

Introduction

I am receiving mails from final year students who want to write an OS as their academic project, since most of their mails are almost same and my replies also same, I decided to create a page for that and this is that. Assume you asked the following questions – How many months it will take to create an OS? What book I have to learn to create an OS?...

Knowledge Required

The following are some of the important area where you should be comfortable. It is not necessary to know in and out of everything. For example – it is not necessary to know all the POSIX C library functions and their arguments; however you should through in core C language.

- C and Assembly (Yes – Assembly knowledge is must)
- Data structures (Array, Linked List, Binary Tree, Hash...)
- Compiler, Linker, Object Files, Executable Files
- Processor Modes – Real and Protected
- Knowledge in BIOS, Interrupt, ISR, DMA, IO etc
- Basic Hardware Knowledge
- OS Concepts

Writing an OS is not an easy task.

If you have knowledge about the above thing then it will take just a year to complete an OS project. :) You should read the following article by Alexei A. Frounze to understand how hard it is, what kind of knowledge needed and the time needed for development of an OS

<http://alexfru.narod.ru/os/skeptically/OSdevSkeptically.html>

Another one from Emmanuel Marty

http://osnews.com/story.php?news_id=1482

<http://www.samueldotj.com/Articles.asp>

Impossible?

No I didn't say that; although developing OS is not an easy task, it is not an impossible one. OS development may take more than what you estimate however it is also possible to complete a working kernel within short frame of time (6 months) by group of students. Once I amazed when I came to know that four Engineering students from some University in Thanjur, Tamilnadu, India has created a kernel as their final project.

Goal=?

OS is composed of many components or modules. Writing all the modules from scratch will take lot of time (decade(s)). And also all modules are not required for all OSes. So decide what kind of OS you want to write. GUI and Shell are some example for programs which are not needed for an OS. Memory Management, Process Management and Scheduler are the important things; around which you may set your goals. Eg – Some new kind of IPC, Special Scheduler, VM.

Design is the first task to be done for any task

Spend some time to finalize your kernel design before starting coding. Set all modules's functions, data structure and how they are going to interact/communicate with other modules before start coding.

Kernel is the most important part in OS design. You may find some links related to kernel design from the following link http://en.wikipedia.org/wiki/Kernel_%28computer_science%29.

Set Environment

What makes OS development harder? – The answer is development/debugging environment. Before start coding but after design set your development environment well.

GCC and NASM/FASM is compiler/assembler used by most hobby OS developers. Try to learn the options supported by these compilers and assemblers. You should also learn how to link Assembly and C source code. Also you may need to learn about MakeFile and linker scripts. You can find most of the tools in the following link <http://www.samuel-dot-j.com/Ace/Links.asp>

Although I am using Windows XP for my development, I suggest using Linux as the host machine for OS development because Linux has lot of open source tools related to kernel development. Although Windows port of those tools are available, those are not up to date or they are broken. Linux comes with all the tools that you want to develop an OS; you just have to select them during the installation.

<http://www.samuel-dot-j.com/Articles.asp>

Debugging

To test and debug your kernel, you need either a real machine or a virtual machine (Virtual PC, VMWARE, Bochs, QEmu). WinImage, FileDisk are some tools help to create image files for virtual machines. You also need to know how to debug your kernel - <http://www.samueldotj.com/Articles.asp>

Management

For a hobby OS developer technical challenge and research is the only thing in custom OS development. If you are creating a kernel for your final year project you should take care of time management also otherwise you can't complete the project. The classic technique of time management is set timeframe for each activity and to complete it within the timeframe. Here are some macro level milestones. (The percentages are of total time within which the objectives should complete.)

- Design (25%) (Reading – 15%, Design – 10%)
- Tools and environment Setup (5%)
- Grub and Hello World Kernel (15%)
- Create Real Kernel (30%)
- Debugging (20%)

Reading

Intel Processor Manual 3 – System Programming

<http://developer.intel.com/design/pentium4/manuals/253668.htm>

Art of Assembly

<http://webster.cs.ucr.edu/AoA/DOS/index.html>

OS Development FAQ

<http://www.osdev.org/osfaq2/>

BonaFide OS Development Documents

<http://www.osdever.net/documents.php?cat=0&sort=1>

Alexei A. Frounze OS Tutorial

<http://alexfru.chat.ru/epm.html>

Alt.OS.Development

<http://groups.google.ca/groups?q=alt.os.development>

OS Development Forum

<http://www.osdev.org/phpBB2/>

File System Specifications

<http://www.forensics.nl/filesystems>

Links

<http://www.samueldotj.com/Ace/Specifications.asp>

<http://www.samueldotj.com/Ace/Links.asp>

<http://www.sunset.dsl.pipex.com/links.html>

Good luck