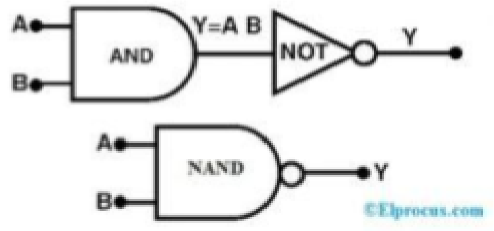
Artificial Power Traces

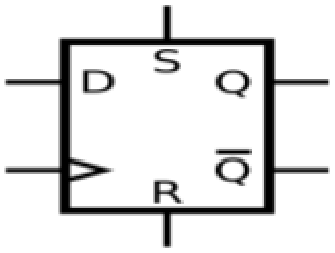
Logic Cells and Their Types

Logic cells are the fundamental building blocks of digital integrated circuits. They perform basic logical operations, such as AND, OR, and NOT, and are used to implement complex combinational and sequential logic circuits. Logic cells are categorized into two main types:

1. **Combinational Logic Cells**:
   - These cells perform operations based on the current input values.
   - Example: NAND gates, XOR gates, and multiplexers.
2. **Sequential Logic Cells**:
   - These cells rely on both current input values and stored past states, typically using feedback mechanisms.
   - Example: Flip-flops (e.g., D flip-flops) and latches.



NAND GATE

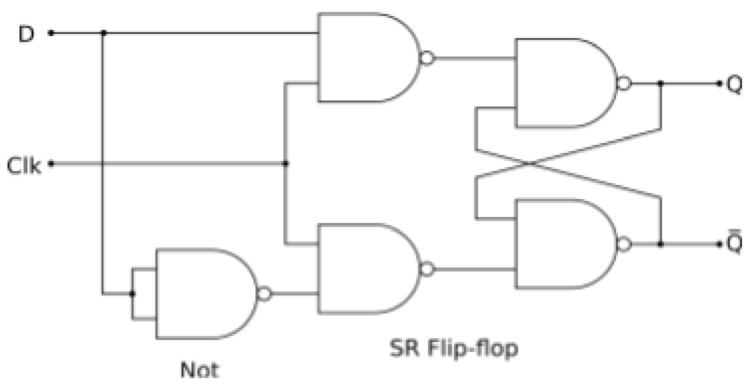


D-flip-flop

From NAND Gate to D Flip-Flop

The **NAND gate** is a universal gate capable of implementing any Boolean function. Its versatility makes it an essential building block for higher-level circuits. Here's how we progress from a NAND gate to a **D flip-flop**:

1. **NAND Gate**:
   - A logic gate that outputs a low signal (0) only when all its inputs are high (1).
   - Its output is given by Y=NOT(A ⋅ B)
2. **Latch (Level-Sensitive Storage)**:
   - By connecting NAND gates in a feedback loop, we create a latch, which can store a single bit of data.
   - A latch maintains its output state until an external signal changes it.
3. **D Flip-Flop (Edge-Sensitive Storage)**:
   - A **D (Data) flip-flop** is built using latches and additional logic to ensure that data is stored only on the clock edge (rising or falling).
   - The D flip-flop stores one bit of data and serves as the foundation for registers.

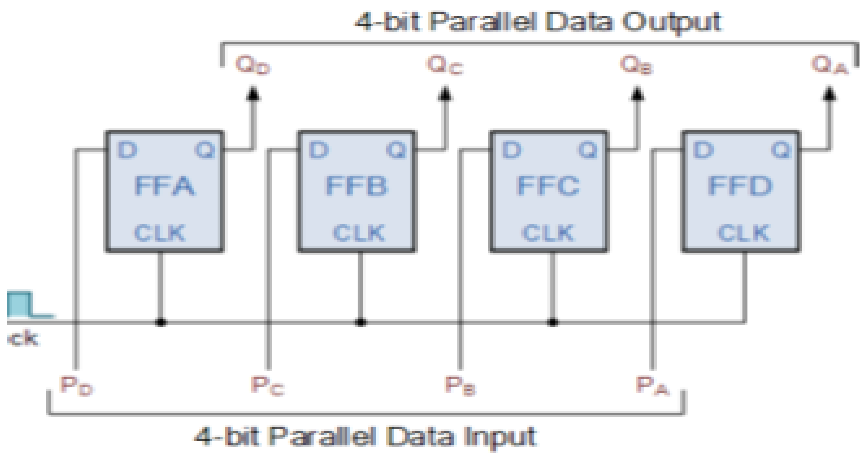


D-flip-flop using NAND gates

From D Flip-Flop to Registers

A **register** is a collection of D flip-flops organized to store multi-bit data. Each flip-flop in the register stores one bit of the intermediate or final computation result. Registers form the backbone of digital systems, enabling temporary data storage and retrieval during operations.

- **Working Mechanism**:
  - When the clock signal triggers, each flip-flop captures the data on its input and holds it until the next clock cycle.
  - Registers can be used for intermediate storage in arithmetic operations, counters, and pipeline stages.



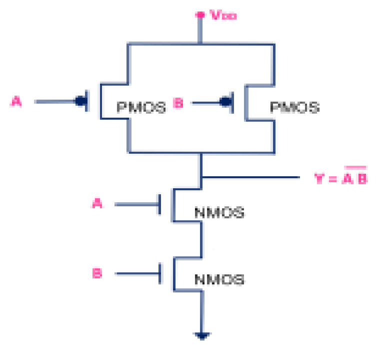
Logic Styles: Example of NAND Gate

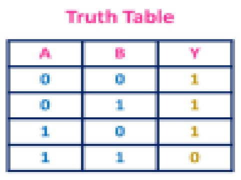
The NAND gate's operation in a digital circuit depends on its implementation style, which affects performance, power, and area. A common implementation style is **CMOS logic**, which uses complementary pairs of transistors.

- **CMOS NAND Gate**:
  - Consists of two NMOS transistors in series and two PMOS transistors in parallel.
  - **Working**:

    When both inputs are high (1), the NMOS transistors conduct, and the output node is pulled low (0).

    In all other cases, the PMOS transistors ensure the output is pulled high (1).





Truth Table

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Lumped Capacitance Model

The **lumped capacitance model** is a simplified representation of the capacitive effects in digital circuits, particularly at the transistor level. It aggregates various intrinsic and extrinsic capacitances into a single equivalent capacitance at a node.

Types of Capacitances in MOS Transistors

1. **Intrinsic Capacitances**:
   - These arise due to the MOSFET's internal physical structure.
   - **Gate Capacitance (Cgs,Cgd)**: Capacitance between the gate and source/drain terminals.
   - **Drain Capacitance (Cds)**: Capacitance between the drain and source.
   - These are significant because they determine the MOSFET's switching behavior.
2. **Extrinsic Capacitances**:
   - These are external to the MOSFET and arise due to parasitic effects.
   - **Interconnect Capacitance**: Capacitance due to wiring between transistors.
   - **Load Capacitance**: Represents the combined effect of the next stage's input capacitance and external components.

External Capacitor Representation

- To simplify modeling, we represent the total capacitance at a circuit node (both intrinsic and extrinsic) as a single **lumped capacitor**.It’s value ranges from 1fF to 1pF.
- This lumped capacitor approximates the combined effects of:
    - Gate capacitance of transistors driven by the node.
    - Capacitance of the wiring (interconnects).
    - Load capacitance at the node.

This approximation is valid for first-order analysis, where high precision is not required.

Charging and Discharging of the Lumped Capacitance

The charging and discharging of this equivalent capacitor define the power consumption and signal transitions in the circuit.

1. **Charging (Logic 0 → Logic 1)**:
    - When the output node transitions from 0 to 1, the PMOS transistor conducts.
    - Current flows from the power supply (VDD) to charge the capacitor.
    - The voltage across the capacitor rises exponentially, following:

$$V(t)=VDD \cdot (1-e^{-t/RC})$$

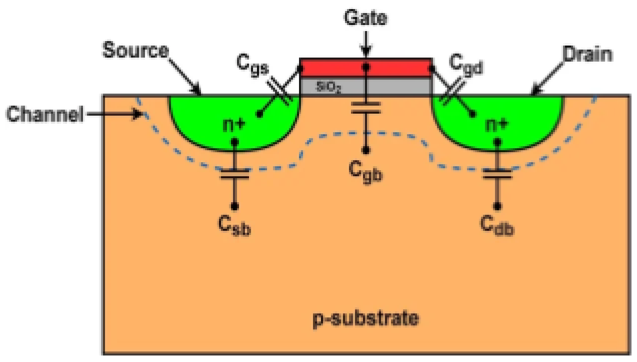
    - Energy consumed during charging is stored in the capacitor:

$$E_{charge}=½\ CV^2$$

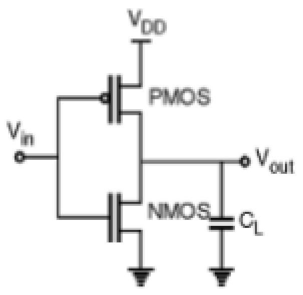
2. **Discharging (Logic 1 → Logic 0)**:
    - When the output node transitions from 1 to 0, the NMOS transistor conducts.
    - Current flows from the capacitor to the ground (GND).
    - The voltage decreases exponentially:

$$V(t)=VDD \cdot e^{-t/RC}$$

    - The energy stored in the capacitor is dissipated as heat during discharge.



Intrinsic Capacitance of N-MOS



External Load Capacitor

Factors Affecting Power consumption in CMOS circuits:

Dynamic Power Consumption: Key Factor

1. **Charging Current**:
   - When a logic gate transitions from 0 to 1, the Lumped capacitor as mentioned before charges.
   - Energy from the power supply is stored in the capacitor, creating an **upward curve** in the power trace during this phase.
   - This corresponds to the **high phase** of the clock cycle, driven by the rising edge of the clock signal.
2. **Discharging Current**:
   - During a transition from 1 to 0, the capacitor discharges, releasing stored energy to the ground.
   - This results in a **downward curve** in the power trace during the **low phase** of the clock cycle, driven by the falling edge of the clock.
3. **Short Circuit Current**:
   - When both NMOS and PMOS transistors conduct briefly during switching, a direct current path forms between the power supply and ground.
   - This causes a **small spike** in the power trace during transitions, adding to dynamic power consumption.

Clock signal

The **clock signal** plays a critical role in driving and synchronizing the transitions of digital circuits, directly influencing the charging and discharging behavior of the lumped capacitance model. Its periodic nature defines the timing of these transitions and shapes the overall power trace profile.

How the Clock Drives Charging and Discharging

1. **High Phase (Rising Edge)**:
   - When the clock transitions to its high phase, it triggers logic gates to switch states (e.g., 0 → 1).
   - This causes the capacitors in the circuit to **charge**, drawing current from the power supply.
   - In the power trace, this phase manifests as a **rise in the power profile**, representing energy being stored in the capacitance.
2. **Low Phase (Falling Edge)**:
   - As the clock enters its low phase, logic gates switch back (e.g., 1 → 0), and the capacitors **discharge**, releasing stored energy to the ground.
   - This phase appears as a **downward curve** in the power trace, reflecting the dissipation of energy from the capacitance.

Impact on the Overall Trace

The clock signal directly determines the timing of transitions, creating a repetitive pattern in the power trace:

- **Regularity**: The periodic nature of the clock ensures that charging and discharging occur at predictable intervals, leading to a repeating waveform in the power profile.
- **Shape**: The sharp changes in current at the rising and falling edges define the characteristic **peaks and troughs** of the trace.
- **Synchronization**: All logic transitions in the circuit are synchronized to the clock, meaning the overall power profile reflects the cumulative effect of these transitions.

Glitches

Glitches are temporary, unintended logic transitions in a circuit. These can occur due to:

- **Mismatched Signal Delays**: Signals propagating through different paths may arrive at logic gates at slightly different times, causing spurious transitions.
- **Logic Hazards**: Incorrect intermediate states arise during combinational logic switching before the circuit stabilizes at the correct output.

Effect on Traces:

- Glitches cause **unexpected current spikes** that add noise-like artifacts to the power trace.
- These artifacts are typically small and irregular but can occur during both charging and discharging phases.
- They complicate analysis by introducing unpredictable variations in otherwise periodic power profiles.

Electronic Noise

Electronic noise refers to small, random fluctuations in the power signal caused by inherent imperfections in electronic components. Common sources include:

- **Thermal Noise**: Random movement of charge carriers due to temperature.
- **Shot Noise**: Variability in current flow due to the discrete nature of charge carriers.
- **Flicker Noise (1/f Noise)**: Long-term variations in the power signal due to imperfections in semiconductor devices.

Effect on Traces:

- Appears as **Gaussian-like deviations** superimposed on the trace, especially at low power levels.
- Unlike glitches or switching noise, electronic noise is random and continuous, making traces appear less smooth and more "fuzzy."
- It reduces the signal-to-noise ratio, complicating the identification of meaningful patterns or peaks.

Switching Noise

Switching noise arises when multiple transistors switch states simultaneously. It is caused by:

- **Simultaneous Switching Output (SSO)**: Multiple logic gates transitioning at the same time, drawing significant current spikes.
- **Power Grid Noise**: Voltage drops or fluctuations in the power supply lines due to sudden changes in current demand.

Effect on Traces:

- Results in **periodic spikes or dips** in the power trace, coinciding with bursts of simultaneous logic transitions.
- Switching noise tends to align with the clock signal but can amplify certain sections of the trace where numerous transitions occur in unison.
- It can mask finer details of the trace by introducing large, overlapping disturbances.

Hamming Distance (HD) Model for Power Trace Simulation

The **Hamming Distance (HD) model** is widely used in simulating power traces for cryptographic side-channel analysis, such as in AES. It models the power consumption based on the number of bit transitions ($0 \rightarrow 1$ or $1 \rightarrow 0$) between two successive intermediate values. This model offers several advantages over the **Hamming Weight (HW) model**, particularly when analyzing operations that depend on transitions rather than static values.

Models for Attacker

1. **Hamming Weight Model**:
    - Estimates power consumption based on the number of '1s' in a single value.
    - Simpler but less realistic for real-world scenarios where power consumption is often influenced by transitions between values.
2. **Hamming Distance Model**:
    - Estimates power consumption based on transitions between two intermediate values during encryption.
    - Reflects **dynamic power consumption**, which is the dominant factor in CMOS circuits.
    - More realistic because it models the energy required to charge and discharge capacitances during transitions.

Advantages of HD Model:

- **Transition-Based**: Captures power consumption due to bit-level changes, which is a more accurate representation of dynamic power dissipation.
- **Clock Synchronization**: Aligns power peaks and troughs with clock cycles, reflecting real-world power traces driven by clocked circuits.
- **Intermediate Values**: Allows analysis after each AES operation (e.g., SubBytes, MixColumns, etc.), providing more granular control for simulation and analysis.

Drawback of this current model is it assumes half of the flipped bits transitioned from 0 to 1 in during clock rising edge and the other half 1 to 0 during clock falling edge.

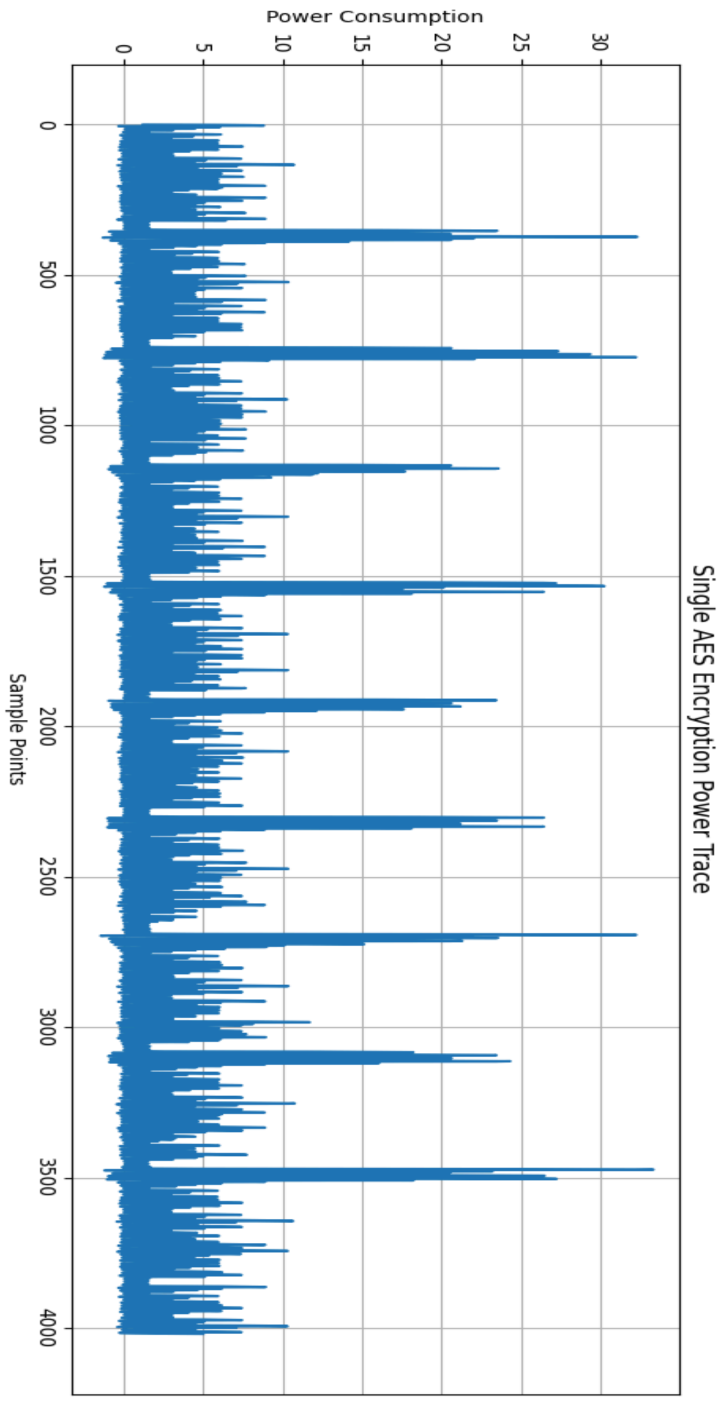
Factors Modeled in HD Traces

1. **Dynamic Power Consumption**:
    - Dominates the power profile due to the charging and discharging of capacitive loads during bit transitions.
    - The clock signal controls the timing of these transitions, creating distinct phases (charging during the rising edge, discharging during the falling edge).
2. **Glitches**:
    - Introduce sporadic power spikes due to unintended transitions.
    - Modeled as random short bursts of power at intermediate steps.
3. **Electronic Noise**:
    - Adds randomness due to inherent device imperfections.
    - Simulated using Gaussian noise to mimic real-world conditions.
4. **Switching Noise**:
    - High-frequency oscillations from simultaneous logic transitions.
    - Represented as high-frequency sinusoidal variations superimposed on the power profile.

Pseudo Code for generation of artificial traces:
(base_power = Hamming DIstance of intermediate values)

```
function generate_power_trace(base_power, num_samples):
    # Step 1: Create time points for a single clock cycle
    time_points = create_evenly_spaced_points(0, 1, num_samples)

    # Step 2: Simulate charging and discharging effects
    charging_curve = base_power * (1 - exponential_decay(time_points, rate=5))  # Charging phase
    discharging_curve = base_power * exponential_decay(reverse(time_points), rate=5)  # Discharging phase

    # Combine charging and discharging to form one full cycle
    power_cycle = combine(charging_curve, discharging_curve)

    # Step 3: Add clock signal effects (square-like behavior)
    clock_amplitude = 0.2 * base_power
    clock_signal = clock_amplitude * square_wave_signal(time_points, frequency=10)

    # Step 4: Add switching noise (high-frequency sine wave)
    switching_frequency = num_samples * 5
    switching_noise = 0.1 * base_power * sine_wave(time_points, frequency=switching_frequency)

    # Step 5: Add random glitches (sporadic power spikes)
    glitch_probability = 0.1
    glitches = generate_random_spikes(size=2 * num_samples, probability=glitch_probability, amplitude=0.05 * base_power)

    # Step 6: Add Gaussian noise for electronic imperfections
    gaussian_noise = generate_gaussian_noise(mean=0, std_deviation=noise_level, size=2 * num_samples)

    # Step 7: Combine all components to form the final power trace
    power_trace = power_cycle + clock_signal + switching_noise + glitches + gaussian_noise

    return power_trace
```

OUTPUT:


Single AES Encryption Power Trace
Power Consumption
Sample Points
0
5
10
15
20
25
30
0
500
1000
1500
2000
2500
3000
3500
4000