Rule	Expression
Commutativity	$X + Y = Y + X$ $X \cdot Y = Y \cdot X$
Associativity	$(X + Y) + Z = X + (Y + Z)$ $(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z)$
Distributivity	$X \cdot Y + X \cdot Z = X \cdot (Y + Z)$ $(X + Y) \cdot (X + Z) = X + Y \cdot Z$
Covering	$X + X \cdot Y = X$ $X \cdot (X + Y) = X$
Combining	$X \cdot Y + X \cdot Y' = X$ $(X + Y) \cdot (X + Y') = X$
Consensus	$X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z$ $(X + Y) \cdot (X' + Z) \cdot (Y + Z) = (X + Y) \cdot (X' + Z)$
Redundancy	$X + X'Y = X + Y$
Generalized Idempotency	$X + X + \cdots + X = X$ $X \cdot X \cdot \cdots \cdot X = X$
DeMorgan's Theorems	$(X_1 \cdot X_2 \cdot \cdots \cdot X_n)' = X_1' + X_2' + \cdots + X_n'$ $(X_1 + X_2 + \cdots + X_n)' = X_1' \cdot X_2' \cdot \cdots \cdot X_n'$
Generalized DeMorgan's	$F(X_1, X_2, \ldots, X_n, +, \cdot) = F(X_1, X_2, \ldots, X_n, \cdot, +)'$
Shannon's Expansion	$F(X_1, X_2, \ldots, X_n) = X_1 \cdot F(1, X_2, \ldots, X_n) + X_1' \cdot F(0, X_2, \ldots, X_n)$ $F(X_1, X_2, \ldots, X_n) = [X_1 + F(0, X_2, \ldots, X_n)] \cdot [X_1' + F(1, X_2, \ldots, X_n)]$
Dual	Interchange $+$ with $\cdot$, and 0 with 1

```
//    !  : Logical NOT, ~ : Bitwise NOT
//    && : Logical AND, & : Bitwise AND
//    || : Logical OR,  | : Bitwise OR
//    ^  : Bitwise XOR
//    ~^ : Bitwise XNOR
//    == : Equality,    === : Case Equality (4-state)
//    != : Inequality,  !== : Case Inequality (4-state)
```

```
module top_module (
  input  logic x1,
  input  logic y1,
  output logic f1
);
  // Module internals here
endmodule

module sub_module (
  input  logic x2,
  input  logic y2,
  output logic f2
);
  // Submodule internals here
endmodule

sub_module sub_inst(.x2(x1), .y2(y1), .f2(f1));
```

```
// Sequential (synchronous) logic
always_ff @(posedge clk or posedge rst) begin
  if (rst)
    out <= 0;
  else
    out <= ~out;
end

// Combinational logic
always_comb begin
  // assignments that depend solely on input combinatorics
end
```

```
interface simple_if (input logic clk);
  logic data;
  modport master (output data);
  modport slave  (input  data);
endinterface
```

Buffer

A	Output
0	0
1	1

NOT

A	Output
0	1
1	0

OR

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

AND

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

NAND

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

NOR

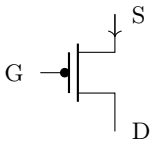
A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

XOR

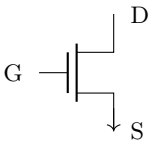
A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

XNOR

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	1



PMOS



NMOS

Parameter	**PMOS**	**NMOS**
Substrate Type	n-type	p-type
Threshold Voltage	Negative	Positive

Gate	PMOS Configuration	NMOS Configuration
OR($A + B$)	Parallel	Series
AND($A \cdot B$)	Series	Parallel

$$N_{MH} = V_{OH} - V_{IH}$$
$$N_{ML} = V_{IL} - V_{OL}$$
$$N_{MH}(A \to B) = V_{OHA} - V_{IHB}$$
$$N_{ML}(A \to B) = V_{ILB} - V_{OLA}$$

t_{pHL}: The time between an input change and the corresponding output change when the output changes from HIGH to LOW.

t_{pLH}: The time between an input change and the corresponding output change when the output changes from LOW to HIGH.

- Note: t_p is measured from the 50% crossing of the input to the 50% crossing of the output

t_f: Time between 90% of output high and 10% output low

t_r: Time between 10% of output low and 90% output high

$$P = cV^2 f$$