EEE204 - Introduction to Embedded Systems

Experiment 1

Objectives

- Become familiar with the MSP430F5529 USB LaunchPad development kit and its basic components.
- Understand the basic anatomy of an assembly language program.
- Become familiar with the process of assembling, uploading, debugging, and executing an assembly language program using CCS.
- Learn how to examine and modify MSP430 memory and register contents.
- Learn how to do arithmetic operations MSP430.

Materials

- Code Composer Studio IDE
- MSP430F5529 USB LaunchPad development kit

PROCEDURE

1. Connect the LaunchPad to the computer via the USB cable provided with it.
2. Start Code Composer Studio (CCS).
3. Create new project **(File→New→CCS Project).**
4. Enter the name of the project.
5. Choose **"MSP430x5xx Family"** as project type and and choose **"MSP430F5529"** as your target device .
6. Select **"Empty-Assembly-only Project"** in **Project template and examples** section.
7. You don't need any more aditional settings, you can click **"Finish"** and create the project.
8. The program will automatically create a **"main.asm"** file. Do not change anything in this file for now, you will write your codes in "**Main loop here**" part.
9. Do the following tasks.

Experimental Work

E1

- Write a program to perform following operation in Assembly.
  23BC+12DE=369A
- Save your program by clicking **(File→Save).**
- Build the project **(Project→Build Project)** .
- Debug the application **(Run→Debug)** .
- Click **Run→Resume** to start the application.
- Observe the CPU registers and memory.
- Click **Run→Terminate** to stop the application and to exit the debugger.

```
;-------------------------------------------------------------------------------
; MSP430 Assembler Code Template for use with TI Code Composer Studio
;
;-------------------------------------------------------------------------------
            .cdecls C,LIST,"msp430.h"       ; Include device header file

;-------------------------------------------------------------------------------
            .def    RESET                   ; Export program entry-point to
                                            ; make it known to linker.
;-------------------------------------------------------------------------------
            .text                           ; Assemble into program memory.
            .retain                         ; Override ELF conditional linking
                                            ; and retain current section.
            .retainrefs                     ; And retain any sections that have
                                            ; references to current section.

;-------------------------------------------------------------------------------
RESET       mov.w   #__STACK_END,SP         ; Initialize stackpointer
StopWDT     mov.w   #WDTPW|WDTHOLD,&WDTCTL  ; Stop watchdog timer


;-------------------------------------------------------------------------------
; Main loop here
;-------------- ------------------------------------------------------------
main:
            mov.w #23BCH,R5
            mov.w #12DEH,R6
            add   R5,R6

            jmp main
;-------------------------------------------------------------------------------
; Stack Pointer definition
;-------------------------------------------------------------------------------
            .global __STACK_END
            .sect   .stack

;-------------------------------------------------------------------------------
; Interrupt Vectors
;-------------------------------------------------------------------------------
            .sect   ".reset"                ; MSP430 RESET Vector
            .short  RESET
```

E2

- Write a program to perform following operation in Assembly.
  4565-2565=2000
- Do the same steps in task 1.

```
; Main loop here
;-------------- ------------------------------------------------------------
main:
            mov.w #4565H,R5
            mov.w #2565H,R6
            sub   R6,R5

            jmp main
```

E3

- Write a program to perform following operation in Assembly.
  2565-4565=E000.
- Do the same steps in task 1.

```
; Main loop here
;-------------- ------------------------------------------------------------
main:
            mov.w #4565H,R5
            mov.w #2565H,R6
            sub   R5,R6

            jmp main
```