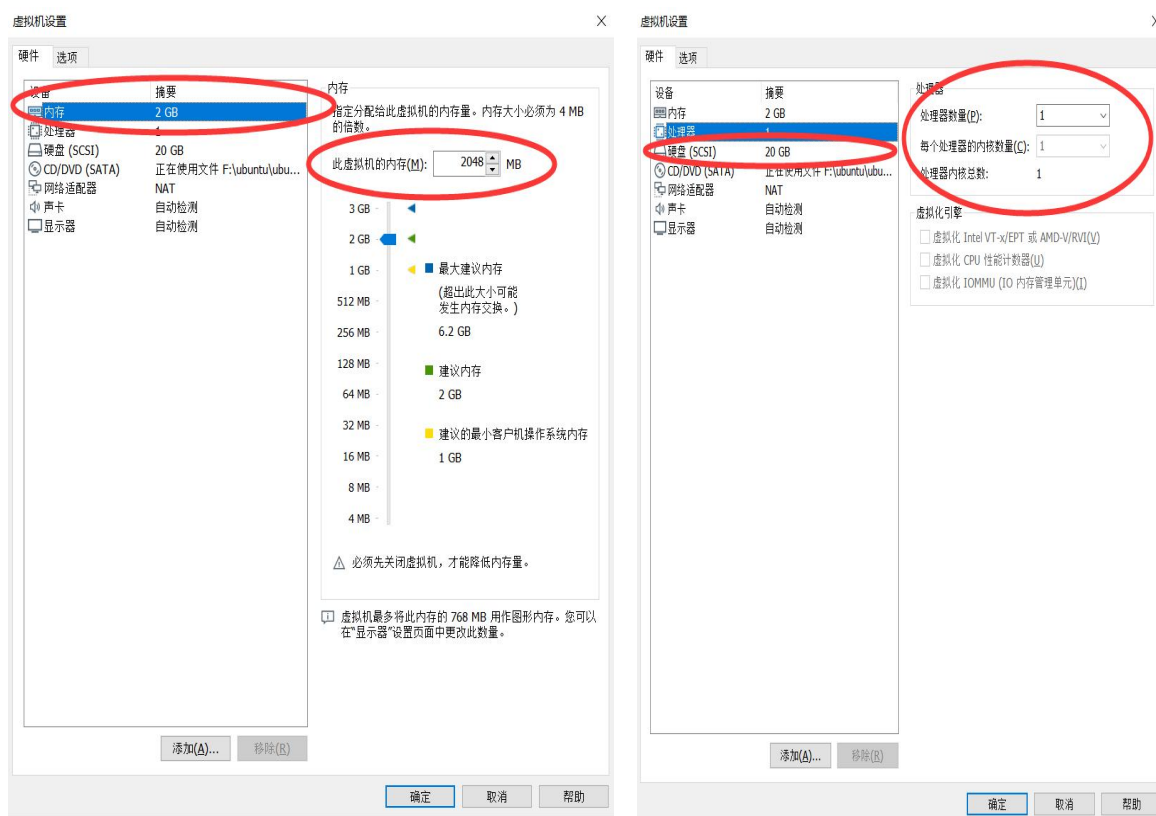


图 1.1.4 不同的硬件配置

二、多个程序在单 CPU（核）下顺序执行运行时间的实验，描述实验环境，记录时间，分析结果

1、实验环境

主要采用 c++进行单线程和多线程程序的编写，实验环境如下：

表 1.2.1 实验环境

| 操作系统 | 编译环境 | 内存容量 | 处理器核数 | 硬盘容量 |
|---------|---------|---------|-------|------|
| Windows | Dev c++ | 2048 MB | 1 | 60GB |
| Ubuntu | G++ | 2048 MB | 1 | 60GB |
| MacOS | G++ | 2048 MB | 1 | 60GB |

2、实验结果

对于实验结果采用折线图的形式表达出来，更容易分析实验结果得出实验结论。横轴是程序运行次数，纵轴是程序运行时间，单位是秒（s）。

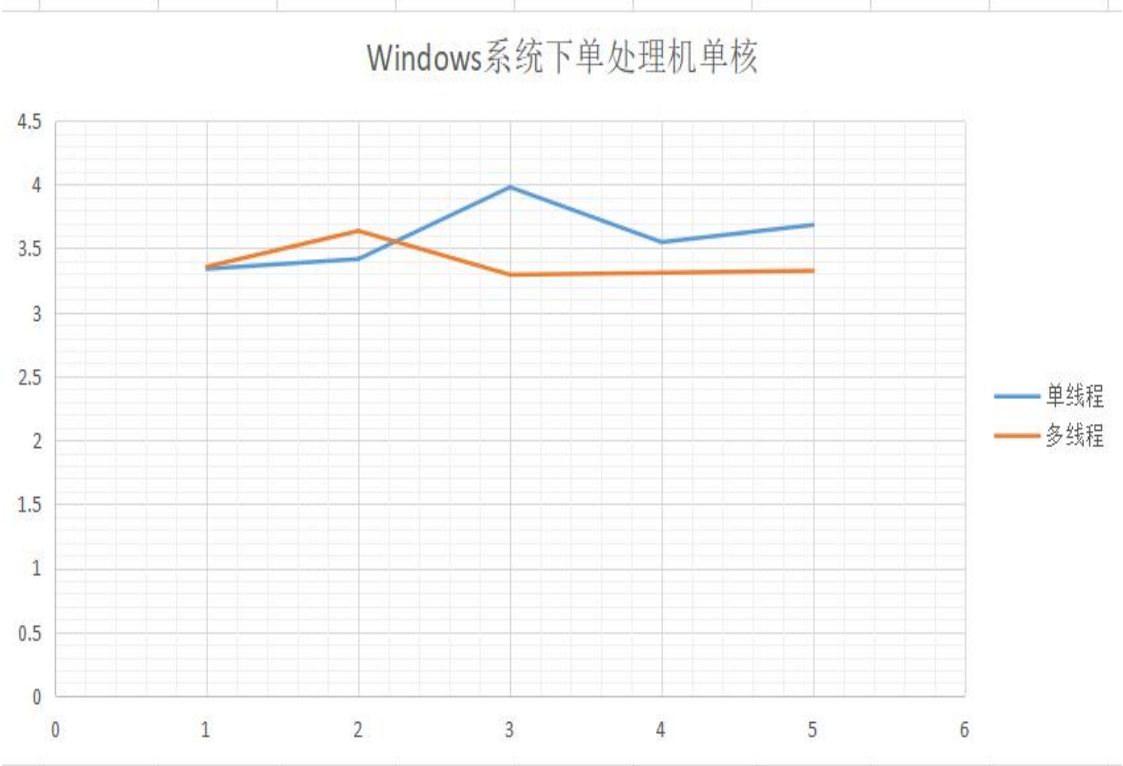


图 1.2.1 Windows 单处理机单核实验运行结果

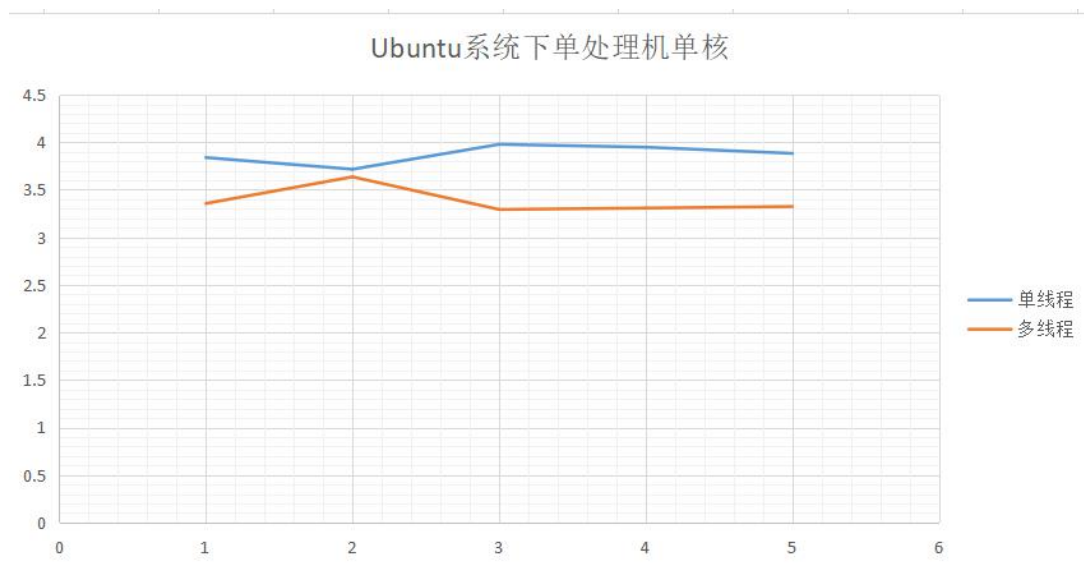


图 1.2.2 Ubuntu 单处理机单核实验运行结果

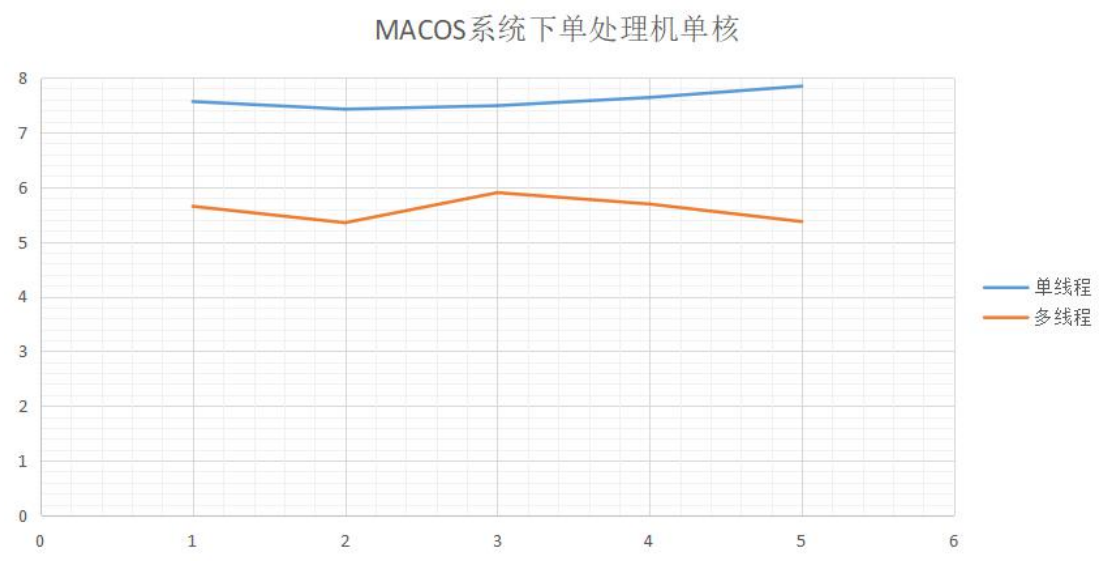


图 1.2.3 Ubuntu 单处理机单核实验运行结果

3、结果分析

通过分析所有图可以发现，不管是单线程还是多线程，在多次运行时，每一次运行的时间不是完全一样的，但是上下波动比较小，运行时间大致一样。

分析以上两个图可以看出，多线程程序的运行时间稍少于单线程。但是有时候多线程时间会多于单线程，可能是因为在单核情况下，线程调度也会花费一定的时间所导致的。

三、多个程序在单 CPU（核）下多道运行以及顺序执行的运行时间的实验，描述实验环境，记录时间，分析结果

1、实验环境

主要采用 c++进行单线程和多线程程序的编写，实验环境如下：

表 1. 3. 1 实验环境

| 操作系统 | 编译环境 | 内存容量 | 处理器核数 | 硬盘容量 |
|---------|---------|---------|-------|------|
| Windows | Dev c++ | 2048 MB | 1 | 60GB |
| Ubuntu | G++ | 2048 MB | 1 | 60GB |
| MacOS | G++ | 2048 MB | 1 | 60GB |

2、实验结果

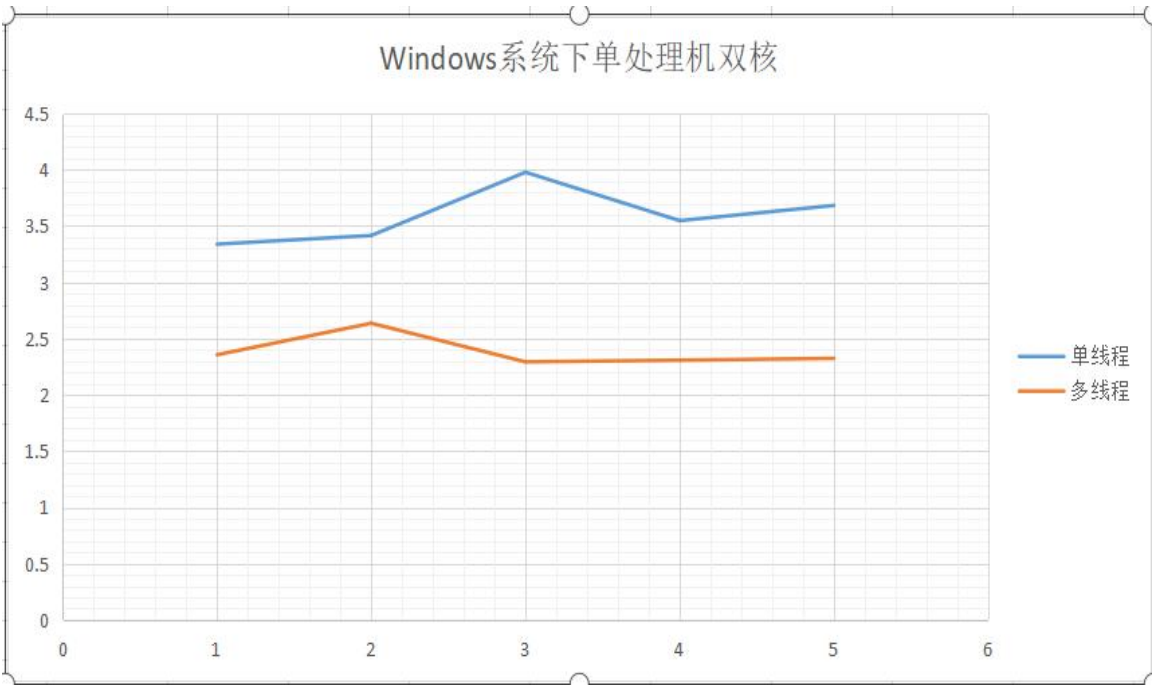


图 1.3.1 Windows 系统下单处理双核实验运行结果

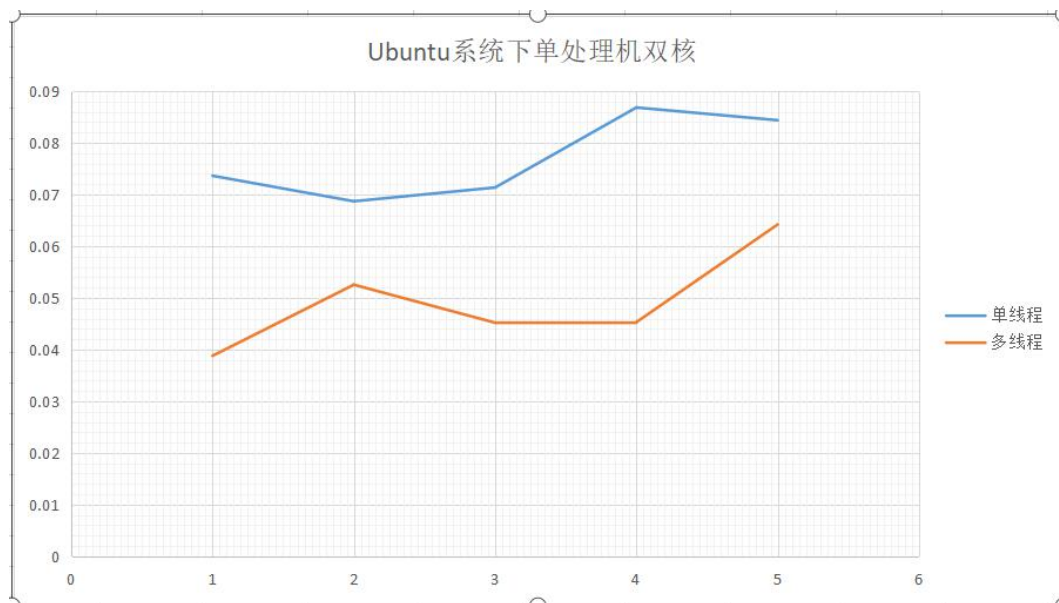


图 1.3.2 Ubuntu 系统下单处理双核实验运行结果

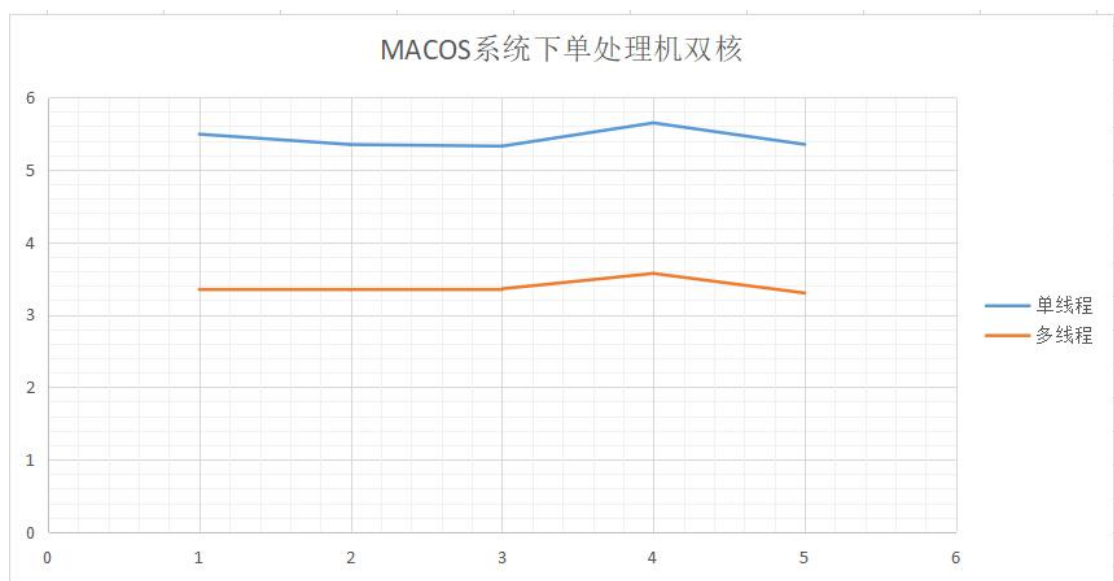


图 1.3.3 MacOS 系统下单处理双核实验运行结果

3、结果分析

通过以上实验结果可以看出来，多线程程序在多核下运行时间小于单线程在多核运行，可以得出，多线程在运行时由于两个线程并行执行，所以节省看时间。还可以看出多线程程序和单线程程序在多次运行时，每一次运行时间不是完全一样的，但是上下波动比较小。

四、体验不同 OS 下的控制接口和操作方式：CUI、GUI、批处理，描述实验环境，记录操作内容和结果，进行对比

1、实验环境

表 1.4.1 实验环境

| 操作系统 | 编译环境 | 内存容量 | 处理器核数 | 硬盘容量 |
|---------|---------|---------|-------|------|
| Windows | Dev c++ | 2048 MB | 1 | 60GB |
| Ubuntu | G++ | 2048 MB | 1 | 60GB |
| MacOS | G++ | 2048 MB | 1 | 60GB |

2、实验内容

2.1 Windows

2.1.1 CUI:

第一步：在 DOS 下进入这个目录

```
C:\Users\qa>cd C:\Users\qa\Desktop
```

第二步：用 g++加上文件名进行编译，会生成一个 a.exe 文件

```
C:\Users\qa\Desktop>g++ 1.cpp
```

第三步：直接输入 a.exe 并且回车，即可执行编译后的文件

```
C:\Users\qa\Desktop>a.exe  
hello word
```

2. 1. 2 GUI

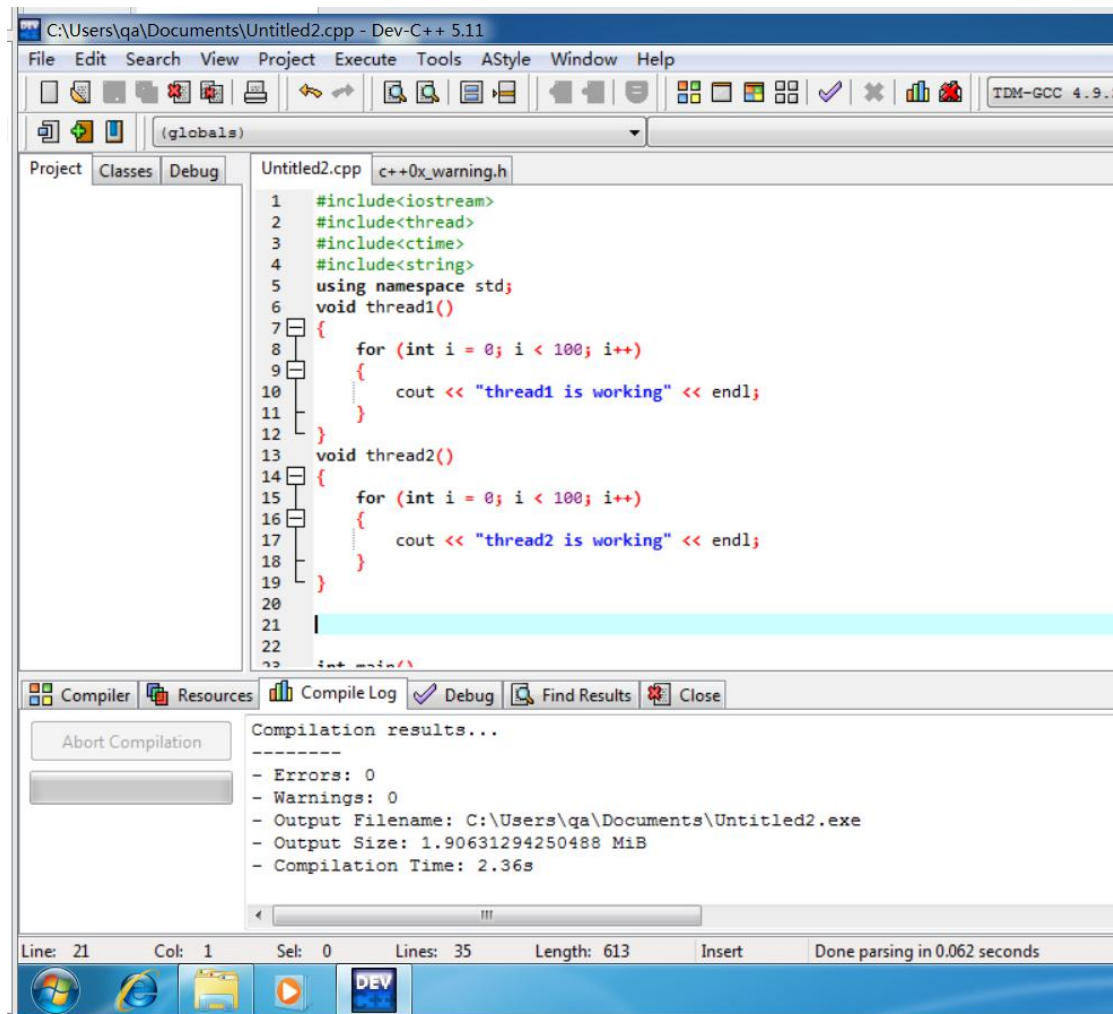


图 1.4.1 Windows GUI

2. 1. 3 批处理

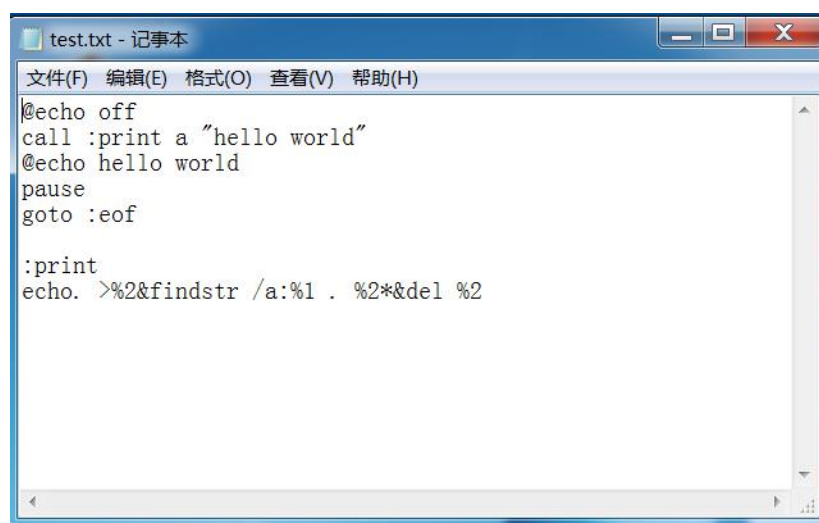


图 1.4.2 Windows 批处理程序

2.2.2 GUI

直接通过 dev 或者 clion 编写程序并且编译运行，与在 windows 中运行基本一致，不再演示


2.2.3 批处理

第一步：用 `ls -l hello.sh` 可以查看是否可执行，如果该文件不可以执行，要先执行语句 `chmod +x hello.sh` 授予执行权限

A terminal window titled '终端' (Terminal) with a dark background. The prompt is 'qa@qa-virtual-machine: ~/桌面/OS'. The user enters 'ls -l hello.sh' and the output is '-rwxrwxrwx 1 qa qa 56 9月 30 17:29 hello.sh'. Then the user enters 'chmod +x hello.sh' and the prompt returns.

```
qa@qa-virtual-machine: ~/桌面/OS
qa@qa-virtual-machine:~/桌面/OS$ ls -l hello.sh
-rwxrwxrwx 1 qa qa 56 9月 30 17:29 hello.sh
qa@qa-virtual-machine:~/桌面/OS$ chmod +x hello.sh
qa@qa-virtual-machine:~/桌面/OS$
```

第二步：输入 `./hello.sh` 运行程序

A small snippet of the terminal showing the command './hello.sh' being entered at the prompt.

```
qa@qa-virtual-machine:~/桌面/OS$ ./hello.sh
```

3、实验结果

通过体验不同 OS 下的控制接口和操作方式：CUI、GUI、批处理，发现在 Ubuntu 和 MacOS 下，两种系统的使用方法基本一样，没有太大差别，对于 Windows 操作系统和其他两种操作系统的使用方法略有不同。

三个系统都可以通过 CUI 在终端进行控制，也可以通过 GUI 进行控制，还可以通过批处理的方式执行多条指令。GUI 的方式可能对于一般用户操作起来更加友好，但是其他方式也是很方便操作的。

五、完成和体验不同操作系统下，相同设备驱动程序安装方法，描述实验环境，记录操作内容和结果，进行对比

1、实验环境

表 1. 5. 1 实验环境

| 操作系统 | 编译环境 | 内存容量 | 处理器核数 | 硬盘容量 |
|---------|---------|---------|-------|------|
| Windows | Dev c++ | 2048 MB | 1 | 60GB |
| Ubuntu | G++ | 2048 MB | 1 | 60GB |
| MacOS | G++ | 2048 MB | 1 | 60GB |

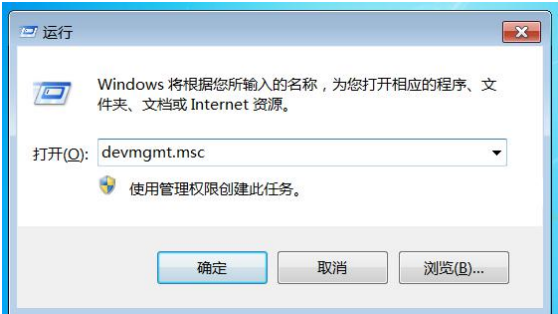
2、实验内容

2. 1 Ubuntu、MacOS:

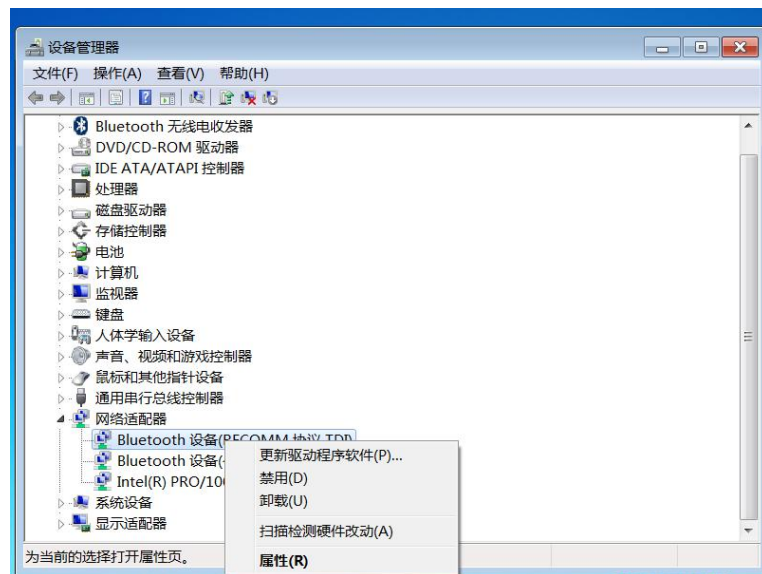
- 第一步：检查是否已安装驱动程序
- 在终端输入\$ lspci 或者\$ grep，利用\$ dmesg 命令可以显示内核识别的所有设备驱动程序。
- 第二步：删除现有存储库（如果存在）
- ```
$ sudo apt-get purge NAME_OF_DRIVER *
```
- 其中NAME\_OF\_DRIVER 是驱动程序的可能名称。还可以将模式匹配添加到正则表达式中以进一步过滤。
- 第三步：将存储库添加到重新研磨器，应在驱动程序指南中指定
- ```
$ sudo add-apt-repository REPOLIST_OF_DRIVER
```
- 应该从驱动程序文档（例如 epel-list）中指定 REPOLIST_OF_DRIVER 。
- 第四步：更新存储库列表。
- ```
$ sudo apt-get update
```
- 第五步：安装软件包。
- ```
$ sudo apt-get install NAME_OF_DRIVER
```
- 第六步：检查安装。
- 运行 lspci 命令（如上所述）以检查驱动程序是否已成功安装。

2. 2 Windows:

- 第一步：点击开始，点击运行，输入 devmgmt.msc，点击确定



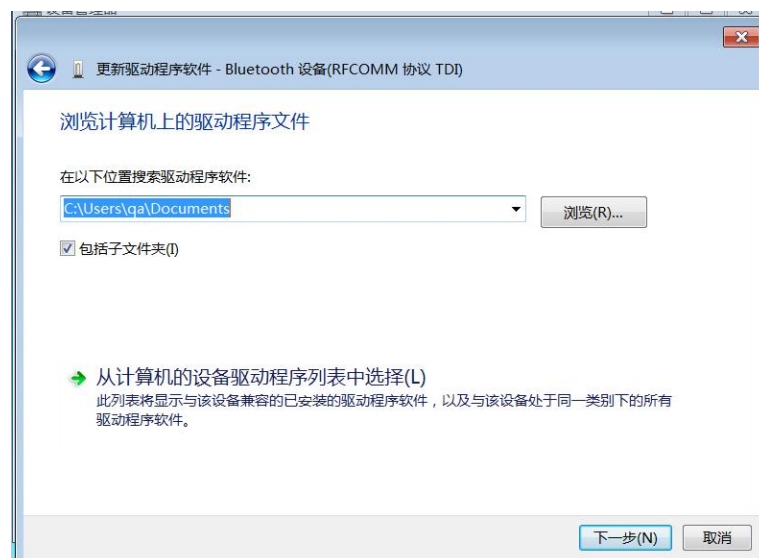
第二步：右击要手动安装驱动的设备，点击更新驱动程序



第三步：点击浏览计算机以查找驱动程序软件



第四步：点击浏览按钮选定要安装的驱动所在位置，也可以手动输入，点击下一步；



第五步：后续步骤与正常安装没有区别，不再赘述

3、实验结论

对于不同系统安装驱动程序的操作大同小异,对于 Ubuntu 或者 MacOS 来说,安装驱动程序需要在终端中输入一些命令来安装,而对于 Windows 来说,安装驱动程序是通过 GUI 界面交互操作实现的,对于平常大多数使用 Windows 用户来说,GUI 的方式可能更方便一点。但是 Ubuntu 和 MacOS 安装驱动程序的方式也很简单。

六、说明 VMware 的用途和作用，猜想一下 VMware 的实现方法

1、用途和作用

利用虚拟机可以在一台电脑上将硬盘和内存的一部分拿出来虚拟出若干台机器，每台机器可以运行单独的操作系统而互不干扰，这些“新”机器各自拥有自己独立的硬盘和操作系统，可以像使用普通机器一样对它们进行分区、格式化、安装系统和应用软件等操作，还可以将这几个操作系统联成一个网络。在虚拟系统崩溃之后可直接删除不影响本机系统，同样本机系统崩溃后也不影响虚拟系统，可以下次重装后再加入以前做的虚拟系统。同时它也是唯一的能在 Windows 和 Linux 主机平台上运行的虚拟计算机软件。虚拟机软件不需要重开机，就能在同一台电脑使用好几个 OS，不但方便，而且安全。虚拟机在学习技术方面能够发挥很大的作用。

VMware 可以在一个窗口里来管理运行的操作系统和应用程序，也可以在运行于桌面上的多台虚拟机之间进行切换。还可以通过网络来实现虚拟机的共享。

2、实现方法

虚拟机是通过软件来模拟出整个计算机的硬件系统。通过虚拟机可以模拟出不同的操作系统。虚拟机通过虚拟化实现了其功能，虚拟出了不同的 OS。

对于内存来说，内存的虚拟化我觉得是从虚拟内存到物理内存的映射，而这里的物理内存也是虚拟的。所以应该就是虚拟内存到虚拟的物理内存再到真正的物理内存，而真正的物理内存是宿主机上的内存空间，宿主机为虚拟的内存分配的部分内存空间。

虚拟机对于 CPU 的虚拟化，我觉得是通过对虚拟的寄存器进行修改来进行的，因为 CPU 的状态是存储在寄存器中的，虚拟机通过对存储 CPU 状态的寄存器进行修改来进行指令的执行。

综上所述，虚拟机对于一个计算机的实现是通过虚拟化来实现的，对于各个部件都进行了虚拟化，来实现一台独立的计算机。

二、操作系统问答题

一、举例说明，OS 由需要转换为计算机系统中必需的软件

在早期没有操作系统的年代，用户对于计算机的操作特别不方便，一般只有专业的计算机专家才能够使用计算机，用户需要直接管理硬件，费时费力。

而操作系统的出现极大的方便了用户，操作系统来管理计算机硬件与软件资源，操作系统来管理和分配内存，决定系统资源供需的优先次序，控制输入设备与输出设备，操作网络与管理文件系统等基本事务。操作系统在计算机程序的辅助下，可以抽象处理计算系统资源提供的各项基础职能，以可视化的手段来向使用者展示操作系统功能，减低计算机的使用难度。

操作系统还提供了图形化界面，用户可以更好更方便的使用计算机。如果没有操作系统，用户使用计算机，开发软件都会特别的不方便，特别困难，就会很少人使用计算机了，所以说，操作系统由需要转换为计算机系统中必需的软件。

二、举例说明，在现代计算机系统只有被 OS 管理和控制的资源才能被用户使用

在没有操作系统的时候，计算机系统的资源完全由用户和用户程序来控制和管理，但是用户非常不方便。又来操作系统后，计算机系统的资源由操作系统控制和管理，用户通过操作系统的服务接口使用这些资源。如果操作系统没有控制和管理这些某些资源，用户就不能通过操作系统的服务获得这些资源的使用。例如，DOS 只能管理 1M 的内存，硬件上装上再多的内存一般用户也无法使用。

三、说明多用户分时系统是如何解决交互性与资源利用率之间的矛盾，说明其创新点

多道批处理系统已经大大地提高了计算机的资源利用率，但是它的致命缺点是缺少交互性。资源利用率和交互性是一对矛盾。如果一台计算机能够连接多个控制台，允许多个用户同时在控制台上进行操作，每个控制台上的用户执行一个程序，形成多个程序并发执行。通过并发程序的分时执行，确保每个用户操作的计算机终端就好像单独的一台计算机。这样就避免了只有一个操作台时，大量的计算机时间被一个用户浪费，同时又克服了多道批处理系统非交互性的缺点。

创新点在于多用户分时系统解决了交互性的问题，用户的请求可以在短时间内得到响应，多个程序并发执行，每个用户操作的终端好像单独的一个计算机一样，既提高了资源利用率，也增加了交互性。

四、举例说明，OS 的生态环境对计算机系统 OS 生存与发展的影响

一个操作的系统的生态环境对于其自身的推广有着很大的影响，只有一个操作系统的生态环境良好，才会有大量的用户去使用该操作系统。

在我看来一个操作系统的生态大致包括这几个方面：应用软件数量和质量、与周围设备的兼容性、体系内硬件类型、对新生设备和原有设备的联系。

对于微软来说，移动设备是一个很大的短板，Windows 的一个缺点就是应用软件比较差，没有足够的应用程序，无法形成一个完整的生态体系。和 iOS 相比，Windows 系统的生态系统过于匮乏，微软的系统架构变更过于频繁，导致了很多程序员不愿意为微软进行移动端软件的开发。

而对于目前国内的操作系统，主要是基于 Linux，可以直接利用 Linux 的软件生态，对于国内操作系统的发展道路还是任重道远的，只有真正掌握了操作系统的核心技术，才能打破美国的垄断，不受制于人。发展一个操作系统需要投入大量的人力物力。