

**System on chip with RISC-V microprocessor and Linux operating system**

**Qiyang Ding, Gan Fang, Yufeng Chen**

**Academic Advisor (Supervisor): Dr. Bowden Mark**

**19/11/2019**

**Declaration**

I confirm that I have read and understood the University’s definitions of plagiarism and collusion from the Code of Practice on Assessment. I confirm that I have neither committed plagiarism in the completion of this work nor have I colluded with any other party in the preparation and production of this work. The work presented here is my own and in my own words except where I have clearly indicated and acknowledged that I have quoted or used figures from published or unpublished sources (including the web). I understand the consequences of engaging in plagiarism and collusion as described in the Code of Practice on Assessment (Appendix L).

**1. Introduction**

**1.1 Background**

In early 1960s, IBM developed four incompatible computers, which included different instruction set architectures, I/O systems and so on. These computers opened a new world for computer, especially computer architecture. Since 1970s, due to the rapid progress of MOS technology, there appears many instruction set architectures and minicomputers, and gradually, complicated instruction set computer (CISC) constructed by Intel such as Intel 8086 and 80386 become the microprocessor with high performance. Nowadays, according to the Jeff Dean and David Patterson, who is the chief researcher in Google and the latter one is the professor of University of California, Berkeley, a new golden age has started in computer architecture and it is relative to the development of machine learning [1]. Until now, MIPS, x86, x86-64, Arm, VLIW and many microarchitectures has exhibited their values in different areas and among these architectures, Reduced Instruction Set Computer (RISC) has gradually come to the center of the stage. It contains less instruction set so that it executes more instructions per program, but it can use less cycles to achieve the program. RISC-V is based on the reduced instruction set architecture and it is developed by David A. Patterson, from University of California, Berkeley. It is an open-source instruction set; thus, it was developed by different people and was applied in many different spheres such as deep learning accelerator and embedded system.

In addition, the other topic of the project is Linux operating system. It experiences five significant foundation, which are Unix, MINIX, GNU, POSIX and Internet. It has become an open-source operating system and because of its characteristics of free and open source, it is widely applied in embedded system, which is also one of the most hopeful direction of RISC-V microprocessor. Therefore, RISC-V and Linux operating system for any release version will be an interesting composition in the embedded system area and also be helpful for the computer architecture or computer system.

**1.2 Objectives**

In this project, RISC-V microprocessor and Linux operating system such as Ubuntu will be the core parts. It aims to construct a system on chip (SoC) to run the transplanted operating system and enable it to act as a minicomputer such as using Internet, getting input from keyboard and so on. This project will hugely improve the understanding of operating system and microprocessor design. It will also help students learn a simple structure of today’s computers. Most importantly, learning software/hardware interface will hugely enable students to write better program or design better digital system. In addition, the microprocessor will be built on the Field Programmable Gate Array (FPGA), which is a powerful integrated chip that allows multiple writes. Hence, it can be a friendly equipment for the students who are learning the knowledge about it. How to test different parts in the system and integrated system is also an significant in this project. Some automatic test tools and GitHub will be used to achieve the function of this.

Contributions: Qiyang Ding is responsible for the development of RISC-V microprocessor and its relative drivers. Gan Fang is responsible for the transplantation of operating system, and Yufeng Cheng is responsible for the boot program of the system.

**2. Materials and Methods**

**2.1 Hardware design**

The FPGA is Digilent 410-292 Nexys 4 DDR Artix-7 Development Board, which includes many ports and DDR2 RAM so that it will be a powerful equipment during the development of microprocessor. The IDE for this FPGA is Vivado 2019.3. It is provided by Xilinx and can help student use FPGA properly. The hardware description language will be Verilog. Moreover, the testbench will be designed for automatic test and validation.

RISC-V core will include five stages pipeline and branch prediction in order to increase the performance of the processor. The CPU will also contain cache, which separated as data cache and instruction cache. Additionally, out of execution (reorder buffer) and superscalar structure will be considered if it can be achieved during specific time to increase the total performance of the microprocessor so that it can support operating system successfully.

Since Xilinx will provide the IP core of the memory controller, this part will be relatively simple, and it will be integrated into the whole project. Except for that, VGA controller, USB controller, Ethernet controller, PS/2 Driver, Serial Port controller, and other controller that can increase the speed of the SoC will also be taken into consideration.

**2.1 Operating system design**

This part of work needs riscv-gnu-toolchain. riscv64-unknown-linux-gnu-gcc is also used in this project. In order to understand the principle of OS design, it is necessary to know the compiling, which consists of preprocessing, compilation, assembly and linking. The first one is preprocessing. Take a C source file as an example, in this stage, all #define and annotations will be deleted, and all macro definitions will be expanded. Then the C source file will be transferred to intermediate file (\*\*\*.i). The second step is compiling. After this step, the intermediate file will become an assembly file. In the next step, the assembly language will be changed to machine code. Finally, the objective files will be linked together and become an executable and linkable file.

Since there are limited sources on the embedded platform, such as ARM MCU, it is impossible to install the compiler on the platform. It means the programmer cannot develop the software in the embedded device. Therefore, the software of embedded platform is generally developed and compiled on the host PC, and then the compiled binary file will be downloaded to the platform. In the host PC, a C library, called newlib, is required. It achieves most functions and its volume is small for an embedded platform.

Before the linux core is transplanted to the embedded platform, a boot loader is needed to initialize the hardware and software environment.

Based on above analysis, developing an embedded operating system is consisted of 8 steps.

i) Download Ubuntu 18.04 and GCC cross compiler, such as arm-linux-gcc.

ii) Download and configurate MINICOM.

iii) Use U-boot as bootloader.

iv) Design the Linux system.

v) Build the root file system using BUSYBOX.

vi) Use JFFS2 file system to create a FLASH disk partition for applications.

vii) Develop software on this Linux system and use QEMU to simulate the hardware environment to run the software.

viii) Transplant the Linux kernel to risc-V.

**3. Deliverables and expected outcomes**

For the demo, people can see Linux operating system running on the SoC. In the file system, many Linux commends can be used. For example, user can enter the file archive using cd commend, can show all files under the current path using ls commend and create a subdirectory using mkdir commend. In the final result, a monitor will be connected to the embedded platform, which means this operating system can display some basic information, such as illustration of configuration information and show the content of text file. Finally, since there is an ethernet port, the operating system can be connected to the internet to download some materials.

For hardware, this project will include a 32 bits RISC-V microprocessor with 5 stage pipeline and branch prediction. It also includes UART controller, DDR controller, network controller, USB controller, BootROM controller, VGA controller and so on. The advanced goal is to design a superscalar structure and use out of order execution. Cache and exception system will also be the advanced goal of hardware in this project.

For software, as mentioned in previous section, this operating system will be developed in Ubuntu 18.04. Moreover, riscv-gnu-toolchain is used to compile code, QEMU is used to simulate hardware environment in the host PC and a C library, called newlib, to provide several basic functions. The advanced goal is to design and transplant a Linux operating system to RISC-V. It can execute some basic operations, display information on monitor and connect to internet.

**4. Reference**

[1] J. Dean, D. Patterson, and C. Young, "A New Golden Age in Computer Architecture: Empowering the Machine-Learning Revolution," *IEEE Micro,* vol. 38, no. 2, pp. 21-29, 2018, doi: 10.1109/MM.2018.112130030.