Goal: Learning how to implement 64x64 signed integer multiplication on a 32-bit cpu.

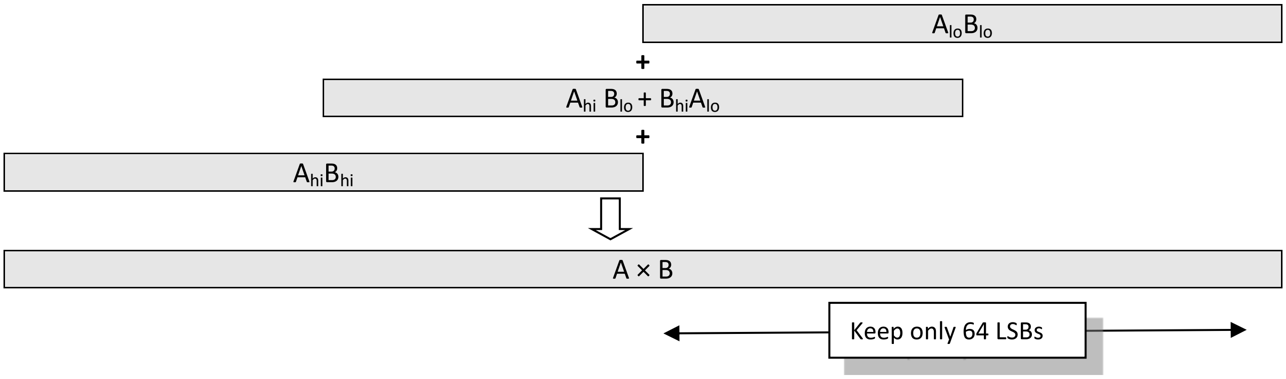
PART I: Background

Most 32-bit CPUs can multiply two 32-bit integers in a single instruction, but 64x64-bit multiplication must be implemented as a sequence of instructions or as a subroutine. When you multiply two 64-bit integers in C, the compiler generates code that calls a library function to do the multiplication. The purpose of this assignment is write our own version of that routine in assembly.

Approach: Multiplying two 64-bit unsigned integers on a 32-bit CPU requires breaking each number into two 32-bit halves, i.e.: $N = 2^{32} \times N_{hi} + N_{lo}$, where N_{hi} and N_{lo} are respectively the most and least significant 32-bit halves. Thus:

$$A \times B = (2^{32}A_{hi} + A_{lo}) \times (2^{32}B_{hi} + B_{lo}) = 2^{64}A_{hi}B_{hi} + 2^{32}(A_{hi}\,B_{lo} + B_{hi}A_{lo}) + A_{lo}B_{lo}$$

Each partial product term requires computing the 64-bit product of two 32-bit numbers. After shifting left 0, 32, or 64 bits according to the corresponding power of 2 scale factor, these partial products are then added to compute the final 128-bit product.



Multiplication of 64-bit integers in C keeps only the least significant 64 bits of the product, so one entire partial product term and the most significant half of two others are not needed. Note that one algorithm works for both signed and unsigned multiplication, since the only difference is in the discarded most-significant half of the product.

PART II: Preparation

1. Download the ZIP file called “Lab Assignments.zip” from the course website on Camino.
2. Unzip the file to your desktop. This should create a folder called “Lab Assignments”. Open the folder.
3. Find and double click on the file called “COEN20.eworkspace”. This will open the EmBitz Integrated Development Environment (IDE) and display the projects for all the lab assignments.
4. If step 3 did not open EmBitz, find the program on the Start Menu and open it. Once EmBitz is open, click on “File > Open” in the upper left-hand corner. In the dialog window that opens, find the pull-down menu in the bottom right and select “EmBitz workspace files”. Then in the middle of the dialog window, navigate to your “Lab Assignments” folder, select the file “COEN20.eworkspace”, and click on “Open”.

PART III: Creating Your Solution

1. Find the project (lab assignment) name in the “Management” panel on the left side of the screen. Make sure that the name is in **boldface**, which indicates that the project is *Active*. If not, right-click on it and select “Activate project”.
2. Expand the project by clicking on the “+” sign immediately to the left of its name. Do the same for any subgroups found within it.
3. Double click on both main.c and SMul64x64inASM.s to view both files. Complete the assembly language function SMul64x64inASM using the C function SMul64x64inC as a guide.
4. To compile the program, right-click on the project name and select “Build”. If there are any error or warning messages displayed, correct the source code of the function and recompile.

   Hint: Function key F7 is a short-cut for “Build” for the Active project.
5. Connect the STM32F4 Discovery board to a USB port on your computer. This provides both power and a download connection to the device. To download the program to the board, click on “Debug” → “Start/stop Debug Session”.

   Hint: Function key F8 is a short-cut for “Debug” →Start/stop Debug Session”.
6. To run the program, click on “Debug” → “Run”. When the program begins to run it will display the first test case and pause. Press the left button to sequence through all the test cases. Verify that your program behaves as expected.
7. To end the debug session, click on “Debug” → Start/stop Debug Session” again.
8. Demonstrate your working program to the Teaching Assistant.
9. Upload your final version of file SMul64x64inASM.s to Camino.