Lecture 14
EE 421 / CS 425
Digital System Design

Spring 2023

Shahid Masud

MIDTERM EXAM
Week of 30 October

Topics

- Fractional Multiplication
- Different Cases of Signed Multiplication in Fractional Binary Numbers
- Booth Encoded Multipliers
- Advantages of Booth Multipliers
- Limitations of Booth Multipliers
- STG Control for Booth Multipliers

Multiplication of Fractions

Convert from decimal to binary $(\frac{3}{4})$

= 0.75

0.75 x 2 = 1.5, keep 1

0.5 x 2 = 1.0, keep 1

0 x 2 = 0 keep 0

And only zeros afterwards

$= 2^{-1} + 2^{-2} + 0 + 0$

= 0.1100; assigning four fractional bits

2’s Complement of Binary Fractional Nos.

- Given binary fractional number = (0.1100)
- Method 1:
- Decide on the number of total bits, eg. 5 bits; Invert all bits; add +1 to LSB
- 2’s Complement = 1.0011
- \+ 1
- ______________________
- = 1.0100
- Method 2:
- Look from right to left; when you Encounter first 1; invert all bits to the left
- For (0.1100), the 2’s Complement = (1.0100)

Question?

- Represent 9/16 using five fractional bits:

- Hint: 9/16 = 0.xxxxx
- Keep multiplying by 2; if answer is greater than 1, keep 1

- Solve: 0.562 x 2 = 1.125, keep 1
- 0.125 x 2 = 0.25, keep 0
- 0.25 x 2 = 0.5, keep 0
- 0.5 x 2 = 1.0, keep 1
- 0 x 2 = 0, keep 0, and same for more terms
- Answer = 0.10010 in binary

Finally, 2’s Complement of “0.10010” is (look right to left) “1.01110” that is (-9/16)

Convert Fraction Number to Binary

- Represent 9/16 using five fractional bits:
- Hint: 9/16 = 0.5625
- Keep multiplying by 2; if answer is greater than 1, keep 1

- Solve: 0.5625 x 2 = **1**.125, keep 1
- 0.125 x 2 = **0**.25, keep 0
- 0.25 x 2 = **0**.5, keep 0
- 0.5 x 2 = **1**.0, keep 1
- 0 x 2 = **0**, keep 0, and same for more terms
- Answer = 0.10010 in binary

Finally, 2's Complement of "0.10010" is (look right to left) "1.01110" that is (-9/16)

Multiplication of Signed Fractions

- Fractions are multiplied like whole numbers, but overflow is not possible
- A 4-bit fractional number is represented as minimum 5-bit fixed point number with MSB holding the sign bit in 2’s Complement format
- The product of two 5-bit numbers will produce 10-bit result
- MSB will be sign-extended (bit replication) for negative multiplicand

Example 1: Positive multiplicand, positive multiplier, fraction multiplication

Show binary multiplication of $(3/4)_{10}$ x $(1/2)_{10}$
Use 5-bits to represent each number

3/4 = 0.1100

1/2 = 0.1000

					0.	1	1	0	0	Positive multiplicand
				x	0.	1	0	0	0	Positive multiplier
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	X	0
0	0	0	0	0	0	0	0	X	X	0
0	0	0	1	1	0	0	X	X	X	1
0	0.	0	1	1	0	0	0	0	0	

Check Multiplier bits one at a time

Insert decimal, count how many bits after decimal in both numbers (4 + 4 = 8 bits in both numbers)

Answer = (0.0110000000) = +(3/8)

Example 2: Negative multiplicand, positive multiplier, fraction multiplication

Show binary multiplication of $(-3/4)_{10}$ x $(3/8)_{10}$
Use 5-bits to represent each number

3/4 = 0.1100
-3/4 = -(0.1100)
= 1.0100

3/8 = 0.0110

Sign bit replication

		1	1	1	1	1	1.	0	1	0	0
						x	0.	0	1	1	0
	1	1	1	1	1	1	0	1	0	0	X
1	1	1	1	1	1	0	1	0	0	X	X
Extra		1	1.	1	0	1	1	1	0	0	0

Negative multiplicand

Negative multiplier
(first 0 only shift, then 1, Add multiplicand)
(next 1, Add Multiplicand, then 0 shifts only)

Insert decimal, count how many bits after decimal in both numbers (4 + 4 = 8 bits in both numbers)
Answer = (11.10111000) , take 2's complement = -(0.01001000) = (-9/32)

Example 3: Positive multiplicand, negative multiplier, fraction multiplication

Show binary multiplication of $(3/4)_{10}$ x $(-3/8)_{10}$
Use 5-bits to represent each number

3/4 = 0.1100

3/8 = 0.0110
-3/8 = -(0.0110)
= (1.1010)

				0.	1	1	0	0
			x	1.	1	0	1	0
				0	0	0	0	0
			0	1	1	0	0	X
		0	0	0	0	0	X	X
	0	1	1	0	0	X	X	X
1	0	1	0	0	X	X	X	X
1.	1	0	1	1	1	0	0	0

Positive multiplicand

Negative multiplier

Check Multiplier bits one at a time

2's Complement of Multiplicand due to 1 MSB

Insert decimal, count how many bits after decimal in both numbers (4 + 4 = 8 bits in both numbers)

Answer = (1.10111000), take 2's Complement = -(0.01001000) = -(9/32)

Example 4: Negative multiplicand, negative multiplier, fraction multiplication

Show binary multiplication of $(-3/4)_{10}$ x $(-3/8)_{10}$
Use 5-bits to represent each number

3/4 = 0.1100 -3/4 = -(0.1100) = 1.0100
3/8 = 0.0110 -3/8 = -(0.0110) = 1.1010

Sign bit replication

				1	1	1	1	1	1.	0	1	0	0	Negative multiplicand
								x	1.	1	0	1	0	Negative multiplier
Extra ←				0	0	0	0	0	0	0	0	0	0	
			1	1	1	1	1	1	0	1	0	0	X	
	1	0	0	0	0	0	0	0	0	0	0	X	X	
	1	1	1	1	1	1	0	1	0	0	X	X	X	
	0	0	0	0	0	1	1	0	0	X	X	X	X	
1	0	1	0	0	0.	0	1	0	0	1	0	0	0	

2's Complement of Multiplicand due to 1 MSB

Insert decimal, count how many bits after decimal in both numbers (4 + 4 = 8 bits in both numbers)
Answer = +(0.01001000) = $2^{-2} + 2^{-5}$ = 0.25 + 0.03125 = 0.28125_{10}

To Remember in Signed Multiplication

- When Multiplicand is Negative, do a sign extension to cover the possible bit-width of Answer

- When Multiplier is Negative, there is a final 2’s Complement Addition Step corresponding to the MSB of multiplier

Algorithmic Improvement in Multipliers

Booth Encoding

Booth Multiplication Process

Booth Encoded Multipliers

Object: To reduce the number of 'Add' steps required in complete multiplication cycle

MSB '1' shows negative number

2's Complement of $7_{10} = (1\ 0\ 0\ 1)_2$

$1 \times 2^0 = 1$

$0 \times 2^1 = 0$

$0 \times 2^2 = 0$

$-1 \times 2^3 = -8$

Allow both +ive and –ive signs to be used in conversion

Decimal value of $(1001)_2 = (-8+1) = -7 =$ **$(\underline{1}\ 0\ 0\ 1)$ or $(-1\ 0\ 0\ 1)$**

Booth's algorithm is valid for both positive and negative numbers in 2's complement format

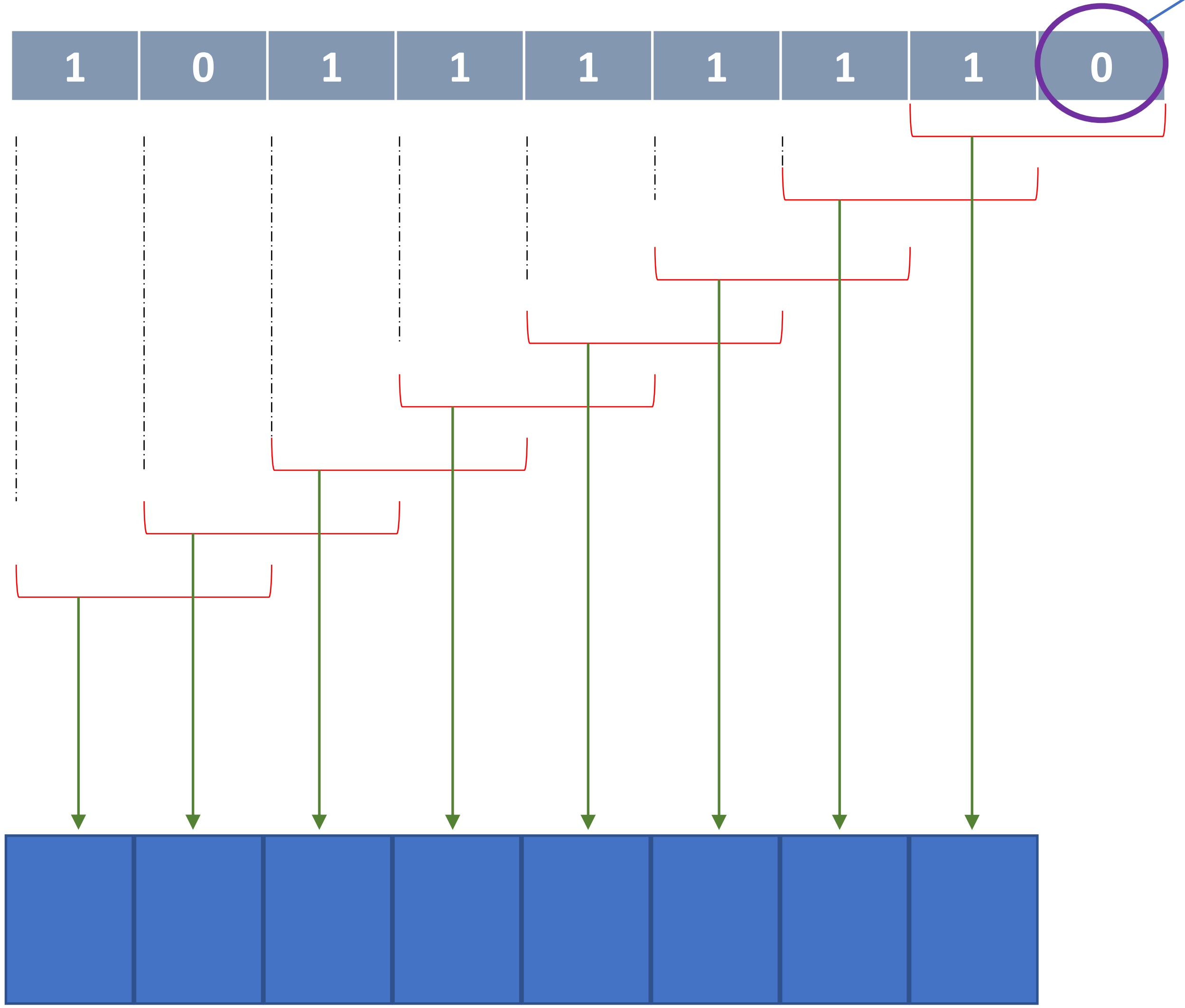
Booth Recoding of a 2's Complement Number

m_i	m_{i-1}	Booth Recoded C_i	Value	Status
0	0	0	0	String of 0s
0	1	1	+1	End of string of 1s
1	0	$\underline{1}$	-1 or $\underline{1}$	Begin string of 1s
1	1	0	0	Midstring of 1s

Booth Recoding of -65_{10}

Append ‘0’ on right, if LSB=1

$-65_{10} =$



2’s Complement notation

+65 = (01000001)
2’s Complement
-65 = (10111111)

m_i	m_{i-1}	Booth Recoded Ci
0	0	0
0	1	1
1	0	$\underline{1}$
1	1	0

$-65_{10} =$

Or

Booth Recoded notation

Question?

Convert decimal number –78 to Booth Encoded format using 8 binary bits

+78 = 01001110
Take 2's Complement
-78 = 10110010

m_i	m_{i-1}	Booth Recoded Ci
0	0	0
0	1	1
1	0	$\underline{1}$
1	1	0

1 0 1 1 0 0 1 0

No need for extra '0' after LSB in this case

10 → -1
01 → 1
00 → 0
10 → -1
11 → 0
01 → 1
10 → -1

Answer = -1 1 0 -1 0 1 -1
After Booth Encoding

Booth Encoded Multiplication

m_i	m_{i-1}	Booth Recoded C_i	Value	Multiplication Action
0	0	0	0	No Operation
0	1	1	+1	Add Multiplicand to Accumulator
1	0	$\underline{1}$	-1 or $\underline{1}$	Subtract Multiplicand from Accumulator (= 2’s Complement Add)
1	1	0	0	No Operation

Booth Multiplication – Example 1

Show Booth Encoded multiplication of 6 x 5, using 4 bits for both numbers

Imagine Zero bit if LSB = 1

Check 2-bits at a time, Right to Left
Shift Left by 1 after every step

6	Multiplicand	Extra		0	0	0	0	0	1	1	0		
x5	Multiplier	bits		0	0	0	0	0	1	0	1	[0]	
				1	1	1	1	1	0	1	0		1[0] = Subtract = Add 2's Compl of Multiplicand to Acc
			0	0	0	0	0	1	1	0	X		01 = Add Multiplicand to Acc
		1	1	1	1	1	0	1	0	X	X		10 = Subtract = Add 2's Compl of Multiplicand to Acc
		0	0	0	0	1	1	0	X	X	X		01 = Add Multiplicand to Acc
								X	X	X	X		00 = No Op, Shift Left by 1
		0	1	0	0	0	1	1	1	1	0		00 = No Op, Shift Left by 1

Answer = (0001 1110) = +(16 + 14) = +30$_{10}$

Booth Multiplication – Example 2

Show Booth Encoded multiplication of 6 x -5, using 4 bits for both numbers

Imagine Zero bit if LSB = 1

Check 2-bits at a time, Right to Left
Shift Left by 1 after every step

6	Multiplicand			0	0	0	0	0	1	1	0	
X -5	Multiplier			1	1	1	1	1	0	1	1	[0]
				1	1	1	1	1	0	1	0	
											X	
		0	0	0	0	0	1	1	0	X	X	
		1	1	1	0	1	1	0	X	X	X	
								X	X	X	X	
		0	0	1	1	1	0	0	0	1	0	

1[0] = Subtract = Add 2’s Compl of Multiplicand to Acc

11 = No Op, Shift Left, Add 0 to Acc

01 = Add Multiplicand to Acc

10 = Subtract = Add 2’s Compl of Multiplicand to Acc

11 = No Op, Shift Left by 1, Add 0 to Acc

Answer = (1110 0010) = Take 2’s Comp = -(0001 1110) = -30_{10}