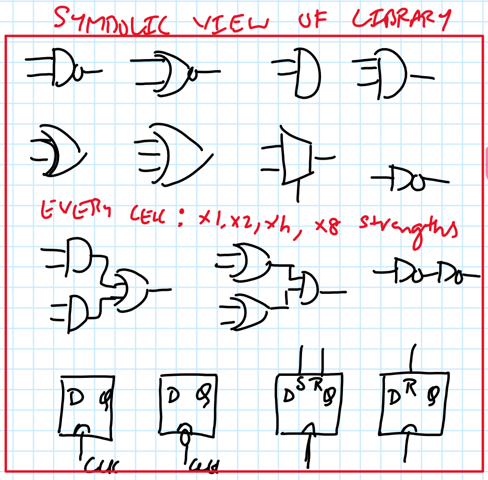
# Lecture-13

Recall that the standard cell library has



LIBRARY VIEWS

* Circuit netlist (Spice netlist .spice files)
* Functional description (.V /.VHD files)
* Layout (Physical layers) (.gdsII or .magic etc)

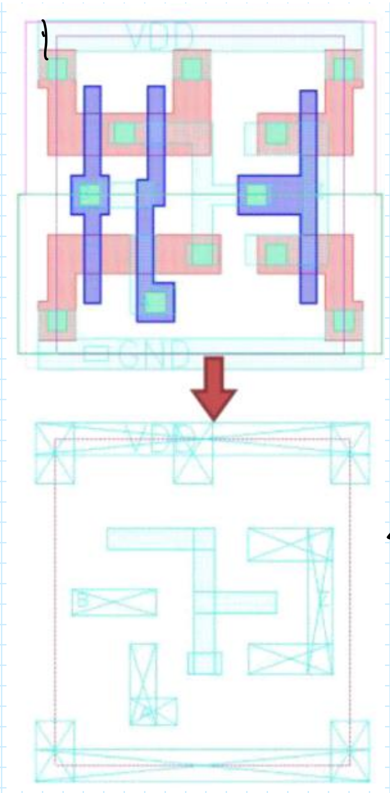
EVERY CELL IN THE LIBRARY WILL HAVE ALL VIEWS

— x — x —

* Recall that after synthesis and physical design, we get the layout and spice netlist from the tool. However spice simulation of such large circuits are not practical (in terms of simulation time)
    - ↳ We spice simulation required for analog simulations (continuous time & continuous amplitude)
    - ↳ But for digital simulation (discrete time & discrete amplitude), we don't need the accuracy & precision of analog simulations.
* Also, when the tool is stitching the top level circuit / layout, the tool needs the following informations
    1) delay of each cell (in a text file)
    2) Power of each cell (in a text file)
    3) Routing (metal traces) information wrt physical coordinates (in a text file)
* 1) & 2) are provided in a standard "LIBERTY FILE" or (.lib) (LIB file)
    3) is provided in a standard "LIBRARY EXCHANGE FORMAT" OR (.lef) (LEF file)

— x —

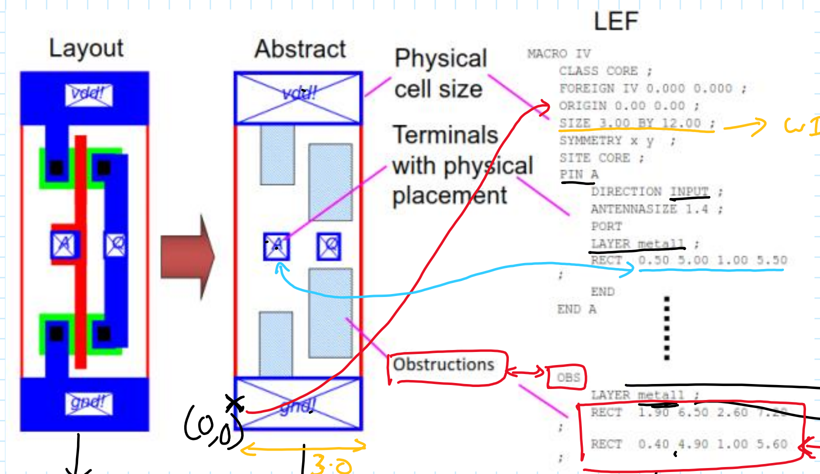
LEF: A LEF file abstracts only the relevant physical information from the layout to be used by the digital tools.



← Actual layout

← Information abstracted from layout for routing purposes.
(only metal info is required for routing !!)

Invertor Example:-



```
MACRO IV
    CLASS CORE ;
    FOREIGN IV 0.000 0.000 ;
    ORIGIN 0.00 0.00 ;
    SIZE 3.00 BY 12.00 ;
    SYMMETRY x y  ;
    SITE CORE ;
    PIN A
        DIRECTION INPUT ;
        ANTENNASIZE 1.4 ;
        PORT
        LAYER metal1 ;
        RECT  0.50 5.00 1.00 5.50
    ;
        END
    END A
          :
    OBS
        LAYER metal1 ;
        RECT  1.90 6.50 2.60 7.20
    ;
        RECT  0.40 4.90 1.00 5.60
    ;
```

→ WIDTH × CELL HEIGHT

(0,0)

3.0

The tool cannot route metal1 on these cordinates at the top level to avoid shorts!!

Actual layout

Abstract information for visualization.

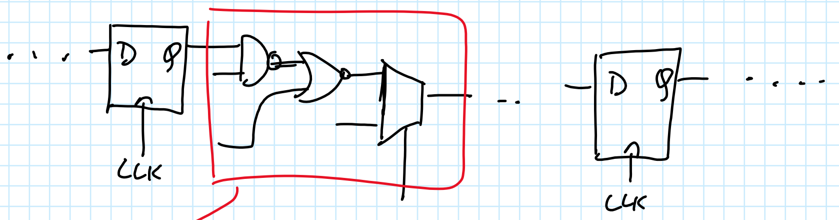
LEF file

∴ LEF file tells the tool (a) what coordinates every cell has a physical i/p and o/p pin to create routing (wiring)
(b) what paths to avoid while routing (wiring) (Obstructions)

— × — ×

LIB FILE

* Consider that based on the HDL file that you have written, the synthesis tool builds the following network of gates

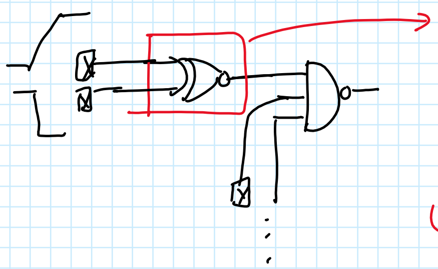


* A design has a (a) timing constraint (max $f_{CLK}$) and/or power constraint.
i.e. logic delays must be optimized.
i.e. can pick a few X4, X8 strength gates for max $f_{CLK}$ (but power↑)

⇒ Depending on constraints, gates have to be chosen from the library by the tool ⇒ The tool must be already aware of delays of the gate!!!

∴ Tool will read .lib file and finds delay of each cell.

* Consider the N/W below,



To decide the strength of this cell, the tool needs two informations (i.e $t_p$ of cell)
i) Rise/fall time @ input of this cell
(ii) output capacitance this cell has to drive.

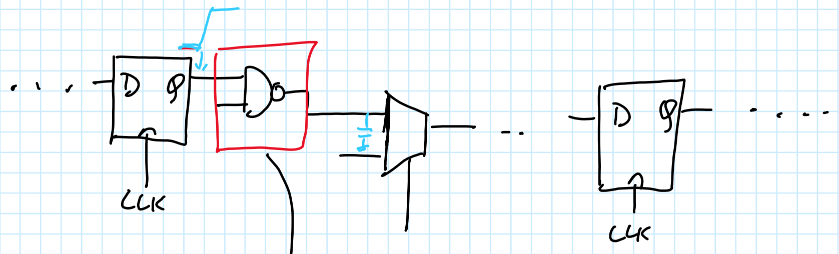
→ (i) is the input specified by designer in this case & known.
(ii) is the 3 i/p NAND gate input capacitance and tool does not have this information ⇒ .LIB must have i/p cap of all pins of all cells.

* Also, the tool must know the delay of the gate for all possible combinations of input rise/falls (also called slew) and output capacitors. This is usually created in the form of a look-up table.

| Inverter prop. delay $t_p$ | | Cap load (pF) | | | |
|---|---|---|---|---|---|
| | | 0.1 | 0.25 | 0.5 | 1 |
| o/p Slew (ps) | 10 | $t_{p1}$ | $t_{p2}$ | $t_{p3}$ | |
| | 100 | | . | – | |
| | 500 | : | | | |
| | 1000 | ( | | | $t_{p16}$ |

Example:-

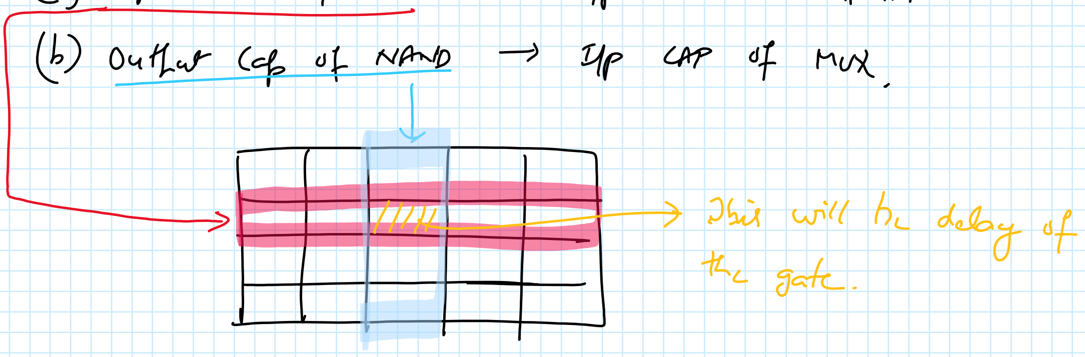
Example :-



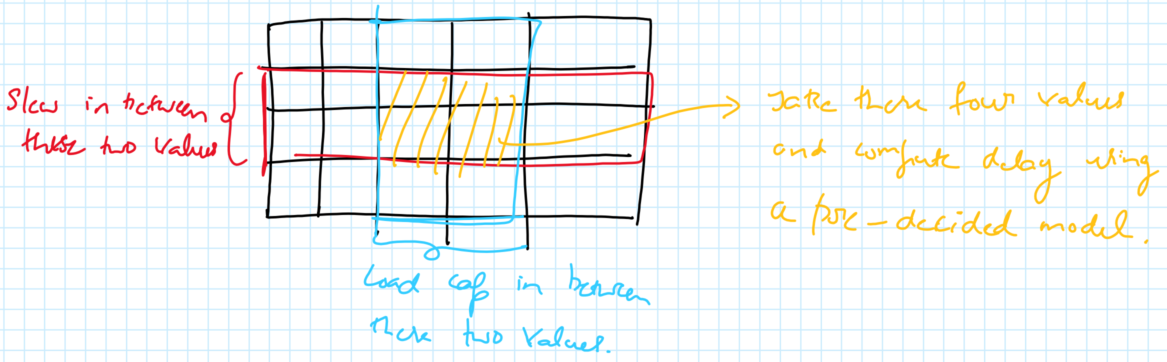
To find delay of gate, the tool goes to the .lib file and maps

(a) Input slew of NAND → o/p RISE time of DFF

(b) Output Cap of NAND → I/p CAP of MUX.



* If the slew and cap do not exactly lie on the table, the tool uses a non-linear delay model (NLDM) to compute the delay.



Slew in between these two values
Take these four values and compute delay using a pre-decided model.
Load cap in between these two values.

∴ .lib file has (a) Input pin capacitance

(b) Timing tables LUT (i/p slew vs o/p cap) for

4 LUTs
- (i) Cell_rise (i.e propagation delay when o/p rises)
- (ii) cell_fall ( " " " falls)
- (iii) rise transition, $t_r$ @ o/p
- (iv) fall_transition, $t_f$ @ o/p

(c) Static/Leakage power (after o/p has settled)

(d) Dynamic power tables LUT (i/p slew vs o/p cap) for

2 LUTs
- (i) rise_power (i.e dynamic power when o/p rises)
- (ii)

[illegible]

2 LUTs $\left\{ \begin{array}{l} \text{(i) rise\_power (i.e dynamic power when o/p rises)} \\ \text{(ii) fall\_power (i.e " " " falls).} \end{array} \right.$

These power are dominated by short circuit power!!

For combinational logic, we have additional info in the .lib file

(e) Setup timing tables (LUT D slew vs CLK slew)

2 LUTs $\left\{ \begin{array}{l} \text{(i) Fall\_constraint (i.e } \uparrow \text{ \& Q falls)} \\ \text{(ii) Rise\_constraint (i.e } \uparrow \text{ \& Q rises)} \end{array} \right.$

(f) Hold timing tables (LUT D slew vs CLK slew)

2 LUTs $\left\{ \begin{array}{l} \text{(i) Fall\_constraint (i.e } \uparrow \text{ \& Q falls)} \\ \text{(ii) Rise\_constraint (i.e } \uparrow \text{ \& Q rises)} \end{array} \right.$