
CSE 220 Computer Organization

Lecture 21

Hussein Badr

Computer Science, Stony Brook University

<http://www.cs.sunysb.edu/~cse220>

Integer Multiplication (1/9)

- Consider unsigned (positive) integers for now.
- Simple grade-school method.
- Form partial products and add.

Example:

	1	2	3		Multiplicand	
x	3	2	1		Multiplier	
<hr/>						
	1	2	3	}	Partial Products	
	2	4	6			<input type="checkbox"/>
	3	6	9			<input type="checkbox"/>
<hr/>						
	3	9	4	8	3	Product

- Size of the final product: six digits max., since our numbers are 3 digits wide.

Integer Multiplication (2/9)

- This method can be used for integers in binary form as well.

Example: $5_{10} \times 6_{10} = 30_{10}$

5_{10} & 6_{10} are 3 bits wide.

$$\begin{array}{r} 101 \text{ multiplicand} \\ \times 110 \text{ multiplier} \\ \hline 000 \\ 101\ \square \\ 101\ \square\ \square \\ \hline 11110 \end{array}$$

This is 30_{10} .

Observe that partial products are either zero or just the multiplicand shifted left by some number of bits.

How do we make use of this?

Here the size of the final product could be up to be six bits.

- Straightforward implementation.
- Since our numbers are 32 bits, we use a 64-bit adder.

Integer Multiplication (3/9)

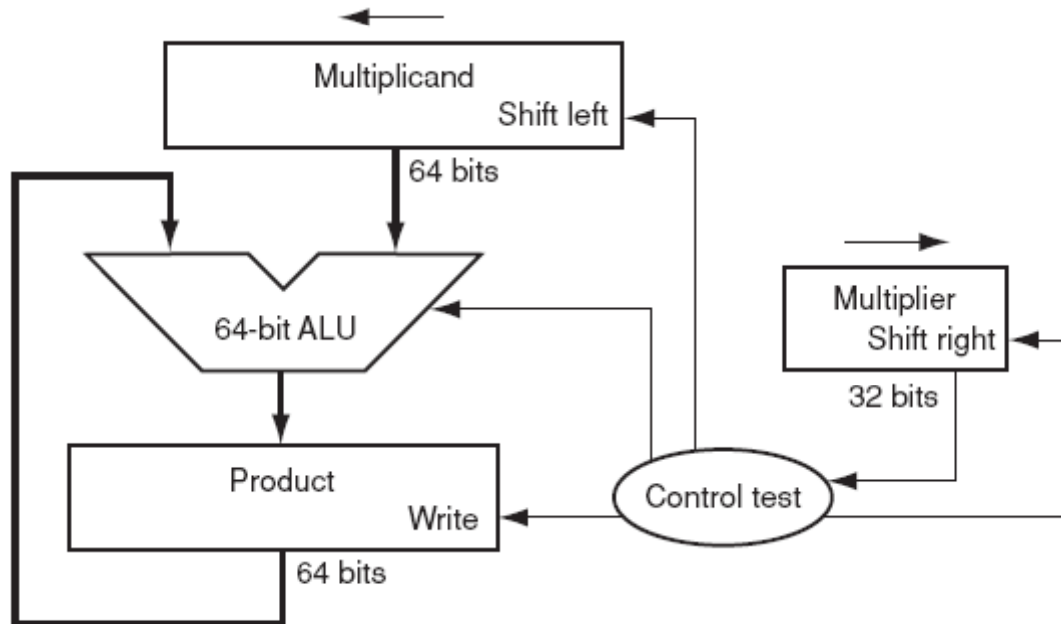
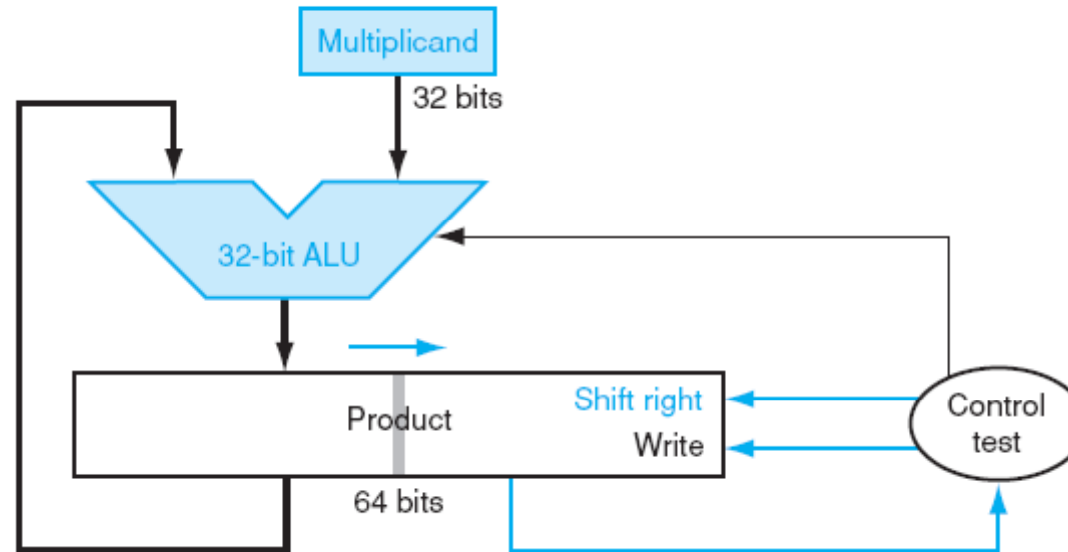


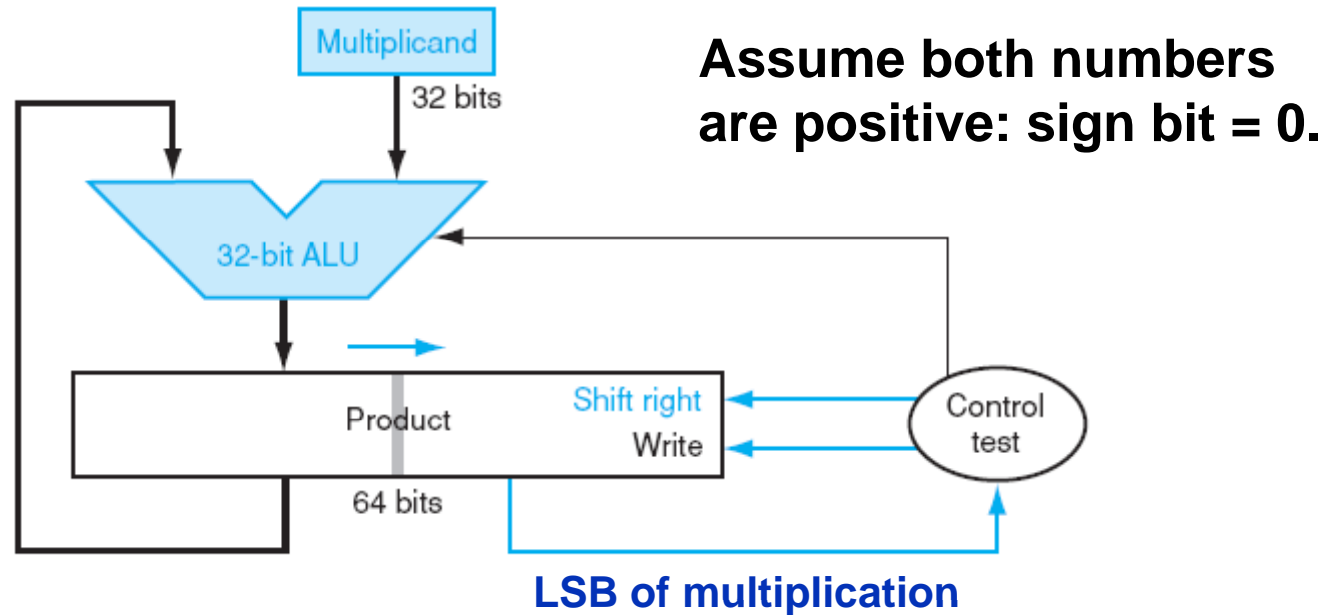
FIGURE 3.4 First version of the multiplication hardware. The Multiplicand register, ALU, and Product register are all 64 bits wide, with only the Multiplier register containing 32 bits. The 32-bit multiplicand starts in the right half of the Multiplicand register and is shifted left 1 bit on each step. The multiplier is shifted in the opposite direction at each step. The algorithm starts with the product initialized to 0. Control decides when to shift the Multiplicand and Multiplier registers and when to write new values into the Product register.

Integer Multiplication (4/9)



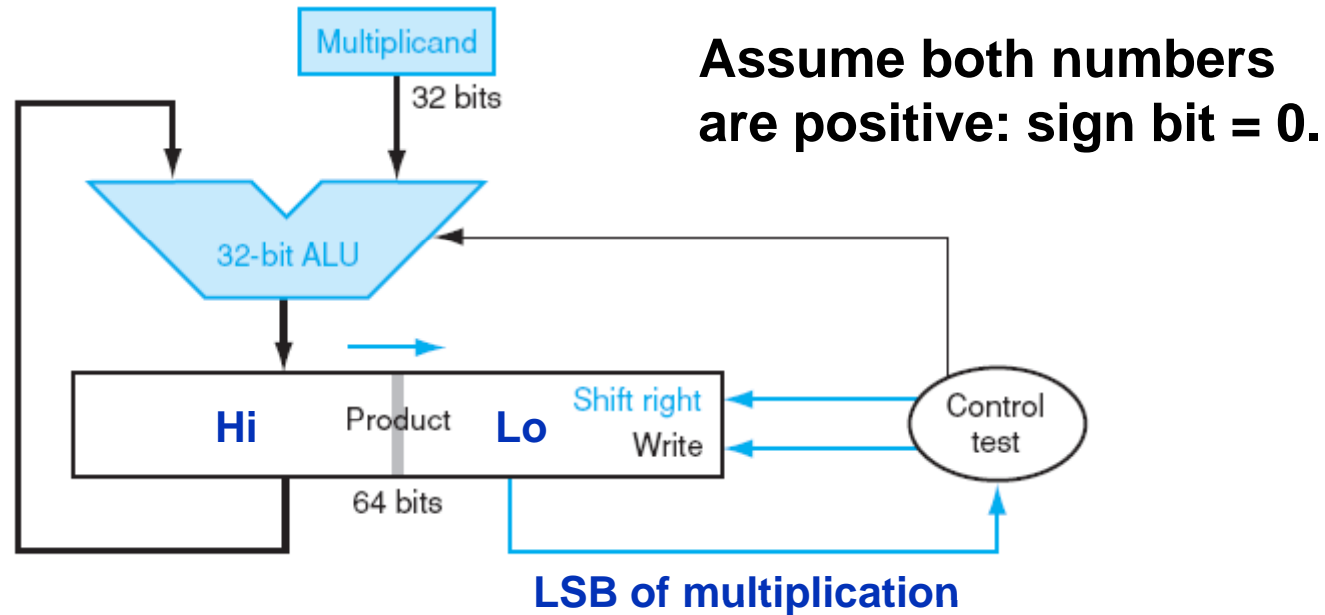
- **Observe that only 32 bits of products are getting added at a time.**
- **So if we shift the product right, then we can make do with only a 32-bit adder.**
- **Everything else is as before.**

Integer Multiplication (5/9)



- Observe that both product and multiplier are shifted right.
- Right half of product register is not used initially
- Why not store multiplier in there?

Integer Multiplication (6/9)



- At each step, we examine the LSB of the product register (multiplier).
- Hi & Lo registers. Hi is the upper half and Lo is the lower half of product.

Integer Multiplication (7/9)

(repeat of figure on preceding three slides)

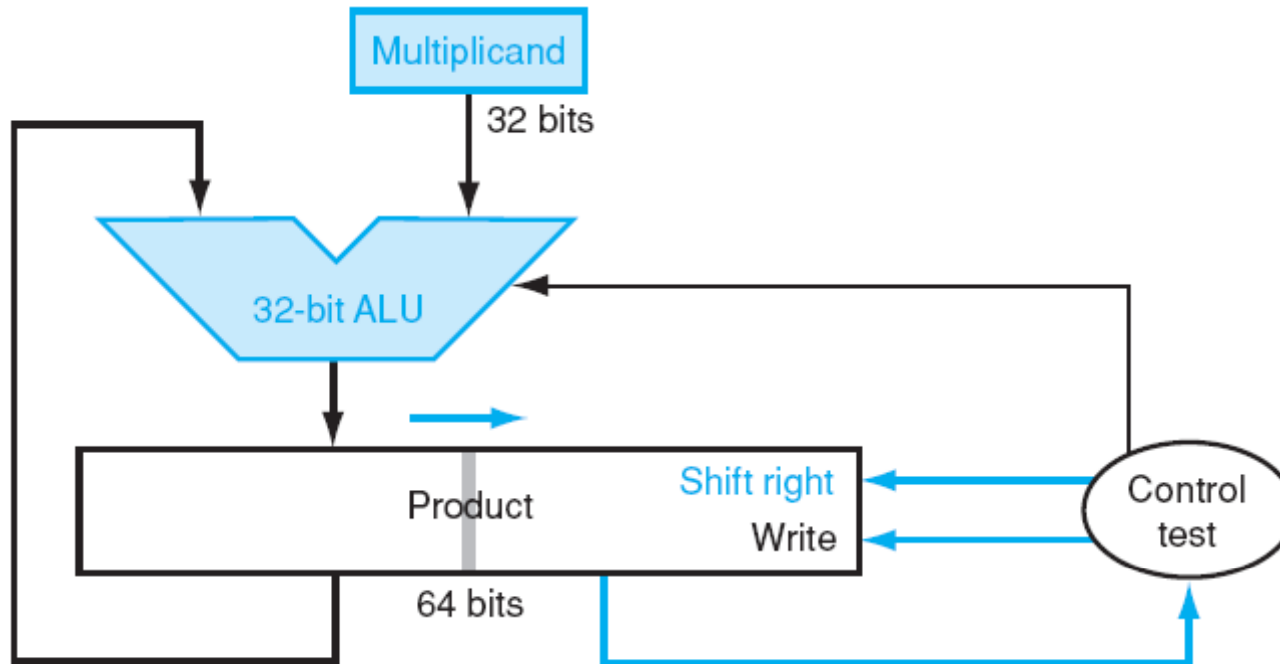
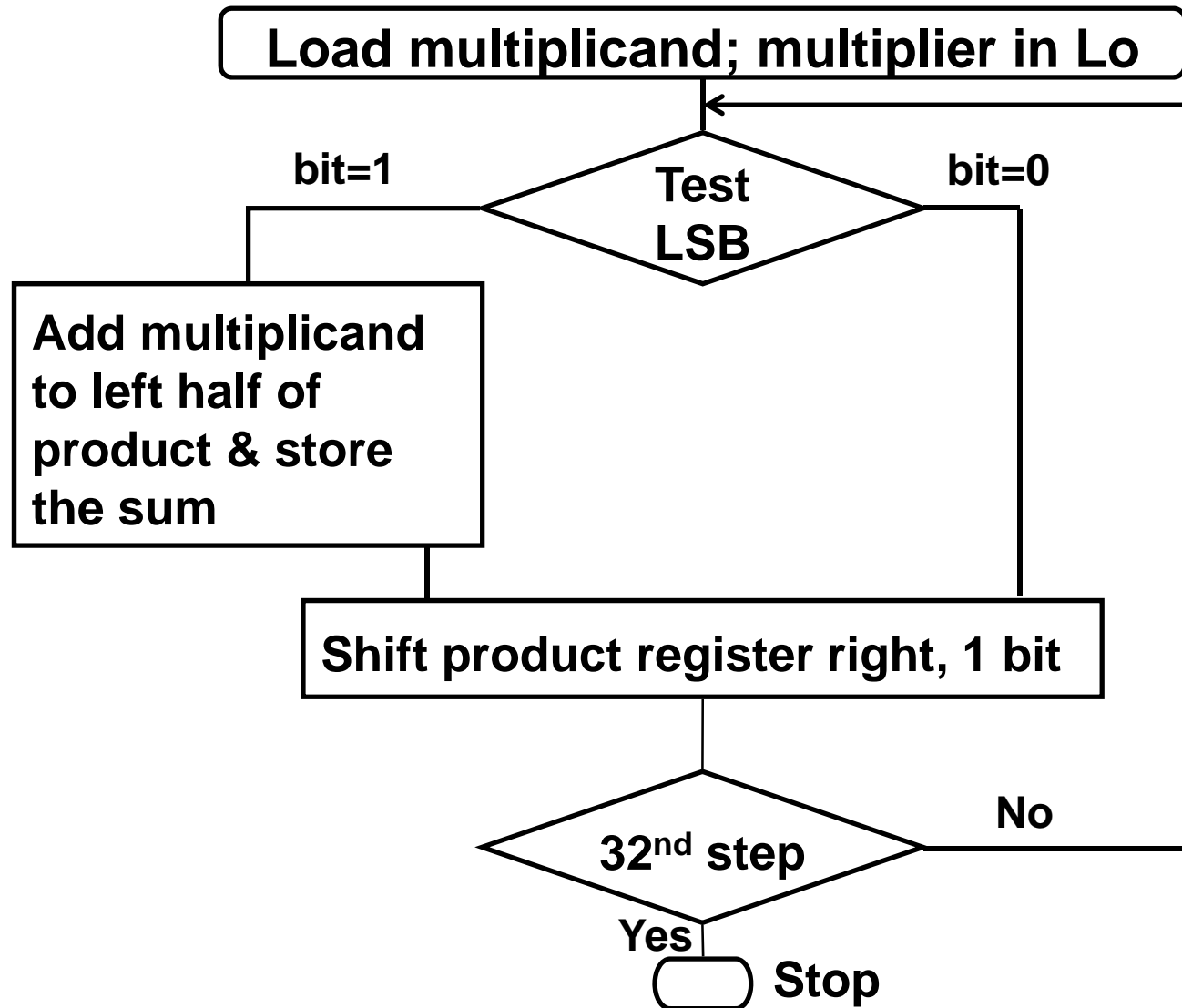


FIGURE 3.6 Refined version of the multiplication hardware. Compare with the first version in Figure 3.4. The Multiplicand register, ALU, and Multiplier register are all 32 bits wide, with only the Product register left at 64 bits. Now the product is shifted right. The separate Multiplier register also disappeared. The multiplier is placed instead in the right half of the Product register. These changes are highlighted in color. (The Product register should really be 65 bits to hold the carryout of the adder, but it's shown here as 64 bits to highlight the evolution from Figure 3.4.)

Integer Multiplication (8/9)

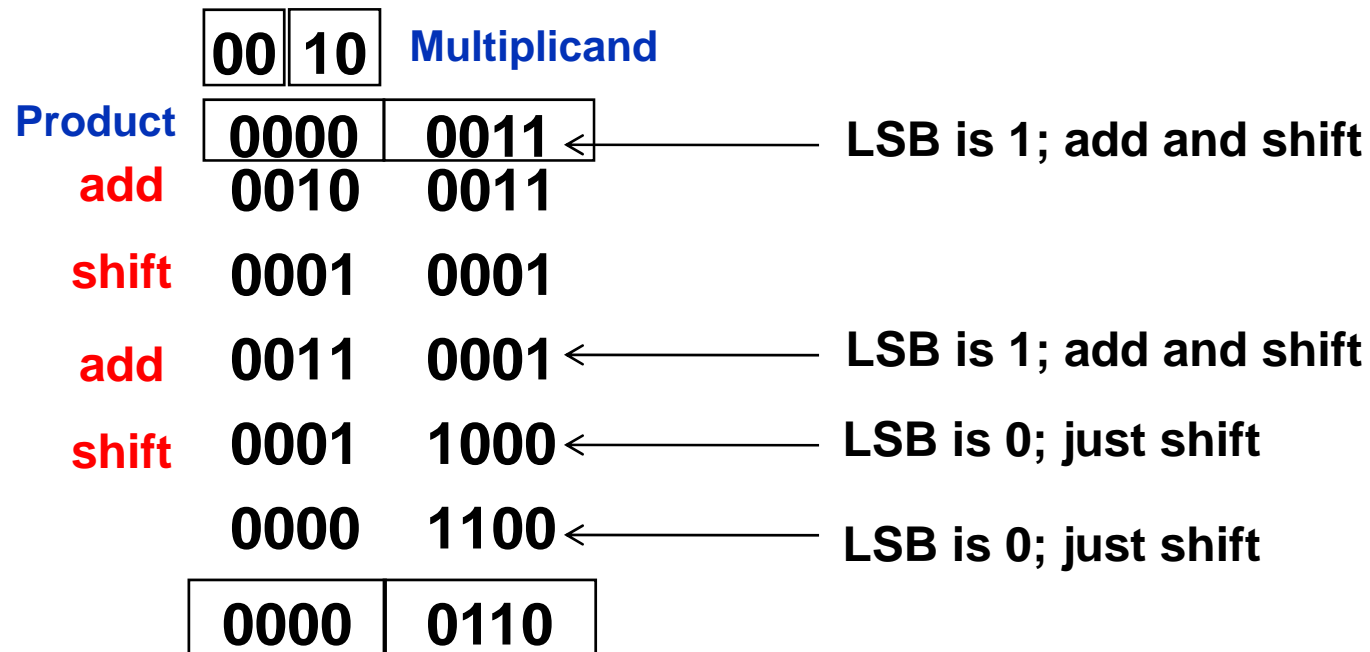


Integer Multiplication (9/9)

Example: Consider $2_{10} \times 3_{10} = 6_{10}$

Let size be 4 bits. $2_{10} = 0010$, $3_{10} = 0011$

Multiplicand = 0010; load 0011 in lower half of product register.



The answer is 6_{10}