DIGITAL DESIGN & COMPUTER ORGANISATION

Multiplication-2

Sudarshan T S B., Ph.D.
Department of Computer Science
& Engineering

DIGITAL DESIGN & COMPUTER ORGANISATION

Multiplication-2

Sudarshan T S B., Ph.D.
Department of Computer Science & Engineering

Multiplication-2

Course Outline

- Digital Design
  - Combinational logic design
  - Sequential logic design
    - Multiplication – 2
- Computer Organisation
  - Architecture (microprocessor instruction set)
  - Microarchitecture (microprocessor operation)

Concepts covered

- Shift-Add Multiplication
- Booth Algorithm

MULTIPLICATION-2

Shift - Add Multiplication

- Array Multiplier
  - Very hardware intensive
  - Expensive & waste of resources
- Multicycle (Shift-Add) Multiplier
  - Add & Shift the multiplicand
  - Iterate the process through the bits of multiplier

```
      1 1 0 1     (13) Multiplicand, M
    X 1 0 1 1     (11) Multiplier, Q
-----------------
      1 1 0 1
    1 1 0 1
  0 0 0 0
1 1 0 1
-----------------
1 0 0 0 1 1 1 1   (143) Product, P
```

Shift Right and Add

```
      1 1 0 1
    X 1 0 1 1
-----------------
  0 0 0 0          Initial result
  1 1 0 1          Multiplier = 1; Add Multiplicand
  -------------
  1 1 0 1
    1 1 0 1        Shift Right
  1 1 0 1          Multiplier = 1; Add Multiplicand
 -------------
1 0 0 1 1 1
  1 0 0 1 1 1      Shift Right
 ------------      Multiplier = 0; No Add
  1 0 0 1 1 1
    1 0 0 1 1 1    Shift Right
  1 1 0 1          Multiplier = 1; Add Multiplicand
 ---------------
1 0 0 0 1 1 1 1    (143) Product, P
```

Shift - Add Multiplication

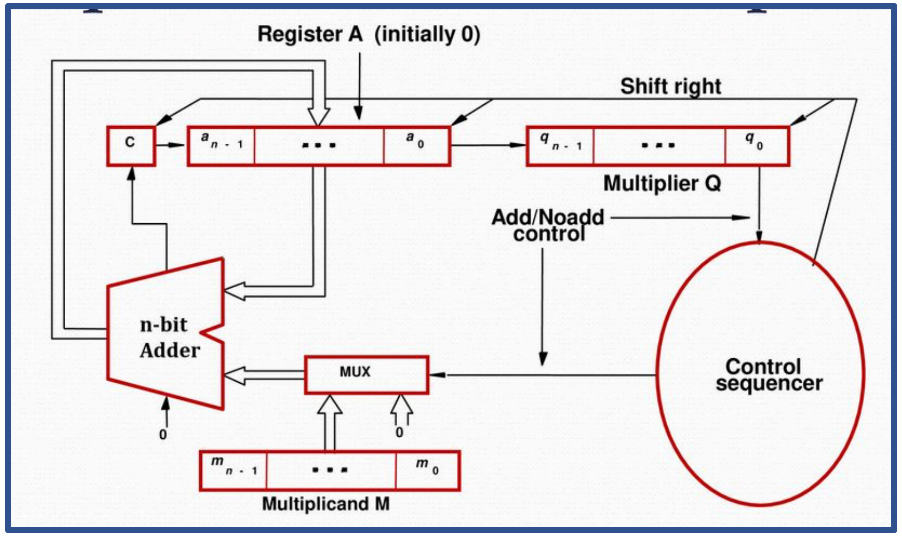
Implementation

- Use an Adder circuitry to do repeated addition
- Multiplication through spatial addition, requires a single n-bit adder doing addition n times.
- After every addition, the result has to be shifted right once.
- Monitor the multiplier bit to add multiplicand to the partial product.
- Repeat the add – shift right process for the number of multiplier bits.
- All the control signals has to be generated for this sequential process appropriately, by a control sequencer.
- Requires n-cycles for n times addition.
- This technique is used for unsigned binary numbers multiplication or positive signed binary numbers multiplication

MULTIPLICATION - 2

Shift - Add Multiplication

- We Require:
  - Accumulator register – A
  - Multiplier – Q
  - Multiplicand – M
  - N-bit Adder
  - Control signals for Shift and Add



M

1 1 0 1

C	A	Q	
0	0 0 0 0	1 0 1 1	
0	1 1 0 1	1 0 1 1	Q_0=1, Add M
0	0 1 1 0	1 1 0 1	Shift Right
1	0 0 1 1	1 1 0 1	Q_0=1, Add M
0	1 0 0 1	1 1 1 0	Shift Right
0	1 0 0 1	1 1 1 0	Q_0=0, **No Add**
0	0 1 0 0	1 1 1 1	Shift Right
1	0 0 0 1	1 1 1 1	Q_0=1, Add M
0	1 0 0 0	1 1 1 1	Shift Right

(143) Product, P

Signed Multiplication

Observation

- Here, if multiplicand(M) is negative or multiplier(Q) is negative, then product(P) is negative.
- If M is negative and Q also is negative, then product is positive
- Hence sign(Product, P) = sign(M) xor sign(Q)

Hence

- Convert M and Q to positive numbers with (n-1) bits
- Multiply positive numbers using shift-add method
- Compute sign; depending on the sign, convert the product accordingly

Alternatively,

- Perform sign-extension on shift in shift-add method
- Use Booth algorithm

MULTIPLICATION - 2

Shift - Add Multiplication

```
        1 1 0 1      (-3) Multiplicand, M
      X 0 1 0 1      (+5) Multiplier, Q
--------------------
  1 1 1 1 1 0 1
  0 0 0 0 0 0
  1 1 1 0 1
  0 0 0 0
--------------------
1 1 1 1 0 0 0 1      (-15) Product, P
```

Consider signed numbers:
1101 → -3
0101 → +5

Product = -15 → 1111 0001

Use Booth Algorithm for both positive and negative M & Q for the following reasons

- To reduce number of additions when sequence of 1's are in multiplier
- To treat both positive and negative 2's complement n-bit operands uniformly

MULTIPLICATION - 2

Booth Multiplier

The method is a school trick:
When multiplying by 9

- Multiply by 10
- Subtract once

Ex: 234 x 9

```
        234 x 10
      -------------
        2340
      -   234
      ------------
        2106
      -----------
```

Apply to binary:
When multiplying by 7

- Multiply by 8
- Subtract one

Ex: 0101 x 0111

```
        0101 x 100 -1
        ---------------
        1111011
        000000
        00000
        0101
        --------------
        0100011 (+35) ✓
        --------------
```

Method is known as Booth Recode
Reduces sequence of 1's to zeros and -1
Thus reduces number of additions

MULTIPLICATION - 2

Booth's Multiplier

Transitions are represented as:

0 0 → 0

0 1 → +1

1 0 → -1

1 1 → 0

Ex:

0 0 0 0 1 1 1 1 1 1 1 0 0 0 1 1 1 0 $\boxed{0}$ (16270)

0 0 0 1 0 0 0 0 0 0 -1 0 0 1 0 0 -1 0 (16270)

No. of add & sub : 10 vs. 4

Ex:

0 1 0 1 0 1 0 1 $\boxed{0}$

=>

+1 -1 +1 -1 +1 -1 +1 -1

Worst case

Ex:

0 1 1 1 1 1 1 1 $\boxed{0}$

=>

+1 0 0 0 0 0 0 -1

Best case

Depending on Bit(i+1) and Bit(i) of the Multiplier, Q following operation can be defined on A:

0 0 → Shift Right (0 x M)

0 1 → Add M (+1 x M)

1 0 → Subtract (2's complement) M (-1 x M)

1 1 → Shift Right (0 x M)

MULTIPLICATION - 2

Booth Multiplier

1101 => -3 Mulitplicand, M
1011 => -5 Multiplier, Q

M: 1 1 0 1

A	Q	Q^{-1}	
0 0 0 0	1 0 1 1	0	
0 0 1 1	1 0 1 1	0	$Q_0 Q^{-1}$= 10, Add 2's Comp of M to A
0 0 0 1	1 1 0 1	1	Shift Right with Sign Extension
0 0 0 0	1 1 1 0	1	$Q_0 Q^{-1}$=11, Shift Right with Sign Ext
1 1 0 1	1 1 1 0	1	$Q_0 Q^{-1}$=01, Add M to A
1 1 1 0	1 1 1 1	0	Shift Right with Sign Extension
0 0 0 1	1 1 1 1	0	Q_0Q^{-1}=10, Add 2's Comp of M to A
0 0 0 0	1 1 1 1	1	Shift Right with Sign Extension

(+15) Product, P

Depending on Q_0 and Q^{-1}

Q_0 Q^{-1}

0 0 → No Add, Shift Right with Sign Extn.(0 x M)
0 1 → Add M to A (+1 x M)
1 0 → Subtract (2's complement) M to A (-1 x M)
1 1 → No Add, Shift Right with Sign Extn. (0 x M)

Booth Multiplier

Points to be noted :

- Handles both positive and negative multipliers uniformly
- Achieves efficiency in the number of additions required when the multiplier has a few large blocks of 1's
- In worst case, the speed of doing multiplication with the Booth algorithm will be same as the normal algorithm
- This is also known as Radix-2 Coding technique

Think about it

- Apply Booth algorithm to the following:

  +M x –Q

  -M x +Q

  +M x +Q

- Implement Booth algorithm using sequential repeated addition circuit or using Array Multiplier circuit.

- Instead of Shift Right Accumulator can be we have Shift Left Multiplicand?

THANK YOU

Sudarshan T S B. Ph.D.,
Department of Computer Science & Engineering

sudarshan@pes.edu

+91 80 6666 3333 Extn 215