|  |
| --- |
| EEL 4742C-12 |
| EEL 4742 Laboratory |
| Experiment #4 |
|  |
| **Daniel Franco, Andrew Mendez**  **3/18/2014** |
|  |

|  |
| --- |
|  |

**Objective:** To develop a C language and an assembly language program that adds, subtracts and multiplies two hexadecimal digits. The resulting answer is then displayed on the LCD display of the MSP430FG4618 experimenter board.

**Apparatus List:**

* Dell Computer
* Monitor
* Keyboard
* Mouse
* CCS software
* MSP430FG4618

**Procedure and/or Design Methodology:**

This experiment had two separate parts to it, which we will describe in order as we worked on them. All programs should run indefinitely and start on a new line each time waiting for user input. We will provide screenshots for each of the programs we worked on.

Part One, C code:

1. Write a C language program that input from the keyboard two hexadecimal numbers via HyperTerminal. Next these two numbers are to be displayed in the HyperTerminal window and on the right most two digits of the LCD display on the experimenter board.
2. Write a C language program that inputs two ASCII hexadecimal digits followed by the “+” symbol to indicate addition. Next, the user enters a second set of two hexadecimal digits that are to be added to the first two hexadecimal digits. The addition result is shown on both the hyperterminal display as well as the LCD display.
3. Write a C language program that inputs two ASCII hexadecimal digits followed by the “−” symbol to indicate subtraction. Next, the user enters a second set of two hexadecimal digits that are to be subtracted from the first two hexadecimal digits. The subtraction result is shown on both the hyperterminal display as well as the LCD display.
4. Write a C language program that inputs two ASCII hexadecimal digits followed by the “\*” symbol to indicate multiplication. Next, the user enters a second set of two hexadecimal digits that are to be multiplied with the first two hexadecimal digits. The multiplication result is shown on both the hyperterminal display as well as the LCD display.
5. Write a program that merges steps 2, 3, and 4 together and uses the ASCII symbols “+”, “-“, and “\*” to determine which arithmetic operation to perform. This program should run indefinitely and start on a new line each time waiting for user input.

Part Two, Assembly code:

1. Write an assembly language program that inputs from the keyboard two hexadecimal numbers via hyperterminal. Next, these two numbers are displayed in the hyperterminal window and on the right most two digits of the LCD display on the experimenter board.
2. Write an assembly language program that inputs two ASCII hexadecimal digits followed by the “+” symbol to indicate addition. Next, the user enters a second set of two hexadecimal digits that are to be added to the first two hexadecimal digits.
3. Write an assembly language program that inputs two ASCII hexadecimal digits followed by the “-” symbol to indicate subtraction. Next, the user enters a second set of two hexadecimal digits that are to be subtracted from the first two hexadecimal digits. The subtraction result is shown on both the HyperTerminal display as well as the LCD display.
4. Write an assembly language program that inputs two ASCII hexadecimal digits followed by the “\*” symbol to indicate multiplication. Next, the user enters a second set of two hexadecimal digits that are to be multiplied with the first two hexadecimal digits. The multiplication result is shown on both the HyperTerminal display as well as the LCD display.
5. Write a program that merges steps 2, 3, and 4 together and uses the ASCII symbols “+”, “-“, and “\*” to determine which arithmetic operation to perform.

**Design Specification Plan:**

C- Programs:

For the C program part of this experiment, we relied on the code that we worked on during previous laboratory sessions, altered it to complete the specified tasks and added the necessary computations for it to accomplish the designated task. For the first program, we simply combined experiments three and four, while for the second part, after reading in the values, we added them together, printed them on to the board, and converted back to ASCII to be able to print on the HyperTerminal. We followed a very similar procedure for parts three and four.

Assembly programs:

Since the assembly programs were designed to complete the same tasks as the C programs, we followed the same plan, with the difference that this time we would be programming in a different language at a level closer to the hardware. The discussion provided to us in the pre-laboratory assignment helped us thoroughly because it gave us a direct idea of what we should do with the hardware in order to get it to complete the assigned tasks.

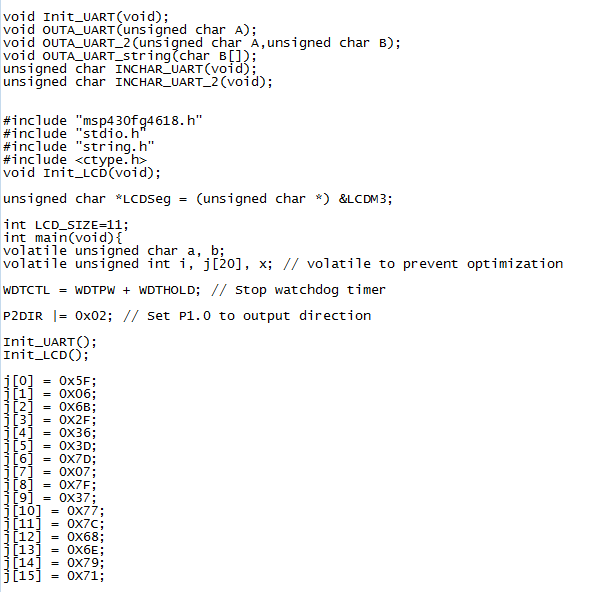
**Test Plan:**

In order to test our code we would be using the MSP-430FG4618 board. By running the code we had created in the Code Composer Studio, we would be able to communicate with the hardware through the Terminal and make sure that the input we had was dealt with correctly and displayed the proper values.

By using the HyperTerminal, we would interact with the board in order to ensure our code was working correctly and it would yield the correct output. This procedure was followed every time we completed a piece of code in any of the languages, C or Assembly.

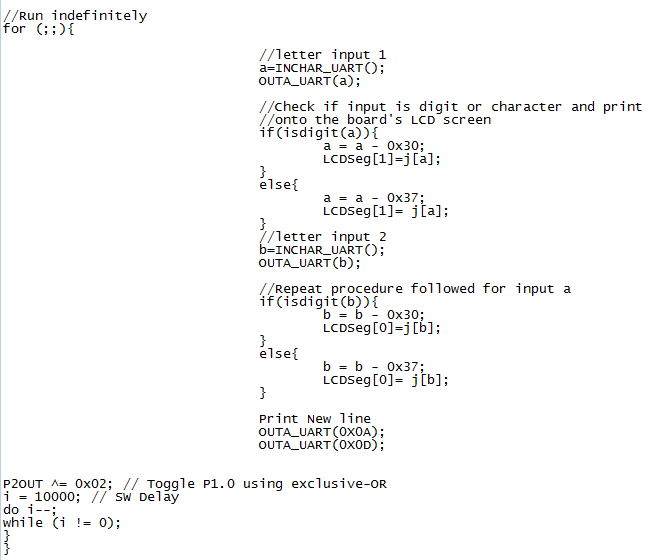
**Source Code:** (We will only be including the primary parts of the codes, if we copy the whole code into here, the report will be too long and we would not want to print too many pages)

For future purposes, this is how we initialized our programs

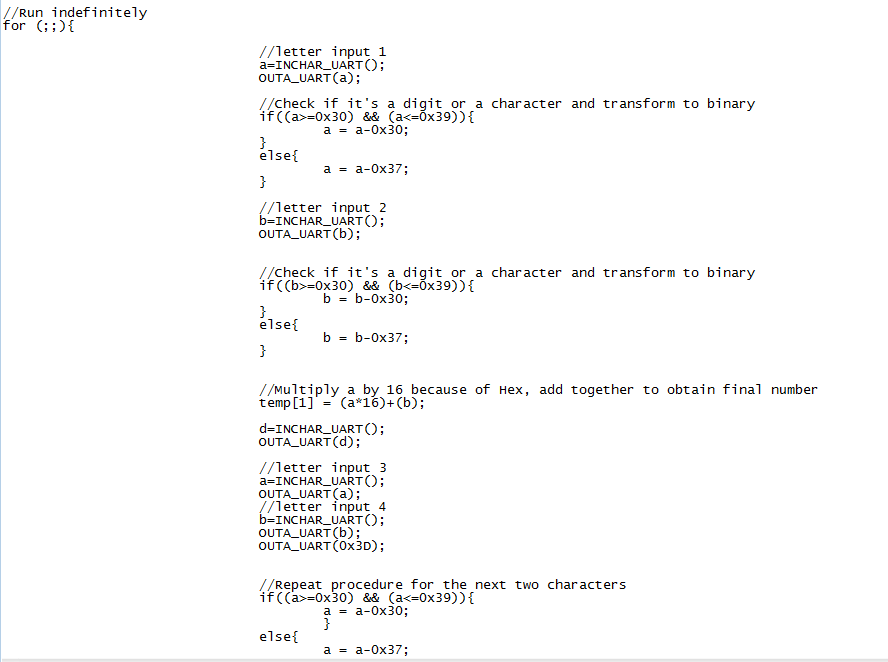


**Part One, C:**

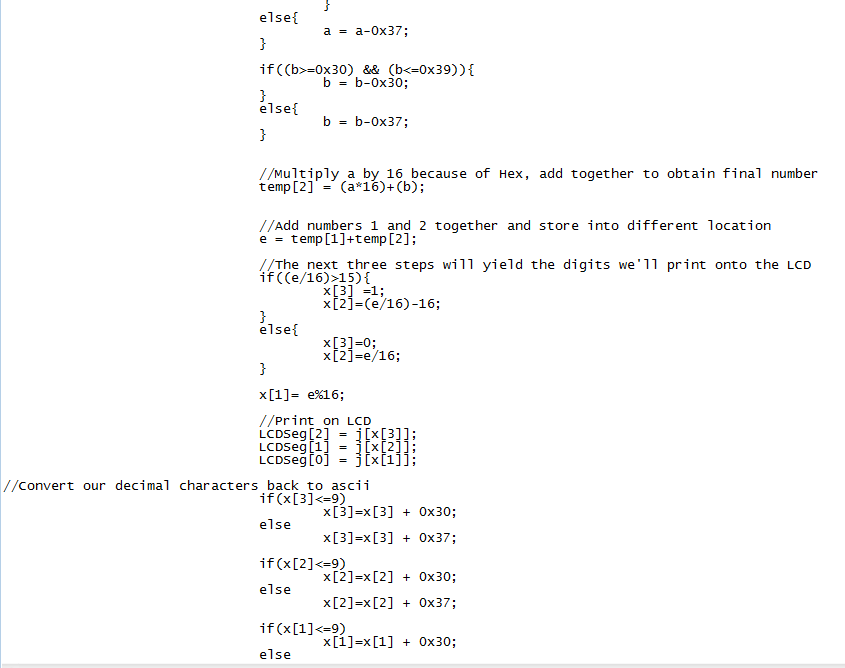
Program 1.



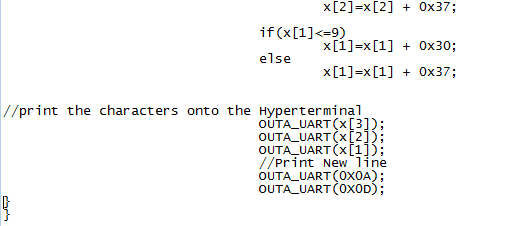
Program 2, Part 1



Program 2, Part 2



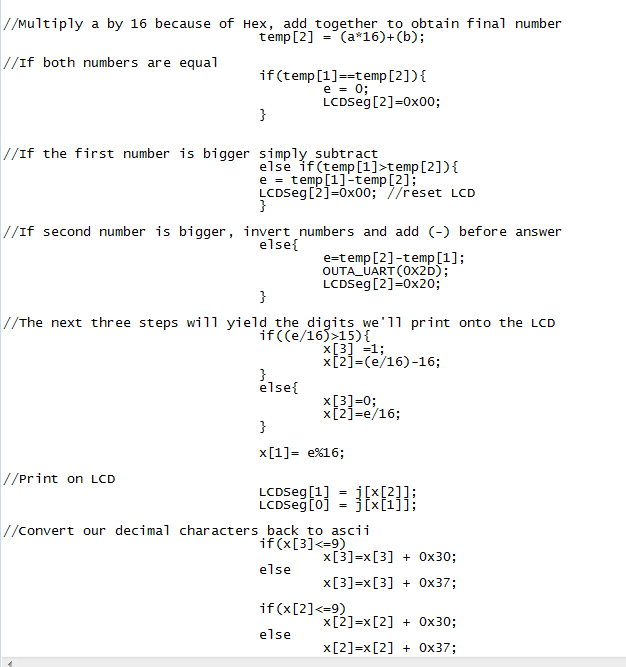
Program 2, Part 3



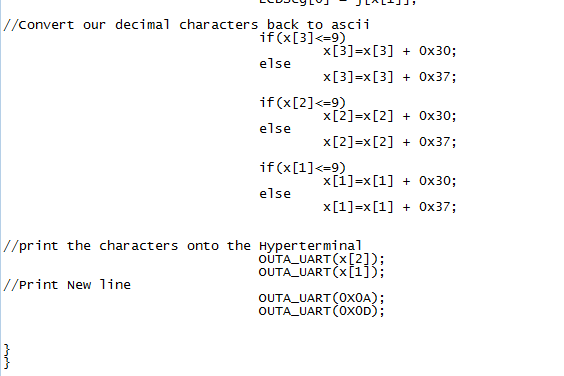
Program 3:

The first part of the code is the same as program 2, it will input two characters or digits and convert them to binary.

Program 3



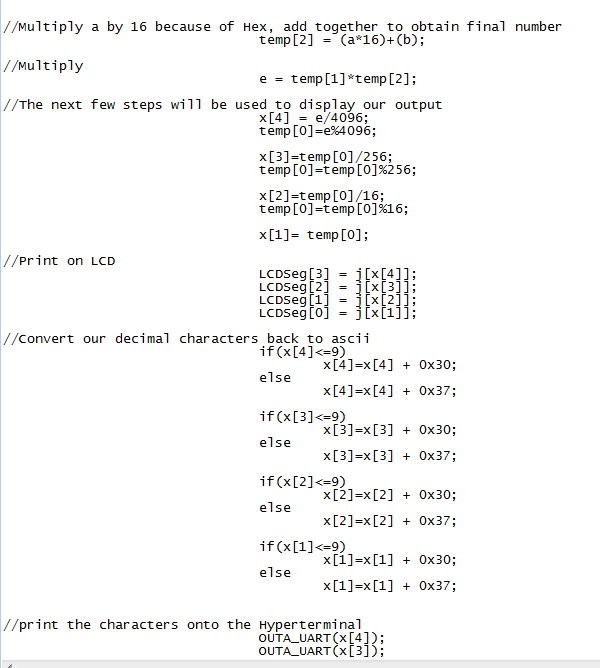
Program 3



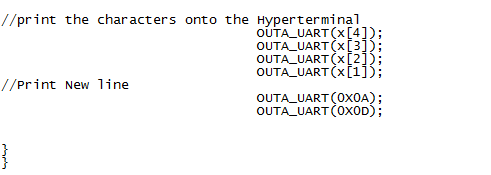
Program 4:

Program 4, as well as program 3 and program 2, starts out by inputting the 4 characters, in sets of 2, and converting them to binary so they can be operated with each other. The following is the code:

Program 4



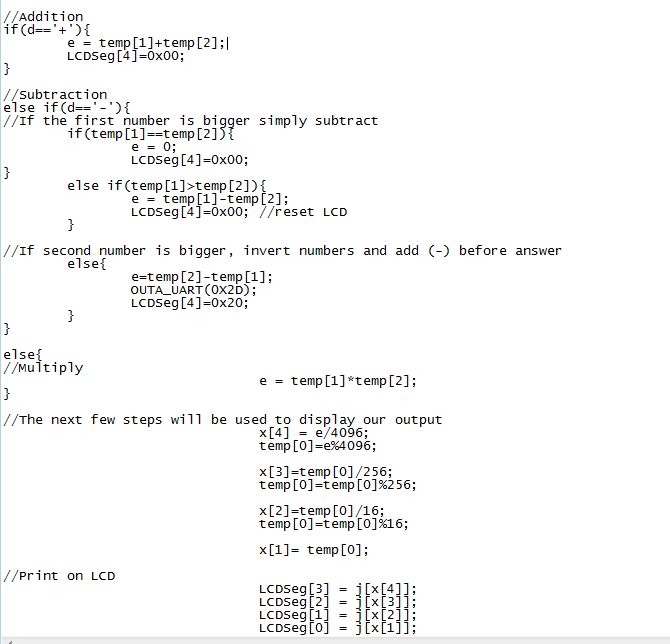
Program 4



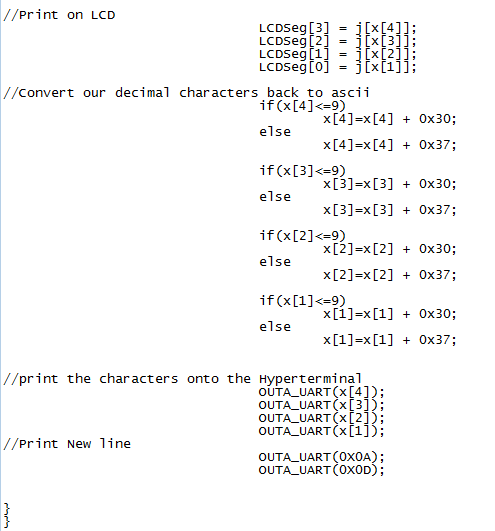
Program 5:

This program, as well as the other 3, starts out by receiving input from the user, so we will not show that part of the code again in order to save space, and paper. There will be a series of ‘if’ statements that will control what happens along the code.

Program 5



Program 5



**Part Two:**

**Conclusion:** Write a conclusion