CMSC 23000	**Operating Systems**	**Homework 2**
Winter 2006		**Due February 3**

1. Consider an 32-bit processor with eight general purpose registers (`r0`–`r7`), with `r4` and `r5` callee-save, `r6` as the frame pointer, and `r7` as the stack pointer. You have been asked to add the following function to a coroutine library for this processor:

   ```
   void yield_to (thread_t *tid);
   ```

   This function suspends the running thread and resumes the thread named by `tid`. Assume that you have the following C data structure for representing a thread state vector:

   ```
   typedef enum { RUNNING, READY, TERMINATED } status_t;
   typedef struct {
       status_t  status;
       word_t    regs[8];
       word_t    pc;
   } tid_t;
   ```

   A pointer for the current thread is kept in the global variable

   ```
   extern tid_t *self;
   ```

   Write the code that implements the `switch_to` function. Note that this function would have to be written in assembler, but you may use C syntax for your code.

2. Assume that a system is running ten reactive processes and one computation-bound process using round-robin scheduling. Furthermore, assume that the reactive processes do a communication operation that blocks for 10ms for every 1ms of computation and assume that it takes 0.1ms to switch contexts.

   (a) What is the CPU utilization when the scheduling quantim is 1ms?

   (b) What is the CPU utilization when the scheduling quantim is 10ms?

   (c) What is the average latency from when a reactive process finishes its communication to when it next gets to run?