ECE 362 – Embedded Operating Systems

Assignment 5

Due March 11th in D2L assignment dropbox

NOTE: Assignment 6 will be due on March 14th before class begins.

1. Create a program that creates 4 threads where each thread prints out its thread Id.

   If you have a runaway process (or several), use the shell command: ps -ef | grep $USER
   to get a list of your running processes. Then type kill -9 pid, where pid is the process you want to kill.

2. Develop a multithreaded program to sort an array of integers.
   - ***DON'T do this until you've solved problem 1!***

   - Your program should run with 4 threads and use the algorithm discussed in class
     - Break the array into four equally sized partitions
     - Sort the partitions separately in different threads
     - Merge the results back together. This is a 2 step process where two threads merge the four partitions into two partitions and then with one thread merge the two partitions into one.
     - Write a simple routine that checks the final array is in sorted order. If the array is sorted print "Sort complete." Otherwise print an error message.

   - Each of the worker threads should use a bubble sort algorithm (intentionally slow, see the pseudo code below).

   - Use an array size of 64,000 elements that you randomly initialize using `rand()`

   - You may find it interesting to try different sized arrays and vary the number of threads. During development you'll want to use a small array, perhaps start with 2 threads, and printout your array results to confirm the sort and merge worked. The final version of your program should only print out the simple message generated by your simple checker.

   - Along with your code, turn in a comparison of how much time the multithread program takes versus running the same problem using a 1 threaded bubblesort algorithm. To determine how much time is required, you might use the unix time program (or "timer" from the first programming assignment).

     Simple bubble routine:

```
for (x = 0 ; x < ( n - 1 ); x++)
  {for (y = 0 ; y < n - x - 1; y++)
    {if (array[y] > array[y+1])
       {temp  = array[y];
       array[y]   = array[y+1];
        array[y+1] = temp;
      }
    }
```