Implementation

Lab Implementation Overview

Software Implementation

We have created a function in c language which is being called in arm assembly. The function we created in c is called “void my\_print(char \*d)” (see appendix). This function takes an reference to char array and print out the string. Arm assembly code has been implemented where my\_print function is being called multiple times (see appendix).

To call the c function in arm assembly you need to import the function. We created our main program named asmmain in arm assembly where we are importing my\_print function by using a statement “import my\_print”. Then we are loading strings “String source” (srcstr) and “String destination” (dststr) in registers. In this asmmain.s we are basically replacing the content of dststr with srcstr and in order to do that we have implemented strcopy function which are coping content of dststr with content of srcstr.

Hardware Implementation

TWR-K60D100M pcb device is used for this lab. User needs to plug-in the device in computer port and press reset button (located on the top left corner beside the device USB port. Now your device is ready. User needs to load the program via uVisison, after working on the project user needs to build the program which can be done by pressing F7 key on the keyboard. For loading the program on the board press the loading icon on uVision interface.

Putty terminal will appear, hit reset button on the board and output will be visible on the putty terminal.

Lab Requirements

1. Code error that was found in mymain.s program.

* We needed to include following two lines of code in asmmain.s which were missing in original program. (see appendix)

LDR r0, =dststr

BL strcopy

2. Content of the addresses and addresses of “srcstr” and “dststr” - (see appendix)

R0 has address of dststr

R1 has address of srcstr

3. Address of main.c and asmmain.s?

Why are address regions for asmmain.s much different than for the data?

4. Calculation for the counter and the loop delay:

Processor speed: 100 MHz (100x10^6 instruction/second)

So to make it delay we need to perform more instruction so that processor take more

time. So we have loading 200x10^6 instruction in r0 register (LDR r0, =0xBEBC200), please see appendix

In side the loop we are incrementing r1 with 1 and comparing with r0 which is a huge number (200x10^6) and it takes takes approximately 10 seconds (accounting the gate delay) to get equat to the value in r0.