BA2 DSD Résumé

Logical operators:

n inputs → 2^n combinations

A	B	A · B	A + B	A ⊕ B = Ā·B + A·B̄
0	0	0	0	0
1	0	0	1	1
0	1	0	1	1
1	1	1	1	0

"not" $\bar{A}$ (¬A)

First ← → last : ¬ (·) · ⊕ +

conjuction "and" · (∧) — disjunction "or" + (∨) — exclusive or (logic exclusion) "xor" ⊕ = $\bar{A}\cdot B + A\cdot\bar{B}$ = $(\neg A \wedge B) \vee (A \wedge \neg B)$

ANSI / IEC

Analog: range of values
Digital: two values (1,0)
0·F 1·T

Result = 1 : minterms — disjunctive Form = sum of products (only sums)
Result = 0 : maxterms — conjunctive Form = product of sums (only products)

↓ : positive edge, ↑ : negative edge

Properties:

$A\cdot A = A$ $A+A=A$ $A\oplus A=0$ $A\cdot 0=0$ $A\cdot 1=A$ $A+0=A$ $A+1=1$ $A\oplus 0=A$ $A\oplus 1=\bar{A}$

$A\cdot(A+B)=A$ $A+(A\cdot B)=A$ $A\cdot\bar{A}=0$ $A+\bar{A}=1$ $A\oplus\bar{A}=1$ $\bar{\bar{A}}=A$

$A\cdot(B+C)=(A\cdot B)+(A\cdot C)$ $A+(B\cdot C)=(A+B)\cdot(A+C)$ $A\cdot(B\oplus C)=(A\cdot B)\oplus(A\cdot C)$ $A\oplus(B\oplus C)=(A\oplus B)\oplus C$

⚠ $A\oplus(B\cdot C)\neq(A\oplus B)\cdot(A\oplus C)$

"NAND" $\overline{A\cdot B}=\bar{A}+\bar{B}$ "NOR" $\overline{A+B}=\bar{A}\cdot\bar{B}$

Karnaugh groups:

2^m minterms/maxterms → m variables don't care (Symetry) ($\bar{E}$)
var E is don't care when 2^{m-1} minterms/maxterms where E=1 and 2^{m-1} where E=0
var. E influences when all 2^m minterms/maxterms are where E=1 or where E=0

Circuits:

SR-Latch, D-Latch, D-FlipFlop

R	S	Q
0	0	Q
–	1	1
1	0	0
0	0	Q
0	1	1
1	–	0

E	Q
1	D
0	Q

Data 0 Store 0

C	Q
↓	D
–	Q

Data 0 clock 0

Clock:

T_1 time logic level 1
T_0 time logic level 0
$T_p = T_1 + T_0$, $f_c = \frac{1}{T_p}$

Representations:

nibble: 4 bits, byte: 8 bits, MSB ... LSB (most significant byte ... least significant byte (index 0)), encoding: thermometer: 00011111b (5 decimal), 00000001b (0 decimal), one-hot: 00100000b, 00000010b, binary [0, 2^k-1]

Sign & Magnitude (sign bit) [$-2^{k-1}+1, 2^{k-1}-1$]: 00b +0, 10b -0, 11010b -5
one's comp. (invert) [$-2^{k-1}+1, 2^{k-1}-1$]: 00b +0, 11b -0, 10101b -5
two's comp. (add. 2 numbers and + carry, −MSB) [$-2^{k-1}, 2^{k-1}-1$]: 00b 0, 11b -1, 10111b -5
excess N [−N, 2^k−N−1]: 00b −N, 01b −N+1
Fixed point: 2^{-1} 2^{-2} 11.01 3,25

Operations:

dec → B=7: 768 B=10 → 2145 B=7

Adder (carry): 212 B=3 + 21 B=3 = 1010 B=3
Subtract (borrow): 112 B=4 − 13 B=4 = 013 B=4

⚠ overflow (input=3 bits, output=4 bits)
⚠ underflow (negative result)

Div.: 1011b / 10 = 0101b
Mult.: 0101b × 11b = 01010 + 0101 = 01111 (overflow)

FSM:

CT: code table — encode IRL to binary — State → Code
TT: transition table — maps current outputs + inputs to new outputs — save to memory on clock ↓
OT: Output table — output Code → IRL encoding

Button: E, F, G, $\bar{B}$

never use the clock, FSM: ∀ inputs possibilities at least (complete) & at most (consistent) one active transition

CT + TT: Medvedev | + OT: Moore, + to output logic: Mealy (input directly connected)

encoding n states: $\lceil \log_2(n) \rceil$ bits (minimum), unused codes: ghost states (don't cares or safe state), POR: power-on-reset (from each state POR + C → Reset State (synchronous, inside transition logic); directly to memory (asynchronous))

waiting new state — clock — current state (as a condition)

Multiplexer:

C	Q
0	a
1	b

$Q = b\cdot c + a\cdot\bar{c}$

Shift-Register:

D → Q → Q → Q → Q, C

None Ideal behavior:

CMOS: complementary metal oxide semiconductor
TTL: transistor to transistor logic, Hazards: gates have delay ⇒ gated clocks don't work, D signal must be stable during t_{setup} and thold → Metastability (either 0 or 1)

Programmable Logic:

ASIC: Application Specific Integrated Circuit, PAL: Programmable Array Logic, OTP: one time programmable (antiFuses, break the wires), GAL: Generic Array Logic (switches: Floating gate transistor)

CPLD: Complex Programmable Logic Device (lot's of GAL + IOB, Sum of products), IOB: Input Output Block, FPGA: Field Programmable Gate Array (Truth Table (LUT), Shift-registers, pass-gates (tri-state buffers)), LUT: Look up Table, P&R software → bit-File (place and route), RAM: random access memory, SOC: System on chip

HDL (hardware description language) —synthesizer→ gates implementation

VHDL:

case-insensitive, comments --, Entity: black-box (has at least 1), Architecture: Functionality, std_logic: a bit, std_logic_vector(n-1 DOWNTO 0): n bits vector, bit value ∈ '0' '1' 'u' 'x' '-' (undefined, unknown (short circuit), don't care), CONSTANT: Fixed signal, SIGNAL: wire,

RTL: register transfer level: From combinational logic - connect with wires (separate sequential logic), Gates: AND NAND OR NOR NOT XOR XNOR, Sensitivity: signals on the right of <= sensity list (should put all), implicit, explicit

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity entity_name is
    generic(k: unsigned;
            L: std_logic := '0'); --default    MSB      LSB
    port(clock, reset: in std_logic;          -- (n-1) downto 0
         Y: out std_logic_vector(0 to 7)); --  0    to   (n-1)
end entity_name;                              LSB      MSB

architecture arch_name of entity_name is
    --Declaration
    type state is (IDLE, UPDATE, DONE); -- local to arch
    --hex X"EF", bin: "11101111", macro: (4 => '0', others => '1')
    signal s_Y_data: std_logic_vector(7 downto 0) := "00000000"; --default value
    signal s_A,s_B,s_C: std_logic;
    signal s_D, s_E: unsigned(5 downto 0);

    constant c_PI: std_logic := '1';

    component component_entity is
        generic(O: integer;
                P: std_logic_vector(2 downto 0));
        port(A,B: in std_logic;
             Y: out std_logic);
    end component;
```

```
begin
    --Description (don't read from output!)
    --Implicit process
    Y <= (s_Y_data(0) or (s_Y_data(2) and (not s_Y_data(3)))) & s_Y_data(6 downto 0);
    s_B <= c_PI when s_Y_data(5)='1' else '-' when s_Y_data(6)='L' else '0';
    with s_Y_data(4) select s_D <= --puts value in s_D
        to_unsigned(0,6) when s_Y_data(0), --tests data[4]=data[0]
        to_unsigned(1,6) when s_Y_data(2),
        to_unsigned(2,6) when others;
    --Explicit process (latest assign matters/don't assign both in implicit and explicit)
    p_process_name: process(reset, clock, s_Y_data(1)) is --sensitivity list
    begin
        -- D-FlipFlop
        --Async reset
        if reset='1' then s_A <='0'; --reset value
        elsif rising_edge(clock) then s_A <= s_B, -- falling_edge(clock)
        end if;
        --Sync reset
        if rising_edge(clock) then
            if reset='1' then s_A <='0';
            else s_A <= s_B xor s_A; --Allowed read and write to signal
            end if;                  --(value not updated right away)
        end if;

        case s_Y_data(1) is
            when s_Y_data(0) => --can use if/else here
                s_E <= to_unsigned(0,6);
                s_A <= '0';
            when s_Y_data(2) => s_E <= to_unsigned(1,6);
            when others => s_E <= k;
        end case;
    end process p_process_name;

    pm_do: component_entity
        generic map(O => 9,
                    P => "001")
        port map(A => s_A, B => s_B, Y => s_C);
end arch_name;
```

Help

Decimal	Bin	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Types