**CMPE-250 Assembly and Embedded Programming**

**Laboratory Exercise Eight**

**Multiprecision Arithmetic**

By submitting this report, I attest that its contents are wholly my individual writing about this exercise and that they reflect the submitted code. I further acknowledge that permitted collaboration for this exercise consists only of discussions of concepts with course staff and fellow students. Other than code provided by the instructor for this exercise, all code was developed by me.

Shubhang Mehrotra

Performed 25th March 2021.

Submitted 1st April 2021.

Lab Section 1

Instructor: Muhammad Shaaban

|  |  |  |
| --- | --- | --- |
| TA: |  | Anthony Bacchetta  Aidan Trabuco |
|  |  | Sam Myers |
|  |  | Payton Burak |

Lecture Section 1

Lecture Instructor: Dr. Roy Melton

**Procedure Screen Capture**

The activity explored the KL05Z Freedom Board further by writing a program to perform Multiprecision Arithmetic operation on 96-bit numbers. The activity was divided into to parts. The first part involved writing an “AddIntMultiU” Subroutine to handle addition of two 96-bit numbers. The subroutine utilized a loop to add numbers by looking at each “word” length of the number at a time.

Additionally, a “GetHexIntMulti” subroutine and a “PutHexIntMulti” subroutine was added to the code to accept and print out the 96-bit numbers in Hexadecimal format. PutHexIntMulti was a simple subroutine which looped over the entire word, concatenated the words together and output it to the screen. GetHexIntMulti is a complex subroutine which accepts user input using the previously define GetChar routine and stores it in a memory address provided to it. Since the user input received is in ASCII codes, it then translates it into Hexadecimal numbers. In doing so, the numbers are saved as zero-extended bytes, which are then required to be packed together to have the required value.

This connection to the board was again established via UART Polling at 9600 Baud rate using PuTTY and its “Serial” connectivity functionality. The code for Multiprecision arithmetic was written and downloaded to the KL05Z board using the Keil IDE. It was then tested against the following cases:

* Simple Addition of two numbers
* Acceptance of both upper-case and lower-case letters
* Detection of invalid hex characters
* Detection of sum overflow.
* Detection of Carry generating addition.

The result of the test run is shown in Figure 1.

Text

Description automatically generated

Figure 1. Result

Figure 1 shows the PuTTY Serial window with the results of the various inputs successfully output to the screen. The code is able to add large hex numbers, detect overflow, detect valid hex input and prompt the user for valid input and is able to perform addition that generates carry, both over the same word length or over the next one.

**Memory Ranges**

For Lab Activity 8, the Listing file and the Map file were generated and analyzed for the memory addresses of the items listed in Table 1, which lists the start addresses and the size occupied by the item in the memory in bytes.

Table 1. Memory Ranges

|  |  |  |
| --- | --- | --- |
| Object | Memory Address  Start | Size  (bytes) |
| AddIntMultiU | 0x0000054E | 44 |
| GetHexIntMulti | 0x000004CC | 110 |
| main  (Reset\_Handler in map) | 0x00000410 | 138 |
| PutHexIntMulti | 0x0000053A | 20 |

The Start Address for the objects were obtained from the Map file. Under the section marked “Image Symbol Table,” is a table which lists the execution address of each entity in the source file. Upon inspection of the table, the start addresses, and the size can be obtained. The End address, if required, can be obtained by adding the start address and the size and subtracting 1.