CmpE213 – Digital Systems Design

Homework 7
ASM and C for the 8051

1. Write a C program that performs the following:

   (a) Declare two unsigned char (1 byte) variables (say x and y), initialize the variables with values within the acceptable ranges (between 0 and 255), add the two variables storing the result in the first (e.g. x = x+y;)

   (b) Declare two unsigned int variables (2 byte), initialize the variables with values within the acceptable ranges, add the two variables storing the result in the first variable,

   (c) Declare two unsigned long variables (4 byte), initialize the variables with values within the acceptable ranges, add the two variables storing the result in the first variable,

   Create a project using the Keil software and add your C program to that project. Build the project and then debug your code. Examine the code using the disassembly window and answer the following questions

   (a) Looking at the assembly translation below each C statement, how many registers, memory locations and total instructions are used to perform each addition (steps a, b, and c)? Use a table to show for each addition in the assembly language translation: 1) number of registers used in the addition, 2) number of "other" memory locations used to store variables, and 3) number of assembly instructions used to compute and store the sum.

   (b) Given the results from part a, how many registers, memory locations, and total instructions would you expect to be used for adding two 8 byte values (assume integer)?

   (c) How did the compiler decide to store the second variable (e.g. y in x = x+y;) used to compute the sums? Why would the compiler store it in this manner?

   In addition to the above questions, please turn in a screen dump of the disassembly window, showing as much of your mixed ASM-C code as possible (don't worry about the other windows).

2. Write a function called *delay()* which creates a delay simply by executing a loop which does nothing (e.g. *for (i=0;i<42;i++){;}*). We'll see uses for this function in class soon. Give this function the following characteristics:

   - Pass the number of millisecond to wait (to be delayed) to the function as an unsigned char (for example, calling *delay(5)* would create a 5mS delay). The function returns nothing (void).
   - Create the delay using a do-while loop whose loop-variable is stored in external data memory.

- VERIFY that passing a 1 to delay() creates a 1mS delay, a 2 creates a 2mS delay, etc, using the Keil debugger (+/- a few uS is OK). Finding the right number of iterations to perform the loop will take some experimentation.
- Define the number of loop-iterations needed to create a 1mS delay as a constant at the beginning of your code and use that constant in your function. That is, instead of using something like:

  do{ blah }while(x < 5280);

  use:

  #define DELAYCYCLES 5280
  do{ blah } while(x <DELAYCYCLES);.

  Using constants in this manner is good programming practice – it makes your code more understandable and makes it easier to change in the future.

Create and simulate your code using the Keil debugger to ensure it works and to find the correct number of loop iterations. Turn in a hardcopy of your code and email a copy of the file containing the function to daryl@ece.umr.edu. Please include the file as an attachment and use the subject header "HWK7 - program". **Please put your name at the top of your program (and anyone you worked with) and comment your code appropriately.**