Homework Assignment #1

Sunday March 3rd @ 5PM

1.Important Information

- **This is an individual assignment**. Any evidence that code was shared between individuals will be treated as academic misconduct.
- All code should have meaningful comments. A 5 point penalty will be levied against each problem that is not commented sufficiently.
- Any code that does not assemble/compile will **receive no points**
- All code is to be written in ARM assembly using the Keil uVision IDE.
- You may use any code that you have developed as a part of an ICE.
- All non-ICE code that is submitted must be written, in its entirety, by you. Submitting code that is not original work of the team/individual will be treated as academic misconduct.
- The maximum point total for the assignment is 50 points

2.Problem Overview

This homework assignment will ask you to create a program that controls the WS2812B LEDs on the ECE353 development platform. Your program will search through a defined area of memory that contains DISPLAY COMMANDS used to create patterns on the WS2812B LEDs.

3.Provided Libraries

In order to aide in the development of your program, you have been provided with an assembly file that contains a function that can set the colors of the WS2812B LEDs.

You should extract the HW1.zip contents into the same directory as your ICE exercises.

4.Behavioral Requirements

The table below describes a list of valid DISPLAY commands.

Command	# of Bytes	Description
LOAD	4	Consists of 4 consecutive bytes in memory that hold the ASCII characters 'LOAD'. In response to the LOAD command, your application should turn all WS2812B LEDs off.
HALT	4	Consists of 4 consecutive bytes in memory that hold the ASCII characters 'HALT'. In response to the HALT command, your application should leave the WS2812B LEDs in their current state and enter an infinite loop.
HANG	8	Consists of 4 consecutive bytes in memory that hold the ASCII characters 'HANG'. The next 4 bytes of data hold an ASCII representation of how many iterations of an empty for loop to turn. The number specified in the delay loop is then multiplied by 10,000 to determine how many iterations of the FOR loop to run. This loop acts as a way to delay the system. Example 'HANG1000' would be an empty for loop that executes 1,000*10,000 times and then exits. Valid wait times MUST be a 4-digit ASCII string that represents a DECIMAL value.
LEDx	10	The first 3 bytes consists of the ASCII characters 'LED'. The 4th character identifies which LED (0-7) is being modified. The remaining 6 ASCII characters indicate the color of the LED. Characters 5&6 indicate the intensity of green, characters 7&8 red, and 9&10 blue.

		This command MUST only result in a color change for the LED identified. All other LEDs should be unchanged. The ONLY valid LED numbers are 0-7. Example to turn on the second LED 'LED2FF8800'

Required Variables

LED_ARRAY
24 bytes reserved in the SRAM used to store the most recent colors of the LEDs. The green color code for LED7 is stored at an offset of 0 from LED_ARRAY. The table below describes the format of the array

ADDRESS	Description
LED_ARRAY + 0	LED7 Green Value
LED_ARRAY + 1	LED7 Red Value
LED_ARRAY + 2	LED7 Blue Value
LED_ARRAY + 3	LED6 Green Value
LED_ARRAY + 4	LED6 Red Value
LED_ARRAY + 5	LED6 Blue Value
And so on.	

Required Functions

You must modify line 60 of main.c. You should replace the following line with your name.

```
printf("LAST NAME, FIRST NAME\n\r");
```

All functions MUST be written to be callee saved.
All functions MUST be written in hw1.s.
Single test case is given for you in main.s. However, unreleased test vectors (including invalid inputs) may be tested against your code for your grade.

hw1_update_leds

Description

Updates the WS2812B LEDs with the current values in the LED_ARRAY. Use the supplied function ***WS2812B_write*** to update the colors. The implementation details of the function are found in hw1_ws2812B.s.

parameters

NONE

Returns

Nothing

hw1_ascii_to_hex

Description

Converts A single ASCII HEX character to its numerical value. Valid characters are 0-9,a-f,A-F.

Parameters

R0 - ASCII Char

Returns

R0 - numerical value. If invalid, return 0xFFFFFFFF

hw1_ascii_to_dec

Description

Converts A single ASCII DECIMAL character to its numerical value. Valid characters are 0-9.

Parameters

R0 - ASCII Char

Returns

R0 - numerical value. If invalid, return 0xFFFFFFFF

hw1_init

Description
Turns off all of the WS2812B LEDs by writing 0x000000 to each LED. Should be called with 'LOAD'

Parameters
NONE

Returns
Nothing

hw1_ledx

Description
Updates the color of the specified LED. All other LEDs should maintain their current color. You should make use of the ***hw1_update_leds*** to update the color of the LEDs.

Parameters
R0 – LED number to be updated. Assume 0 is the left most LED and 7 is the right most LED.

R1 – An unsigned 32bit number. Bits 31-24 are unused. Bits 23-0 represent the color value to write to the LED.

Returns
Nothing

hw1_wait

Description
Delays the examination of the next memory address by a variable amount of time (hang).

parameters
R0 – A 32-bit unsigned number representing the number of iterations of an empty for loop that must be executed.

Returns
Nothing

hw1_search_memory

Description

This function will search through memory a byte at a time looking for valid DISPLAY commands. When a valid DISPLAY command is found, carry out the corresponding behavior described in the DISPLAY command table above. Memory that holds invalid commands are ignored. This function will examine memory until and HALT command is found.

Your implementation of hw1_search_memory should make use the functions defined above when possible. You may add any additional functions that you feel are necessary.

Parameters

R0 : The starting address of the memory that contains the commands to update the LEDs.

Returns

Nothing

5.Assignment Submission

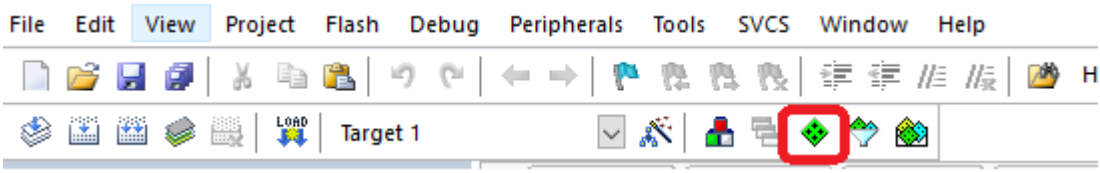
You will submit your entire project, including all source files, as a **ZIP** archive to the drop box on the course web site.

Make sure that you have modified main.c with your name before you submit the code.

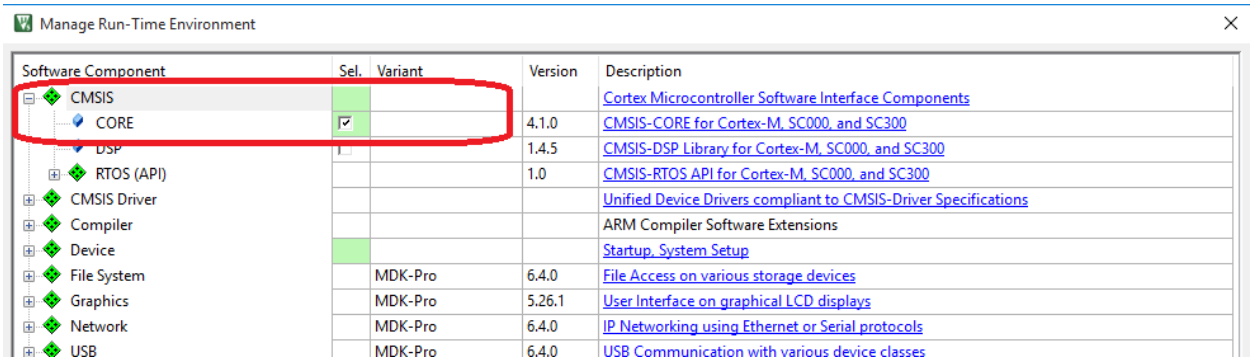
6.Keil uVision Setup

If your Keil uVision project won't compile, make sure that you have your uVision environment set up correctly.

Select 'Manage Run Time Environment' from the icons near the top of the screen (shown in red).



Expand the CMSIS item and select CORE.



7.Grading Rubric

As mentioned before, any evidence that code was shared between individuals will be treated as academic misconduct. You should always submit what you have done instead of submitting nothing. Late submission will result in 10 point deduction per day for this assignment.

Points	Description
30	All 8 LEDs are functioning properly with given RGB values
5	HANG command properly functioning
5	Functions are written as callee saved
5	Code properly functions against invalid input cases
5	Documentation style (commenting)
50	