Chapter 4

* Adder – in its simplest form, a digital electronic device that performs the operation of addition on two binary digits (the …. And addend), known as binary adder
* Half-adder – is an implementation of an adder that has provision only for the input of addend and augend bits as is capable of generating sum and carry output
* Full adder – has provisions for inputs of addend, augend, and carry bits. It is capable of operations sum and carry outputs. These adders maybe cascaded when it is desired to add binary words greater than one bit in length by connection together the carry inputs and outputs of adjacent stages.
* 0 + 0 = 0  
  1 + 0 = 1  
  0 + 1 = 1  
  1 + 1 = 0, carry 1
* Gate - device that performs a basic operation on electrical signals. Its accepts one or more input signals and produces a signal output signal.
* Circuits – gates combined to perform more complicated tasks.
* Boolean expressions - uses Boolean algebra, a mathematical notation for expres two-valued logic
* Logic diagrams – a graphic representation of a circuit; each gate has its own symbol
* Truth tables– a table showing all possible input value and the associated output vales
* Gate Types – typically, logic diagrams are black and white with games distinguished only by their shape
  + Not
    - Accepts one input signal (0 or 1 ) and returns the opposite signal as output

|  |  |
| --- | --- |
| * + - A | * + - X |
| * + - 0 | * + - 1 |
| * + - 1 | * + - 0 |

* + - A = X'
  + And
    - Accepts two input signals

|  |  |  |
| --- | --- | --- |
| * + - A | * + - B | * + - X |
| * + - 0 | * + - 0 | * + - 0 |
| * + - 0 | * + - 1 | * + - 0 |
| * + - 1 | * + - 0 | * + - 0 |
| * + - 1 | * + - 1 | * + - 1 |

* + - Boolean Expression: x = a \* b
  + Or
    - Accepts two input signals
    - Opposite of And Gate

|  |  |  |
| --- | --- | --- |
| * + - A | * + - B | * + - X |
| * + - 0 | * + - 0 | * + - 0 |
| * + - 0 | * + - 1 | * + - 1 |
| * + - 1 | * + - 0 | * + - 1 |
| * + - 1 | * + - 1 | * + - 1 |

* + - Boolean expression is X = A + B
  + Xor – exclusive or
  + Nand – not and
  + Nor – not or
* If ((a1 = b1) and (c1 < d1))

|  |  |  |
| --- | --- | --- |
| * A | * B | * X |
| * 1 | * 0 | * 0 |
| * 0 | * 1 | * 0 |
| * 0 | * 0 | * 0 |
| * 1 | * 1 | * 1 |

* IF ((A1 = B1) OR (C1 < D1))

|  |  |  |
| --- | --- | --- |
| * A | * B | * X |
| * 0 | * 1 | * 1 |
| * 1 | * 0 | * 1 |
| * 0 | * 0 | * 0 |
| * 1 | * 1 | * 1 |

* Full Adder Truth Table

|  |  |  |  |
| --- | --- | --- | --- |
| * A | * B | * SUM | * CARRY |
| * 0 | * 0 | * 0 | * 0 |
| * 0 | * 1 | * 1 | * 0 |
| * 1 | * 0 | * 1 | * 0 |
| * 1 | * 1 | * 0 | * 1 |

* The carry out goes into the next adder, if they are cascaded.
* All the addition is happening in the ALU of the CPU
* Example
  + Addend: A0, A1
  + Aguend: B0, A1
  + Result: D0, D1
  + LSB (least significant bit): D0
  + NMSB (next most significant bit): D1