Ho Chi Minh City University of Technology

Faculty of Computer Science and Engineering
Course: Computer Architecture Lab (CO2008)

Tutorial

Handling external files

Ho Chi Minh City, October 2024

Contents

1 Introduction

- **The main purpose of this session is to help you get familiar with reading and writing file.**
- This is a tutorial, students do not need to submit any exercise

2 Read and write a file

To allocate memory, please refer to the following syscall:

```
li $v0, 9 # system call code for dynamic allocation
li $a0, 24 # $a0 contains number of bytes to allocate
```

After the above system call, $v0 contains the first address in heap memory that is allocated. Then, accessing the allocated memory can be done by lw/sw, for example:

```
# Trying to write to allocated space
addi $t0, $zero, 2021
sw $t0, 0($v0)
```

The followings are instructions used to access a file (open/close/read/write):

```
# Sample MIPS program that writes to a new file.
# by Kenneth Vollmar and Pete Sanderson
.data
fout: .asciiz "testout.txt" # filename for output
msg1: .asciiz "Before read: "
msg2: .asciiz "After read: "
buffer_write: .asciiz "The quick brown fox jumps over the
lazy dog.\n"
buffer_read: .asciiz "----------------------------------------\n"
.text
##############################################################
# Open (for writing) a file that does not exist
li $v0, 13 # system call for open file
la $a0, fout # output file name
li $a1, 1 # Open for writing (flags are 0: read, 1: write)
li $a2, 0 # mode is ignored
syscall # open a file (file descriptor returned in $v0)
move $s6, $v0 # save the file descriptor
##############################################################
# Write to file just opened
li $v0, 15 # system call for write to file
move $a0, $s6 # file descriptor
la $a1, buffer_write # address of buffer from which to write
li $a2, 44 # hardcoded buffer length
```

```
    syscall # write to file
    ##############################################################
    # Close the file
    li $v0, 16 # system call for close file
    move $a0, $s6 # file descriptor to close
    syscall # close file
    ##############################################################
    ##############################################################
    # Open (for reading) a file
    li $v0, 13 # system call for open file
    la $a0, fout # input file name
    li $a1, 0 # Open for reading (flags are 0: read, 1: write)
    li $a2, 0 # mode is ignored
    syscall # open a file (file descriptor returned in $v0)
    move $s6, $v0 # save the file descriptor
    ##############################################################
    # Read from file
    li $v0, 14 # system call for read
    move $a0, $s6 # file descriptor
    la $a1, buffer_read # address of buffer read
    li $a2, 44 # hardcoded buffer length
    syscall # read file
```

Note that the file name is the PATH to the file in your filesystem. In the case of the above example, since the file has already been in the same folder as MARS, you only need to use the name of the file.