CSCI 2500 Assignment 4

Carry Lookahead Adder

Due Date: Wednesday, November 1st, 11:59:59 PM

For this assignment you will implement a *carry lookahead adder* as described in cla.pdf. This adder improves upon the partial adder that you implemented in Lab 7. In that lab you constructed a 4-bit ripple carry adder. Now you must build a 64-bit carry lookahead adder. This can be achieved by building a "hierarchy" of progressively larger groups, sections, etc. See the PDF for more information. Additional details may be found starting on page B-38 in your textbook. Your CLA must also support the subtraction operation. This is discussed on page B-29 of your textbook. To summarize, recall that in order to negate a number in two's complement we have to invert all of the bits and then add 1. This can be achieved by setting the initial carry-in to 1 instead of 0. Your program must support hexadecimal (base 16) input. C natively supports hexadecimal; see the documentation for the `%x` specifier for `printf()`/`scanf()`. You should also be using the bitwise operations C exposes as mentioned in Lab 7.

I recommend that you start with a simple ripple carry adder so you can check your answers. It will also be helpful to quickly enable debugging output to check that your *generates* and *propagates* (see the PDF) are what you expect.

Sample output below.

```
bash-3.2$ ./cla
Enter A (hex):
1
Enter B (hex):
2
Add (0) or subtract (1):
1

A is 0000000000000001 or 1
B is 0000000000000002 or 2
Inverting 2
B (bin) : 1111111111111111111111111111111111111111111111111111111111111101

Calculate sum, S:

A (bin) : 0000000000000000000000000000000000000000000000000000000000000001
B (bin) : 1111111111111111111111111111111111111111111111111111111111111101
S (bin) : 1111111111111111111111111111111111111111111111111111111111111111

S is ffffffffffffffff or -1

bash-3.2$ ./cla
Enter A (hex):
```

```
1
Enter B (hex):
1
Add (0) or subtract (1):
0

A is 0000000000000001 or 1
B is 0000000000000001 or 1

Calculate sum, S:

A (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000000001
B (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000000001
S (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000000010

S is 0000000000000002 or 2

bash-3.2$ ./cla
Enter A (hex):
1
Enter B (hex):
1
Add (0) or subtract (1):
1

A is 0000000000000001 or 1
B is 0000000000000001 or 1
Inverting 1
B (bin) : 11111111111111111111111111111111111111111111111111111111111111111111111111111110

Calculate sum, S:

A (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000000001
B (bin) : 11111111111111111111111111111111111111111111111111111111111111111111111111111110
S (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000000000

S is 0000000000000000 or 0

bash-3.2$ ./cla
Enter A (hex):
15
Enter B (hex):
3
Add (0) or subtract (1):
0

A is 0000000000000015 or 21
B is 0000000000000003 or 3

Calculate sum, S:

A (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000010101
B (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000000011
S (bin) : 00000000000000000000000000000000000000000000000000000000000000000000000000011000
```

```
S is 0000000000000018 or 24

bash-3.2$ ./cla
Enter A (hex):
15
Enter B (hex):
3
Add (0) or subtract (1):
1

A is 0000000000000015 or 21
B is 0000000000000003 or 3
Inverting 3
B (bin) : 11111111111111111111111111111111111111111111111111111111111111100

Calculate sum, S:

A (bin) : 00000000000000000000000000000000000000000000000000000000000010101
B (bin) : 11111111111111111111111111111111111111111111111111111111111111100
S (bin) : 00000000000000000000000000000000000000000000000000000000000010010

S is 0000000000000012 or 18

bash-3.2$ ./cla
Enter A (hex):
deadbeef
Enter B (hex):
123abc
Add (0) or subtract (1):
0

A is 00000000deadbeef or 3735928559
B is 0000000000123abc or 1194684

Calculate sum, S:

A (bin) : 00000000000000000000000000000000110111101010110110111111011101111
B (bin) : 00000000000000000000000000000000000000000000000100100011101010111100
S (bin) : 00000000000000000000000000000000110111101011111111111100110101011

S is 00000000debff9ab or 3737123243

bash-3.2$ ./cla
Enter A (hex):
deadbeef
Enter B (hex):
123abc
Add (0) or subtract (1):
1

A is 00000000deadbeef or 3735928559
B is 0000000000123abc or 1194684
Inverting 1194684
```

```
B (bin) : 11111111111111111111111111111111111111111111101101110001010100011

Calculate sum, S:

A (bin) : 00000000000000000000000000000001101111010101101101111101110111
B (bin) : 11111111111111111111111111111111111111111111101101110001010100011
S (bin) : 00000000000000000000000000000001101111010011011100001000011001l

S is 00000000de9b8433 or 3734733875

bash-3.2$ ./cla
Enter A (hex):
de1ec7ab1ebeef
Enter B (hex):
5ca1ded
Add (0) or subtract (1):
0

A is 00de1ec7ab1ebeef or 62521287748009711
B is 0000000005ca1ded or 97132013

Calculate sum, S:

A (bin) : 00000000110111100001111011000111101010110001111010111110111011l1
B (bin) : 00000000000000000000000000000000000010111001010000111011110l101
S (bin) : 00000000110111100001111011000111101100001110100011011100110llll00

S is 00de1ec7b0e8dcdc or 62521287845141724

bash-3.2$ ./cla
Enter A (hex):
de1ec7ab1ebeef
Enter B (hex):
5ca1ded
Add (0) or subtract (1):
1

A is 00de1ec7ab1ebeef or 62521287748009711
B is 0000000005ca1ded or 97132013
Inverting 97132013
B (bin) : 11111111111111111111111111111111111111101000110101111000100001001O

Calculate sum, S:

A (bin) : 00000000110111100001111011000111101010110001111010111110111011l1
B (bin) : 11111111111111111111111111111111111111101000110101111000100001001O
S (bin) : 00000000110111100001111011000111101001010101010010100001000000010

S is 00de1ec7a554a102 or 62521287650877698
```