COMS W4995-02
Languages for Embedded System Design
Homework 2

Prof. Stephen A. Edwards
Columbia University

Assigned September 16, 2002
Due October 2, 2002

You may submit the solutions either on paper or electronically, but not both. If you submit it electronically, send a single file that is text, PostScript, PDF, or Word. Don't send multiple files or an archive such as tar or zip. I just print out whatever you submit and read it; I don't try to run the programs.

Make sure your name appears at the beginning of the file you send.

I want the paper or electronic versions at the beginning of class (4:10 PM EDT) on the due date. This applies to both on-campus and CVN students.

1. **(5 points)** Book, Exercise 9-3: List three things currently missing from Java that are needed by most embedded software systems.

2. **(10 points)** Does marking a method or block of code synchronized in Java guarantee that other threads cannot access the synchronized object? If so, explain how. If not, explain what else is needed to guarantee this.

3. **(15 points)** The Java wait() method releases all locks owned by its thread before suspending. Why does it do this? What would happen if did not?

4. **(50 points)** Concurrent Java programs

   (a) **(10 points)** Write a threaded FIFO buffer for integers in Java.

   (b) **(20 points)** Use it in a simple program with two threads. One thread should put successive integers (i.e., 1, 2, 3, etc.) in a buffer. The other should repeatedly remove an integer from the buffer and print it.

   (c) **(20 points)** Modify your program to create two copies of the sequence generator, each feeding into the same buffer. Does your program print 1 1 2 2 3 3 or something else? Modify it so it does.

   The point of this exercise is not to write a program that happens to work, but one that will always work regardless of the Java implementation.

   Do not use `sleep()`.

   In all cases, I want to see your source code as well as program output. Also, please mention the type of machine (e.g., Windows, Solaris, Linux, etc.) you used to compile and run Java.

5. **(10 points)** Define priority inversion. Give a scenario in which it occurs (i.e., describe processes, their priorities, and their actions).

6. **(10 points)** What is the major difference between earliest-deadline first and rate-monotonic scheduling? Which would be easier to implement?