CS3500 LAB 1

Krutarth Patel EE23B137

7 August 2025

Brief

In this assignment we setup xv6-riscv, a toy operating system. The tasks involved were:

1. Setup Docker
2. Write a program to print out 'Hello ⟨ Your Roll No ⟩! Welcome to CS3500'
3. Compile the code as a user program and verify correctness

Details

0.1 Setup Docker

I was using Arch Linux so there was no official documentation on how to install docker on my system. However the latest package was available in the AUR repository and it worked just fine.

0.2 Write a program ...

```
.section .data
# data section, the string to be printed
# is defined here
msg:    .asciz "Hello EE23B137! Welcome to CS3500"

.section .text
# code starts here
.global main

main:
    # syscall: write(fd=1, buf=msg, count=14)
    li a7, 16          # syscall number for write
    li a0, 1           # file descriptor 1 (stdout)
    la a1, msg         # address of the message
    li a2, 33          # length of the message
    ecall              # make the syscall

    # syscall: exit(status=0)
    li a7, 2          # syscall number for exit
    li a0, 0           # exit code 0
    ecall              # make the syscall
```

- Figuring out the arguments required for the write() and exit() was tricky. I guessed that it was the same as in Linux and it worked.
- The syscall numbers were different for xv6. I found them in `kernel/syscall.h`

0.3 Compile the code ...

Apply the following patch to `Makefile`

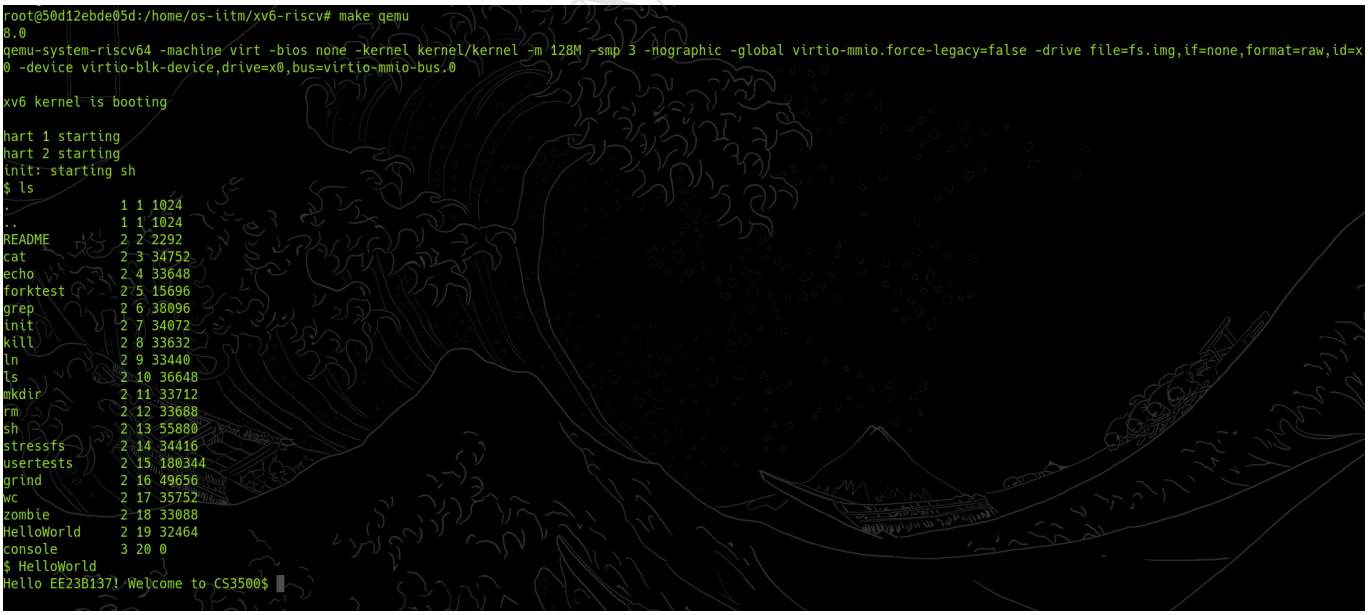
```
diff --git a/Makefile b/Makefile
index fab7bc9..347e257 100644
--- a/Makefile
+++ b/Makefile
@@ -109,7 +109,10 @@ $U/usys.S : $U/usys.pl

 $U/usys.o : $U/usys.S
         $(CC) $(CFLAGS) -c -o $U/usys.o $U/usys.S
-
+
+$U/HelloWorld: $U/HelloWorld.S
+        $(CC) $(CFLAGS) -o $U/HelloWorld $U/HelloWorld.S
+
 $U/_forktest: $U/forktest.o $(ULIB)
         # forktest has less library code linked in - needs to be small
         # in order to be able to max out the proc table.
@@ -142,6 +145,7 @@ UPROGS=\
         $U/_grind\
         $U/_wc\
         $U/_zombie\
+        $U/_HelloWorld\

 fs.img: mkfs/mkfs README $(UPROGS)
         mkfs/mkfs fs.img README $(UPROGS)
```

The whole `UPROGS` business was difficult to understand. I had to look how other user programs were being compiled and guess my way to this solution.

Proof



How to run

1. Place the HelloWorld.S file inside the `user/` directory
2. Apply the above patch to your `Makefile`
3. Compile and run using `$ make qemu` inside the docker container