DigiPen
INSTITUTE OF TECHNOLOGY

CS100 #07

# Arithmetic Unit

Vadim Surov

# Introduction

- ALU is a combinational digital electronic circuit that performs arithmetic and bitwise operations on integer binary numbers.

- A central processing unit (CPU), floating-point unit (FPU) or graphics processing units (GPU) may contain multiple ALUs.

# ALU

- Data: binary numbers as operands for operators
- Operators:
  - Arithmetic: add, subtract, increment, …
  - Bitwise logical
  - Bit shift
- Status: supplemental information about the result of an operation
  - Ex: Overflow, which indicates the result of an arithmetic operation has exceeded the numeric range of output.

# Review: Binary Addition

- Binary addition follows a similar procedure as decimal addition:
  - When adding numbers we add each column separately.
  - Then if the addition of the digits in the column generates a carry the carry is then added to the next column to the left.
- Thus, a column (other than the first one) may receive a carry from the column to its right and can generate a carry as well.

# Review: Binary Addition

- The binary addition table showing the carry underlined:

| + | 0 | 1 |
|---|---|---|
| 0 | 00 | 01 |
| 1 | 01 | 10 |

- Consider the second column in the following addition:

| **Example** | | | |
|---|---|---|---|
| Carry | Carry out:1 | Carry in :1 | |
| | 0 | 1 | 1 |
| | 0 | 1 | 1 |
| Sum | 1 | 1 | 0 |

# Half Adder

- In order to build the device one has to take a closer look to the table of addition.
- The binary addition table showing the carry underlined can be consider to be the merger of two separate tables, one representing the sum and the second representing the carry:

# Half Adder

| + | 0 | 1 |
|---|---|---|
| *0* | 00 | 01 |
| *1* | 01 | 10 |

=

| ***sum*** | 0 | 1 |
|---|---|---|
| *0* | 0 | 1 |
| *1* | 1 | 0 |

+

| ***carry*** | 0 | 1 |
|---|---|---|
| ***0*** | 0 | 0 |
| ***1*** | 0 | 1 |

# Half Adder

- The sum table is identical to the one generated by an XOR gate.

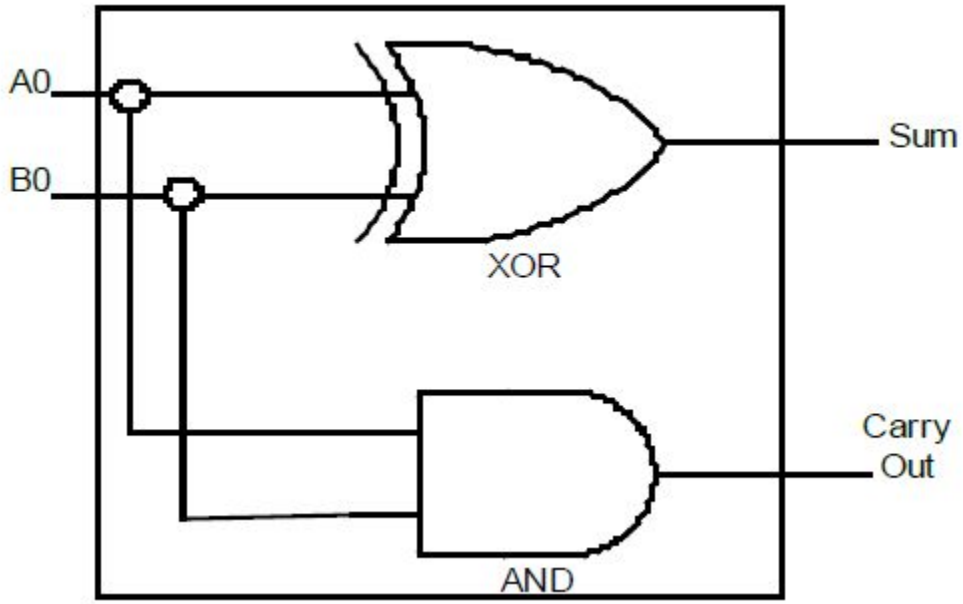
| XOR | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 0 |

- And the carry table is identical to the one generated by an AND gate.

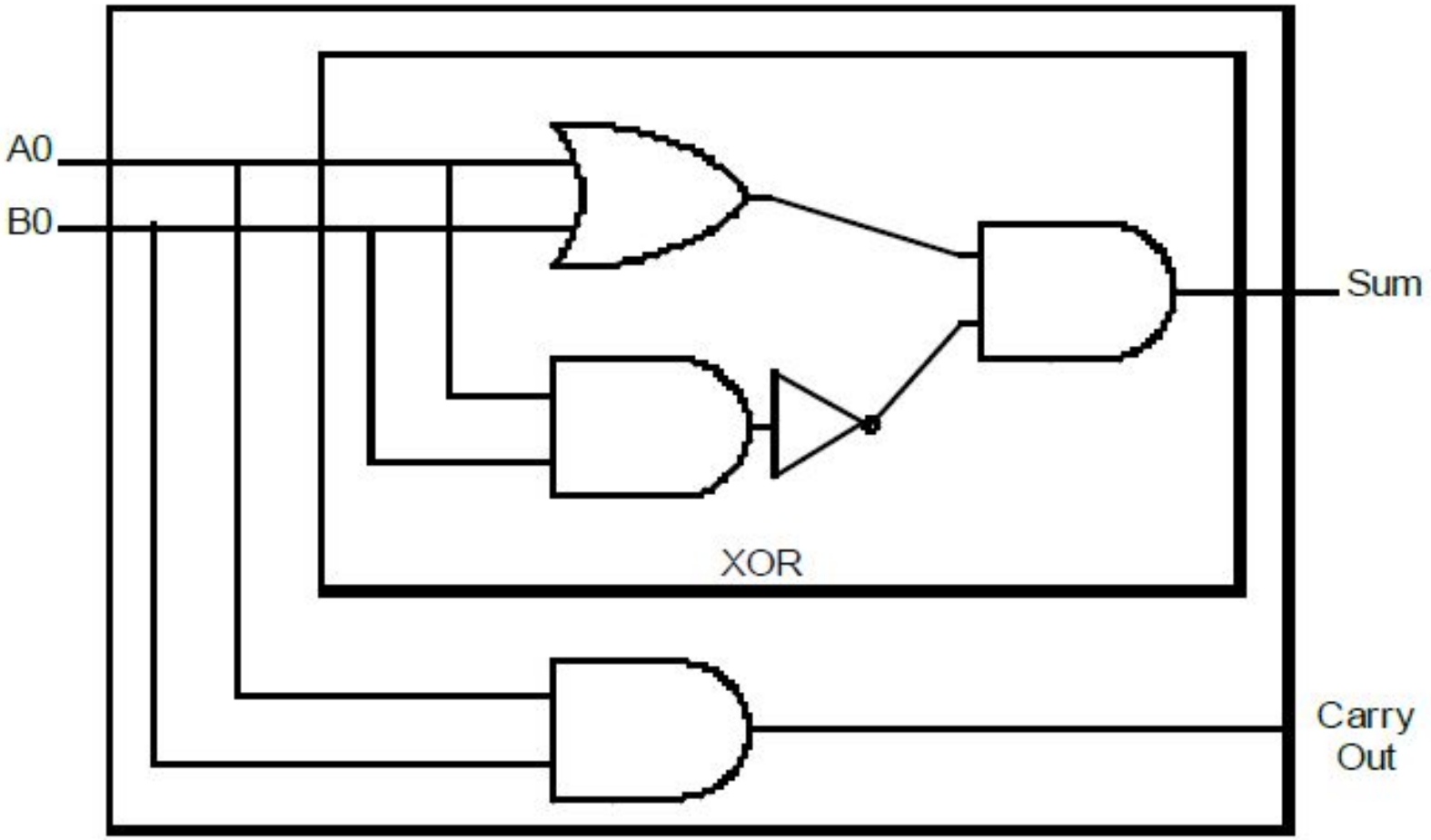
| AND | 0 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |

# Half Adder

- The following diagram shows how the XOR and the AND gates are coupled together in order to generate the first version of the 2-bit adder we are trying to build.
- This early version is called a half adder:

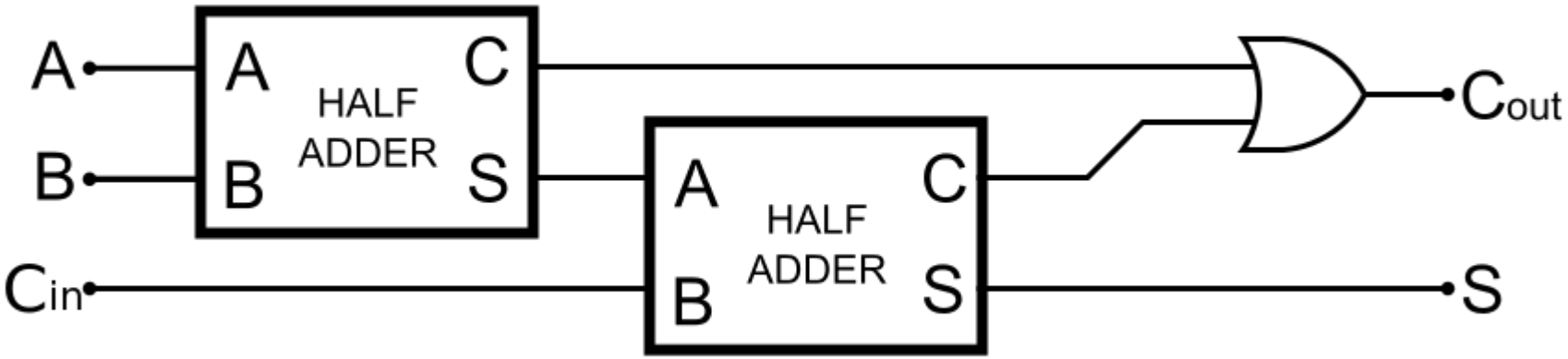


# Half Adder



# Full Adder

- The solution to the problem is obtained by using two half adders connected together with an OR gate as shown in the diagram below:



# Full Adder

- The logic expressions for this full adder are:
  - $S = (A \text{ XOR } B) \text{ XOR } C_{in}$
  - $C_{out} = AB + C_{in}(A \text{ XOR } B)$

- By applications of boolean algebra, the second statement equivalent to
  - $C_{out} = AB + BC_{in} + AC_{in}$

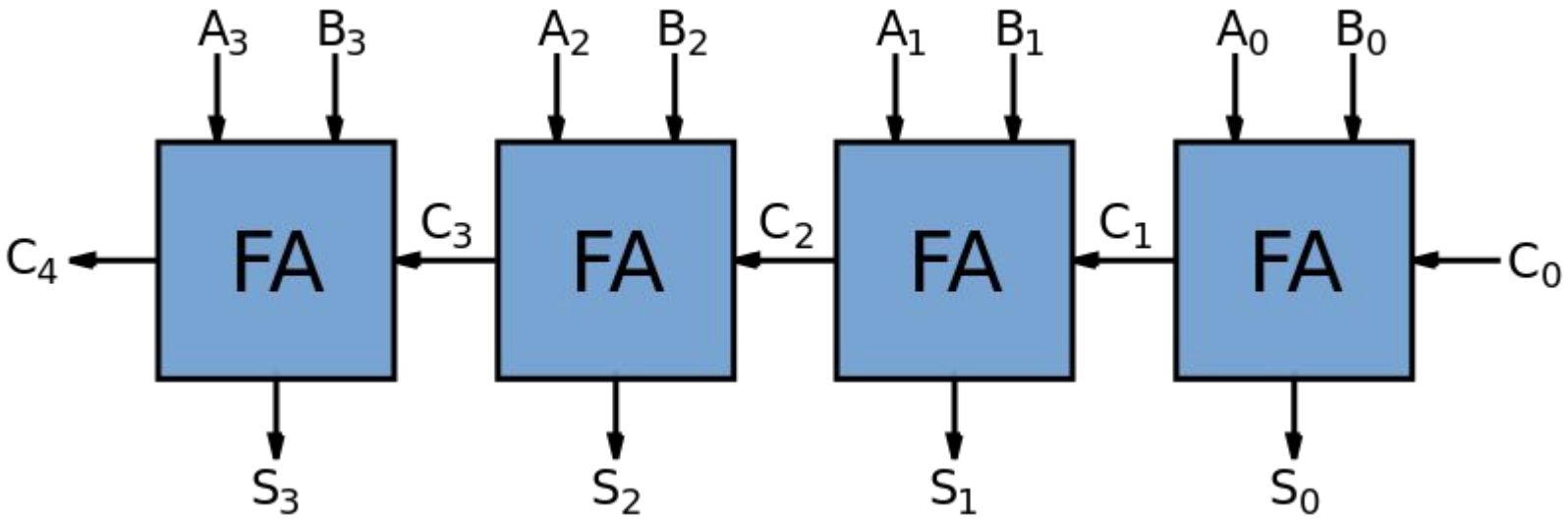
# Truth Table For A Full Adder

| 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 |

Binary number of 1s in input

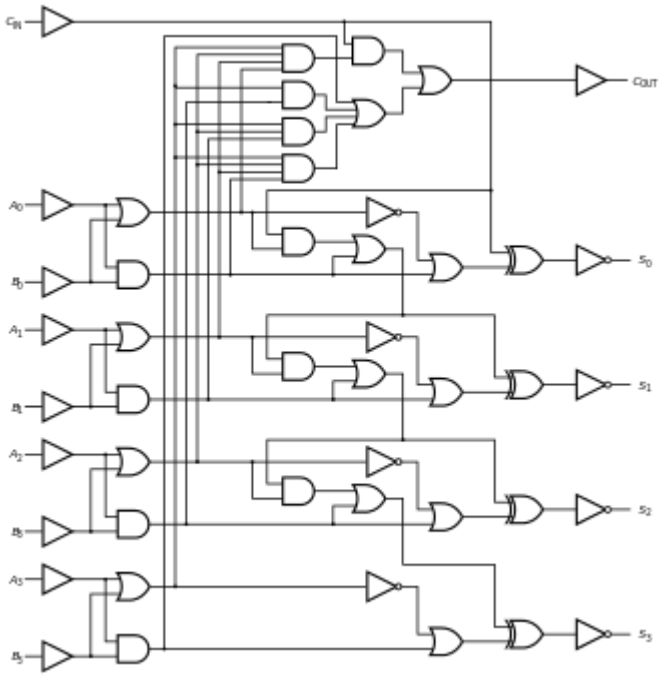
# N-bit Adder

- Several full adders can be combined together to add numbers that are more than one bit wide.
- The following diagrams represent a 4-bit adder:

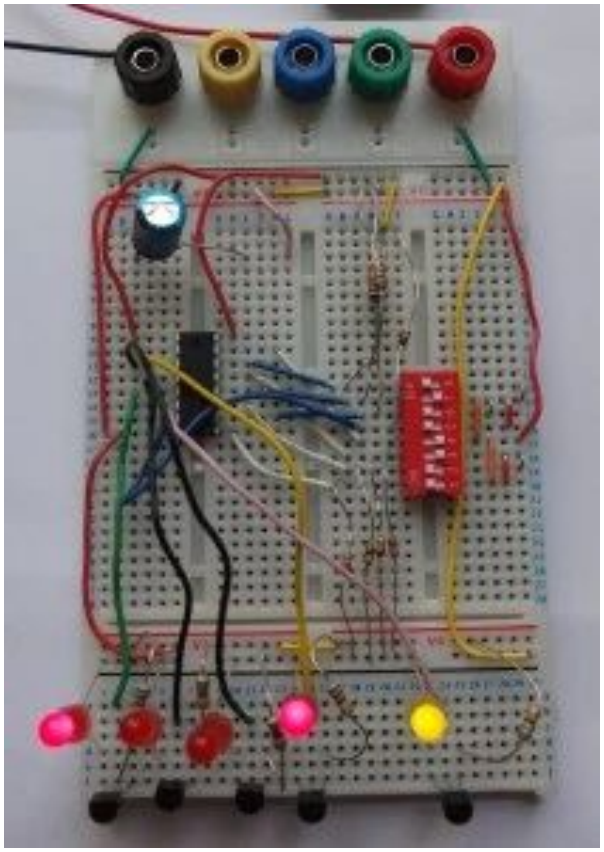


# 4008

- The 4008 is a 4-bit full adder that takes two 4-bit binary numbers, A[3:0] and B[3:0] and carry-in signal, $C_{in}$ and adds them to produce a 5-bit result, made up of a sum, S[3:0] and a carry-out.

# The 4008 On A Board



Source:
https://simplestcomputer.wordpress.com/route-map/stage-2-arithmetic-handling-numbers/full-adder/

# References


- https://en.wikipedia.org/wiki/Arithmetic_logic_unit
- https://en.wikibooks.org/wiki/Practical_Electronics/Adders