Design Representations and CAD Tools

(4541.554 Introduction to Computer-Aided Design)

School of EECS

Seoul National University

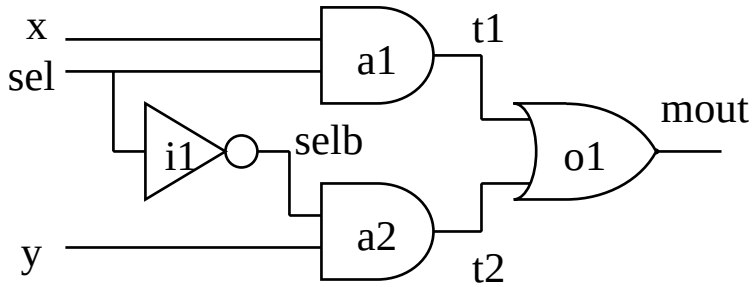
Design Representation

- **We focus on HW designs.**
- **Behavioral Representation**
  - **Behavior of outputs in terms of inputs and time**
  - **Independent of implementation**
  - **Chart, state diagram, state table, HDL(Hardware Description Language; VHDL, Verilog)**
  - **Example:**

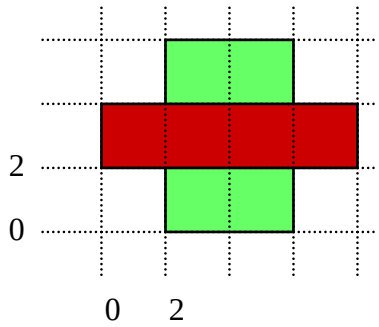
```
if sel = '1' then
   mout <= x;
else
   mout <= y;
end if;
```

- **Structural Representation**
  - **Components and their connections**
  - **Represents implementation**
  - **Schematic diagram, HDL, netlist language (EDIF)**
  - **Example:**

```
i1: inv port map (selb, sel);
a1: and2 port map (t1, x, sel);
a2: and2 port map (t2, y, selb);
o1: or2 port map (mout, t1, t2);
```



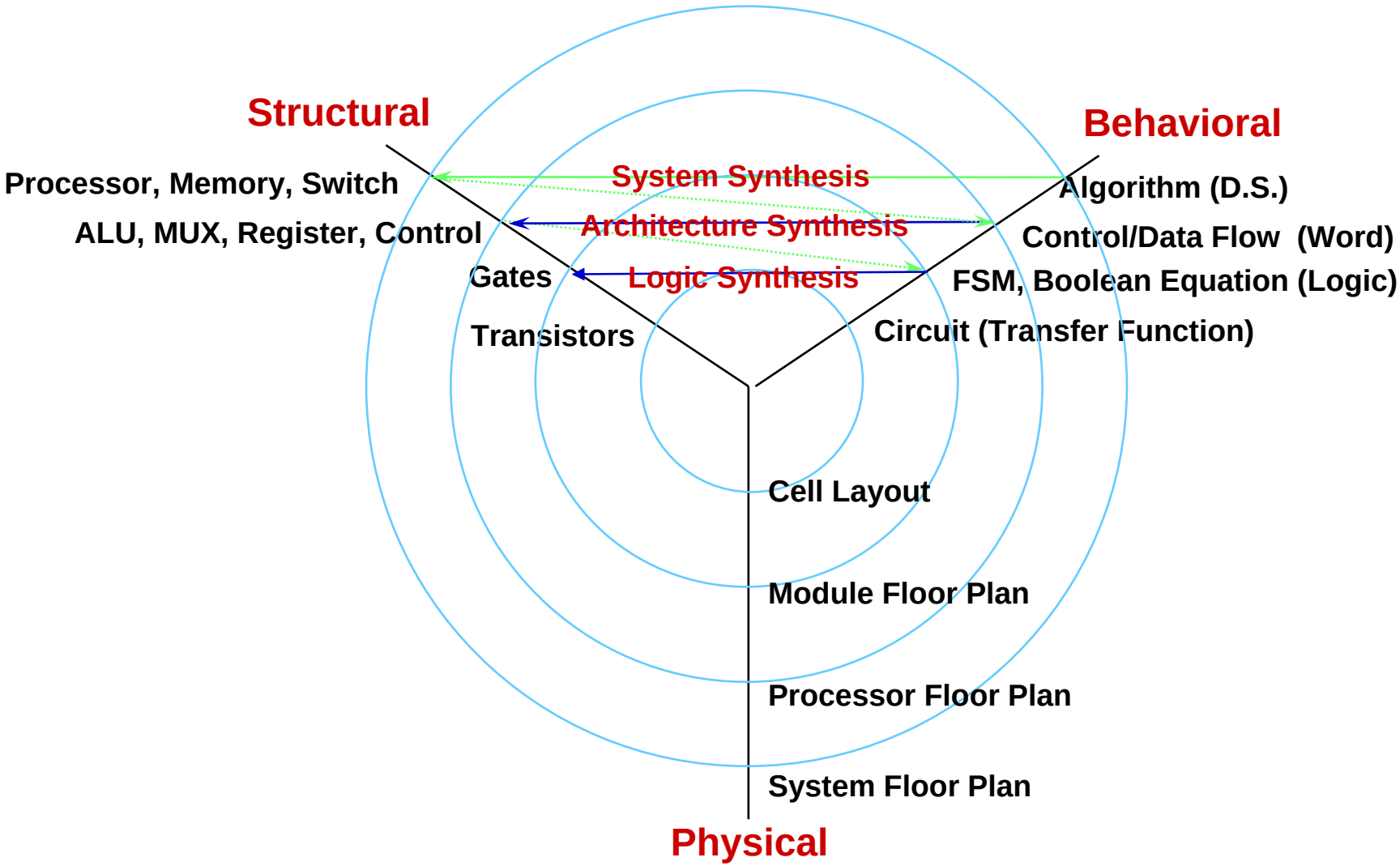
- **Physical Representation**
  - **Geometrical representation**
  - **Graphical image, layout language (CIF, GDSII)**
  - **Example: CIF**



L ND;
B 4 6 4 3;
L NP;
B 8 2 4 3;

l_x l_y x y

- **Y-Chart**

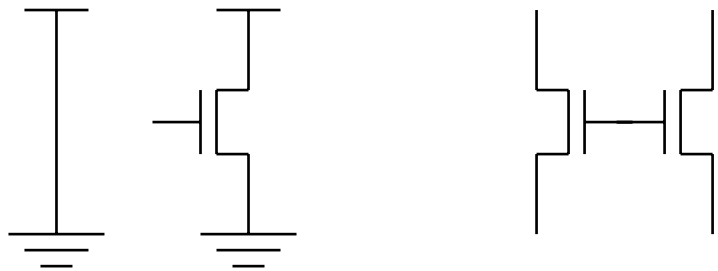

Structural
Behavioral
Processor, Memory, Switch
ALU, MUX, Register, Control
Gates
Transistors
System Synthesis
Architecture Synthesis
Logic Synthesis
Algorithm (D.S.)
Control/Data Flow (Word)
FSM, Boolean Equation (Logic)
Circuit (Transfer Function)
Cell Layout
Module Floor Plan
Processor Floor Plan
System Floor Plan
Physical

CAD Tools

- **Taxonomy of CAD Tools for HW design**
  - **Synthesis tools**
    - **Transform from a higher-level description to a lower-level description satisfying functionality and constraints**
  - **Estimation tools**
    - **Estimate performance, area cost, power consumption, etc.**
    - **Used for manual design space exploration or integrated into synthesis tools**
  - **Verification tools**
    - **Check to see if the functionality and constraints are satisfied**
    - **Used to check synthesis results**
  - **Design management tools**
    - **Design version control, data dependency management, tool integration, format conversion, etc.**

- **Synthesis Tools**
  - **Silicon compiler**
    - **Goal is to generate a mask layout from behavioral specification in HDL as a single-step process**
    - **Complexity is too high**
  - **Whole process is divided into multiple steps.**
  - **Each step consists of synthesis and verification**
  - **There can be debug cycles within each step or across multiple steps.**
  - **From the viewpoint of systems design**
    - **System synthesis (HW-SW co-synthesis)**
      - **Partitioning**
      - **Software synthesis**
      - **Hardware synthesis**
      - **Interface Synthesis**
  - **Looking at hardware design only**
    - **Architecture synthesis**
    - **Logic synthesis**
    - **Layout synthesis**

- **Verification Tools**
  - **Physical design verification**
    - **Used mainly for custom design**
    - **Less important for ASIC design due to correct-by-construction concept**
    - **DRC (Design Rule Checking)**
      - **Dimensions of each pattern, distance between patterns, enclosure, extension**
    - **ERC (Electrical Rule Checking)**
      - **Short circuit, floating input, tied output, fan-out**



- **Circuit extraction**
  - **Connectivity extraction**
    - **Extract derived layer**

      tran: D & P & (! B)

      dwire: D & (! tran)

      PDcut: P & D & B
    - **Merge signals connected together**
    - **Connect transistors to signals**
  - **Parameter extraction**
    - **Parasitic capacitance**
    - **Parasitic resistance**
    - **Transistor size: W/L**
- **Connectivity verification**
  - **LVS (Layout Versus Schematic)**

- **Simulation**
  - **Most popular method for verification**
  - **In general, exhaustive test is impossible.**
  - **Simulation at every levels of synthesis process**
  - **Depending on the levels,**
    - **Circuit**
    - **Logic (switch, gate)**
    - **Functional**
    - **Behavioral**
    - **Mixed-level**
    - **Mixed-signal**
  - **Depending on simulation algorithms**
    - **Event-driven simulation**
    - **Oblivious simulation**
    - **Compiled-code simulation**
    - **Cycle-based simulation**

- **Formal verification**
  - **Complete verification is possible.**
  - **Prove logically that implementation (synthesis result) satisfies or is equivalent to specification (synthesis input)**
  - **Complexity is high and it still takes too much time in some case.**
  - **The user must represent the design in a formal way, which can be a difficult task**
  - **Examples:**
    - **Model checking**
    - **Equivalence checking**

- **Timing verification**
  - **Compute delay along critical paths**
  - **Need to find critical paths excluding false paths, which is a very difficult task**
  - **Can be used in combination with cycle-based simulation**