REGISTER-TRANSFER LEVEL (1)

- Register-transfer level (RTL): the most common abstraction level encountered in digital design

```
data_in                 data_out
|   coeff. ROM   | R2 | R1 | R3 | R4 |
  +-------------+----+----+----+----+
  |             |    |    |    |    |
  |             |    |    |    |    |
  +-------------+----+----+----+----+
```

REGISTER-TRANSFER LEVEL (2)

- Signals start either in a register or a primary input.
- Signals end either in a register or a primary output.
- Registers change their value at the rising edge of the clock (implemented by positive edge-triggered flipflops).
- Signals have time to propagate through combinational logic until next rising edge of clock.
- Hardware behaves as a finite state machine (FSM), where the registers hold the current state and the combinational logic computes the next state.
**FSM STRUCTURE (MEALY)**

Primary inputs → Combinational logic → Primary outputs

- Next state
- Current state
- Memory

One clock is assumed and register contents are updated every clock cycle.

**FSM STRUCTURE (MOORE)**

Primary inputs → Combinational logic → Primary outputs

- Next state
- Current state
- Memory

**EDGE-TRIGGERED FLIP-FLOP**

- The edge-triggered flip-flop is the most frequently used memory logic in sequential circuitry.
- It requires the data only to be stable within the setup and hold margins.
- One has a robust design style when an entire circuit exclusively uses positive (or negative) edge-triggered flip-flops.

**POSITIVE EDGE-TRIGGERED FLIP-FLOP**

- D
- Master latch
- Slave latch
- i₂
- Q
- clock signal
- $\phi = 0$
- $\phi = 1$
- $\phi = 0$
- $\phi = 1$

Only the input value at the rising clock transition is captured!
MASTER-SLAVE OPERATION

• $\Phi = 0$: master latch output follows input $D$; slave latch keeps its value.
• Rising edge of $\Phi$: value of $D$ now propagates to slave’s output.
• $\Phi = 1$: master latch keeps its value; slave latch is transparent and follows master.

STANDARD-CELL-BASED DESIGN (1)

• The designer is given a *library* of elementary circuits called *cells*.
• The design of library cells (from transistors) is a specialized task and is done by a dedicated group or company.
• Library information consists of:
  – cell function and simulation models (delays!)
  – cell layouts.

STANDARD-CELL-BASED DESIGN (2)

• Designing amounts to delivering a *netlist* of cells for layout.
• The netlist is generally obtained by means of *logic synthesis*.
• The netlist is mapped on a layout by means of *placement* and *routing*.

STANDARD-CELL LAYOUT (1)

• Cells use a limited number of metal layers.
• Power wires run horizontally through the cells using the same pitch; horizontal cell *abutment* can be applied.
• Other wires have *terminals* on the top and/or bottom of the cell; wiring channels realize the interconnection of terminals according to netlist.
STANDARD-CELL LAYOUT (2)

Abutment of Cells 1 & 2

STANDARD-CELL LAYOUT (3)

- Cells are organized in multiple rows with wiring channels in between.
- Routing can use metal layers on top of cells.
- An entire chip will in general consist of multiple blocks of standard cells, as well as memories, data paths, etc.
- Place-and-route software is used fit the standard cells inside the area delimited by large blocks such as memories.

STANDARD-CELL LAYOUT (4)

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%).

**PROPAGATION DELAY**  
**RISE/FALL TIMES (2)**

- \( V_{in} \)  
- \( V_{out} \)  
- \( t_{pHL} \)  
- \( t_f \)  
- \( t_{pLH} \)

**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.

**The standard-cell library has cells AND2D1, AND2D2, AND2D4:**
- What is the difference between the cells?
CASCADED DRIVER CIRCUIT

In combinational logic signals need to pass a sequence of gates, each contributing some amount of delay.

PATH DELAY

Max. path delay should be less than one clock period

CRITICAL PATH

- Critical path = path which creates longest delay
LOGIC REWRITES

- Deep logic
- Shallow logic

Shallow logic, in general, requires more area.

TIME-AREA TRADE-OFF IN LOGIC SYNTHESIS

- In general, one can make a circuit faster by using more area.
- Example: a carry-ripple adder is small but relatively slow; a carry-lookahead adder is faster, but larger.
- In logic synthesis one constrains the clock period and the synthesis tool will choose the implementation that fits the constraint.

LOGIC OPTIMIZATION

- Logic synthesis programs transform Boolean expressions into logic gate networks in a particular library.
- Optimization goals: minimize area, meet delay constraint.

TECHNOLOGY-INDEPENDENT OPTIMIZATIONS

- Works on Boolean expression equivalent.
- Estimates size based on number of literals.
- Uses factorization, resubstitution, minimization, etc. to optimize logic.
- Technology-independent phase uses simple delay models.
TECHNOLOGY-DEPENDENT OPTIMIZATIONS

- Maps Boolean expressions into a particular cell library.
- Mapping may take area and delay into account.
- May perform some optimizations in addition to simple mapping.
- Allows more accurate delay models.

In state-of-the-art design, technology-dependent optimization continues during layout (placement and routing):
- Think of buffering to compensate for parasitic delays.

CLOCK TERMINOLOGY

- Clock edge: rising or falling transition.
- Duty cycle: fraction of clock period for which clock is active (e.g., for active-low clock, fraction of time that clock is 0).

MEMORY-ELEMENT PARAMETERS

- Setup time: time before clock edge during which data input must be stable.
- Hold time: time after clock event for which data input must remain stable.

CLOCK SKEW (1)

- Clock must arrive at all memory elements in time to load data.
- The maximum difference between clock arrival times is the clock skew.
CLOCK SKEW (2)

- Clock skew values larger than the flip-flop input-output delay lead to malfunctioning: some computations will be based on the next state rather than the current state.

\[ \delta > \epsilon \]

Is this a hold-time or a setup-time violation?

What can we do about it?

CLOCK DISTRIBUTION

- Goals:
  - deliver clock to all memory elements with acceptable skew;
  - deliver clock edges with acceptable sharpness.

- Clocking network design is one of the greatest challenges in the design of a large chip.

H-TREE

CLOCK TREE

- In order to balance the delay from the clock source to the flip-flops, clock trees are used.
- They use optimal buffer sizing principles.
- In current-day practice clock trees are generated during layout as wiring delay is significant.