CS 310 Homework 1: Serial Port Driver

Spring 2021

February 23, 2021

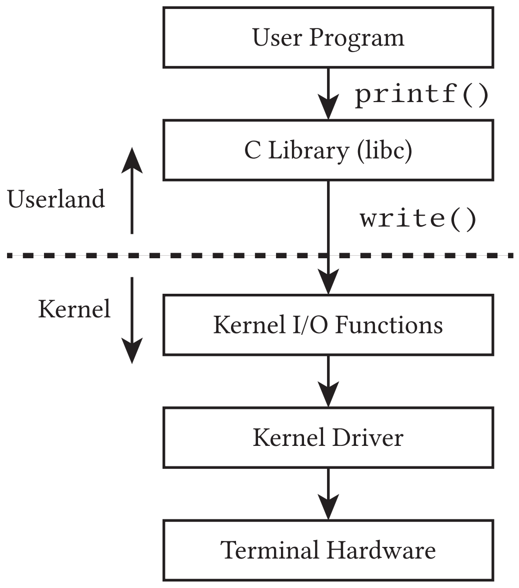
1 Introduction

In this homework assignment, you're going to build a serial port driver for your kernel that allows it to print to the terminal. This homework will be done in your `pios` repo, not in Linux. To write characters to the serial port, we will use a memory-mapped IO register `MU_IO`. It is located at address `0x3F215040` (`0xFE215004` on the Pi 4). All you need to do to send a character over the serial port in the emulator is write the character to the `MU_IO` register.

On the course website, I have linked to C file called `rprintf.c` that includes a function called `esp_printf`, an OS-free version of `printf`. You can use `esp_printf` mostly the same way as you would use regular texttprintf—it supports `%c`, `%d`, `%x`, etc. It doesn't support printing floating point, but we shouldn't need that.

There is one minor difference between `esp_printf` and regular `printf`—the interface to the terminal driver. The regular interface to the terminal driver is shown below. `printf` normally calls `write` to write a character string to the terminal, but this won't work for us because we haven't implemented `write` yet. Instead, `esp_printf` lets us pass a function as the first argument. The function that we pass will print a single character to the terminal. So a typical call to `esp_printf` looks like this:

```
esp_printf(putc, "Current Execution Level is %d\r\n", getEL());
```



The deliverables of this homework assignement are:

1. Create a new file in the `src` directory called `serial.c`. In `serial.c` write a function called `putc` that prints a single character to the serial port:

   ```
   void putc(int data);
   ```

2. Use `esp_printf` to print out the current execution level to the terminal.