Lab 3 Sign-Off Sheet

Name: _____ uname:_____

Name: _____ uname:_____

Q1. Write a function called **`CPU_work(int time)`** which causes the Arduino to use about *time* ms. (So **`CPU_work(2)`** would use 2ms of CPU time.) It should do this with a "for" loop. Make sure your subroutine allows for more than 2^15 – 1 iterations of the busy loop. Modify the code in task1 to:

a. Call **`CPU_work(900)`** and **`CPU_work(100)`** to get a signal with 1s period and a 10% duty cycle.
b. Adjust the CPU_work function as needed to get the period reasonably close to 1 second (say within a couple of milliseconds).

Q2. For this part you will do two demos for the lab instructor.

a. Demonstrate the tasks running as described above. Briefly discuss your answers to Q3 with the lab instructor.
b. Change Task1's CPU utilization to 55ms and run the code. Using the scope or logic analyzer, set things up so that you can convince your lab instructor that both tasks are *always* meeting their deadlines (or not). Or, alternatively, explain why you can't show they are (or are not) meeting their deadlines.

Q3. Get tasks T1, T2, and T3 working as described above where T1 and T3 are using one IO pin (T3 setting it high and T1 setting it low once CPU_work() has finished). Again, you will do two demonstrations for your lab instructor and answer one question. Try to have your numbers picked for part b and to have thought about part c before you ask your lab instructor to come over.

a. Show your lab instructor the working code you wrote for the above.
b. Modify the CPU_work duration of one of the tasks so that a deadline is being just barely missed. RM scheduling theory should help. Show your lab instructor and explain how you know a deadline is being missed.
c. Explain to your lab instructor what is happening when a deadline is missed. In particular, what is going on when task 3 releases its new task instance when the previous instance of task 1 is still running? Your lab instructor will ask about the semaphore and the pin's value.

G4. Using the scope, demonstrate this working code to your lab instructor. Explain what has changed from when the delay code was in the ISR and explain why deferred interrupts could be useful.

G5. Demonstrate that you can change GPIO pin 21 to toggle every 2ms as detailed in the how to build an RTOS application example above.

G6. Demonstrate the deferred interrupt example to your instructor.

G7. Show the GSI that your robot can accomplish these tasks in addition to your scope output.