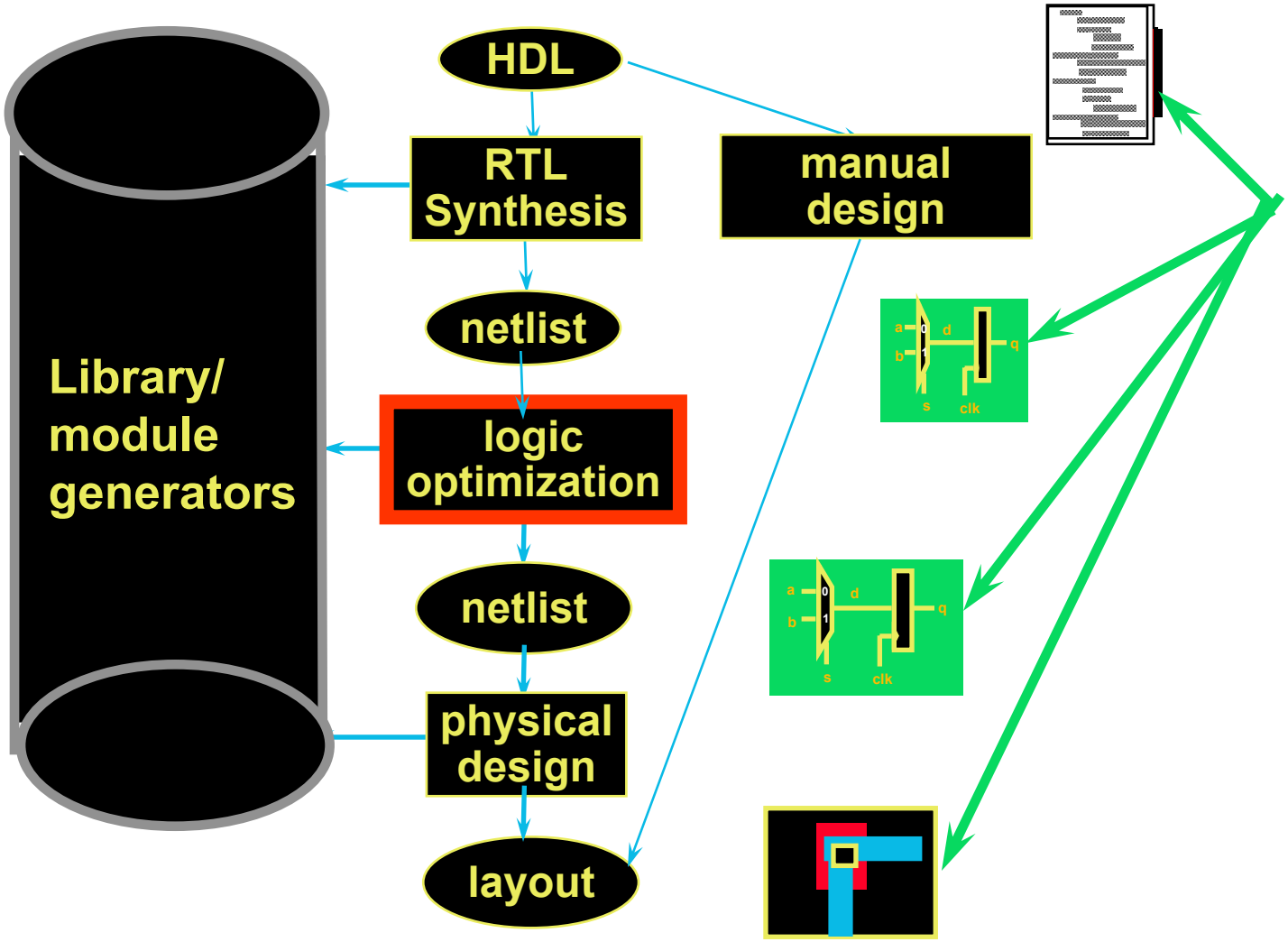
VLSI CAD Flow: Logic Synthesis,

6.375 Lecture 13

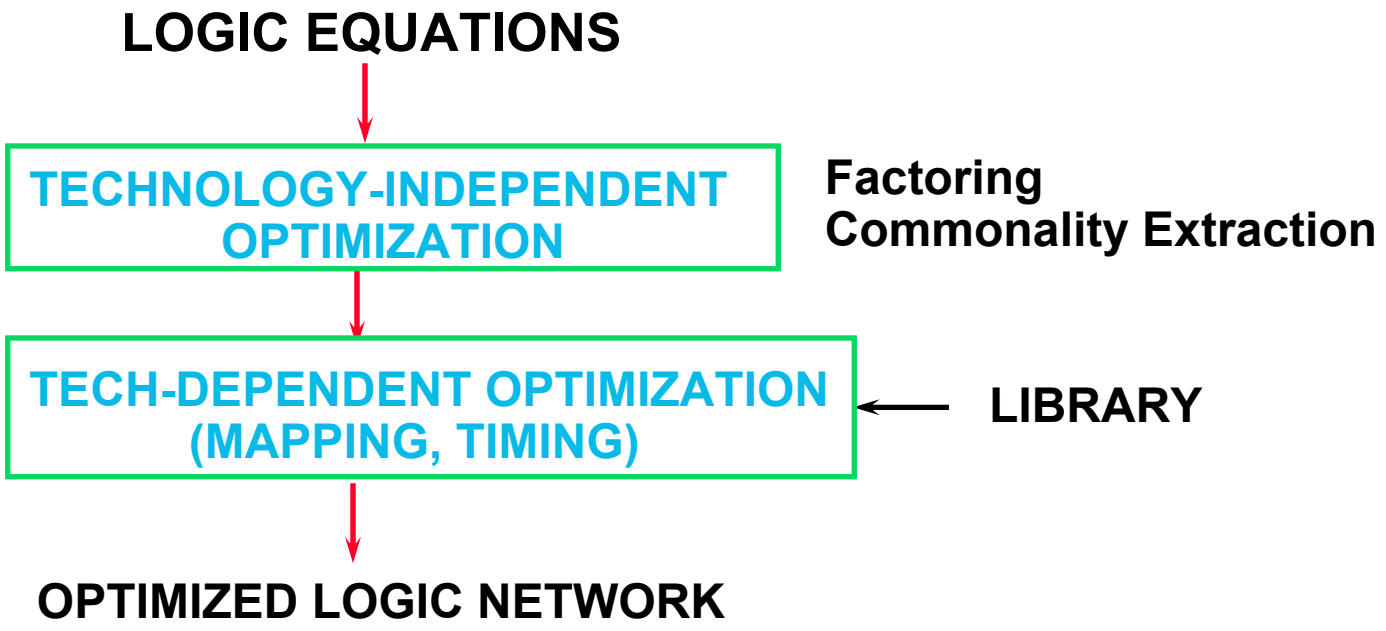
by Ajay Joshi

(Slides by S. Devadas)

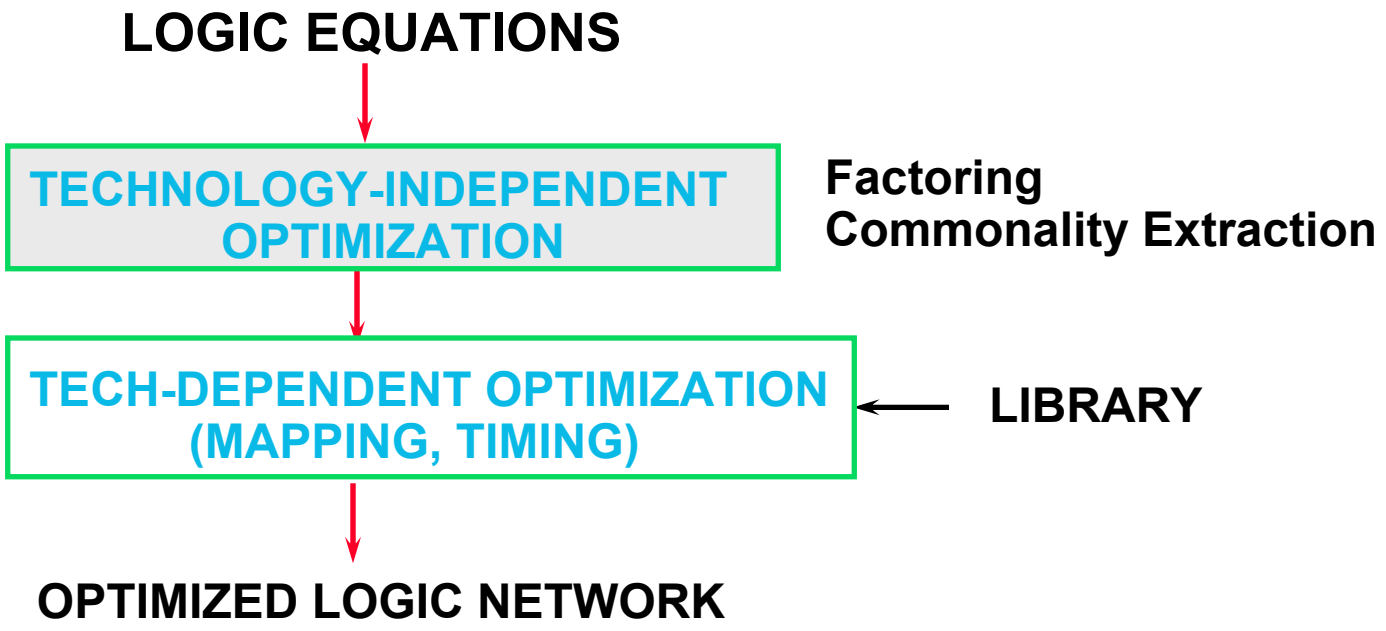
RTL Design Flow



Logic optimization flow



Logic optimization flow



Why logic optimization?

Transistor count redution	→	**AREA**
Circuit count redution	→	**POWER**
Gate count (fanout) reduction	→	**DELAY (Speed)**

Area reduction, power reduction and delay reduction improves design

Boolean Optimizations

Involves:

- **Finding common subexpressions.**
- **Substituting one expression into another.**
- **Factoring single functions.**

$$F = \begin{cases} f_1 = AB + AC + AD + AE + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\,\overline{E} \\ f_2 = \overline{A}B + \overline{A}C + \overline{A}D + \overline{A}F + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\,\overline{F} \end{cases}$$

Find common expressions

$$F = \begin{cases} f_1 = A(B + C + D + E) + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\,\overline{E} \\ f_2 = \overline{A}(B + C + D + F) + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}\,\overline{F} \end{cases}$$

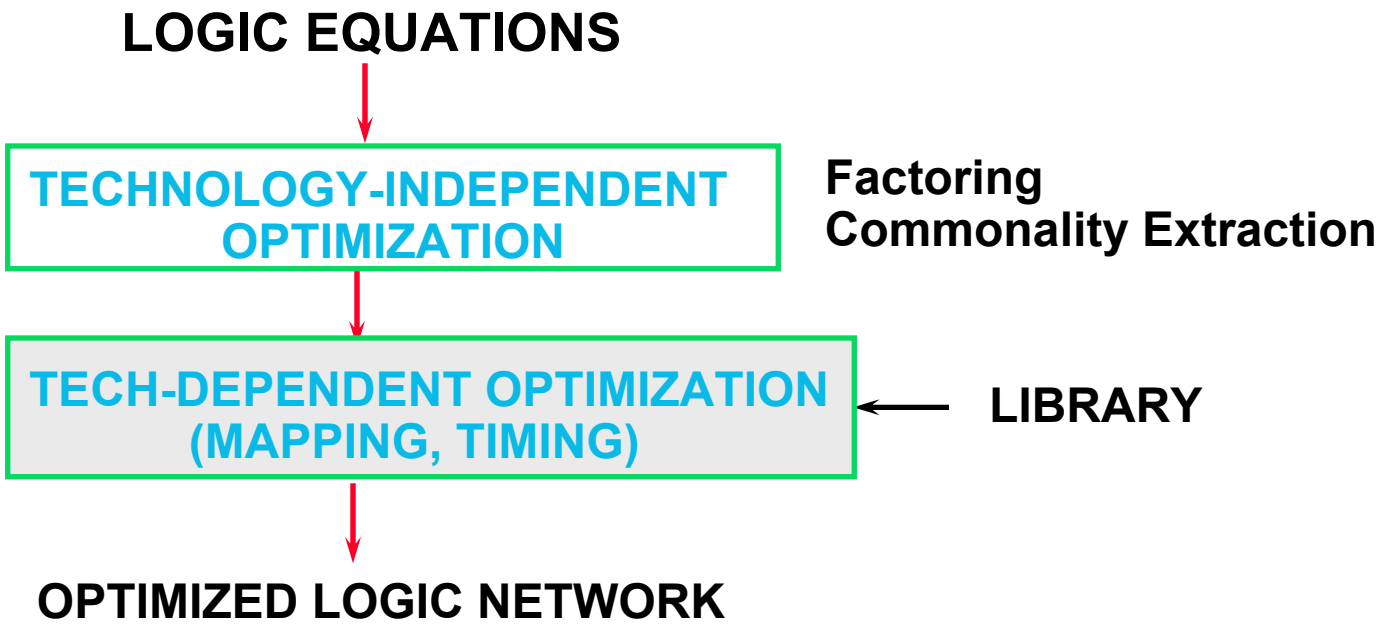
Extract and substitute common expression

$$G = \begin{cases} g_1 = B + C + D \\ f_1 = A(g_1 + E) + \overline{A}\,\overline{E}\,\overline{g_1} \\ f_2 = \overline{A}(g_1 + F) + \overline{A}\,\overline{F}\,\overline{g_1} \end{cases}$$

Algebraic Optimizations

- Algebraic techniques view equations as polynomials
- Rules of polynomial algebra are used
- For e.g. in algebraic substitution (or division) if a function $f = f(a, b, c)$ is divided by $g = g(a, b)$, a and b will not appear in f / g
- Boolean algebra rules are not applied

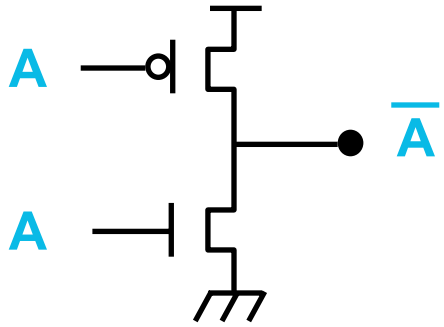
Logic optimization flow

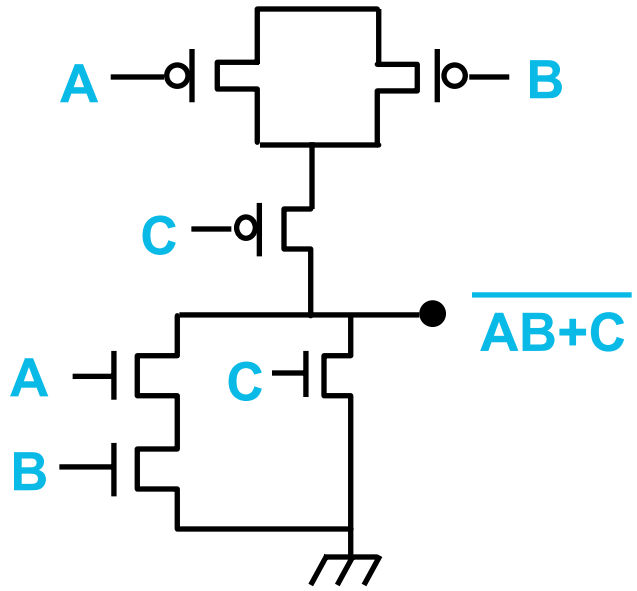


“Closed Book” Technologies

A standard cell technology or library is typically restricted to a few tens of gates e.g., MSU library: 31 cells

Gates may be NAND, NOR, NOT, AOIs.

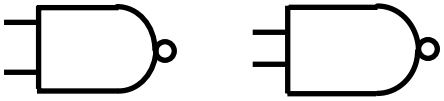
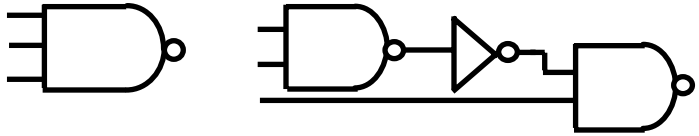
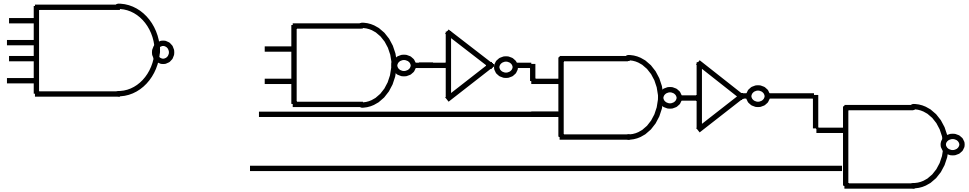
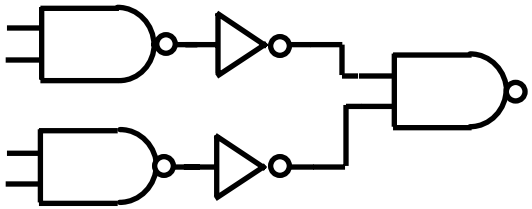




Standard cell library

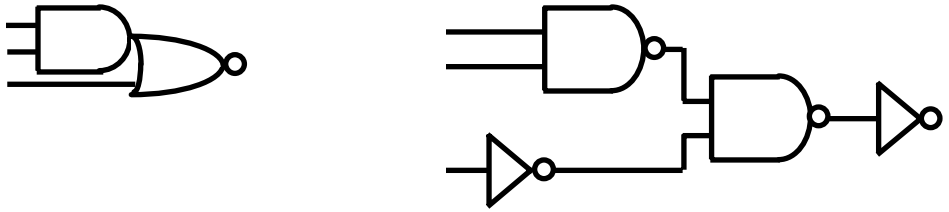
- For each cell
  - Functional information
  - Timing information
    - Input slew
    - Intrinsic delay
    - Output capacitance
  - Physical footprint
  - Power characteristics

Sample Library

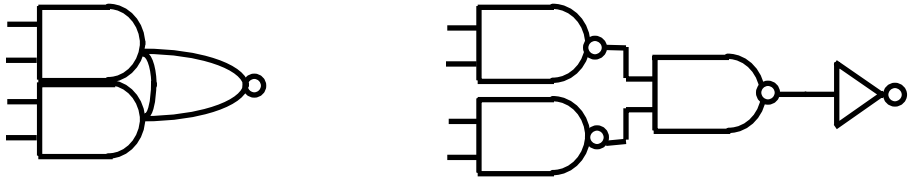
INVERTER	2	
NAND2	3	
NAND3	4	
NAND4	5	
		

Sample Library - 2

AOI21 4



AOI22 5



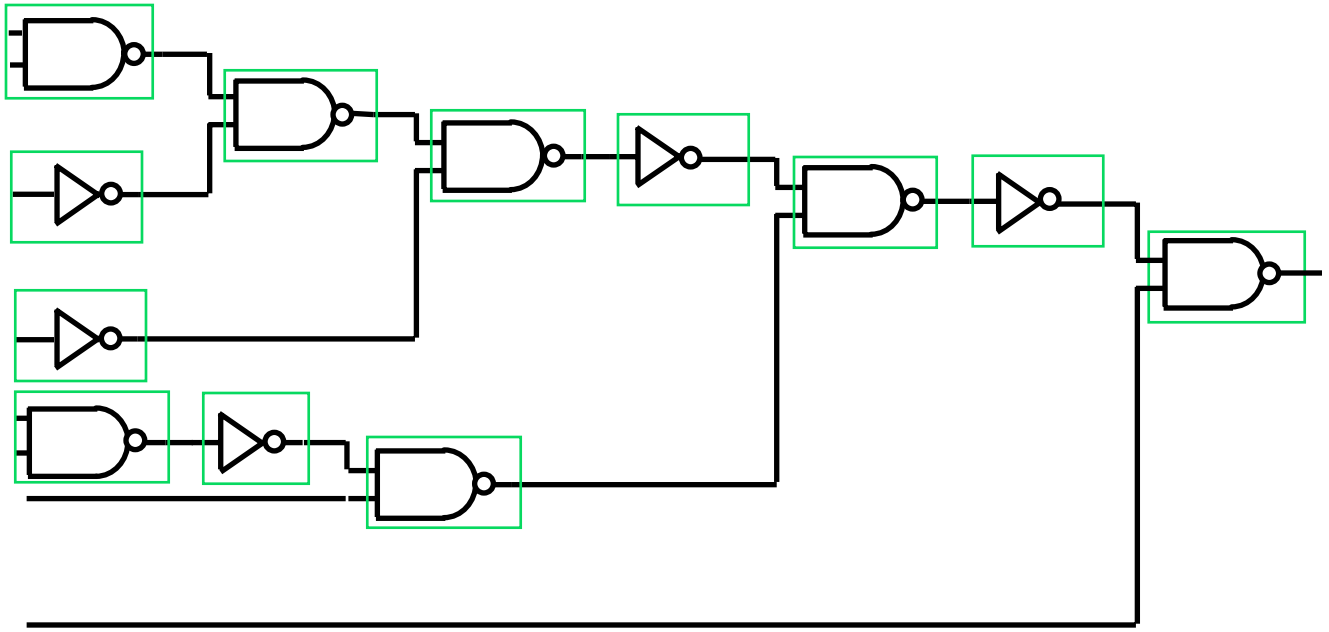
Mapping via DAG* Covering

- Represent network in canonical form
  ⇒ subject DAG
- Represent each library gate with canonical forms for the logic function
  ⇒ primitive DAGs
- Each primitive DAG has a cost
- Goal: Find a minimum cost covering of the subject DAG by the primitive DAGs

* Directed Acyclic Graph

Trivial Covering

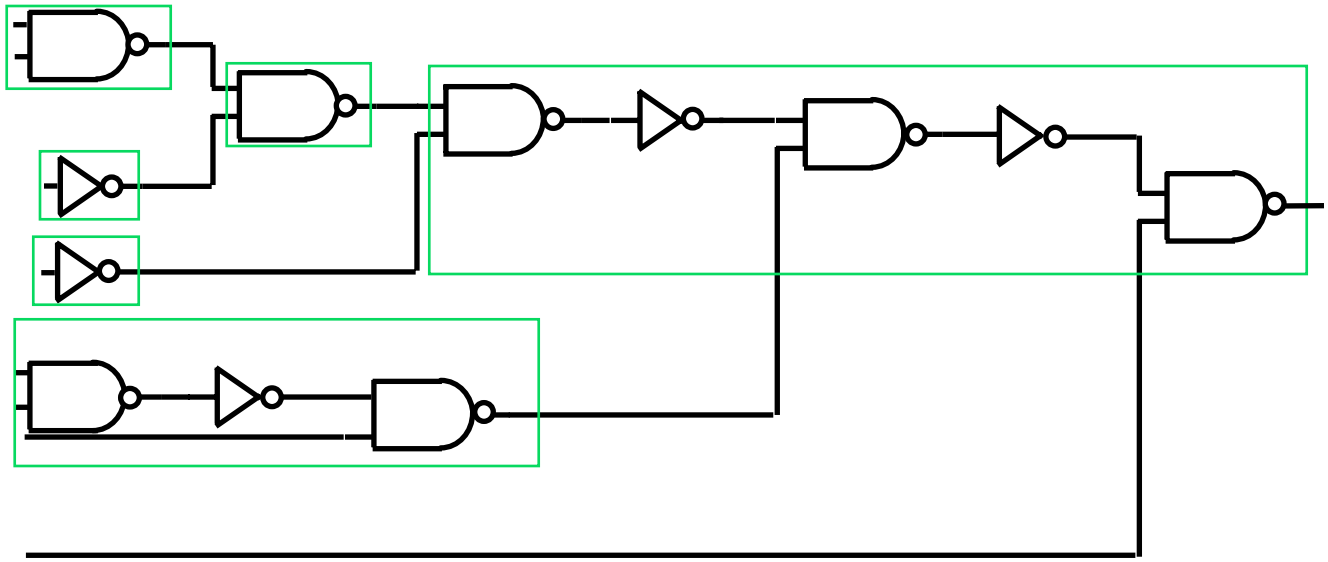
Reduce netlist into ND2 gates → subject DAG



7 NAND2 = 21
5 INV = 10

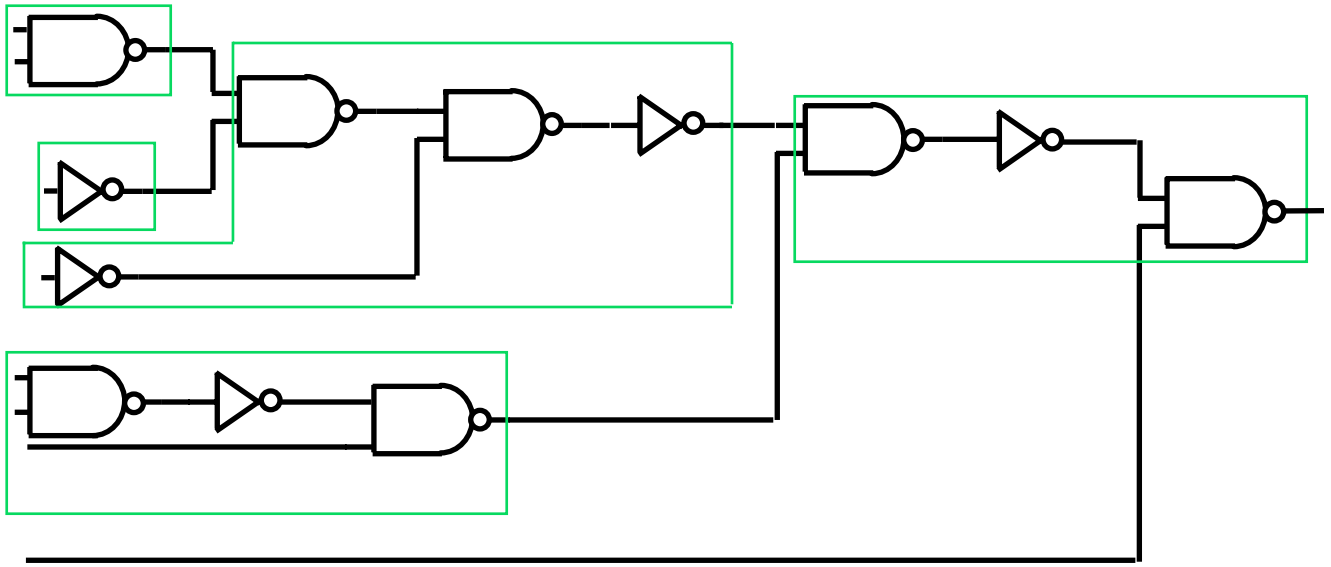
31 (area cost)

Covering #1



2 INV	= 4
2 NAND2	= 6
1 NAND3	= 4
1 NAND4	= 5
	19 (area cost)

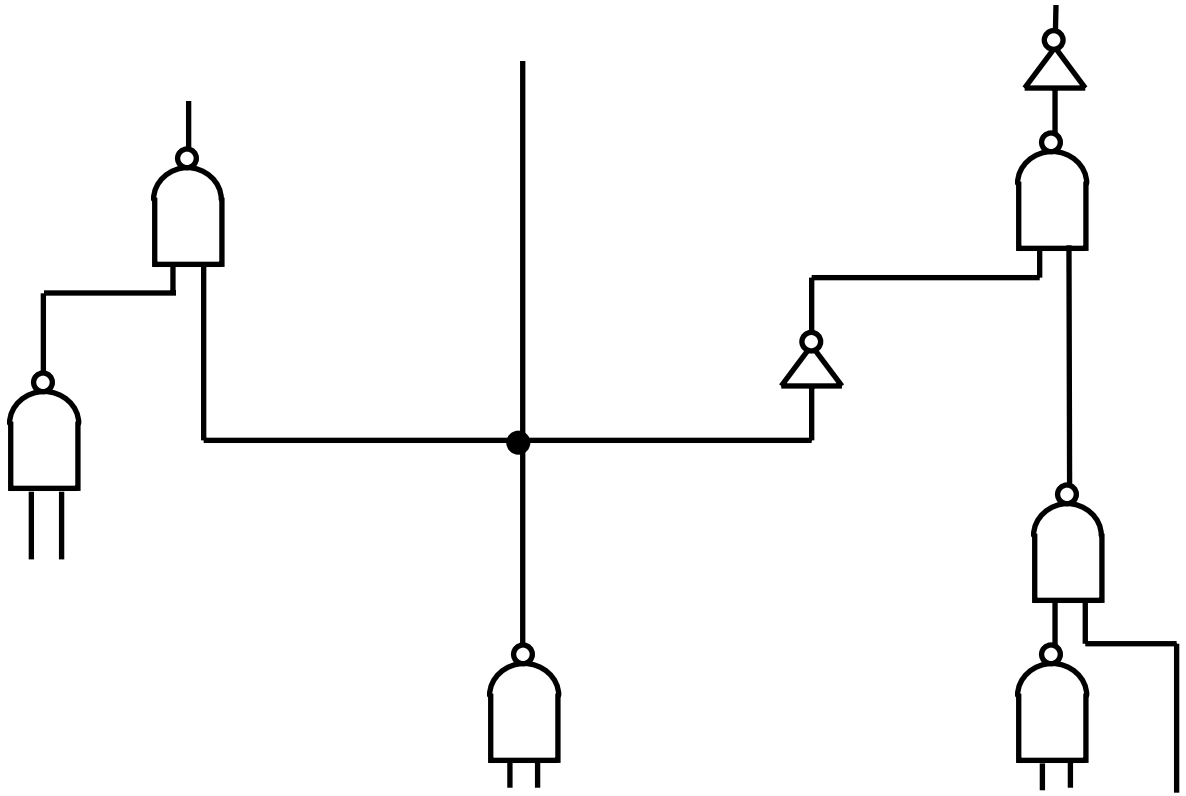
Covering #2



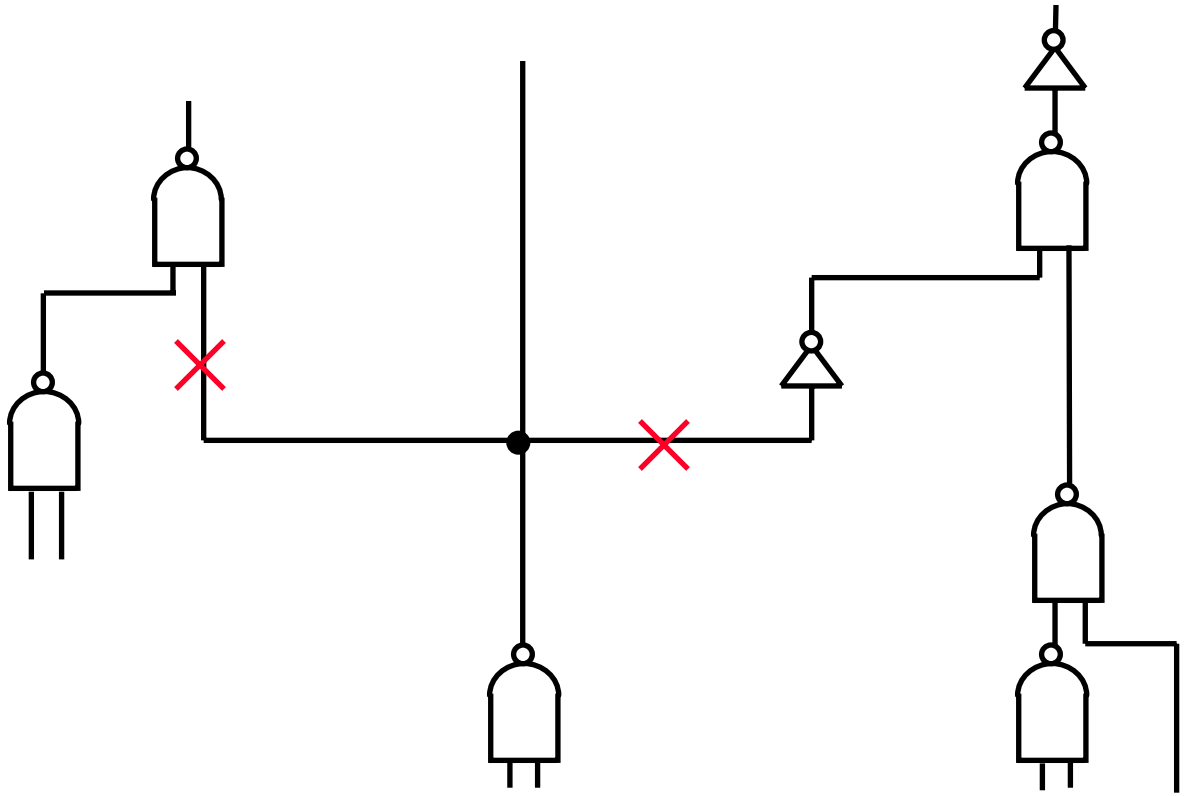
1 INV = 2
1 NAND2 = 3
2 NAND3 = 8
1 AOI21 = 4

17 (area cost)

Multiple fan-out

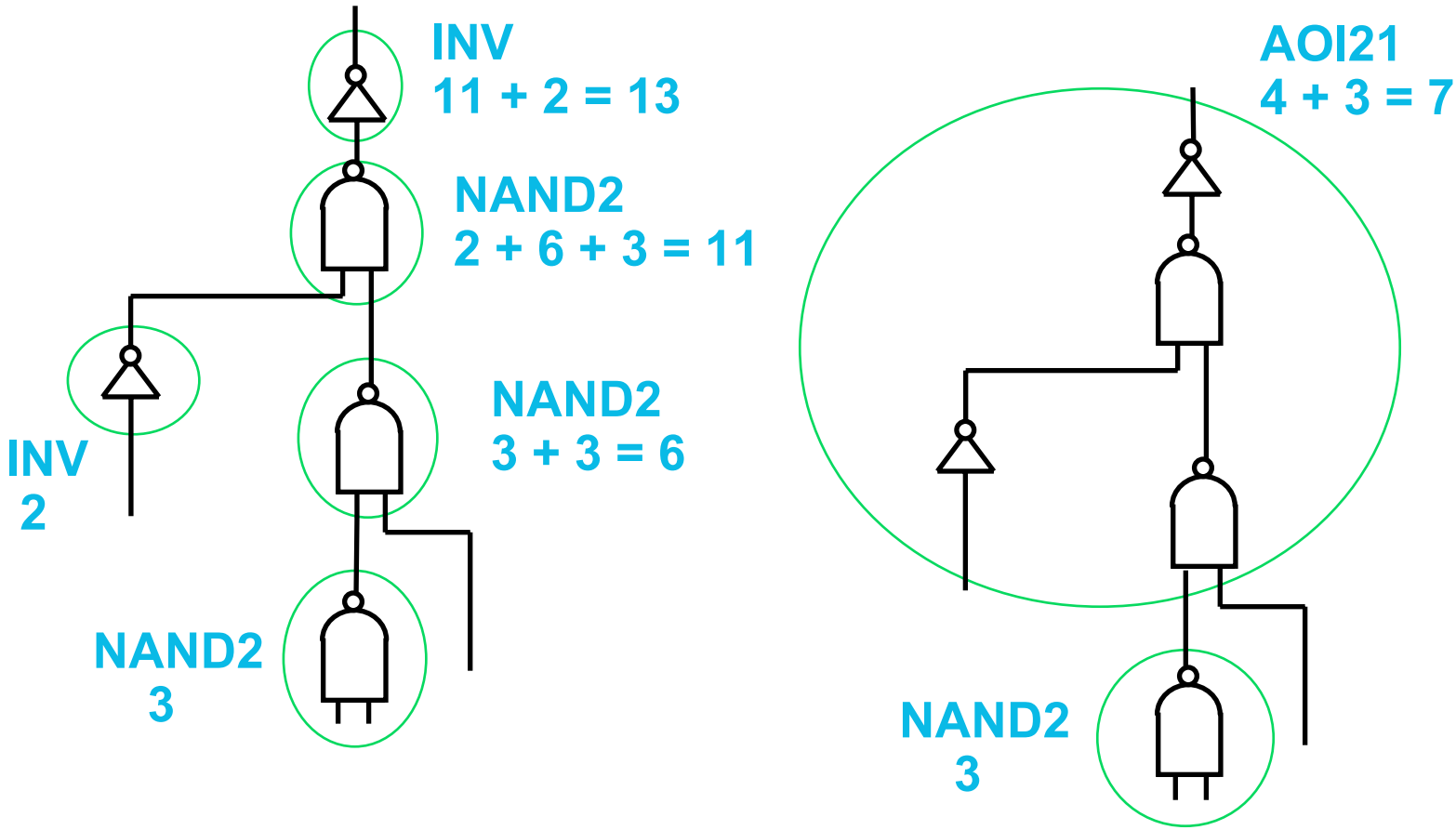


Partitioning a Graph



- Partition input netlist into a forest of trees
- Solve each tree optimally
- Stitch trees back together

Optimum Tree Covering



DAG Covering steps

- Partition DAG into a forest of trees
- Normalize the netlist
- Optimally cover each tree
  - Generate all candidate matches
  - Find optimal match using dynamic programming

Summary

- Logic optimization is an important step in the design flow
- Two-step flow
  - Technology independent optimization
  - Technology dependent optimization
- Advantages of logic optimization
  - Reduce area
  - Reduce power
  - Reduce delay

For more details…

Refer to Srinivas Devadas' slides for 6.373

http://csg.csail.mit.edu/u/d/devadas/public_html/6.373/lectures/