CMPE-250 Assembly and Embedded Programming

Laboratory Exercise 8

Multiprecision Arithmetic

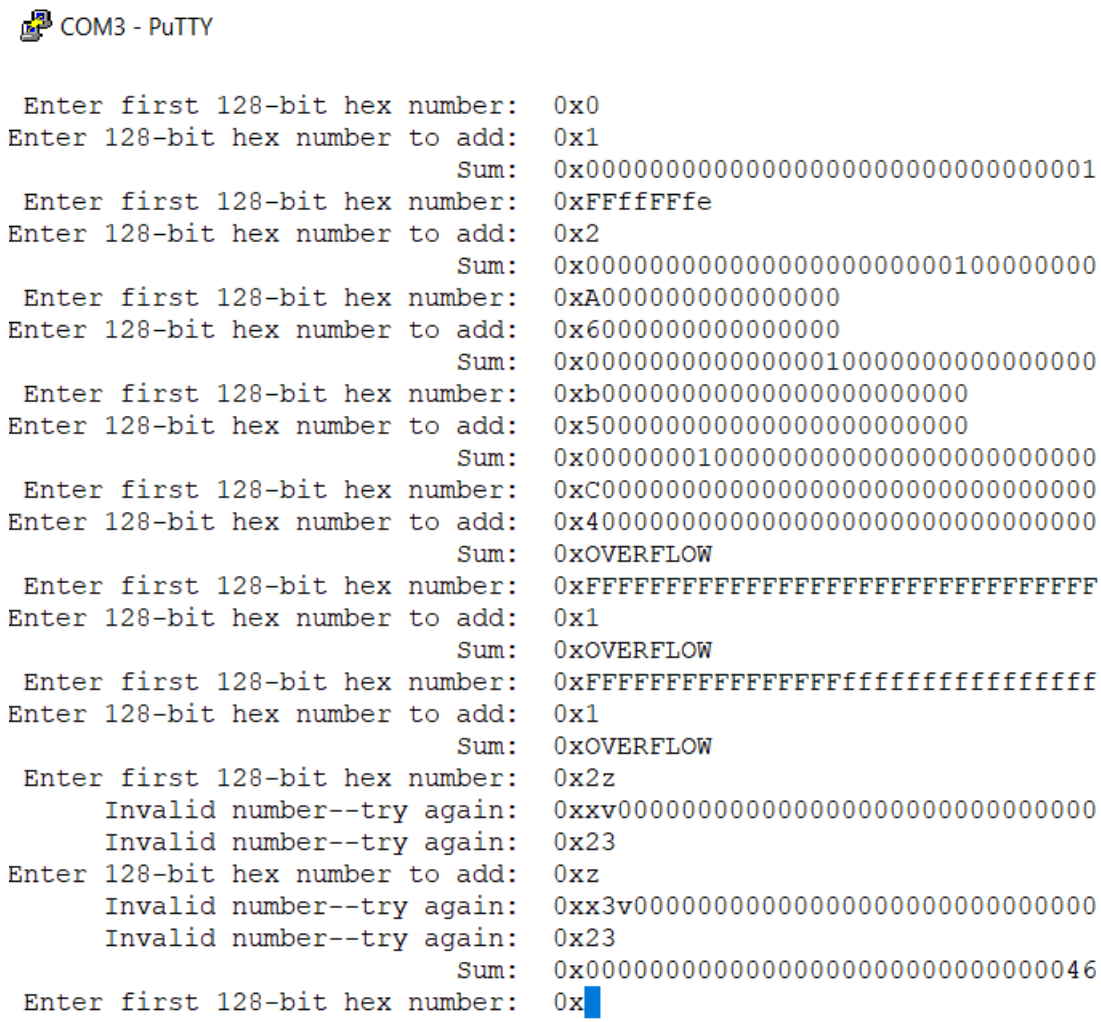
By submitting this report, I attest that its contents are wholly my individual writing about this exercise and that they reflect the submitted code. I further acknowledge that permitted collaboration for this exercise consists only of discussions of concepts with course staff and fellow students. Other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

Andrei Tumbar
Submitted: 10-27-20

Lab Section:	5
Instructor:	Gordon Werner
TA:	Tianran Cui
	Anthony Bacchetta
Lecture Section:	1
Lecture Instructor:	Melton

Demonstration

A screen capture of the terminal output of the final program was taken. The screen capture was taken after the final demonstration was performed.



```
COM3 - PuTTY

 Enter first 128-bit hex number:  0x0
Enter 128-bit hex number to add:  0x1
                            Sum:  0x00000000000000000000000000000001
 Enter first 128-bit hex number:  0xFFffFFfe
Enter 128-bit hex number to add:  0x2
                            Sum:  0x00000000000000000000000100000000
 Enter first 128-bit hex number:  0xA000000000000000
Enter 128-bit hex number to add:  0x6000000000000000
                            Sum:  0x00000000000000010000000000000000
 Enter first 128-bit hex number:  0xb00000000000000000000000
Enter 128-bit hex number to add:  0x500000000000000000000000
                            Sum:  0x00000001000000000000000000000000
 Enter first 128-bit hex number:  0xC000000000000000000000000000000
Enter 128-bit hex number to add:  0x400000000000000000000000000000
                            Sum:  0xOVERFLOW
 Enter first 128-bit hex number:  0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Enter 128-bit hex number to add:  0x1
                            Sum:  0xOVERFLOW
 Enter first 128-bit hex number:  0xFFFFFFFFFFFFFFFFffffffffffffffff
Enter 128-bit hex number to add:  0x1
                            Sum:  0xOVERFLOW
 Enter first 128-bit hex number:  0x2z
      Invalid number--try again:  0xxv000000000000000000000000000000
      Invalid number--try again:  0x23
Enter 128-bit hex number to add:  0xz
      Invalid number--try again:  0xx3v000000000000000000000000000000
      Invalid number--try again:  0x23
                            Sum:  0x00000000000000000000000000000046
 Enter first 128-bit hex number:  0x
```

Figure 1: Terminal screen capture

The first operation tests that basic adding functionality works with 16-byte integers. This test is an important baseline to verify that the numbers were stored in little endian format. The next addition tests both the case-insensitive support along with the carry bit from the first word to the second. The next two cases also test that the carry bit is functional between each of the word boundaries.

The next two inputs will nexts that is a carry on the final bit is set, the program will indicate that the addition operation overflowed.

Finally the last two addition operations test whether invalid input will properly re-prompt the user for both the addend and augend.

Subroutine addresses

Table 1: Code section offsets and endings.

Subroutine	Address	Size (bytes)
main	0x00000410	140
AddIntMultiU	0x0000049d	56
GetHexIntMulti	0x000004d5	116
PutHexIntMulti	0x00000549	26