CS 35L

Software Construction Laboratory

Lecture 5.2

31st October, 2019

Logistics

- Assignment 5
  - Due on November 4th
- Hardware requirement for Week 8
  - Seeed Studio BeagleBone Green Wireless Development Board
- Assignment 10 Sheet
  - Teams of 2
  - https://docs.google.com/spreadsheets/d/1PVqVMEEsHjmmj9YLyqz5K4wU-k0Dwwm2iO9uS1Zq1wk/edit?usp=sharing

Assignment 10 Presentations and Reports

- If you presented in Weeks 3,4,5 or 6, you have 3 weeks from the date of your presentation to submit your report and presentation
- If you presented in Weeks 7 or 8, you have 2 weeks from the date of your presentation to submit your report and presentation
- If you presented in Week 9, you have 1 week from the date of your presentation to submit your report and presentation.
- If you presented in Week 10, you have till Week 10 Friday to submit your report and presentation

Final Exam

- Common Final Exam
- Sunday, December 8th from 3:00-6:00pm
- Room - TBD

Review - Previous Lab

- System Calls
  - Processor modes
    - User Mode
    - Kernel Mode
  - System Calls
  - System Call Overhead
  - Examples of System Calls

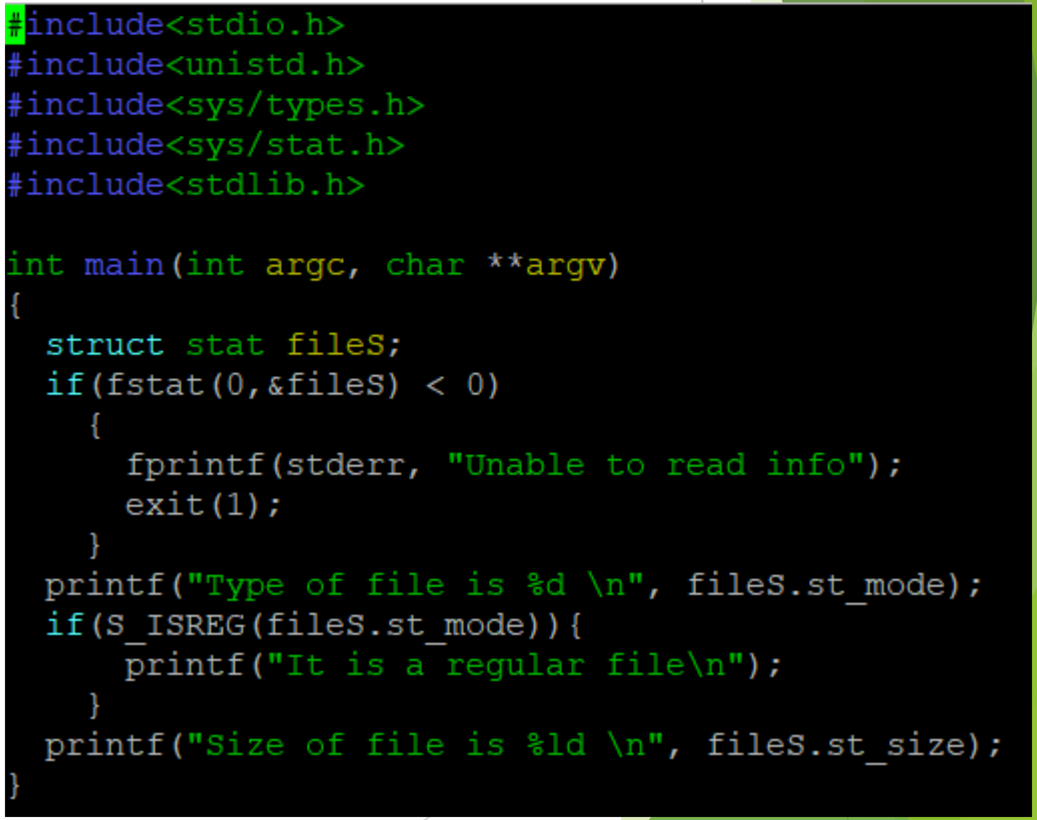
System Call Programming

Time and *strace*

- **time [options] command [arguments...]**
- Output:
  - –real 0m4.866s: elapsed time as read from a wall clock
  - –user 0m0.001s: the CPU time used by your process
  - –sys 0m0.021s: the CPU time used by the system on behalf of your process
- **strace: intercepts and prints out system calls.**
- -$ strace –c ./tr2b ‘AB’ ‘XY’ < input.txt
- -$ strace –c ./tr2u ‘AB’ ‘XY’ < input.txt

Fstat() demo

- Man 2 stat for additional information
- Check if it is a regular file or piped input through S_ISREG
- Use lseek() in case file is grown in size and set the file offset to the current location



```
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<stdlib.h>

int main(int argc, char **argv)
{
 struct stat fileS;
 if(fstat(0,&fileS) < 0)
  {
   fprintf(stderr, "Unable to read info");
   exit(1);
  }
 printf("Type of file is %d \n", fileS.st_mode);
 if(S_ISREG(fileS.st_mode)){
   printf("It is a regular file\n");
  }
 printf("Size of file is %ld \n", fileS.st_size);
}
```

fsanitize

- Checks for memory leaks (address mode) and runtime exceptions (undefined mode)
- Make sure to have /usr/local/cs/bin/ prepended to the path (works with the latest version of gcc)
- Compile using
  gcc –o out program-with-potential-memory-leak.c –
  fsanitize=address –static-libasan
  OR
  gcc –o out program-with-potential-undefined-behavior.c –
  fsanitize=undefined –static-libubsan

Assignment 5 - Laboratory

- Write tr2b and tr2u programs in 'C' that transliterates bytes. They take two arguments 'from' and 'to'. The programs will transliterate every byte in 'from' to corresponding byte in 'to'
  - ./tr2b 'abcd' 'wxyz' < bigfile.txt
    - Replace 'a' with 'w', 'b' with 'x', etc
  - ./tr2b 'mno' 'pqr' < bigfile.txt
- tr2b uses getchar and putchar to read from STDIN and write to STDOUT.
- tr2u uses read and write to read and write each byte, instead of using getchar and putchar. The nbyte argument should be 1 so it reads/writes a single byte at a time.
- Test it on a big file with 5,000,000 bytes
  - $ head --bytes=# /dev/urandom > bigfile.txt

Assignment 5 - Laboratory

- Review:
  - int ch = getchar()
    - NOTE: getchar() returns an Integer, not a character
  - putchar(ch)
  - int numRead = read(STDIN_FILENO, ch, size)
  - int numWritten = write(STDOUT_FILENO, ch, size)

Tr2b.c

- Write a main function which accepts arguments
  - main(int argc, const char* argv[])
- Check for the length of arguments
  - Retrieve first argument in char * from, second argument in char * to
  - Compare the lengths of from and to; If not same, throw an error and exit
  - You can use strlen to get lengths
- To throw an error, write to stderr using library functions
- Check if ‘from’ has duplicates
  - In a loop, take input from stdin (till you reach eof of stdin) using getchar()
  - Check if the character you just retrieved is a part of from; if yes then put the corresponding character in stdout with putchar()

Tr2u.c

- Repeat the same procedure as in tr2b.c except replace:
  - getchar() with read
  - putchar() with write

Assignment 5 - Homework

- Rewrite sfrob using system calls (sfrobu)
- sfrobu should behave like sfrob except:
  - If stdin is a regular file, it should initially allocate enough memory to hold all data in the file all at once
- Functions you'll need: *read*, *write*, and *fstat* (read the man pages)
- Measure differences in performance between sfrob and sfrobu using the *time* command
- Estimate the number of comparisons as a function of the number of input lines provided to sfrobu

Assignment 5 - Homework

- Run your program on inputs of varying numbers of input lines, and estimate the number of comparisons as a function of the number of input lines.
- Varying number of input lines => number of words
- Number of comparisons => keep a counter in the frobcmp() function to check how many times it is being called
- Use the time command to compare the overall performance of sfrob, sfrobu and sfrobu –f
- Measure any differences in performance between sfrob and sfrobu using the time command.

Assignment 5 - Homework

- Refer *to Read*, *Write*, *Open*, *Close* System Calls
- Reserved File Descriptors
  - 0 – stdin
  - 1 – stdout
  - 2 – stderr
- int fstat(int fd, struct stat *buf)
  - Returns information about the file with the descriptor fd into buf

Additional Information

- www.cs.uregina.ca/Links/class-info/330/SystemCall_IO/SystemCall_IO.html
- courses.engr.illinois.edu/cs241/sp2009/Lectures/04-syscalls.pdf
- www.bottomupcs.com/system_calls.xhtml

Assignment 10 - Presentations

- Today's Presentation
  - Xiangyu Wan
  - Xintong Liu
- Next Class
  - Shriniket Buche
  - Tyler Szeto

Questions?