Homework 0: Preperations



Due Friday by 11:59pm **Points** 30 **Submitting** a file upload **Available** until Feb 1 at 11:59pm

Updated: corrected the example output on 01/25/21.

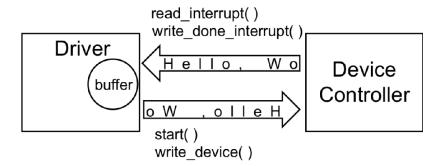
Read the **Lab Tutorial** before starting this homework.

Objective

The purpose of this assignment is to make sure you are prepared to succeed on the programming projects in this class. You will need to know how to test code on the pyrite server, create a basic makefile, write a C program, and debug using gdb and valgrind.

Problem Statement

Create a device driver that interfaces with a simulated device controller. The driver receives interrupts that indicate a character has been read from the device and echos it back to the device by calling write_device(). The device controller only has the capacity to write one character at a time, so you must buffer incoming characters and wait for a call to write_done_interrupt() before writing the next character.



You must use a <u>ring buffer (https://towardsdatascience.com/circular-queue-or-ring-buffer-92c7b0193326)</u>. The size of the buffer must

1 of 3 28/01/2021, 9:15 PM

be input to your program as a command line argument, this means you will need to allocate the array for the buffer dynamically.

Resources

Here is the device controller simulator, you are not allowed to modify these files in any way.

device-controller.h

device-controller-simulator.c

Here is the starting point for your code. Following this template, your solution should not be more than about 60 lines of code.

echo-driver.c

Requirements

Your submission must include:

- 1. A makefile that compiles the code with just the command make. It must be tested on the pyrite.cs.iastate.edu server.
- 2. The completed echo-driver.c file (see resources below).
- 3. An example (text or screen capture) of you using a breakpoint in gdb while running your program.
- 4. An example (text or screen capture) of you running valgrind on your program. The valgrind report should show no memory leaks. Although memory is automatically freed when a program exits, for the purpose of this problem we will consider any memory not explicitly freed by the program as a memory leak.

Example Output

The output should be the following:

\$./echo-driver 8
Hello, World!
Hello, WodH, d!
Hello, WodH, d!
HlWld!
Helo, Wo
1, Wol!el Wor!
11W!He

2 of 3 28/01/2021, 9:15 PM

\$./echo-driver 16
Hello, World!
Hello, World!
Hello, d!
Hello, d!
HlWld!
Helo, Wo
1, Wol!el Wor
\$./echo-driver 32
Hello, World!

3 of 3 28/01/2021, 9:15 PM