Uinxed-Kernel Project Outline

Hello! This project, Uinxed, is a Unix-like operating system kernel developed from scratch by our team, aiming to follow modern computer architecture and advanced system design concepts. We are committed to building an efficient, stable and scalable kernel while maintaining code clarity and maintainability to meet the needs of different users and developers.

To compile the Uinxed project, you first need to prepare a suitable development environment. This project supports compilation on Unix-like operating systems, covering common Linux distributions such as Debian, Ubuntu, Kali, ArchLinux, and Alpine. Before starting to compile, make sure to install the following tools: Make, GCC (recommended version is 13.3.0 and above), QEMU, Xorriso, clang-format, and clang-tidy. These tools are used for project building, code compilation, simulation testing, ISO image file construction, code formatting, and static analysis.

After completing the environment preparation, you can follow the steps below to compile the project. First, clone the project code to your local computer by executing the “git clone https://github.com/ViudiraTech/Uinxed-Kernel.git/” command, and switch to the project root directory, i.e. “cd Uinxed-Kernel”. Next, run the “make” command to start the compilation process. Our Makefile has been carefully configured for you, covering all aspects of project compilation to ensure that you can successfully generate kernel files.

Makefile is the core build script of this project. It is not only responsible for organizing and managing the compilation process of the project, but also provides a variety of practical functions to meet the development needs in different scenarios. Makefile has a core compilation function. By defining a series of compilation rules and dependencies, it can automatically detect changes in source code files and recompile only the changed parts, thereby greatly improving compilation efficiency. In the project, Makefile will call the GCC compiler to compile all C language source files (.c) according to predefined compilation options and parameters to generate corresponding target files (.o). At the same time, it will also process assembly language source files (.s), assemble them into target files, and finally link all target files together to generate the kernel file UxImage. In addition to basic compilation functions, Makefile also supports the generation of bootable ISO image files. By executing the “make” command, Makefile will not only compile the kernel, but also automatically call the xorriso tool to package the kernel file and related files such as the boot program into an ISO image file named Uinxed-x64.iso. This image file can be used for testing and running on the QEMU virtual machine or other hardware platforms that support UEFI boot. To facilitate your testing of the kernel in a virtual machine environment, Makefile provides a run target. Simply execute the “make run” command, and Makefile will use the QEMU simulator to load and run the previously generated ISO image file, quickly start the Uinxed kernel, and allow you to debug and verify without actual hardware. In addition, Makefile also integrates code formatting and static analysis functions. With the clang-format tool, you can format the C language and assembly language source code in the project by running the “make format” command to ensure the consistency of the code style. At the same time, using the clang-tidy tool, executing the “make check” command can perform static analysis on the code to help you find potential code quality issues and security vulnerabilities.

Testing the Uinxed kernel in a virtual machine environment is the most convenient and safe way. We recommend using the QEMU virtual machine emulator, which is an open source and powerful virtualization tool that supports multiple hardware platforms and operating system architectures. After completing the project compilation and generating the ISO image file, you only need to execute the “make run” command in the project root directory. This command automatically calls QEMU and loads the Uinxed-x64.iso image file for startup. During the QEMU startup process, you will see the kernel's boot information and initialization process. If the kernel can start and run normally, congratulations, this indicates that your compilation and build work is successful. In the virtual machine, you can test and verify the various functions of the kernel, such as checking whether the kernel can correctly initialize the hardware device, whether it can handle interrupts and exceptions normally, and so on.

When you want to run the Uinxed kernel on actual hardware, some additional preparation is required. First, you need to write the kernel file to UEFI boot storage media, such as a USB flash drive or hard drive. First, connect the USB flash drive or hard drive to your computer and convert its partition table format to GPT. Simultaneously, create a FAT32 formatted ESP (EFI system partition) to store the files required for UEFI boot. Then, copy the EFI folder from the "assets/Limine" folder in your project directory to the root directory of the ESP partition. Next, copy the compiled kernel file UxImage to the "EFI/Boot" folder in the ESP partition. Finally, set your computer's boot mode to UEFI mode and ensure that the CSM (Compatibility Support Module) function is disabled. Then, boot your computer using the prepared boot media to run the Uinxed kernel on the actual hardware platform. You can also boot the kernel in Legacy mode, but this is not recommended. When running the kernel on actual hardware, always ensure hardware compatibility and stability to avoid potential system failures or data loss risks.

The Uinxed project adheres to the concepts of open source, collaboration and innovation, and aims to contribute to the field of operating system kernel development. Our team members uphold the love and pursuit of technology, and are committed to exploring the cutting-edge technology of modern computer system design. Through continuous research and practice, we integrate advanced design concepts into the implementation of the kernel, providing users with an efficient, stable and scalable operating system kernel. At the same time, we focus on the clarity and maintainability of the code so that other developers can easily understand and participate in the development of the project, and jointly promote the development and progress of the project.

This project is licensed under the Apache 2.0 open-source license, and all source code and related resources are shared open-source within this framework. This means you are free to use, modify, and distribute the project code, but you must comply with the obligations stipulated in the license. For example, when distributing derivative works based on this project code, you must retain the original license notice and use the same open-source license to ensure the openness and sharing of the code. We respect intellectual property rights and ensure that all code and resources in the project comply with relevant laws and regulations and do not infringe on the rights of any third party.

While our team is committed to ensuring the stability and reliability of the Uinxed project, various risks and uncertainties may still exist during its use. This project is provided "as is" without any express or implied warranties of any kind, including but not limited to warranties of merchantability, fitness for a particular purpose, and non-infringement. We assume no liability whatsoever for any direct, indirect, incidental, special, or consequential damages or losses that may result from the use of this project. Users assume all risks associated with using this project and must assess and make decisions based on their own circumstances and technical capabilities. We sincerely hope that the Uinxed project can provide valuable reference and assistance to operating system kernel development enthusiasts and professionals. If you encounter any problems or have any questions while using this project, please feel free to contact us through the contact information provided in the project, we will be happy to provide you with support and assistance.