Topics

- Combinational logic functions
- Static complementary logic gate structures

Combinational logic expressions

- Combinational logic: function value is a combination of function arguments.
- A logic gate implements a particular logic function.
- Both specification (logic equations) and implementation (logic gate networks) are written in Boolean logic.

Gate design

- Why designing gates for logic functions is non-trivial:
  - may not have logic gates in the library for all logic expressions;
  - a logic expression may map into gates that consume a lot of area, delay, or power.
- In both cases, one deviates from standard-cell-based design: full-custom design.

Completeness

- A set of functions $f_1, f_2, \ldots$ is complete iff every Boolean function can be generated by a combination of the functions.
- NAND is a complete set; NOR is a complete set; \{AND, OR\} is not complete.
- Transmission gates are not complete.
- If your set of logic gates is not complete, you can't design arbitrary logic.
Static complementary gates

- **Complementary**: have complementary pull-up (p-type) and pull-down (n-type) networks.
- **Static**: do not rely on stored charge.
- Simple, effective, reliable; hence ubiquitous.

Static complementary gate structure

Pull-up and pull-down networks:

Inverter

NAND gate
NOR gate

AOI/OAI gates

- AOI = and/or/invert; OAI = or/and/invert.
- Implement larger functions.
- Pull-up and pull-down networks are compact: smaller area, higher speed than NAND/NOR network equivalents.

AOI example

Pull-up/pull-down network design

- Pull-up and pull-down networks are duals, i.e. series connections in one network become parallel connection and vice versa.
- To design one gate, first design one network, then compute dual to get other network.
Dual network construction

Topics
- Electrical properties of static combinational gates:
  - transfer characteristics;
  - delay;
  - power.
- Effects of parasitics on gate.
- Driving large loads.
- Switch logic (partially from Chapter 4).

Logic levels
- Solid logic 0/1 defined by $V_{SS}/V_{DD}$.
- Inner bounds of logic values $V_L/V_H$ are not directly determined by circuit properties, as in some other logic families.

Logic level matching
- Levels at output of one gate must be sufficient to drive next gate.
- $V_{OL} < V_{IL}$ and $V_{OH} > V_{IH}$!
Transfer characteristics

- Transfer curve shows static input/output relationship—hold input voltage, measure output voltage.

Logic thresholds

- Choose threshold voltages at points where slope of transfer curve = -1.
- Inverter has a high gain between \( V_{IL} \) and \( V_{IH} \) points, low gain at outer regions of transfer curve.
- Note that logic 0 and 1 regions are not equally sized.

Noise margin

- Noise margin = voltage difference between output of one gate and input of next. Noise must exceed noise margin to make second gate produce wrong output.
- In static gates, \( t=\infty \) voltages are \( V_{DD} \) and \( V_{SS} \), so noise margins are \( V_{DD}-V_{IH} \) and \( V_{IL}-V_{SS} \).
Delay

Assume ideal input (step), RC load.

Propagation delay
Rise/fall times (1)

- **Propagation delay** is measured between the 50% transition moment of input and the 50% transition of output; it is the average of *high-to-low* and *low-to-high* delays.
- **Rise time** is measured from the 10% to the 90% points in the output transition.
- **Fall time** is the reverse (from 90% to 10%).

Delay assumptions

- Assume that only one transistor is on at a time. This gives two cases:
  - rise time, pull-up on;
  - fall time, pull-up off.
- Assume resistor model for transistor. Ignores saturation region and mischaracterizes linear region, but results are acceptable.
Resistive model for transistor

- Average V/I at two voltages:
  - maximum output voltage
  - middle of linear region
- Voltage is $V_{ds}$, current is given $I_d$ at that
  drain voltage. Step input means that $V_{gs} = V_{DD}$ always.

Resistive approximation

Inverter delay circuit

- Load is resistor + capacitor, driver is
  resistor.

Inverter delay with $\tau$ model

- $\tau$ model: gate delay based on RC time
  constant $\tau$.
- $V_{out}(t) = V_{DD} \exp\{-t/(R_n+R_L)/C_L\}$
- $t_f = 2.2 (R_n+R_L)/C_L$
- For pull-up time, use pull-up resistance.
Power consumption analysis

- Almost all power consumption comes from switching behavior.
- Static power dissipation comes from leakage currents.
- Surprising result: power consumption is independent of the sizes of the pull-ups and pull-downs.

Power consumption circuit

- Input is square wave.

Power consumption

- A single cycle requires one charge and one discharge of capacitor: \( E = C_L(V_{DD} - V_{SS})^2 \).
- Clock frequency \( f = 1/t \).
- Power = \( E \cdot f = f \cdot C_L(V_{DD} - V_{SS})^2 \).
- Resistance of pull-up/pull-down drops out of energy calculation.

Driving large loads

- Sometimes, large loads must be driven:
  - off-chip;
  - long wires on-chip.
- Sizing up the driver transistors only pushes back the problem—driver now presents larger capacitance to earlier stage.
Optimal sizing

- Use a chain of inverters, each stage has transistors larger than previous stage.
- Optimal number of stages $n_{opt} = \ln(C_{big}/C_g)$.
- Driver sizes are exponentially tapered.

Switch logic

- Can implement Boolean formulas as networks of switches (think of relay networks used in pre-transistor times).
- Can build switches from MOS transistors—transmission gates.
- Transmission gates do not amplify but have smaller layouts.
Behavior of n-type switch

n-type switch has source-drain voltage drop when conducting:
– conducts logic 0 perfectly;
– introduces threshold drop into logic 1.

n-type switch driving static logic

Switch underdrives static gate, but gate restores logic levels.

n-type switch driving switch logic

Voltage drop causes next stage to be turned on weakly.

Behavior of complementary switch

Complementary switch produces full-supply voltages for both logic 0 and logic 1:
– n-type transistor conducts logic 0;
– p-type transistor conducts logic 1.
Switching logic signals

\[ ab' + a'b \]

Topics

- Pseudo-nMOS gates.
- DCVS logic.
- Domino gates.
- Wire delay.

Pseudo-nMOS

- Uses a p-type as a resistive pull-up, n-type network for pull-downs.
Characteristics

- Consumes static power.
- Has much smaller pull-up network than static gate: clear area advantage; also shorter pull-up time (just one transistor R).
- Switching speed advantage: don’t need to charge gate capacitances of pull-ups.
- Pull-down time is longer because pull-up is “fighting”.

Output voltages

- Logic 1 output is always at \( V_{DD} \).
- Logic 0 output is above \( V_{SS} \).
- \( V_{OL} = 0.25 \left( V_{DD} - V_{SS} \right) \) is one plausible choice.
- Using 0.5 \( \mu m \) parameters, 3.3V power supply:
  - \( W_p/L_p : W_n/L_n = 3.9 \).

DCVS logic

- DCVSL = differential cascode voltage switch logic.
- Static logic—consumes no dynamic power.
- Uses latch to compute output quickly.
- Requires true/complement inputs, produces true/complement outputs.

DCVS structure
**DCVS operation**

- Exactly one of true/complement pull-down networks will complete a path to the power supply.
- Pull-down network will lower output voltage, turning on other p-type, which also turns off p-type for node which is going down.

**DCVS evaluation**

- Pull-up network with its larger p-type transistors is eliminated: smaller load for previous stage.
- No static current.
- The complement of each signal is available (saves inverters).
- The complementary pull-down networks can share transistors.

---

**DCVS example with shared transistors**

Out = xor(A, B)


---

**Precharged or dynamic logic**

- Precharged logic uses stored charge to help evaluation.
- Precharge node, selectively discharge it.
- Requires multiple clock phases for evaluation.
Domino logic

- Uses precharge clock to compute output in two phases:
  - precharge;
  - evaluate.
- Is not a complete logic family—cannot invert.

Domino gate structure

- Domino phases
  - Controlled by clock $\phi$.
  - **Precharge**: p-type pull-up precharges the storage node; inverter ensures that output goes low.
  - **Evaluate**: storage node may be pulled down, so output goes up.

- Domino buffer
  - Output inverter is needed for two reasons:
    - make sure that outputs start low, go high so that domino output can be connected to another domino gate;
    - protects storage node from outside influence.
Domino effect

Gate outputs fall in sequence:

- gate 1
- gate 2
- gate 3

Using domino logic

- Can rewrite logic expression using De Morgan’s Laws:
  - \((a + b)' = a'b'\)
  - \((ab)' = a' + b'\)
- Add inverters to network inputs/outputs as required.

Charge sharing in domino gates

\[ \phi \quad V_{DD} \]
\[ a : 0 \rightarrow 1 \]
\[ = 0 \]
\[ \phi \quad V_{SS} \]

Part of the charge provided in precharge phase moves away, degrading output level.

Domino and stored charge

- Charge can be stored in source/drain connections between pull-downs.
- Stored charge can be sufficient to affect precharge node.
- Can be averted by precharging the internal pull-down network nodes along with the precharge node.
Evaluation domino logic (1)

- Area advantage: no pMOST network.
- Speed advantage:
  - gate output only connects to half the number of transistors in the next gate: less load capacitance.
  - can be designed to switch early in the voltage swing, instead of halfway.
  - even more improvement when precharging is delayed between gates (waveform propagation).

Evaluation domino logic (2)

- Is used in high-speed data paths.
- Higher speed comes at the expense of more power consumption.

Wire delay

- Wires have parasitic resistance, capacitance.
- Parasitics start to dominate in deep-submicron wires.
- Distributed RC introduces time of flight along wire into gate-to-gate delay.

RC transmission line

- Assumes that dominant capacitive coupling is to ground, inductance can be ignored.
- Elemental values are $r_i$, $c_i$. 

\[ \begin{align*}
V_{in} & \quad + \quad c_1 \quad r_1 \quad c_2 \quad r_2 \quad c_3 \quad r_3 \quad \ldots \quad c_n \quad r_n \quad V_{out} \\
- & \quad -
\end{align*} \]
Elmore delay (1)

- It is e.g. used to evaluate various choices during automatic placement and routing.
- A first-order approximation: pretend that all resistors charge all capacitances behind them sequentially.

\[ \delta_E = \sum_{\text{all resistors } i} r_i \left( \sum_{\text{all capacitances from } r_i \text{ output}} c_j \right) \]

Elmore delay (2)

- Can be computed as sum of sections:
  \[ \delta_E = \sum_{i=1}^{n} r_i (n-i)c = \frac{1}{2} rc(n^2 - n) \]
- Delay grows as square of wire length (too pessimistic).
- Main advantage: easy to compute.

Wire sizing

- Wire length is determined by layout architecture, but we can choose wire width to minimize delay.
- Wire width can vary with distance from driver to adjust the resistance which drives downstream capacitance.

Optimal wire sizing

- Wire with minimum delay has an exponential taper.
- Optimal tapering improves delay by about 8%.
Approximate tapering

Can approximate optimal tapering with a few rectangular segments.

RC trees

Generalization of RC transmission line.

RC crosstalk

- Crosswalk slows down signals---increases settling noise.
- Two nets in analysis:
  - aggressor net causes interference;
  - victim net is interfered with.

Aggressors and victims
Wire cross-section

- Victim net is surrounded by two aggressors.

Crosstalk delay vs. wire aspect ratio

- Increased spacing (S/H)

Crosstalk delay

- There is an optimum wire width for any given wire spacing---at bottom of U curve.
- Optimum width increases as spacing between wires increases.

RLC transmission lines

- Inductance of long wires cannot be neglected.
- Analysis considers a wire as an RLC ladder network rather than an RC ladder.
- Value of damping factor $\xi$ is important.
- One can calculate the optimal number and strength of buffers.