Homework Quiz (HW #2)
February 16, 1999
CS152 Computer Architecture and Engineering

This quiz covers one of the problems from homework #2.
Good Luck!

<table>
<thead>
<tr>
<th>Your Name:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SID Number:</td>
<td></td>
</tr>
<tr>
<td>Discussion Section:</td>
<td></td>
</tr>
</tbody>
</table>

Total:
A better multiply for the Single Instruction Computer

In the homework, you were introduced to the “Subtract and branch if negative” instruction:

\[
\text{sbn } a, b, c \quad \text{# Mem}[a] = \text{Mem}[a] - \text{Mem}[b]; \text{ If Mem}[a] < 0 \text{ goto } c;
\]

Problem 3.30 asked you to develop a multiply routine that was not particularly efficient. Consider the following alternative for a 16-bit computer. This performs \( c = a \times b \), modifying \( b \) in the process:

\[
c = 0;
\text{for (index = 32768; index } \geq 1; \text{ index } = \text{index } / 2) \{ \\
\quad \text{If (b } \geq \text{ index) } \{ \ b = b - \text{index}; \ c = c - \text{index } \times \text{a}; \ } \}
\}
\]
\[
c = -c;
\]

To implement this, we will assume a macro facility that lets us generate groups of instructions many times.

1. Create a macro called “inner” which performs the inner instructions of our multiply. This takes 4 memory locations as input: \( a, b, \text{index}, \) and \( \text{scaled}_a \), and computes:

\[
\text{if (b } \geq \text{ index) } \{ \ b = b - \text{index}; \ c = c - \text{scaled}_a; \}
\]

Assume one temporary memory location (temp). Note that your code can contain an empty label called “exit;” at the end.

\[
\text{inner}(b, c, \text{index, scaled}_a, \text{temp}):
\]

2. Create a macro called “double” which puts \((2\times\text{Mem}[in]) \Rightarrow \text{Mem}[out]\), using \( \text{Mem[temp]} \) as a temporary location. This must not alter the “in” memory location:

\[
\text{double}(\text{out, in, temp}):
\]
3. Using your macros, write a routine “multiply” that will take a and b as input, and place the result in c. **Make this work for 4 bit variables.** Assume that the memory location “one” contains the constant 1. Also, assume that you can use your macros like instructions:

For instance:  
\[
\text{double}(\text{two, one, mytemp}) \\
\text{inner}(\text{q, r, other, two, anothertemp})
\]

The macro facility expands each macro call into the set of instructions in the macro definition, while substituting the actual arguments for formal ones. Thus, for instance, the first call to “double” above will substitute \text{two}⇒\text{out}, \text{one}⇒\text{in}, and \text{mytemp}⇒\text{temp} in your definition of double.

*Hint: you must first create a bunch of constants like 2 and (2 * a), then use them.*

Multiply_4_bit:

4. If you were to make a 16-bit version of this, what is the worst-case number of cycles that it would take? Is this much better than the multiply from the homework?