Project 3: A Serial Console

Due March 27

1 Overview

You will alter GeekOS to run a shell on serial port 1 (COM1), just as it runs a shell on the screen and keyboard. To do so, you will provide a serial port driver for GeekOS and modify the process data structures to remember which console is to be used.

2 Goal

The primary goal of this assignment is to develop an understanding of hardware devices in an operating system, how processes can wait for input from devices, and a bit about initializing hardware in a driver.

3 Reference

QEMU is currently set up so that stdin and stdout are redirected to the COM1 serial port. Any bytes you send to COM1 will be printed in the linux terminal, any key presses (when the terminal is active) are sent to COM1.

Most of the COM1 registers have been initialized by QEMU to the correct values, however you will be responsible for setting appropriate registers to allow COM1 hardware interrupts. Information on how to do this can be found at `http://en.wikibooks.org/wiki/Serial_Programming/8250_UART_Programming`.

The code in keyboard.c may also prove useful as a guide, however there are extra complications with the keyboard (scan code conversion, Test_Input, ...) that won't be necessary for the serial input.

4 On boot

Print `"Welcome to the serial console!\n"` to the serial port.

5 Rules

Alter main.c so that 2 shells are created on boot, one on screen and keyboard, and another on the serial port COM1.

The serial shell and all its descendants should send output and receive input from the serial port.

Just like the keyboard, you should enable COM1 interrupts and read data from the serial port in response to the interrupt. Do not poll.

6 Hints

Modify thread structures as appropriate in order to track which processes use the serial console. If you choose to modify the Kernel_Thread structure, put all modifications at the bottom - geekos expects esp to be the first entry in the structure.

Recall that there are in effect 2 print functions - one for the kernel and one for the user. Both should be modified to enable serial printing.

src/libc/conio.c contains the code for user input/output. You should not need to modify any of the code here, instead, make changes to the system calls which are used by this code. Anything cursor related should be ignored by the serial port which has no cursor.

7 Notes

Invoking "exit" from the serial console shall not halt the OS. (Unlike exiting the init shell, which will.)

You do not need to handle backspace. Backspace requires a notion of cursor position, and this is closely tied to the screen.c implementation in geekos.