HOMEWORK V
ECE 3710 — Microcomputer Hardware and Software
Utah State University
Due date: **October 18, 2013**

Problem 1. Write a function/subroutine named `getStream` that when called receives a stream of bytes from the serial port and stores the bytes on the stack. The first byte of the data stream will tell you how many bytes are to follow. After you've stored that many bytes, return from the function/subroutine. The first byte of the stream (the count) should be pushed on the stack after you've pushed the rest of the stream. Assume a 400 byte stack (you can receive up to 255 bytes so you have to use byte-alignment). Configure the uC to use a 15.38 MHz clock and 9600 baud for UART. This should be done in both C and assembly.

Problem 2. You're working on a project that requires more serial ports than are available on the microcontroller. Rather than simply adding more serial ports, your manager wants you to use unused I/O ports to implement RS-232 in software (the accountants say that the time it will take you to implement RS-232 is less than the cost of adding additional UART chips—you are obviously an underpaid, new engineer). Having given your two week notice (who wants to listen to accountants, after all?), you are still obligated to implement a transmission function that transmits a byte at a given baud rate. Let's define the function as `rs232TX(unsigned char data, unsigned int baudRate)`. You have a line driver connected to `PB.3` that will output -10 V when it receives a one and +10 V when it receives a zero (these are the levels used for your RS-232 implementation). The byte needs to be transmitted with one start and one stop bit.

Problem 3. Create a C function, `void MSDelay(unsigned int itime)`, that generates *itime* milliseconds of delay on the Keil simulator. Describe how you determined the proper bounds for the loop(s) and any relevant simulator settings (target board, clock, etc.). Include a screen capture that demonstrates your function.

Problem 4. (EXTRA CREDIT) You are to implement the Base64 encoder function in C. In Base64 encoding three arbitrary bytes are converted into four ASCII characters (see the *Base64* entry on Wikipedia). Your function should take two inputs: the first, *s*, is a pointer to the data to be encoded, while the second, *f*, is a pointer to where the converted values should be stored.