Programming Assignment #4 - String Manipulation and Assembly
CSE30 - Computer Organization and Systems Programming
Winter 2011

Overview

The goal of this assignment is recode the LED lighting functions from PA#3 in assembly. You are given completely functional C code that operates the LEDs and performs correct string input using the Capsense. Your job is to replace the C code in the select functions with ARM assembly code.

Project Description

The big idea for this assignment is inlining assembly code. This can be done directly in the PSoC creating C code.

You can add assembly code straight into your C by using the `asm` keyword. The example below shows you how you would write your own `strcpy` function using assembly for the function body.

```
void my_strcpy(char *src, char *dst) {
asm("loop: LDRB  r3, [r0], #1 \n \t"
      "STRB  r3, [r1], #1 \n \t"
      "CMP   r3, #0 \n \t"
      "BNE   loop \n \t");
     return;
}
```

Notes:

- You need to put each assembly instruction on a separate line of code wrapped in “”.
- The function follows the standard argument passing procedure that we discussed in class. The first argument is in `r0`, the second is in `r1` and so on.
- Do not try to return from functions yourself. The compiler does some housekeeping at the beginning and end of the function that won’t occur if you return early. This will likely cause strange behavior.

Programming Assignment

Write the assembly code for the following LED functions.

```
void turnLED2On(char * string1, char * string2, char * string3, int *condition);
```

LED2 will light up if any of the strings are a permutation of any other string currently in memory (excluding the empty string). This means that any two of the strings have the same length and exactly the same characters. For example, if String1 = “ABC” and

String2 = “BCA” then LED2 should light up. The function sets `condition` to TRUE/FALSE depending on the values of the three strings.

```
void turnLED3On(char * string1, char * string2, char * string3, int * condition);
```

LED3 should light up if any of the strings are equal to any other strings in memory (again excluding the empty string). Write a function turnLED3On that sets `condition` TRUE/FALSE depending on the values of the three strings.

```
void turnLED4On(char * string1, char * string2, char * string3, int * condition);
```

LED4 should light up if any of the strings are a substring of any of the other strings. For example, if String2 = “123” and String3 = “ab123c”, then String2 is a substring of String3. Write a function turnLED4On that sets `condition` TRUE/FALSE depending on the values of the three strings.

Notes:

- These functions are slightly different than those that were defined in the last programming assignment. The previous return value is now passed as an argument.
- You are provided a fully functional project written in C. Your assembly code must adhere exactly to the C code given to you.