CSE2003: System Programming (Spring 2009)

Programming Assignment #1:
Multiplication of two 32-bit signed integers

Due: Fri March 27, 11:59PM

1. Introduction

The purpose of this assignment is to become more familiar with the binary representation of signed integers and to understand what happens during the multiplication of two signed integers.

2. Problem specification

2.1 Overview

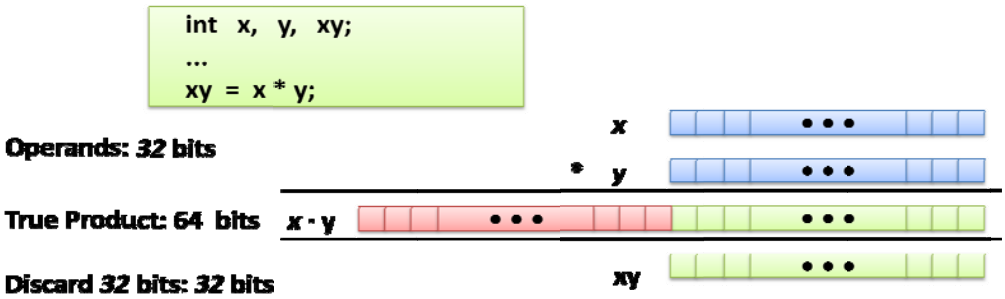
Write a C function named `mult32bits()` which receives two signed integers and computes the multiplication of those integers. The prototype of `mult32bits()` is as follows:

```
void mult32bits (int x, int y, int *xyh, int *xyl);
```

The first two arguments, `x` and `y`, represent the multiplicand and the multiplier, respectively. The function `mult32bits`() should store the result of `x * y` in the memory locations pointed to by `xyh` and `xyl`. Since the multiplication of two signed 32-bit integers produces a 64-bit result, the high-order 32 bits should be saved in the memory location pointed to by `xyh` and the low-order 32bits in the memory location designated by `xyl`.

2.2 Backgrounds

When two integers `x` and `y` are represented as `w`-bit two's complement numbers, their product `x * y` requires as many as `2w` bits to represent in two's complement form. Since signed integers are represented as 32 bits in 32-bit machines (including PCs running a 32-bit version of Microsoft Windows or Linux), the multiplication of two signed integers produces a 64-bit result. In C, however, signed multiplication is performed by truncating the 64-bit product to 32 bits (For further details, please refer to Section 2.3.5 of the textbook).



Instead of truncating the high-order 32 bits, `mult32bits()` stores the high-order 32 bits into `xyh` (the red part in the above figure), and the low-order 32 bits into `xyl` (the green part in the above figure). In this way, you can get the true 64-bit product.

2.3 Restrictions

When you implement the function `mult32bits()`, you should use only `int` type variables. In addition, you are allowed to use only integer arithmetic and logical operations inside `mult32bits()`.

2.4 Verification of your result

Since a "`long long`"-type integer consists of 64 bits in 32-bit machines, another way to obtain the true product of two 32-bit integers is to cast the result into the `long long` type, as shown in the following code example.

```
int x, y;
long long z;
...

z = (long long) x * y;
```

Therefore, your result should be identical to the result obtained by the above code example. More specifically, at the end of calling `mult32bits (x, y, &xyh, &xyl), xyh` should have the value of `(int) (z >> 32)` and `xyl` should have the value of `(int) (z & 0xffffffff)`.

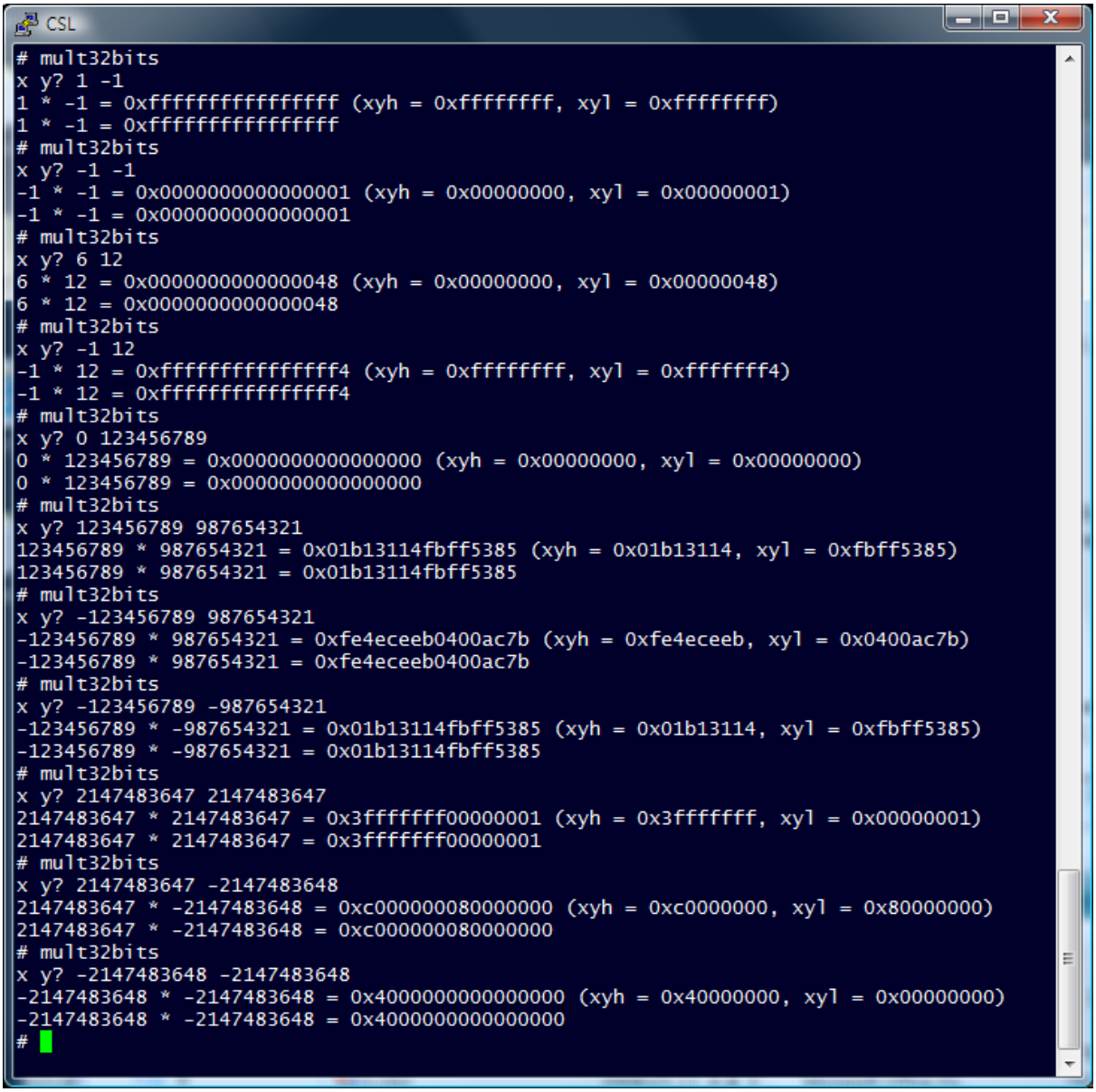
Please be careful when the result is a negative number. The result also should be represented in two's complement form.

3. Example

The following code shows the basic skeleton of your program and how to verify your result.

```
#include <stdio.h>

void mult32bits (int x, int y, int *xyh, int *xyl)
{
       /* Only use int type here */
       ...

       *xyh = ...
       *xyl = ...
}

int main()
{
       int x, y, xyh, xyl;
       long long z;

       printf ("x y ? ");
       scanf ("%d %d", &x, &y);

       mult32bits (x, y, &xyh, &xyl);

       /* The following outputs should be same */
       printf ("%d * %d = 0x%08x%08x (xyh = 0x%08x, xyl = 0x%08x)\n",
                     x, y, xyh, xyl, xyh, xyl);
       printf ("%d * %d = 0x%016llx\n", x, y, (long long) x * y);
}
```

Some sample runs:



4. Hand in instructions

- Make sure you have included your name and the student ID in the header comment of your program.
- The source file name which contains the function `mult32bits()` should be "`YourStudentID.c`" (e.g., `2008310123.c`).
- "`YourStudentID.c`" file should contain only the source code of `mult32bits()`. Do not include any other functions such as `main()`.
- Prepare a separate document in PDF format (most preferred, but other formats, such as .txt, .doc, and .hwp, are also allowed), which explains the design and implementation of your code. The document should be named "`YourStudentID.pdf`".
- Send a mail to [jinsookim at skku.edu] AND [cse2003skku at gmail.com] (Yes! I have created an account at gmail for this course!) with attaching two files, "`YourStudentID.c`" and

"`YourStudentID.pdf`". The subject line of the mail should be in the following format:

```
[CSE2003] PA#1, YourStudentID, YourName
```

5. Logistics

- You will work on this assignment alone.
- The submission status will be posted on the course homepage at http://csl.skku.edu/CSE2003S09.
- Only the assignments submitted before the deadline will receive the full credit. 25% of the credit will be deducted for every single day delay.
- Any attempt to copy others' work will result in heavy penalty (for both the copier and the originator). Don't take a risk.

Good luck!

Jin-Soo Kim
Computer Systems Laboratory
School of Information and Communication Engineering
Sungkyunkwan University