

COMP2611: Computer Organization

Arithmetic Logic Unit II

- ❑ You will learn the following in this tutorial:
 - ❑ Multiplication, Booth's algorithm
 - ❑ Division

Arithmetic Logic Unit II

Review of Arithmetic logic Units

- Multiplication
- Booth's Algorithm
- Division

Exercises

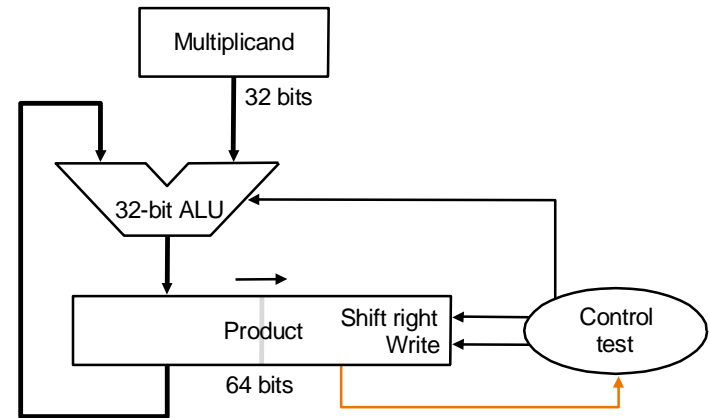
❑ 32-bit ALU

❑ Two registers:

- ❑ **Multiplicand register: 32 bits**

- ❑ **Product register: 64 bits**

(right half also used for storing multiplier)



❑ Operations:

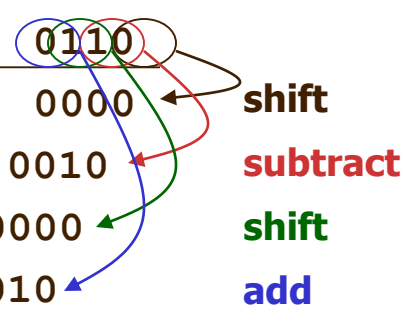
- ❑ The right half of the product register is initialized to the multiplier, and its left half is initialized to 0
- ❑ The two right-shifts at each step for version 2 are combined into only a single right-shift because the product and multiplier registers have been combined

- ❑ Multiplication of two signed numbers +3 and +6 (0011 and 0110)

Iteration	Multiplicand (M)	Product (P)	Remark
0	0110	0000 0011	Initial state
1		<u>0110</u> 0011 <u>0011</u> 0001	Left(P) = Left(P) + M P = P >> 1
2		<u>1001</u> 0001 <u>0100</u> 1000	Left(P) = Left(P) + M P = P >> 1
3		<u>0100</u> 1000 <u>0010</u> 0100	No operation P = P >> 1
4		<u>0010</u> 0100 <u>0001</u> 0010	No operation P = P >> 1

Multiplier

- Let's consider multiplying 0010_2 and 0110_2

	Convention	Booth
Multiplicand	0010	0010
Multiplier	\times 0110	
	+ 0000	+ 0000
	+ 0010	- 0010
	+ 0010	+ 0000
	+ 0000	+ 0010
Product	= 0001100	= 0001100

Idea of Booth's Algorithm

- Looks at two bits of multiplier at a time from right to left
- Assume that shifts are much faster than adds
- Basic idea to speed up the calculation: **avoid unnecessary additions**

Example (Booth's Algorithm)

7

- ❑ Multiplication of two signed numbers +2 and +6 (0010 and 0110)

Multiplier			
Iteration	Multiplicand (M)	Product (P)	Remark
0	0010	0000 0110 0	Initial state
1		0000 0110 0	No operation
		0000 0011 0	P = P >> 1
2		1110 0011 0	Left(P) = Left(P) - M
		1111 0001 1	P = P >> 1
3		1111 0001 1	No operation
		1111 1000 1	P = P >> 1
4		0001 1000 1	Left(P) = Left(P) + M
		0000 1100 0	P = P >> 1

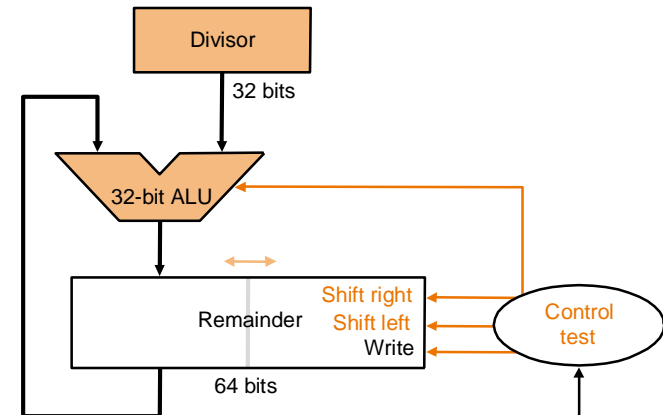
❑ 32-bit ALU

❑ Two registers:

- ❑ **Divisor register: 32 bits**

- ❑ **Remainder register: 64 bits**

(right half also used for storing quotient)



❑ Operations:

- ❑ 32-bit divisor is always subtracted from the left half of remainder register
 - The result is written back to the left half of the remainder register
- ❑ The right half of the remainder register is initialized with the dividend
 - Left shift remainder register by one before starting
- ❑ The new order of the operations in the loop is that the remainder register will be **shifted left one time too many**
 - Thus, **final correction step:** must **right shift back only the remainder** in the left half of the remainder register

- Division of a 4-bit unsigned number (0111) by another one (0011)

Iteration	Divisor (D)	Remainder (R)	Remark
0	0011	0000 0111 0000 1110	Initial state $R = R \ll 1$
1		<u>1</u> 101 1110 <u>0000</u> 1110 0001 1100	Left(R) = Left(R) - D Undo ($L(R) = L(R) + D$) $R = R \ll 1, R_0 = 0$
2		<u>1</u> 110 1100 <u>0001</u> 1100 0011 1000	Left(R) = Left(R) - D Undo $R = R \ll 1, R_0 = 0$
3		<u>0</u> 000 1000 <u>0001</u> 0001	Left(R) = Left(R) - D $R = R \ll 1, R_0 = 1$
4		<u>1</u> 110 0001 <u>0001</u> 0001 0010 0010	Left(R) = Left(R) - D Undo $R = R \ll 1, R_0 = 0$
extra		0001 0010	$Left(R) = Left(R) \gg 1$

Remainder

correction

Quotient

Arithmetic Logic Unit II

Review of Arithmetic logic Units

- Multiplication
- Booth's Algorithm
- Division

Exercises

- ❑ **Question 1:** According to the multiplication hardware – refined version, do multiplication of two unsigned number 5 x 7 (0101 and 0111), fill in the table below.

Iteration	Multiplicand (M)	Product (P)	Remark
0	0101		Initial state
1			
2			
3			
4			

- ❑ **Question 2:** According to Booth's algorithm, do multiplication of two signed number +2 and -3 (0010 and 1101), fill in the table below.

Iteration	Multiplicand (M)	Product (P)	Remark
0	0010		Initial state
1			
2			
3			
4			

- ❑ **Question 3:** According to the division hardware – improved version. Divide 8 (1000) by 3 (0011), fill in the table below.

Iteration	Divisor (D)	Remainder (R)	Remark
0	0011		Initial state
1			
2			
3			
4			
extra			

- ❑ Today we have reviewed:
 - ❑ Multiplication, Booth's algorithm
 - ❑ Division