# EE-421: Digital System Design

## Doing MATHS on an FPGA

Arithmetic Circuits for Number Crunching:
Adders,
Subtractors,
Multipliers and Dividers

Dr. Rehan Ahmed [rehan.ahmed@seecs.edu.pk]

NUST
SCHOOL OF ELECTRICAL ENGINEERING &
COMPUTER SCIENCE (SEECS)

# Roadmap

**Arithmetic Circuits**

- Addition
- Subtraction
- Multiplication
- Division


Allegory of Arithmetic, from "Margarita Philosophica," 1504 by Gregor Reisch

# ADDITION and SUBTRACTION

# Design Tradeoffs

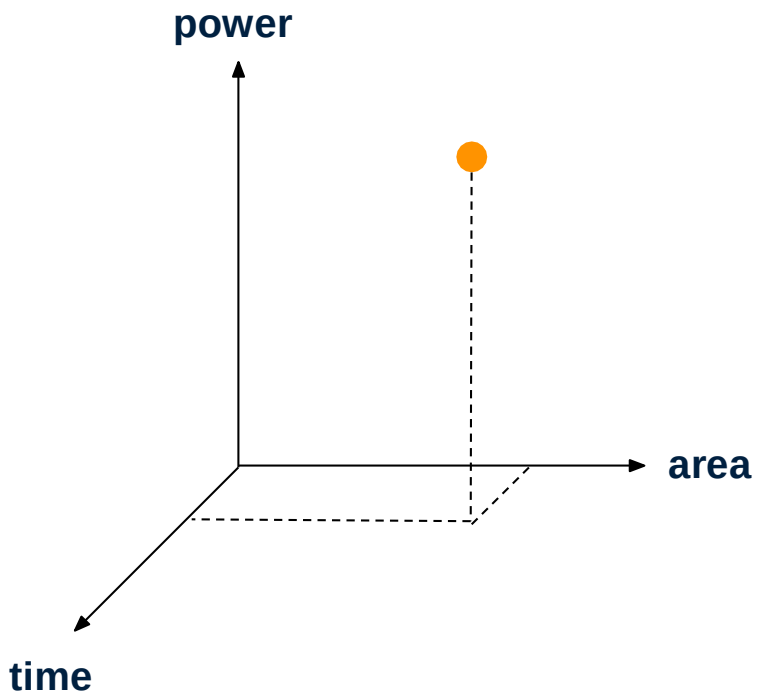
There are many ways to build adders (or any function).

**Which is the right implementation?**

Depends on your system's requirements

**Optimization Metrics:**

- Speed
- Power
- Area



# Single Cycle vs. Multi Cycle Arithmetic

Arithmetic units can be written as:

1. Combinational Blocks
   - Outputs depend only on inputs
   - Results available in one clock-cycle

2. Multi-Cycle
   - Result is computed over multiple cycles
   - Can be much smaller (require fewer logic resources)
   - Can be pipelined for higher clock frequency

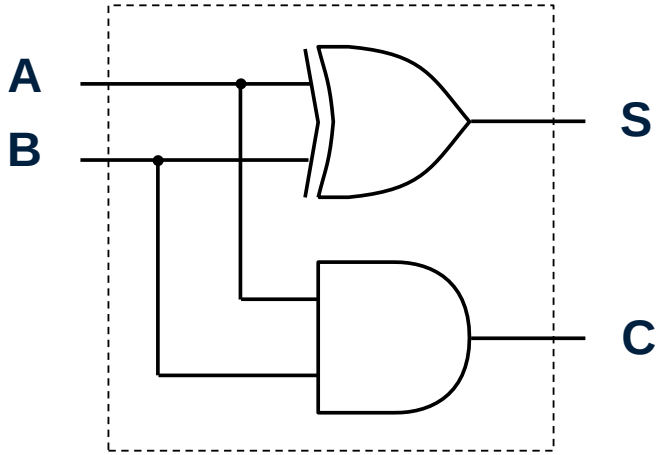
# Common Adder Architectures

- Carry Ripple/Propagate Adder
- Carry Select Adder
- Carry Look-ahead Adder
- Carry Save Adder

# Quick Review of Half and Full Adders

# Review: 1-Bit (Half) Adder

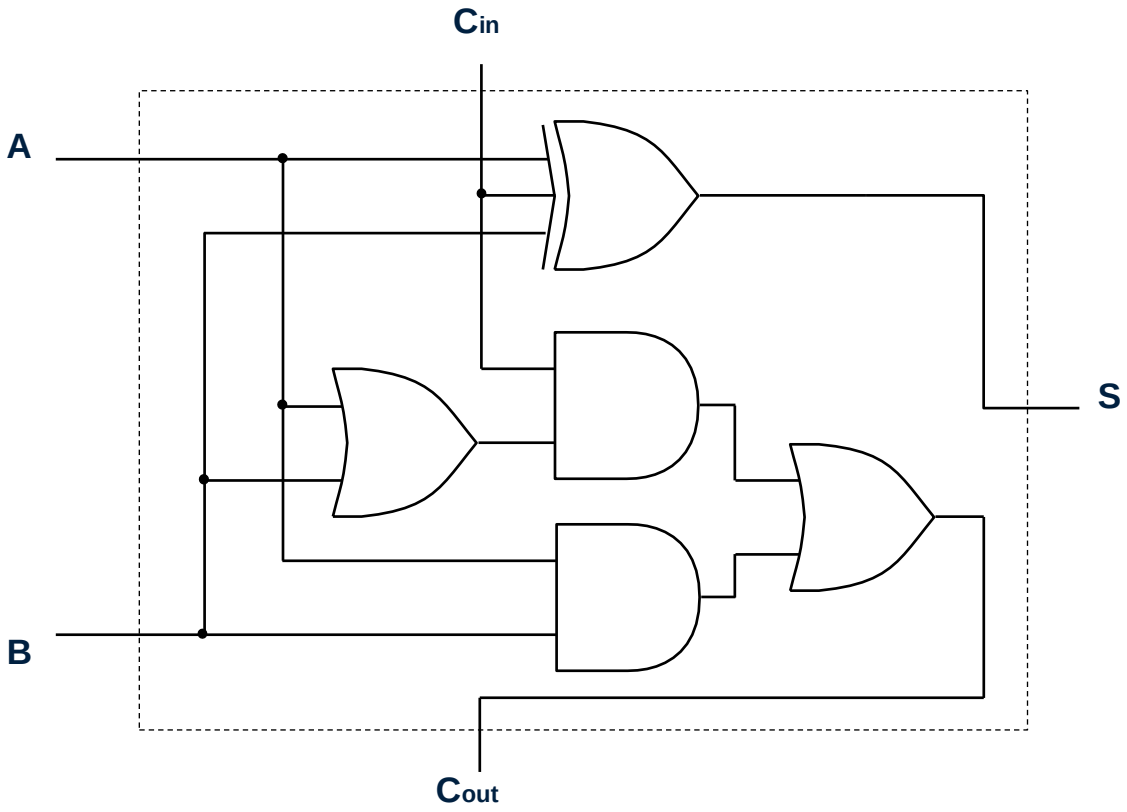
| A | B | C | S |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |




**What if we want to add more than one bit?**

# Review: Full Adder

| A | B | $C_{in}$ | $C_{out}$ | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |



# Aside: How do we perform Subtraction?

# Aside: Subtraction

**Can perform subtraction with addition**

$$A - B = A + (-B)$$

Recall, Two's Complement Numbers:

$$-B = \overline{B} + 1$$

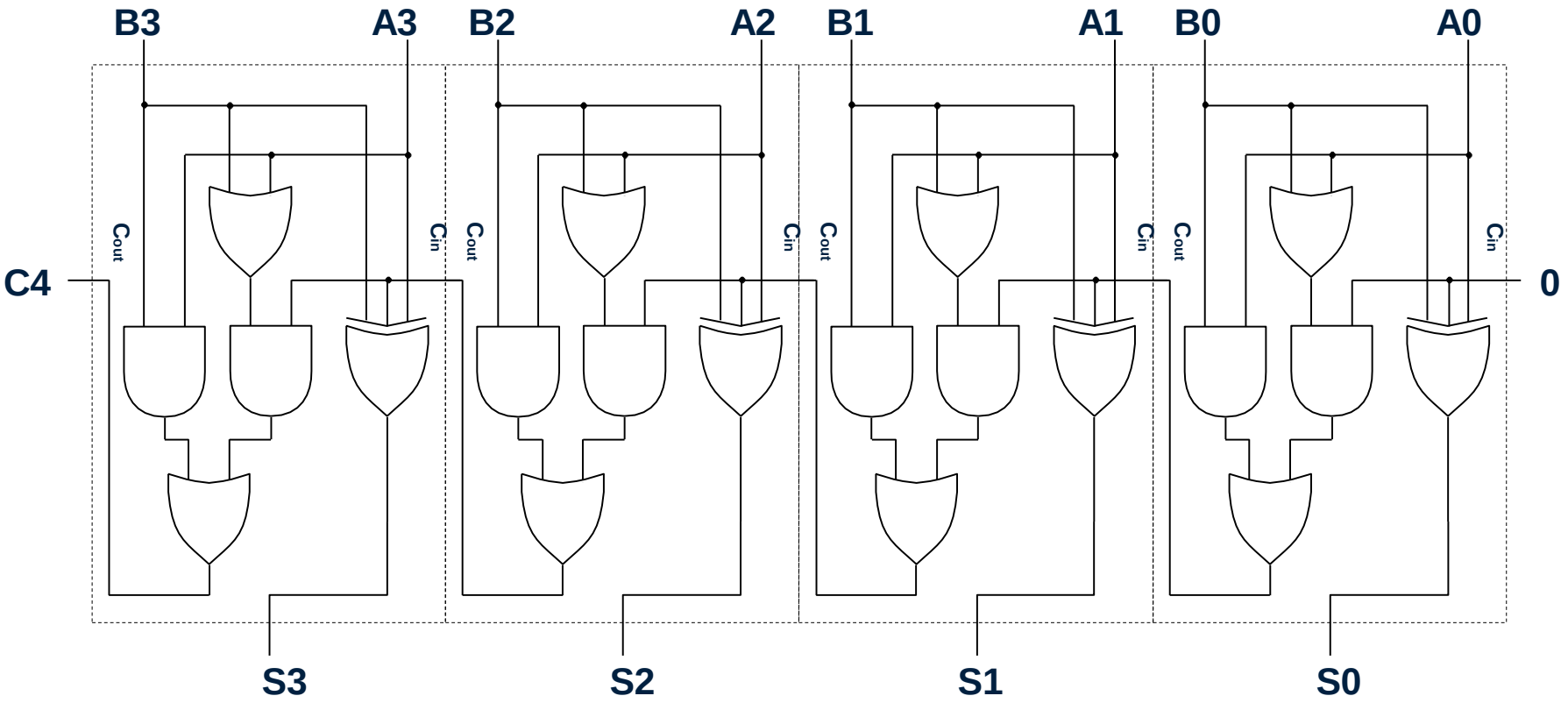
(Invert all bits of **B** and add 1)

$$A - B = A + \overline{B} + 1$$

# Carry Propagate/Ripple Adder

# Carry Propagate Adder (CPA)

**Connect Full Adders to make Carry Propagate Adder (Ripple Adder)**

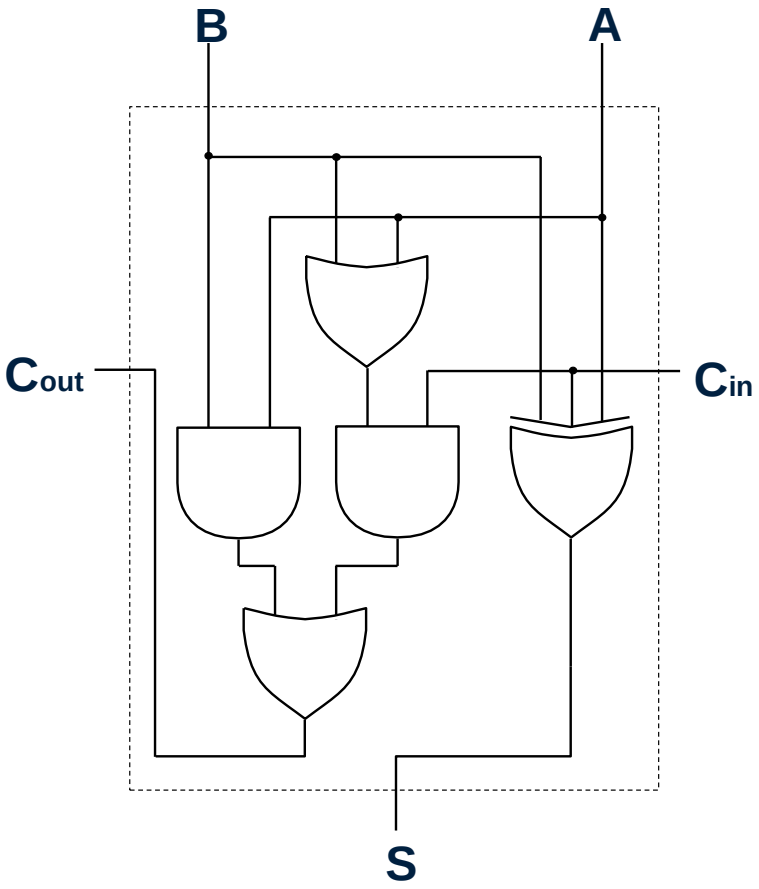



- Right-most stage is least-significant bit (LSB)
- Carry-out of previous stage feeds into Carry-In of next stage
- Can extend to any number of bits
  - not entirely true, we'll see that later…

# How fast is this Ripple Carry Adder?

# Delay of a Full Adder

**What is the critical path delay of a Full Adder?**




Assume all gates have the same **gate propagation delay $t_{PD}$**

If inputs **A**, **B**, **$C_{in}$** arrive at time 0,

- **S** is ready after **$t_{PD}$**

  A, B, $C_{in}$ ↠ XOR Gate ↠ S

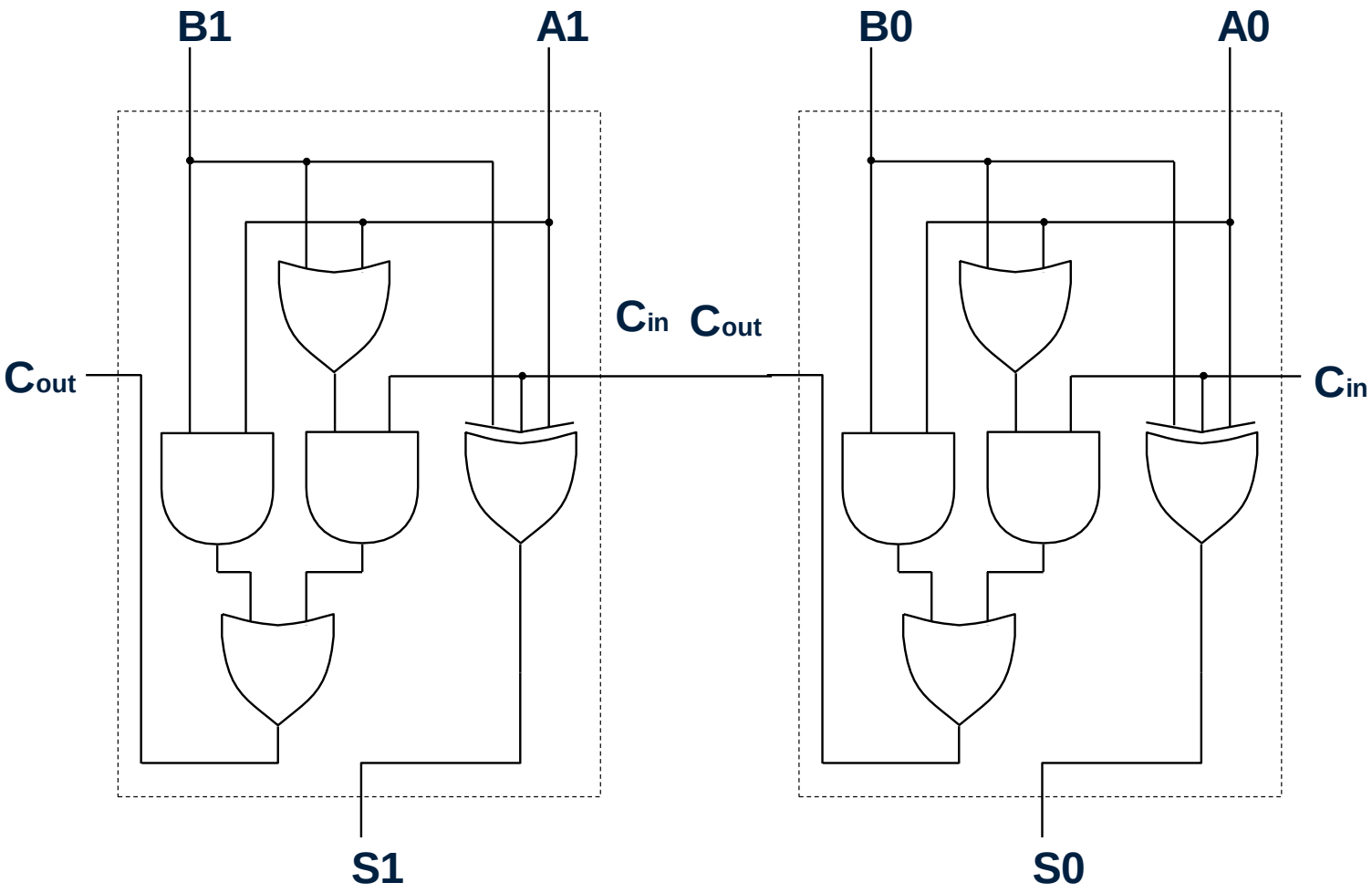
- **$C_{out}$** is ready after **3 $t_{PD}$**

  A, B ↠ OR ↠ AND ↠ OR ↠ $C_{out}$

**Critical Path Delay is 3 $t_{PD}$**

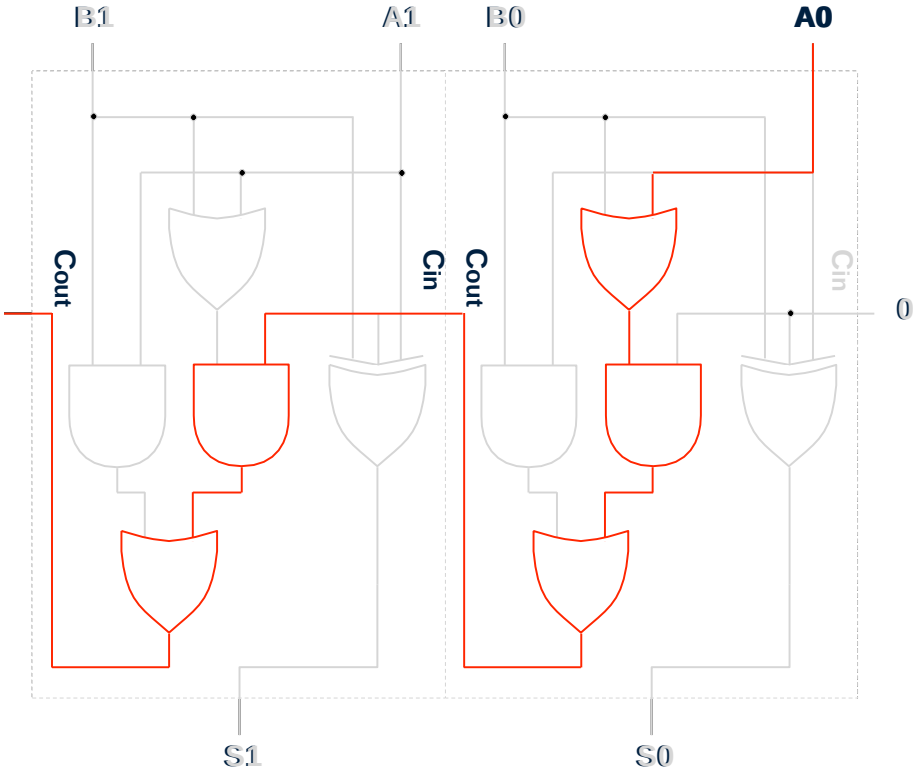
# Delay of Carry Propagate Adder

## Consider 2-bit Carry Propagate Adder



# Delay of Carry Propagate Adder

## Consider 2-bit Carry Propagate Adder




Inputs **A0, A1**, **B0, B1** arrive at time 0,

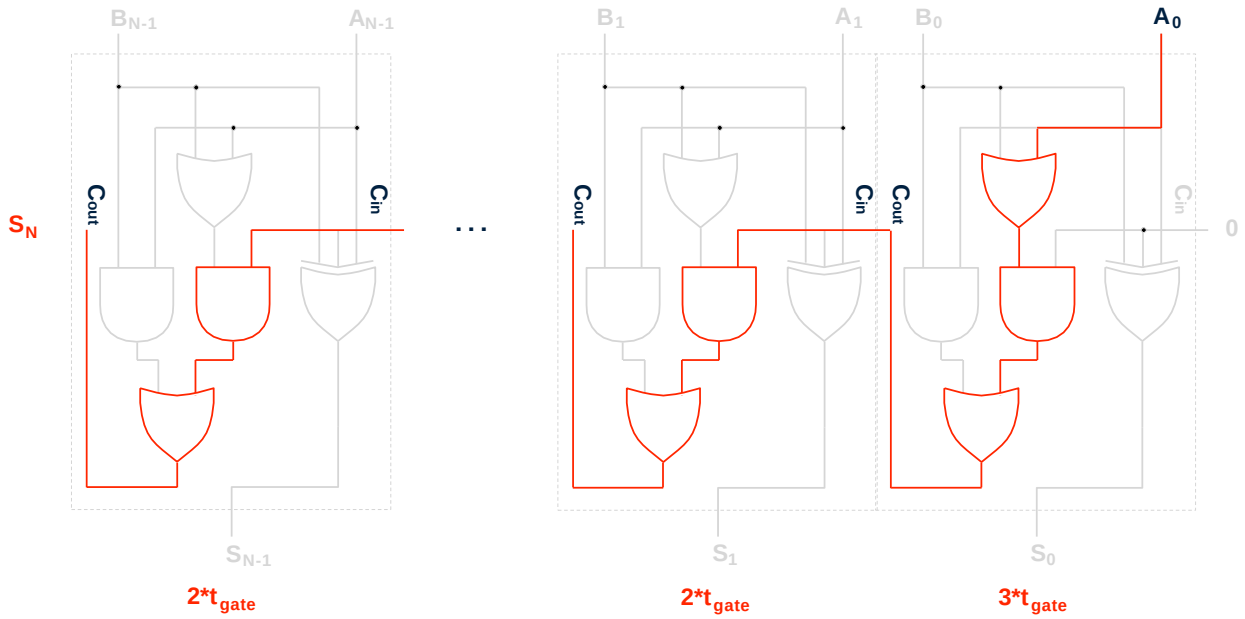
- $\mathbf{C_{out}}$ of first bit is ready after **3 $t_{PD}$**
- Delay of next $\mathbf{C_{out}}$ is ready after another **2 $t_{PD}$**

**Critical Path Delay is 5 $t_{PD}$**

# Delay of Carry Propagate Adder

**Delay for an N-bit Carry Propagate Adder is**

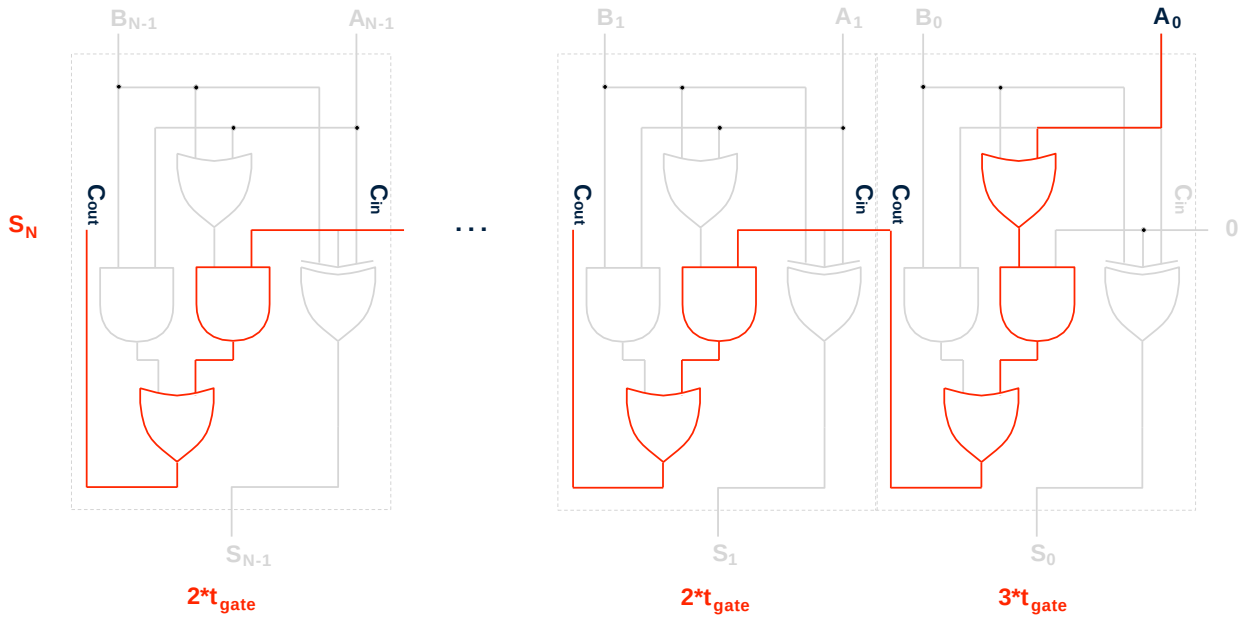
$$2(N-1)t_{PD} + 3t_{PD}$$




**Delay is proportional to N ↠ SLOW for large N**

# Problem with Carry Propagate Adder

Carry Propagate Adders are slow because the high-order (sum) bits need to wait for the carry-in from lower-order bits

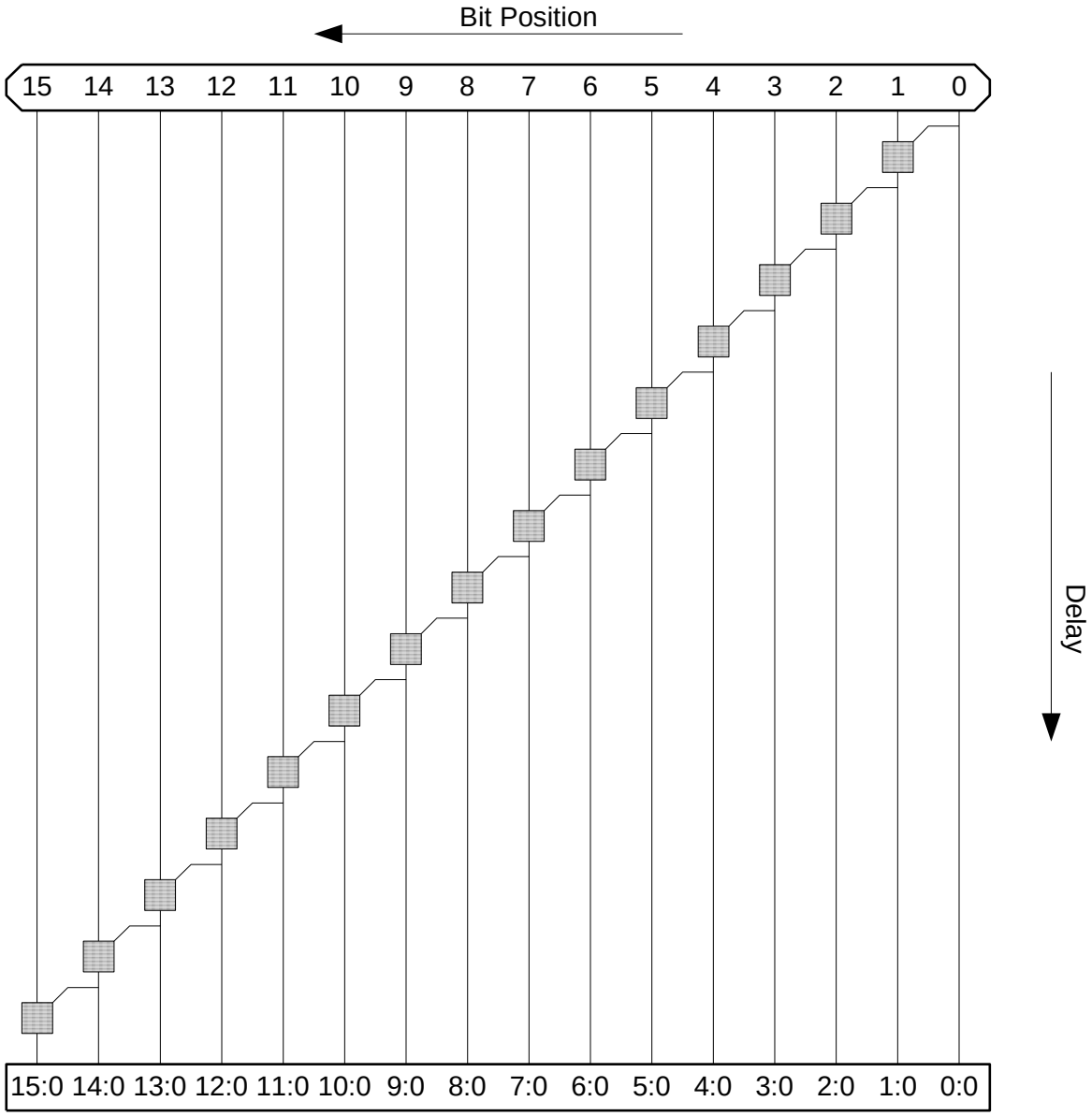



**Delay is proportional to N ↠ SLOW for large N**

*When 32- or 64-bit numbers are used, this delay may become unacceptably high !!!*

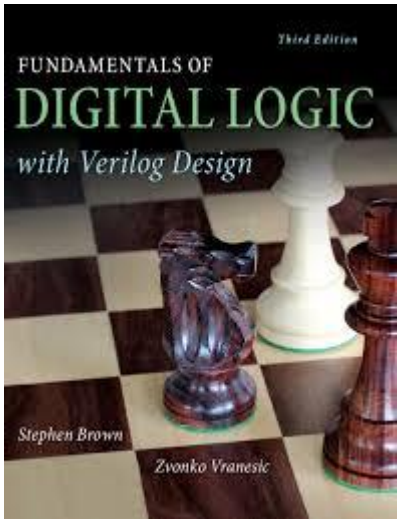
# Problem with Carry Propagate Adder

*When 32- or 64-bit numbers are used, this delay may become unacceptably high !!!*



# Recommended Reading

- Digital System Design with Verilog HDL, 3/e, b **S**tephen Brown and **Z**vonko Vranesic. [**S&Z**]
  - S&Z,
    - Chapter-3
      - 3.2



# THANK YOU