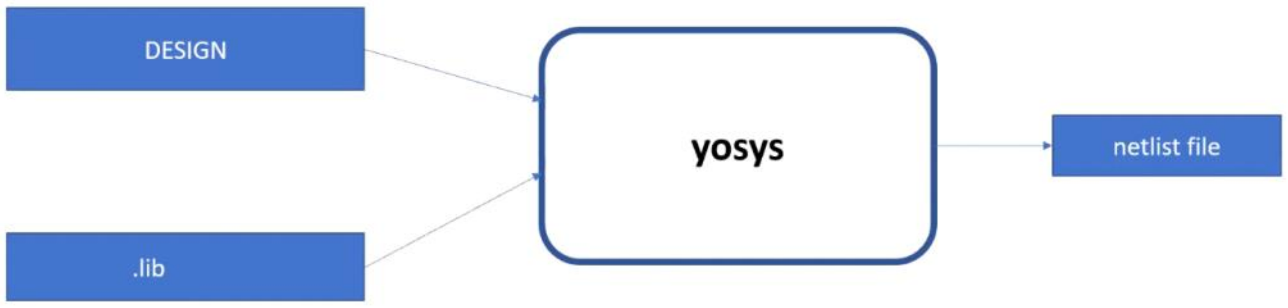
Introduction to Yosys

Monday, May 12, 2025 12:41 PM

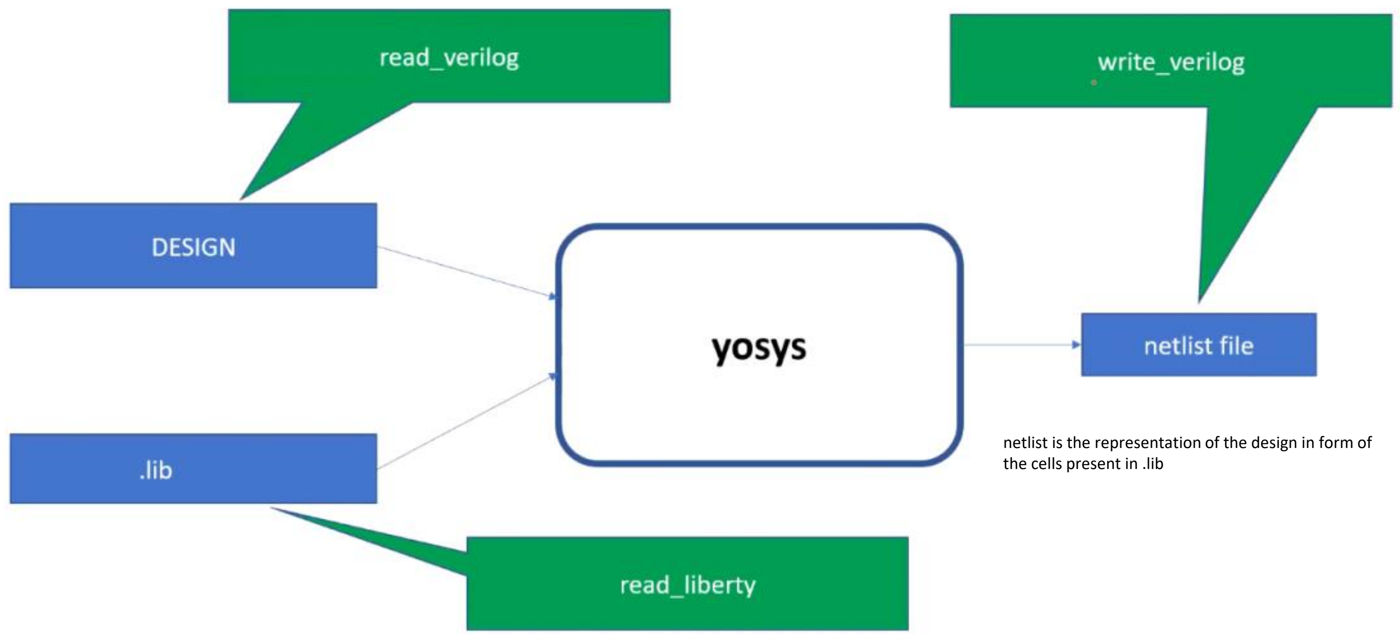
Introduction to yosys

Synthesizer

- Tool used for converting the RTL to netlist
- **Yosys** is the synthesizer used in this course

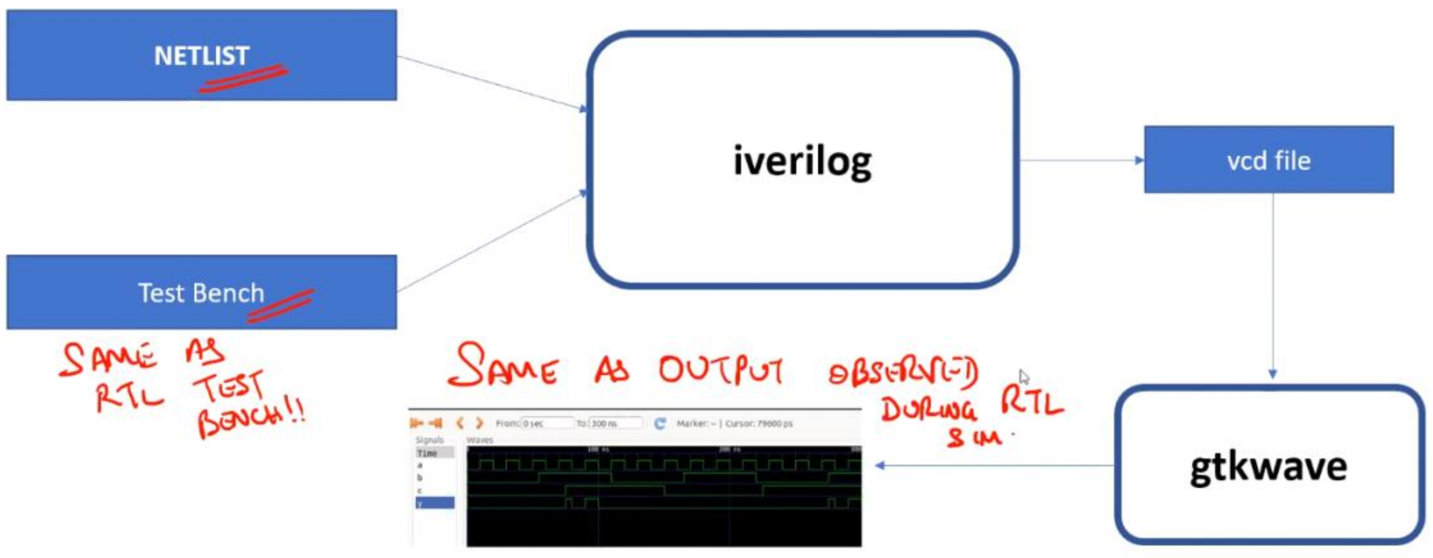


Yosys setup



netlist is the representation of the design in form of the cells present in .lib

Verify the synthesis



NOTE

The set of Primary inputs / primary outputs will remain same between the RTL design and Synthesized netlist → Same Test bench can be used !!

Logic Synthesis

RTL Design

- RTL Design
  - Behavioral representation of the required specification

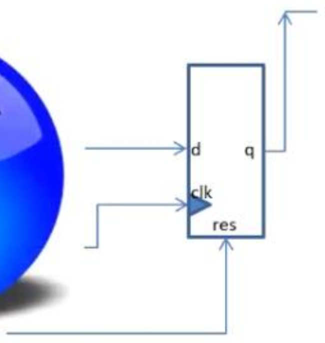
```
module sample_code (
input clk,rst ,
output result,done);

always @ (posedge clk ,posedge rst)
  if(rst)
    ----
   else
    ---
endmodule
-------------------------------------------
-------------------------------------------
```

VERILOG..
HDL

```
module sample_code (
input clk,rst ,
output result,done);

always @ (posedge clk ,posedge rst)
  if(rst)
   ----
  else
   ---
endmodule
-------------------------------------------
-------------------------------------------
```

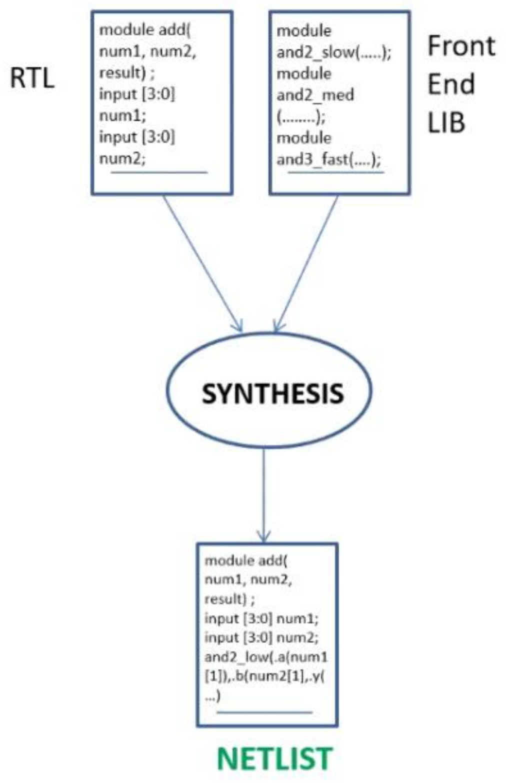


Digital Logic Circuit

RTL CODE

Synthesis

- RTL to Gate level translation
- The design is converted into gates and the connections are made between the gates
- This is given out as a file called netlist



```
module add(
num1, num2,
result) ;
input [3:0]
num1;
input [3:0]
num2;
```

```
module
and2_slow(.....);
module
and2_med
(........);
module
and3_fast(....);
```

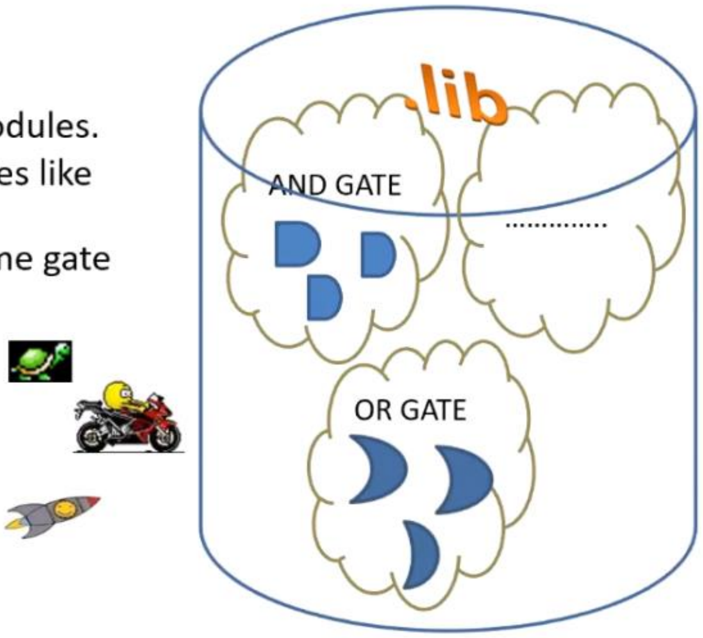
```
module add(
num1, num2,
result) ;
input [3:0] num1;
input [3:0] num2;
and2_low(.a(num1
[1]),.b(num2[1],.y(
...)
```

What is .lib

- .lib
  - Collection of logical modules.
  - Includes basic logic gates like And, Or , Not, etc…
  - Different flavors of same gate
    - 2 input And gate
      - Slow
      - Medium
      - Fast
    - 3 input And gate
      - Slow
      - Medium
      - Fast
    - 4 input And gate

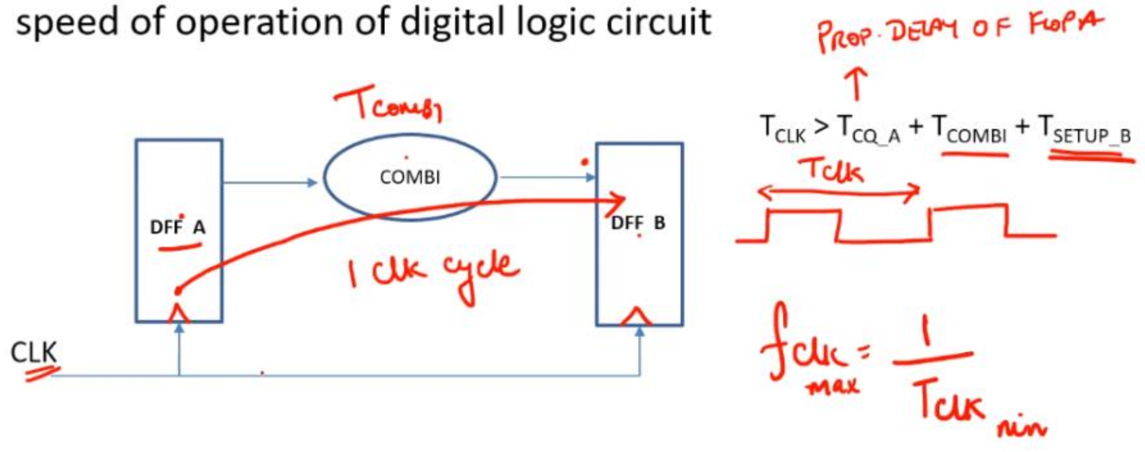
……………..

……………..



Why different flavours of gate

- Combinational delay in logic path determines the maximum speed of operation of digital logic circuit



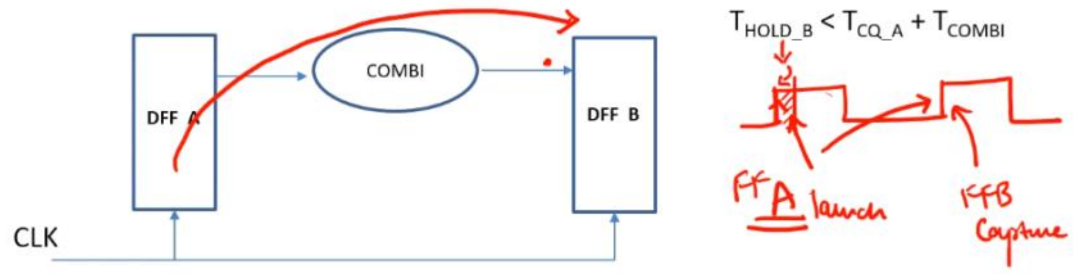
$$T_{CLK} > T_{CQ\_A} + T_{COMBI} + T_{SETUP\_B}$$

$$f_{clk_{max}} = \frac{1}{T_{clk_{min}}}$$

- So we need cells that work fast to make T_{COMBI} small
- Are faster cells sufficient ?

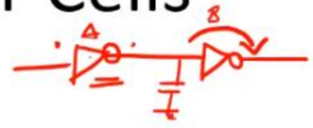
Tclk >= tpd1 + tcomb + tsu2
th2 <= tpd1 + tcomb

Why we need slow cells ?



- To ensure that there are no "HOLD" issues at DFF_B , we need cells that work slowly ☺ !!!
- Hence we need cells that work fast to meet the required performance and we need cells that work slow to meet HOLD
- The collection forms the lib

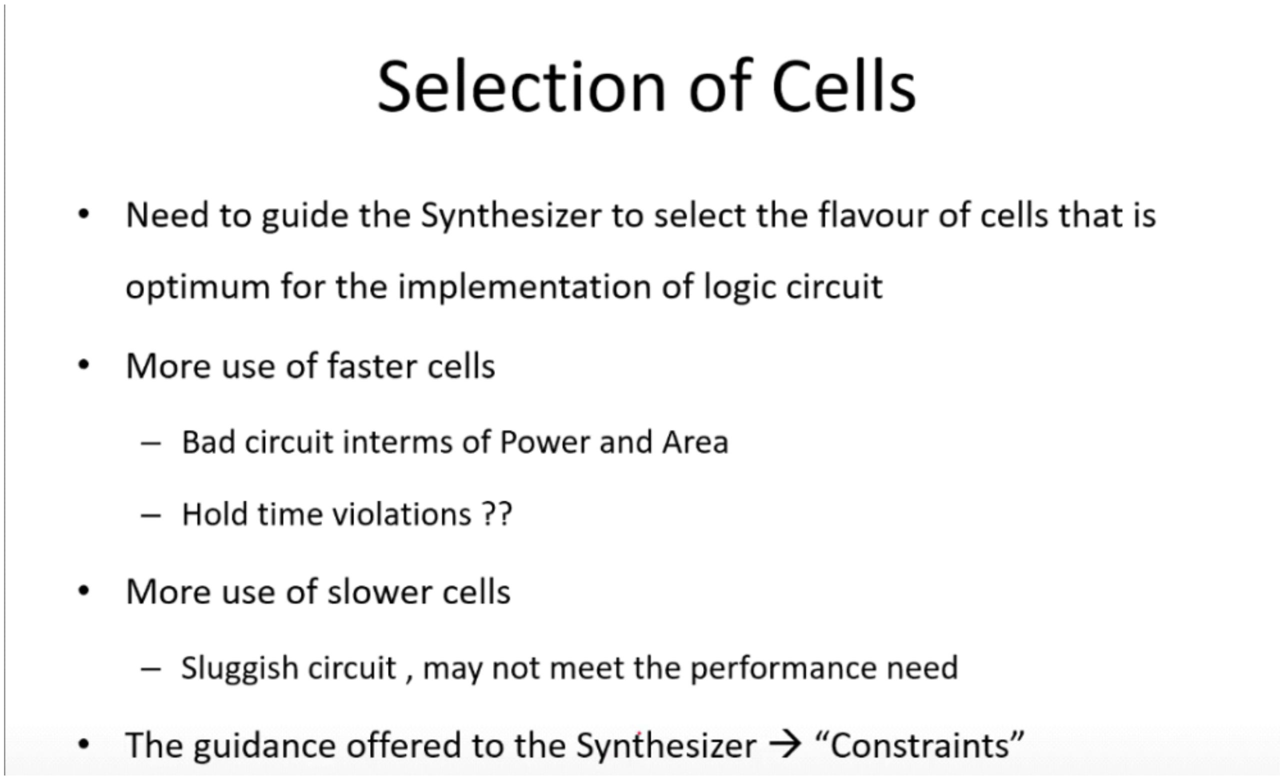
Faster Cells vs Slower Cells



- Load in Digital Logic circuit → Capacitance
- Faster the charging / discharging of capacitance → Lesser the cell delay

$\beta = \mu C_{ox} \frac{W}{L}$ $\quad I_{ds} = \frac{\beta}{2}\left[(V_{gs} - V_{th})^2\right]$ $\quad W\uparrow, \beta\uparrow, I_{ds}\uparrow, \text{delay}\downarrow$

  - To charge / discharge the capacitance fast , we need transistors capable of sourcing more current ⟶ WIDE TRANSISTORS
  - Wider transistors -> Low Delay -> More Area and Power as well !!
  - Narrow transistors -> More Delay -> Less Area and Power
  - Faster cells donot come free , they come at penalty of area and power

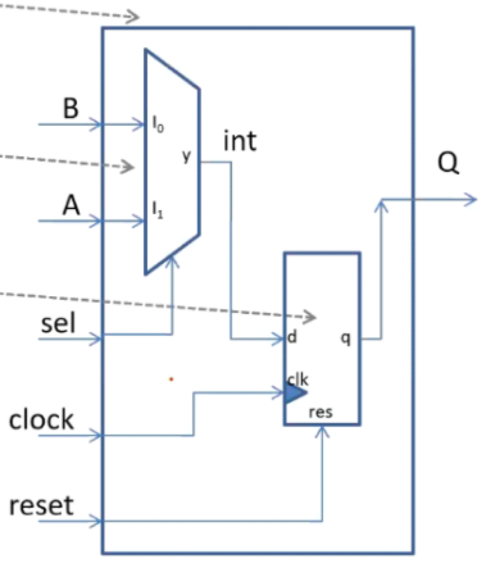


Selection of Cells

- Need to guide the Synthesizer to select the flavour of cells that is optimum for the implementation of logic circuit
- More use of faster cells
  - Bad circuit interms of Power and Area
  - Hold time violations ??
- More use of slower cells
  - Sluggish circuit , may not meet the performance need
- The guidance offered to the Synthesizer → "Constraints"

Synthesis (Illustration)

```
module (A,B,sel,clock, reset,Q)
input A,B,sel,clock,reset;
output Q;
wire  int;
assign int = sel ? A :B;
always @(posedge clock or posedge reset)
     begin
       if (reset)
       begin
            Q <= 1'b0;
       end
       else if (clk)
       begin
            Q <= int;
       end
     end
endmodule
```



The circuit on the right is created from RTL using the gates available in the .Lib and given out as Netlist.