Multi-tasking OS

ECE598: Advanced Operating Systems – Homework 9

Spring 2015

Due: Wednesday, 6 May 2015, 5pm

This homework involves taking the multi-tasking OS and improving it a bit.

1. **Download the homework code template**
   - Download the code from:
     `http://web.eece.maine.edu/~vweaver/classes/ece598_2015s/ece598_hw9_code.tar.gz`
   - Uncompress the code. On Linux or Mac you can just
     `tar -xzvf ece598_hw9_code.tar.gz`
2. **Memory Allocation (5pts)**

   Convert the memory allocation code found in `memory.c` to be a "find next" allocator rather than "find first". You will want to modify the `find_free()` routine.
3. **Scheduling (5pts)**

   Currently the scheduler in `scheduler.c` is a simple round-robin scheduler, which schedules all ready jobs equally. This includes scheduling the idle task.

   Modify the scheduler (`schedule()`) to not schedule process id #0 (the idle task) unless no other jobs are available to run.
4. **Extra Credit (Up to 5 extra pts)**

   Do something additional. Below are just some suggestions for things you can do.
   - Easy
     - Add support for the REBOOT system call.

       To reset the Pi, we will program the hardware watchdog to reset the machine if no one updates the watchdog counter for 1/16 of a second. Then we'll not bother to update the counter, so it resets.

       To do this, in the `syscalls.c` file you will want to find the REBOOT entry point and add code that does the following:
       * Set the `PM_WDOG` register to have the value `PM_PASSWORD` logical-ored with 1.
       * Set the `PM_RSTC` register to have the value `PM_PASSWORD` logical-ored with `PM_RSTC_WRCFG_`

       To test, issue the `reset` command in the shell.
   - Medium

    - Modify the system timer in timer.c to have a context switch time of 100Hz rather than the current 2Hz.

      This will require updating the code you wrote in Homework #3, so you might want to review what you did there.

      Also update the code so that the LED still blinks at the 2Hz rate, and that the "time" command still reports the proper time.

  - Hard
    - Write your own userspace program and run it.
      See the userspace directory for examples of how to do this. Note, the current build of userspace code uses the `xxd` utility which is available on Linux but probably not on Windows.

5. **Submit your work**

   - Run `make submit` in your code directory and it should make a file called `hw9_submit.tar.gz`. E-mail that file to me.