`xcell`: a cell library characterizer for combinational and state-holding gates

Rajit Manohar
Computer Systems Lab, Yale University
`https://csl.yale.edu/`
`https://avlsi.csl.yale.edu/`

Tool: `https://github.com/asyncvlsi/xcell`

Yale

AVLSI

Liberty file for cell library (Synopsys "`.lib`" file)

- Used by numerous tools in an EDA flow
  - ❖ Timing analysis
  - ❖ Power analysis
  - ❖ Logic synthesis
  - ❖ … and any tool that uses timing/power analysis!
    - ▸ Placer, router, …

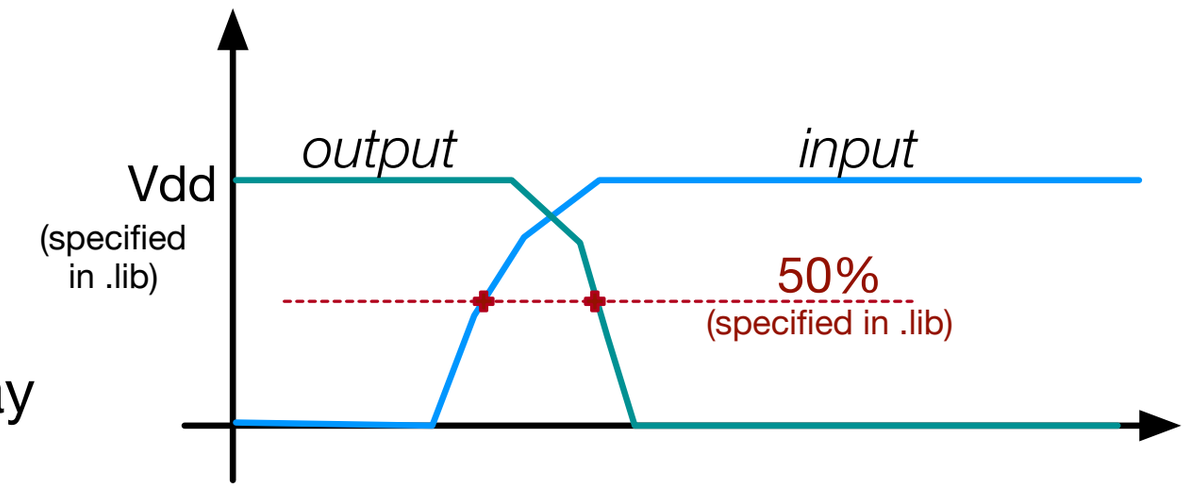
- Usually created by the standard cell library provider
  - ❖ Companion to a cell library
  - ❖ Commercial tools exist to generate .lib files from the physical implementation of a cell library
    E.g. Synopsys SiliconSmart

What is in a .lib file?

- Operating conditions
  - ❖ Supply voltage, temperature, etc.
- Cell definitions
  - ❖ Name, area, power/ground pins
  - ❖ Leakage power
  - ❖ I/O pins for the cell
    - Pin type (input/output/bi-directional)
    - Input capacitance
    - Logic equation (“function”) for output pins
    - Delay arcs
      - “when input pin X rises, output falls after delay D”
    - Internal power

Example: delay calculation, non-linear delay model

- Setup test circuit
  - ❖ SPICE netlist for the gate
  - ❖ Output load capacitance
  - ❖ Input transition time (slew rate)
  - ❖ Run SPICE simulation, extract delay
- Sweep
  - ❖ Electrical scenarios
    - ▸ Output load capacitance
    - ▸ Input transition time
  - ❖ Logical scenarios: values of *other* input pins, input change causing output change
- Example: NAND2 (A,B), 7 transition time options, 10 load capacitance options
  - ❖ 70 (electrical) x 4 (logical) = 280 SPICE scenarios



State-holding gates

- Non-standard state-holding gates for asynchronous logic
  - Example: C-element, inputs A, B; output Y
    - Y = 1 if A = 1 & B = 1
    - Y = 0 if A = 0 & B = 0
    - Y is state-holding otherwise
  - Logic equation in .lib: **Y** = A*B | **Y** * !(!A*!B)
    - Note: synchronous tools will view this as an error!
- Characterization of general state-holding gates
  - Compute a trajectory through the state-space (see paper); up to 4 steps
    - Set output to a well-defined value (Step 1)
    - Take the gate to a state-holding state (Step 2)
    - Take the gate to a state-holding state right before output changes (Step 3)
    - Switch input (Step 4)

Using xcell

- Cell descriptions: two options
  - ❖ Use ACT to specify gates
    - ▸ Gate sizing can be specified
    - ▸ ACT auto-generates SPICE netlist

```
defcell NAND2X1 (bool? A, B; bool! Y)
{
  prs { A & B => Y- }
  sizing { Y{-1} }
}
```

  - ❖ Specify SPICE netlist path in config file; use ACT “external module”

```
defcell NAND2X1 (bool? A, B; bool! Y);
```

- Operating mode: specified in config file
- Calibration points: specified in config file
- SPICE simulator to use: specified in config file
  - ❖ Currently supports HSPICE and Xyce
- List of cells to characterize: using a simple ACT file that instantiates all the cells to be characterized (examples in repo)

Yale

AVLSI

Summary

- `xcell` can generate .lib files for
  - ❖ Combinational gates
  - ❖ State-holding gates required for asynchronous logic

- `xcell` generated .lib files have been used for a few years
  - ❖ For asynchronous static timing and power analysis
    - ‣ Uniform format used for all gates (state-holding + combinational)
      - ❖ As inputs to `abc/yosys` for technology mapping
        … for multiple technology nodes (130nm, 65nm, 28nm)

- Future work
  - ❖ Support special .lib formats for state-holding gates used in synchronous logic
  - ❖ More SPICE simulations! (variation modeling, better delay models, …)

Yale

AVLSI