Computer Organization and Architecture

Module 6

Computer Arithmetic

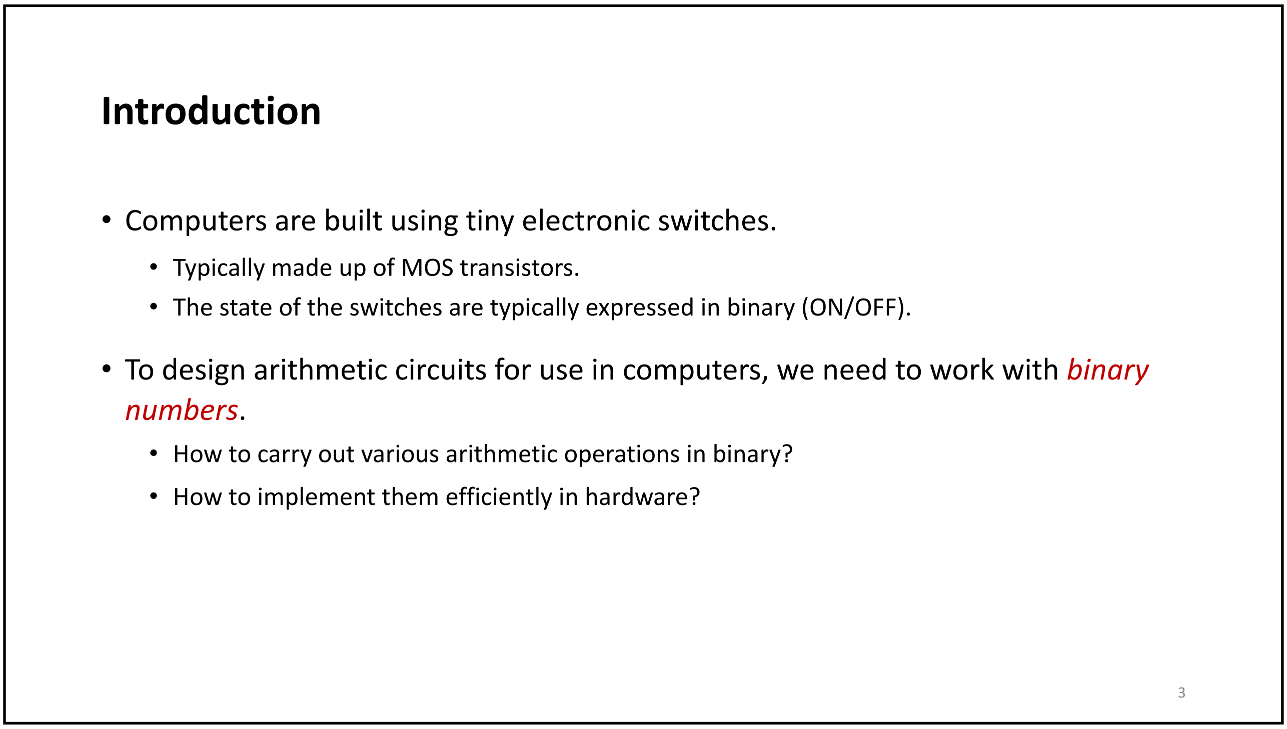
Prof. Indranil Sengupta

Dr. Sarani Bhattacharya

Department of Computer Science and Engineering

IIT Kharagpur

Computer Arithmetic

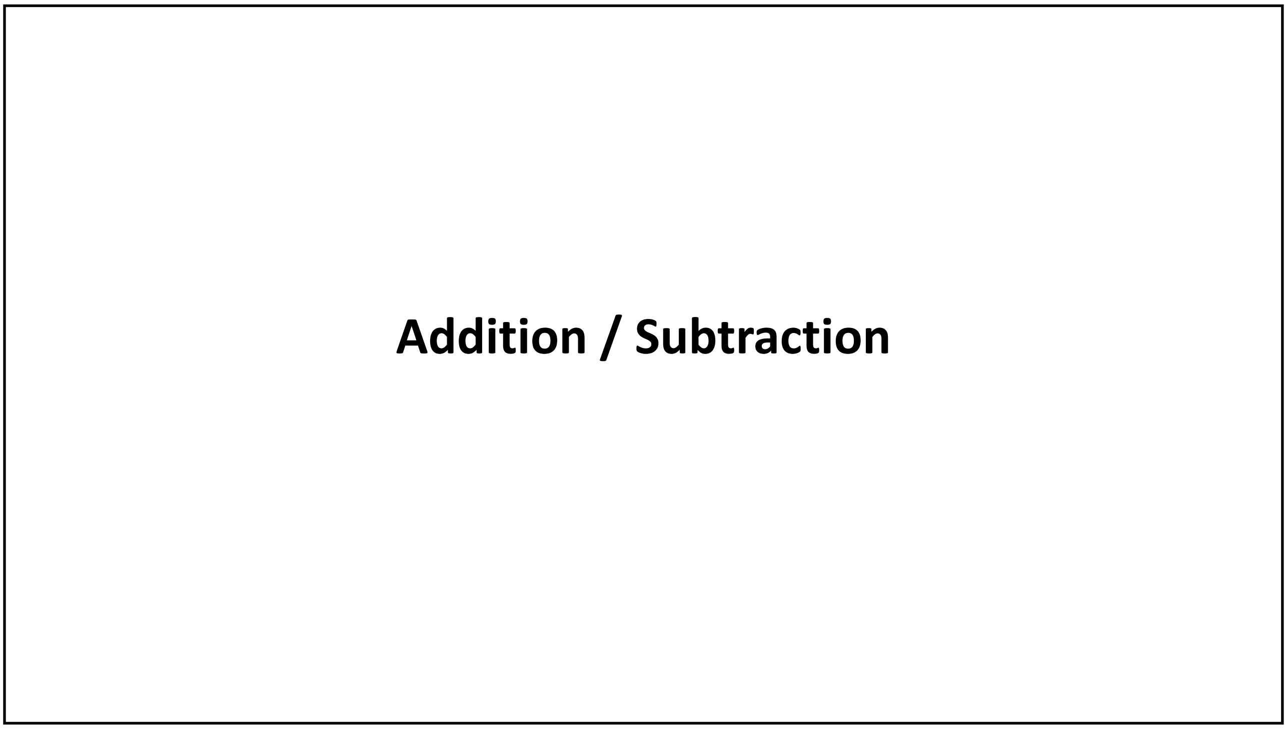


Introduction

- Computers are built using tiny electronic switches.
  - Typically made up of MOS transistors.
  - The state of the switches are typically expressed in binary (ON/OFF).
- To design arithmetic circuits for use in computers, we need to work with *binary numbers*.
  - How to carry out various arithmetic operations in binary?
  - How to implement them efficiently in hardware?

3

3



Addition / Subtraction

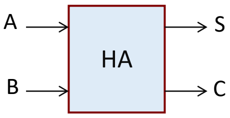
4

Addition of Two Binary Digits (Bits)

- When two bits *A* and *B* are added, a sum (*S*) and carry (*C*) are generated as per the following truth table:

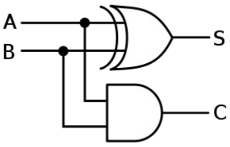
Inputs		Outputs	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

0 + 0 = 00
0 + 1 = 01
1 + 0 = 01
1 + 1 = 10



HALF ADDER

S = A'.B + A.B' = A ⊕ B
C = A.B



Addition of Multi-bit Binary Numbers

```
   0 0 1 0 1 1 0  ← Carry
   0 1 0 1 0 1 1  ← Number A
 + 0 0 0 1 0 0 1  ← Number B
 ---------------
   0 1 1 0 1 0 0  ← Sum S
```

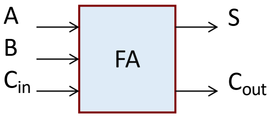
```
   1 1 1 1 1 1 0  ← Carry
   0 1 1 1 1 1 1  ← Number A
 + 0 0 0 0 0 0 1  ← Number B
 ---------------
   1 0 0 0 0 0 0  ← Sum S
```

- At every bit position (stage), we require to add 3 bits:
  - 1 bit for number A
  - 1 bit for number B
  - 1 carry bit coming from the previous stage

WE NEED A FULL ADDER

Full Adder

Inputs			Outputs	
A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



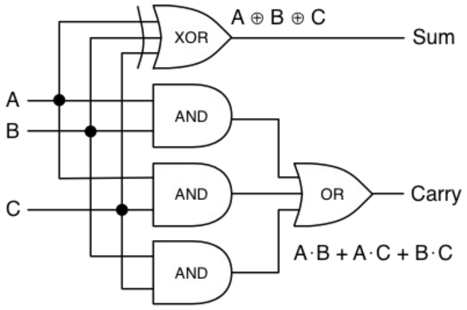
$$S = A'.B'.C_{in} + A'.B.C_{in}' + A.B'C_{in}' + A.B.C$$
$$= A \oplus B \oplus C_{in}$$

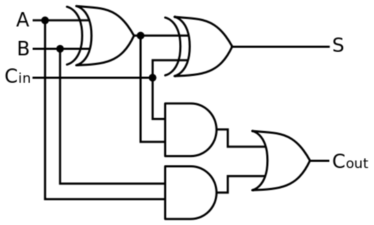
$$C_{out} = B.C_{in} + A.C_{in} + A.B + A.B.C_{in}$$
$$= A.B + B.C_{in} + A.C_{in}$$

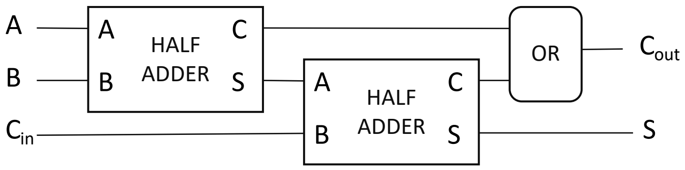
7

7

Various Implementations of Full Adder





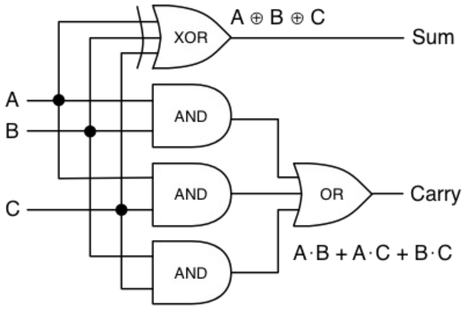


8

8

- **Delay of a full adder:**
  - Assume that the delay of all basic gates (AND, OR, NAND, NOR, NOT) is δ
  - Delay for Carry = 2δ
  - Delay for Sum = 3δ

    (AND-OR delay plus one inverter delay)

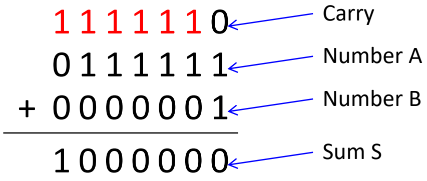


Parallel Adder Design

- We shall look at the various designs of *n*-bit parallel adder.
  a) Ripple carry adder
  b) Carry look-ahead adder
  c) Carry save adder
  d) Carry select adder

Ripple Carry Adder

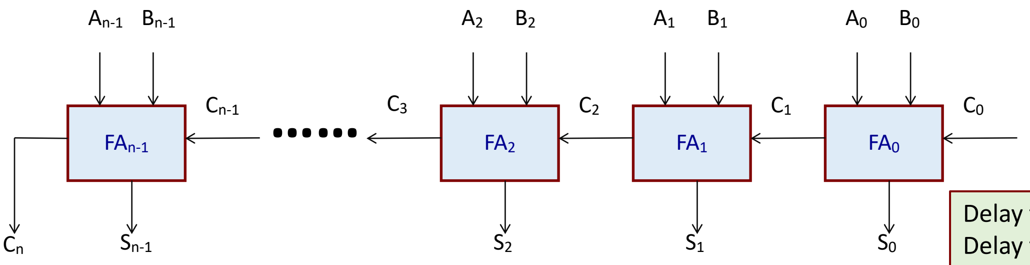
- Cascade *n* full adders to create a *n*-bit parallel adder.
- Carry output from stage-*i* propagates as the carry input to stage-(*i*+1).
- In the worst-case, carry ripples through all the stages.



```
  1 1 1 1 1 1 0   Carry
  0 1 1 1 1 1 1   Number A
+ 0 0 0 0 0 0 1   Number B
---------------
  1 0 0 0 0 0 0   Sum S
```

11

11



Delay for C_1 $= 2\delta$
Delay for C_2 $= 4\delta$
Delay for C_{n-1} $= 2(n-1)\delta$
Delay for C_n $= 2n\delta$

Two numbers: $A_{n-1}\ldots A_2A_1A_0$ and $B_{n-1}\ldots B_2B_1B_0$
Input carry: C_0
Sum: $S_{n-1}\ldots S_2S_1S_0$
Output carry: C_n

Delay for S_0 $= 3\delta$
Delay for S_1 $= 2\delta + 3\delta = 5\delta$
Delay for S_2 $= 4\delta + 3\delta = 7\delta$
Delay for S_{n-1} $= 2(n-1)\delta + 3\delta = (2n+1)\,\delta$

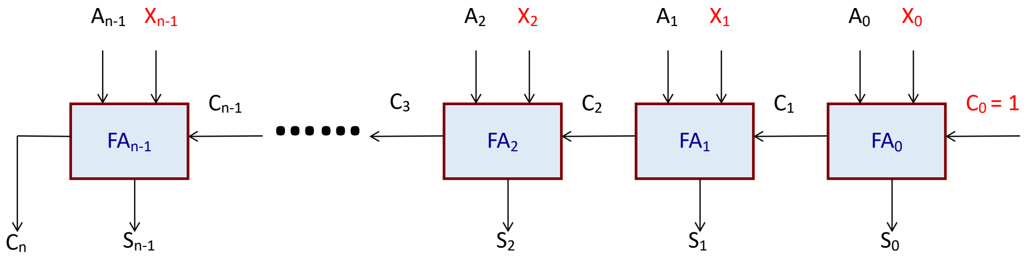
Delay is proportional to n

12

12

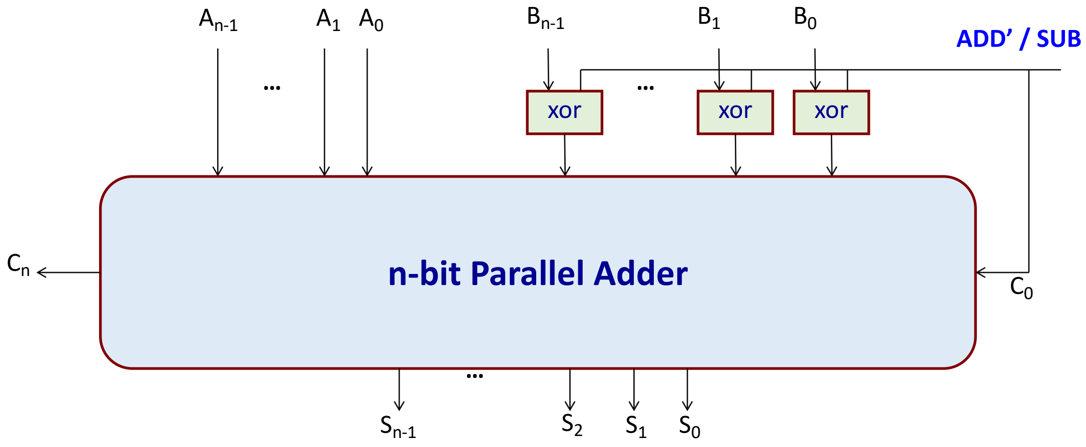
How to Design a Parallel Subtractor?

- **Observation:**
  - Computing *A* - *B* is the same as adding the 2’s complement of *B* to *A*.
  - 2’s complement is equal to 1’s complement plus 1.
  - Let $X_i = B_i'$.



A Parallel Adder/Subtractor

xor gate is a controlled inverter


A_{n-1}
A_1
A_0
B_{n-1}
B_1
B_0
ADD’ / SUB
…
…
xor
xor
xor
n-bit Parallel Adder
C_n
C_0
…
S_{n-1}
S_2
S_1
S_0