**Harold’s Boolean Algebra**

**Cheat Sheet**

12 September 2021

**Boolean Algebra**

|  |  |  |  |
| --- | --- | --- | --- |
| **Boolean Expression** | **Law or Rule** | **Equivalent Circuit** | **Description** |
|  | Annulment  (OR) | universal parallel circuit | A in parallel with closed = “CLOSED” |
|  | Annulment  (AND) | universal series | A in series with open = “OPEN” |
|  | Identity  (OR) | universal parallel | A in parallel with open = “A” |
|  | Identity  (AND) | universal series circuit | A in series with closed = “A” |
|  | Idempotent  (OR) | idempotent parallel circuit | A in parallel with A = “A” |
|  | Idempotent  (AND) | idempotent series circuit | A in series with A = “A” |
|  | Double Negation |  | NOT NOT A (double negative) = “A” |
|  | Complement  (OR) | complement parallel circuit | A in parallel with NOT A = “CLOSED” |
|  | Complement  (AND) | complement series circuit | A in series with NOT A = “OPEN” |
|  | Commutative  (OR) | absorption parallel circuit | A in parallel with B = B in parallel with A |
|  | Commutative  (AND) | absorption series circuit | A in series with B = B in series with A |
|  | Distributative  (OR) |  | Permits the multiplying or factoring out of an expression |
|  | Distributative  (AND) |  |
|  | Associative  (OR) |  | Allows the removal of brackets from an expression and regrouping of the variables |
|  | Associative  (AND) |  |
|  | Absorptive  (OR) |  | Enables a reduction in a complicated expression to a simpler one by absorbing like terms |
|  | Absorptive  (AND) |  |
|  | Absorptive  (Derived) |  | Reduces a complicated expression to a simpler one by absorbing compliment term |
|  | De Morgan’s Theorem  (NOR) | | Invert and replace OR with AND |
|  | De Morgan’s Theorem  (NAND) | | Invert and replace AND with OR |

Source: <https://www.electronics-tutorials.ws/boolean/bool_6.html>

**Boolean Logic Gates**

|  |  |  |  |
| --- | --- | --- | --- |
| **Boolean Logic** | **Notation** | **Gate** | **Description** |
| **IDENTITY** | 1  T  True |  | On, Tautology, High voltage (typically +5V) |
| **NULL** | 0  F  ⊥  False | **GND**  Learn more | Circuit Playground: G is for Ground | Adafruit ... | Off, Contradiction, Low voltage (typically 0V) |
| **Input** | A, B, C, D |  | Line, Wire, Connects to |
| **Output** | W, X, Y, Z |  | Line, Wire, Connects from |
| **AND** | A ∧ B  A ∩ B | AND symbol | AND, BUT, Multiply, Conjunction, Intersection |
| **OR** | A ∨ B  A ∪ B  A | B | OR symbol | Inclusive-OR, Add, Disjunction, Union |
| **NOT** |  | NOT symbol | NOT, Invert, Negation, Change, Difference |
| **NAND** | A ⊼ B  A | B\* | NAND symbol | Not AND |
| **NOR** | A ⊽ B  A ↓ B | NOR symbol | Not OR |
| **XOR** | A ⊕ B  A ⊻ B | XOR symbol | Exclusive-OR, Both A and B are different |
| **XNOR** | A ⊙ B | XNOR symbol | Exclusive-NOR, Both A and B are the same |

**Boolean Logic Truth Tables**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Inputs** | | **Outputs** | | | | | | | | |
| **A** | **B** | **AND** | **NAND** | **OR**  **+** | **NOR** | **XOR**  ⊕ | **XNOR**  ⊙ | **NOT** | **VCC**  **1** | **GND**  **0** |
| **0** | **0** | 0 | 1 | 0 | 1 | 0 | 1 | A=1 | 1 | 0 |
| **0** | **1** | 0 | 1 | 1 | 0 | 1 | 0 | A=1 | 1 | 0 |
| **1** | **0** | 0 | 1 | 1 | 0 | 1 | 0 | A=0 | 1 | 0 |
| **1** | **1** | 1 | 0 | 1 | 0 | 0 | 1 | A=0 | 1 | 0 |

**Blank Truth Tables**

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Inputs** | | | **Output** | |
| **A** | **B** | **C** | **X** | **Y** |
| **0** | **0** | **0** |  |  |
| **0** | **0** | **1** |  |  |
| **0** | **1** | **0** |  |  |
| **0** | **1** | **1** |  |  |
| **1** | **0** | **0** |  |  |
| **1** | **0** | **1** |  |  |
| **1** | **1** | **0** |  |  |
| **1** | **1** | **1** |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Inputs** | | | | **Output** | | |
| **A** | **B** | **C** | **D** | **X** | **Y** | **Z** |
| **0** | **0** | **0** | **0** |  |  |  |
| **0** | **0** | **0** | **1** |  |  |  |
| **0** | **0** | **1** | **0** |  |  |  |
| **0** | **0** | **1** | **1** |  |  |  |
| **0** | **1** | **0** | **0** |  |  |  |
| **0** | **1** | **0** | **1** |  |  |  |
| **0** | **1** | **1** | **0** |  |  |  |
| **0** | **1** | **1** | **1** |  |  |  |
| **1** | **0** | **0** | **0** |  |  |  |
| **1** | **0** | **0** | **1** |  |  |  |
| **1** | **0** | **1** | **0** |  |  |  |
| **1** | **0** | **1** | **1** |  |  |  |
| **1** | **1** | **0** | **0** |  |  |  |
| **1** | **1** | **0** | **1** |  |  |  |
| **1** | **1** | **1** | **0** |  |  |  |
| **1** | **1** | **1** | **1** |  |  |  |

**Karnaugh Mapping (K-Map)**

|  |  |  |  |
| --- | --- | --- | --- |
| 2-Bit  K-Map | | **A** | |
| 0 | 1 |
| **B** | 0 |  |  |
| 1 |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3-Bit  K-Map | | **AB** | | | |
| 00 | 01 | 11 | 10 |
| **C** | 0 |  |  |  |  |
| 1 |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4-Bit  K-Map | | **AB** | | | |
| 00 | 01 | 11 | 10 |
| **CD** | 00 |  |  |  |  |
| 01 |  |  |  |  |
| 11 |  |  |  |  |
| 10 |  |  |  |  |

2x2 Group

1x4 Group

**K-Map Rules**

1) Circle only 1s (ones) and don’t cares for Sum of Products (SOP), .

a. Circle only 0s (zeros) and don’t cares for Product of Sums (POS), .

b. Don’t cares may be used or ignored.

2) No diagonals, only horizontal or vertical connections.

3) Group only adjacent cells in groups with powers of 2 (1x1, 1x2, 2x1, 2x2, 2x4, 4x2, 1x4, 4x1).

4) Make groups as large as possible.

5) Must group all 1s (ones) for SOP or all 0s (zeros) for POS.

6) Overlapping is allowed.

7) Wrapping around all edges allowed, both top-bottom edges and left-right edges.

8) Fewest groups possible (OPTIMAL).

9) For each circle, determine which inputs do not contribute to the logic (is both 0 and 1).

10) Write down equation as a SOP,