

B10815013

范恩琪

四資工二甲

## Computer Organization Project 5 - GNU Toolchain and RISC-V ISA

### Report

---

In this project, we are required to write a simple Makefile to build my own executable program, library, compiled objects, and RISC-V assembly codes from C++ projects.

My first source code is a Fibonacci number program, after comparing with both elf-gcc and linux-gnu-gcc, from what I see the most noticeable difference is that with elf-gcc, there are more lines of assembly code than linux-gnu-gcc. And after comparing using static, it seems that the result of linux-gnu using static is much longer than the one not using static, but for the result of elf, the length is the same for both static and not static.

For my second source code, I make a program to check if a number is an odd or even number, and after comparing with both elf-gcc and linux-gnu-gcc, it is the same as my first source code, the most noticeable difference is that elf-gcc produces much longer code than linux-gnu-gcc. And after comparing using static, it the result of linux-gnu using static is much longer than the one not using static, but the length for elf result using static and not static is the same.

For my third source code, I make a program to check the length of a string, and after comparing with both elf-gcc and linux-gnu-gcc, it can be concluded that by using elf-gcc, the assembly code is much longer than linux-gnu-gcc. And after comparing using static, it the result of linux-gnu using static is much longer than not using static, but the result of elf has the same length for both static and not static.

## Makefile for source1:

```
CC = g++

all: a.out source1.exe linux_source1.o elf_source1.o linux_source1_static.o elf_source1_static.o

a.out: source1.cpp
    $(CC) source1.cpp

source1.o: source1.cpp
    $(CC) -c -Wall -g source1.cpp

source1.i: source1.cpp
    $cpp source1.cpp > source1.i

source1.s: source1.i
    $(CC) -S source1.i

source1.exe: source1.cpp
    $(CC) source1.cpp -o source1.exe

linux_source1.o: source1.cpp
    $riscv32-unknown-linux-gnu-g++ source1.cpp -o linux_source1.o
    $riscv32-unknown-linux-gnu-objdump -d linux_source1.o

elf_source1.o: source1.cpp
    $riscv32-unknown-elf-g++ source1.cpp -o elf_source1.o
    $riscv32-unknown-elf-objdump -d elf_source1.o

linux_source1_static.o: source1.cpp
    $riscv32-unknown-linux-gnu-g++ source1.cpp -o linux_source1_static.o -static
    $riscv32-unknown-linux-gnu-objdump -d linux_source1_static.o

elf_source1_static.o: source1.cpp
    $riscv32-unknown-elf-g++ source1.cpp -o elf_source1_static.o -static
    $riscv32-unknown-elf-objdump -d elf_source1_static.o

clean:
    rm -f source1.o a.out source1.exe linux_source1.o elf_source1.o linux_source1_static.o elf_source1_static.o
```

## Makefile for source2:



```
CC = g++

all: a.out source2.exe linux_source2.o elf_source2.o linux_source2_static.o elf_source2_static.o

a.out: source2.cpp
    $(CC) source2.cpp

source2.o: source2.cpp
    $(CC) -c -Wall -g source2.cpp

source2.i: source2.cpp
    $cpp source2.cpp > source2.i

source2.s: source2.i
    $(CC) -S source2.i

source2.exe: source2.cpp
    $(CC) source2.cpp -o source2.exe

linux_source2.o: source2.cpp
    $riscv32-unknown-linux-gnu-g++ source2.cpp -o linux_source2.o
    $riscv32-unknown-linux-gnu-objdump -d linux_source2.o

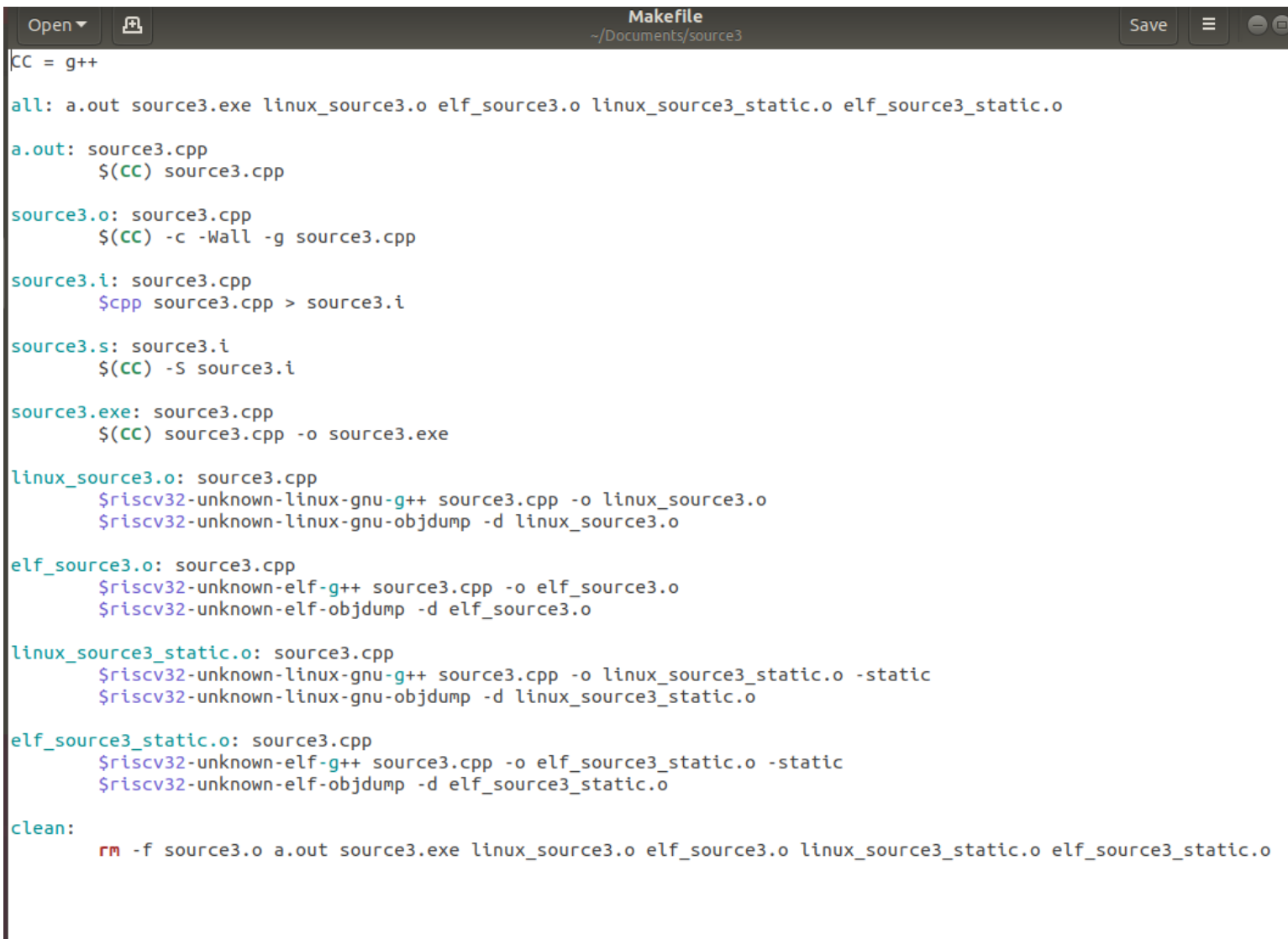
elf_source2.o: source2.cpp
    $riscv32-unknown-elf-g++ source2.cpp -o elf_source2.o
    $riscv32-unknown-elf-objdump -d elf_source2.o

linux_source2_static: source2.cpp
    $riscv32-unknown-linux-gnu-g++ source2.cpp -o linux_source2_static.o -static
    $riscv32-unknown-linux-gnu-objdump -d linux_source2_static.o

elf_source2_static: source2.cpp
    $riscv32-unknown-elf-g++ source2.cpp -o elf_source2_static.o -static
    $riscv32-unknown-elf-objdump -d elf_source2_static.o

clean:
    rm -f source2.o a.out source2.exe linux_source2.o elf_source2.o linux_source2_static.o elf_source2_static.o
```

## Makefile for source3:



```
CC = g++

all: a.out source3.exe linux_source3.o elf_source3.o linux_source3_static.o elf_source3_static.o

a.out: source3.cpp
    $(CC) source3.cpp

source3.o: source3.cpp
    $(CC) -c -Wall -g source3.cpp

source3.i: source3.cpp
    $cpp source3.cpp > source3.i

source3.s: source3.i
    $(CC) -S source3.i

source3.exe: source3.cpp
    $(CC) source3.cpp -o source3.exe

linux_source3.o: source3.cpp
    $riscv32-unknown-linux-gnu-g++ source3.cpp -o linux_source3.o
    $riscv32-unknown-linux-gnu-objdump -d linux_source3.o

elf_source3.o: source3.cpp
    $riscv32-unknown-elf-g++ source3.cpp -o elf_source3.o
    $riscv32-unknown-elf-objdump -d elf_source3.o

linux_source3_static.o: source3.cpp
    $riscv32-unknown-linux-gnu-g++ source3.cpp -o linux_source3_static.o -static
    $riscv32-unknown-linux-gnu-objdump -d linux_source3_static.o

elf_source3_static.o: source3.cpp
    $riscv32-unknown-elf-g++ source3.cpp -o elf_source3_static.o -static
    $riscv32-unknown-elf-objdump -d elf_source3_static.o

clean:
    rm -f source3.o a.out source3.exe linux_source3.o elf_source3.o linux_source3_static.o elf_source3_static.o
```

## Reference:

[1] GCC and Make: Compiling, Linking and Building C/C++ Applications

URL: [https://www3.ntu.edu.sg/home/ehchua/programming/cpp/gcc\\_make.html](https://www3.ntu.edu.sg/home/ehchua/programming/cpp/gcc_make.html)

[2] 12 Linux GNU Binary Utilities Binutils Commands with Examples

URL: <https://www.thegeekstuff.com/2017/01/gnu-binutils-commands/>

[3] The RISC-V Instruction Set Manual,

URL: <https://github.com/riscv/riscv-isa-manual/releases/download/Ratified-IMAFDQC/riscv-spec-20191213.pdf>