Additional Slides to De Micheli Book

Sungho Kang

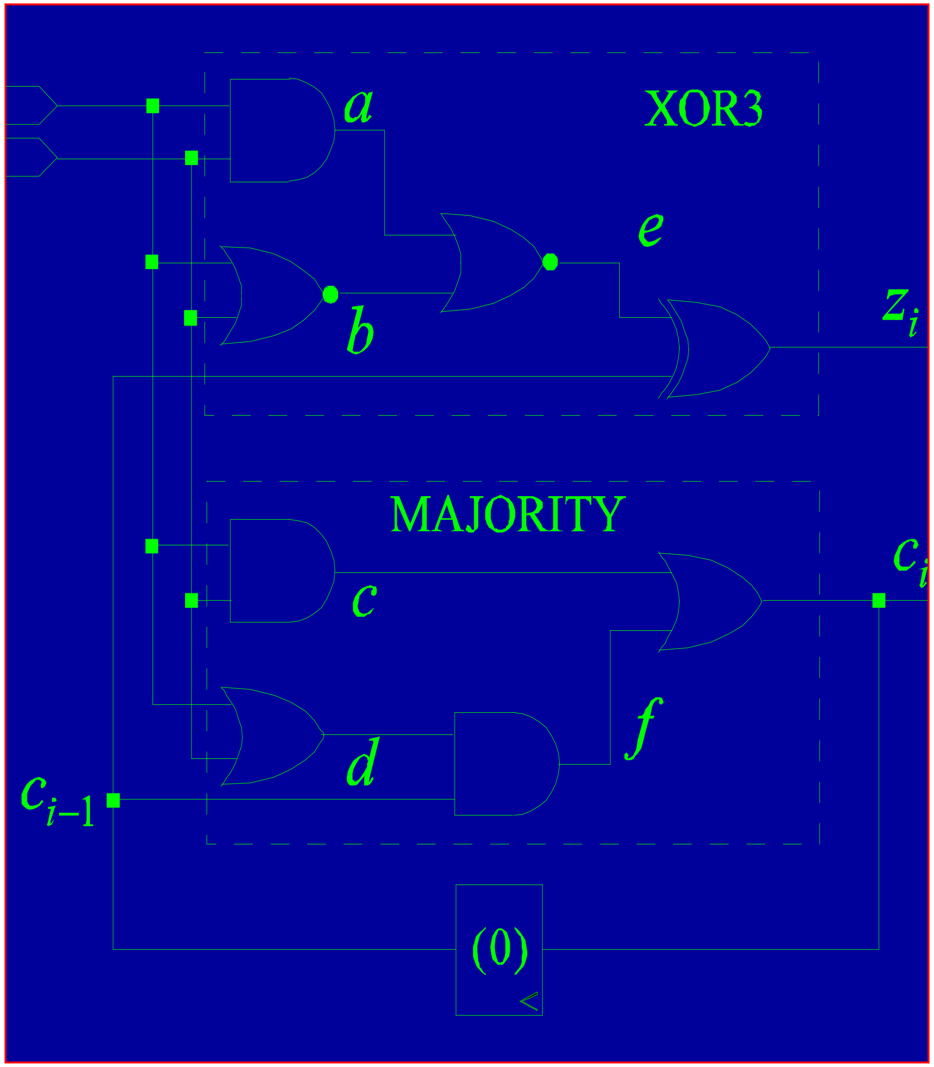
Yonsei University

Design Style - Decomposition

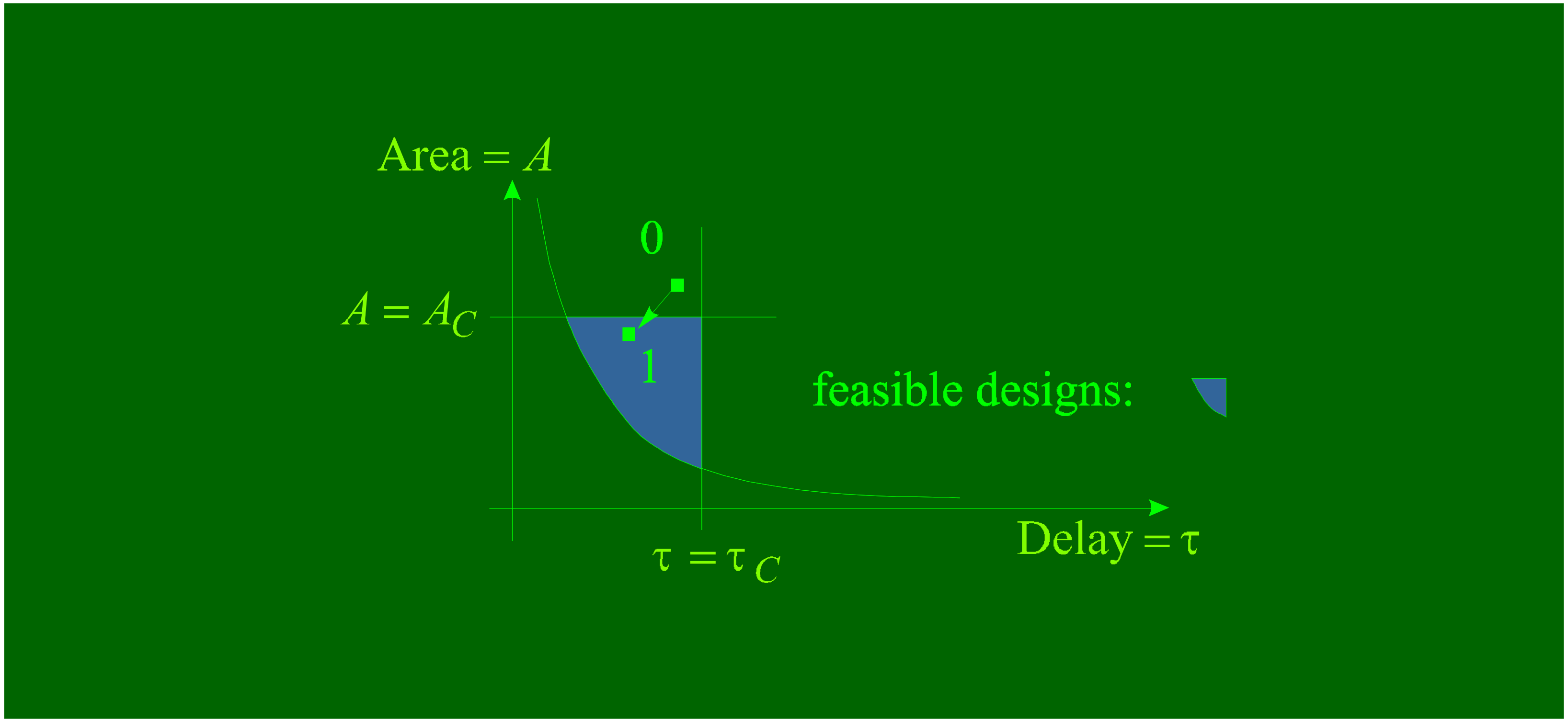
- Behavioral Synthesis
  - Resource allocation; Pipelining; Control flow parallelization; Communicating Sequential Processes; Partitioning
- Sequential Synthesis
  - Register Movement and Retiming; State Minimization; State Assignment; Synthesis for Testable FSM's; State Machine Verification
- Logic Synthesis
  - Extraction of combinational logic to HDL; Two-level minimization; Algebraic Decomposition; Multilevel Logic Minimization; Synthesis for Multifault Testability; Test Generation via Minimization; Technology mapping; Timing optimization
- Technology Mapping
  - Mapping to Library of Logic Gates; Timing Optimization
- Physical Design Synthesis
  - Cell Placement; Routing; Fabrication; Engineering Changes

Area vs Delay:The Bit-Serial Adder

- A typical **tradeoff** is area versus delay
- With just one full adder, this circuit can do 32-bit addition
- But, it is 32x slower than a parallel adder (32 full adders, 1 bit output per clock cycle)

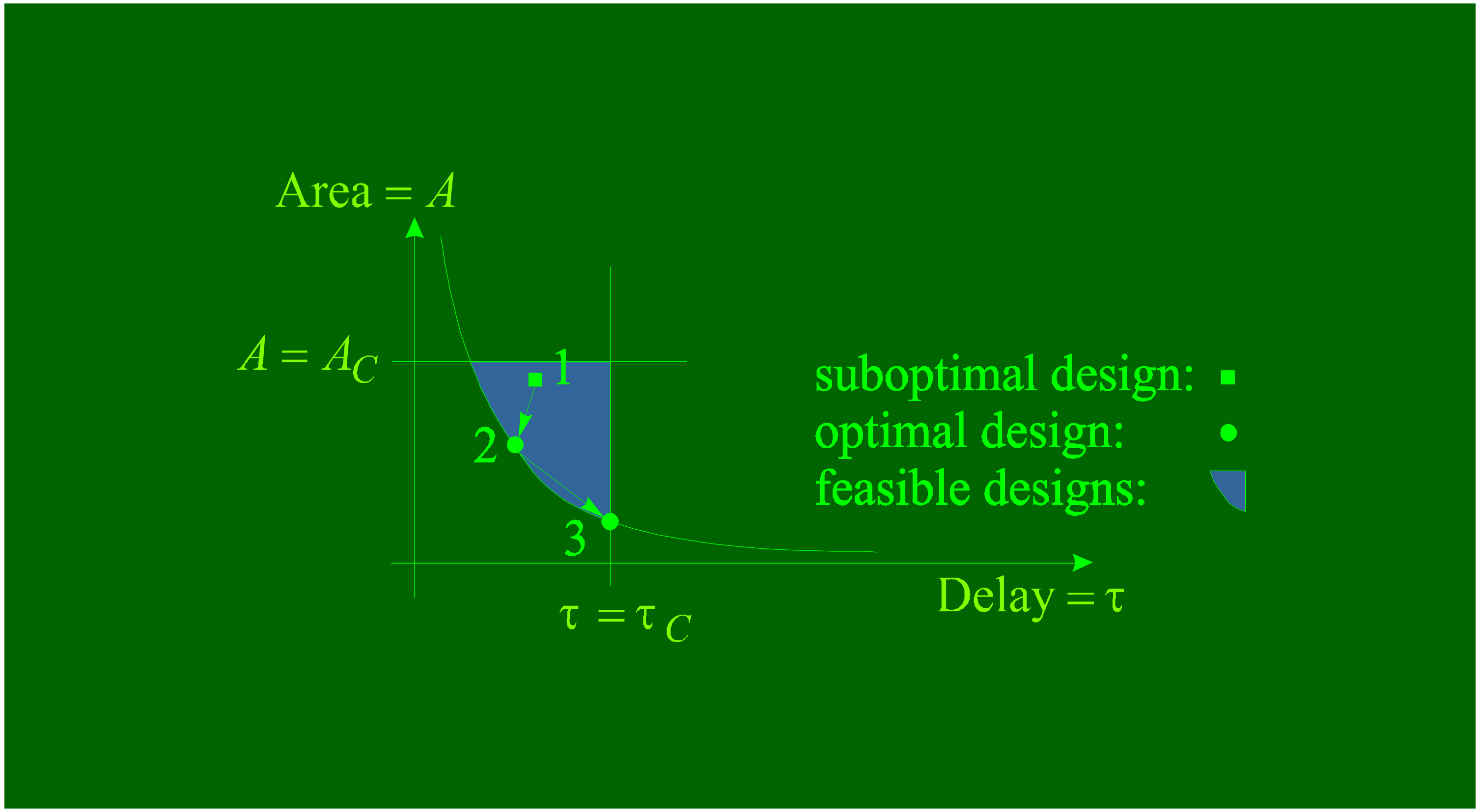


Design Tradeoff Curve



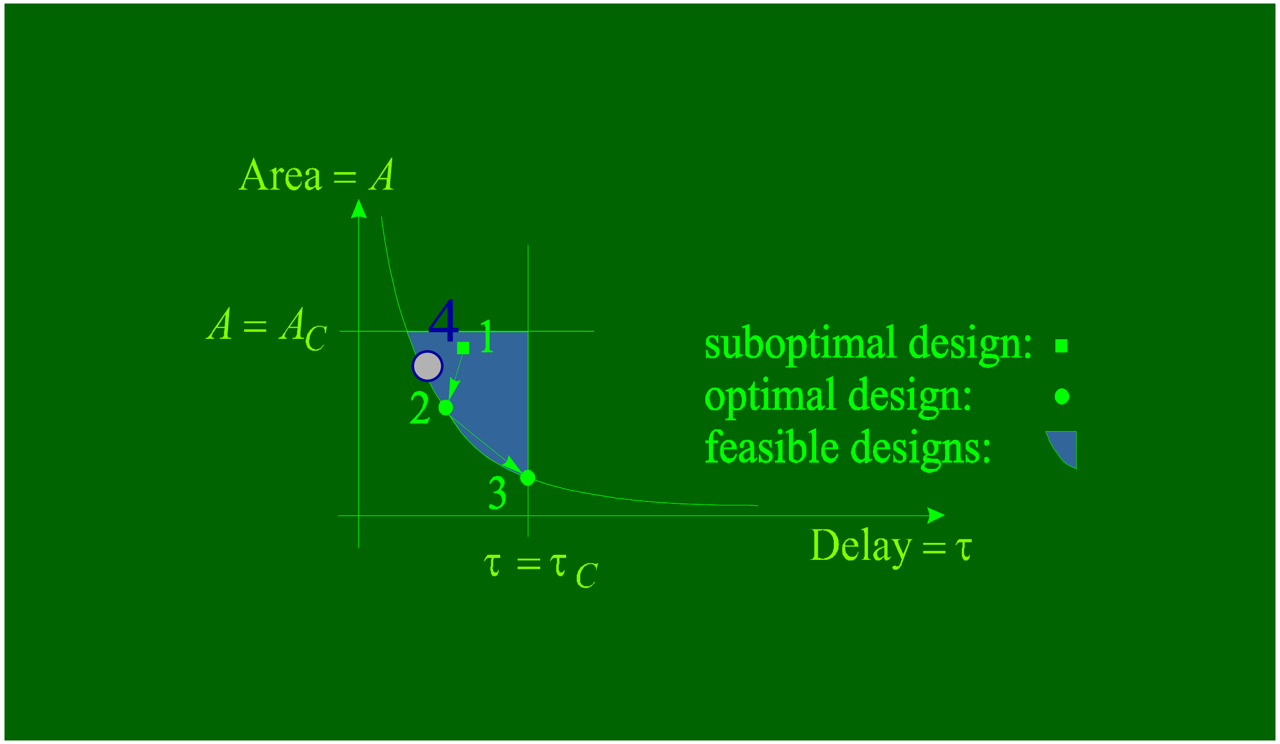
- **Holding other factors constant, the Area vs Delay tradeoff curve is typically parabolic**
- **The first design requirement is meet *Constraints* on Chip Area and Critical Path Delay (0 to 1)**

Design Tradeoffs



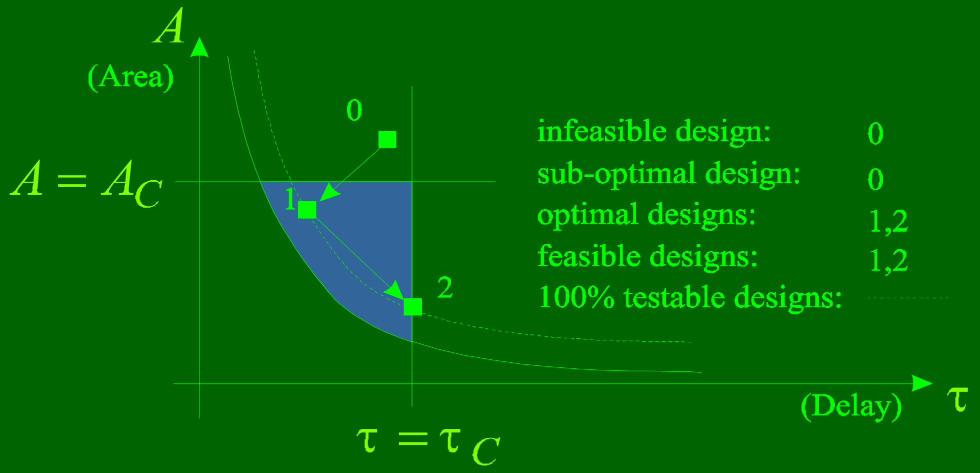
- **The next priority is to optimize a feasible design**
- **Design 2 is optimal, in the sense that area and delay cannot both be decreased from this point**
- **Tradeoff is now necessary, according to *Policy***

Design Tradeoffs



- A typical design ***Policy*** is to optimize area subject to a delay constraint (2 to 3)
- Often a preferred policy is to optimize delay subject to an area constraint (2 to 4)

Prioritizing Testability



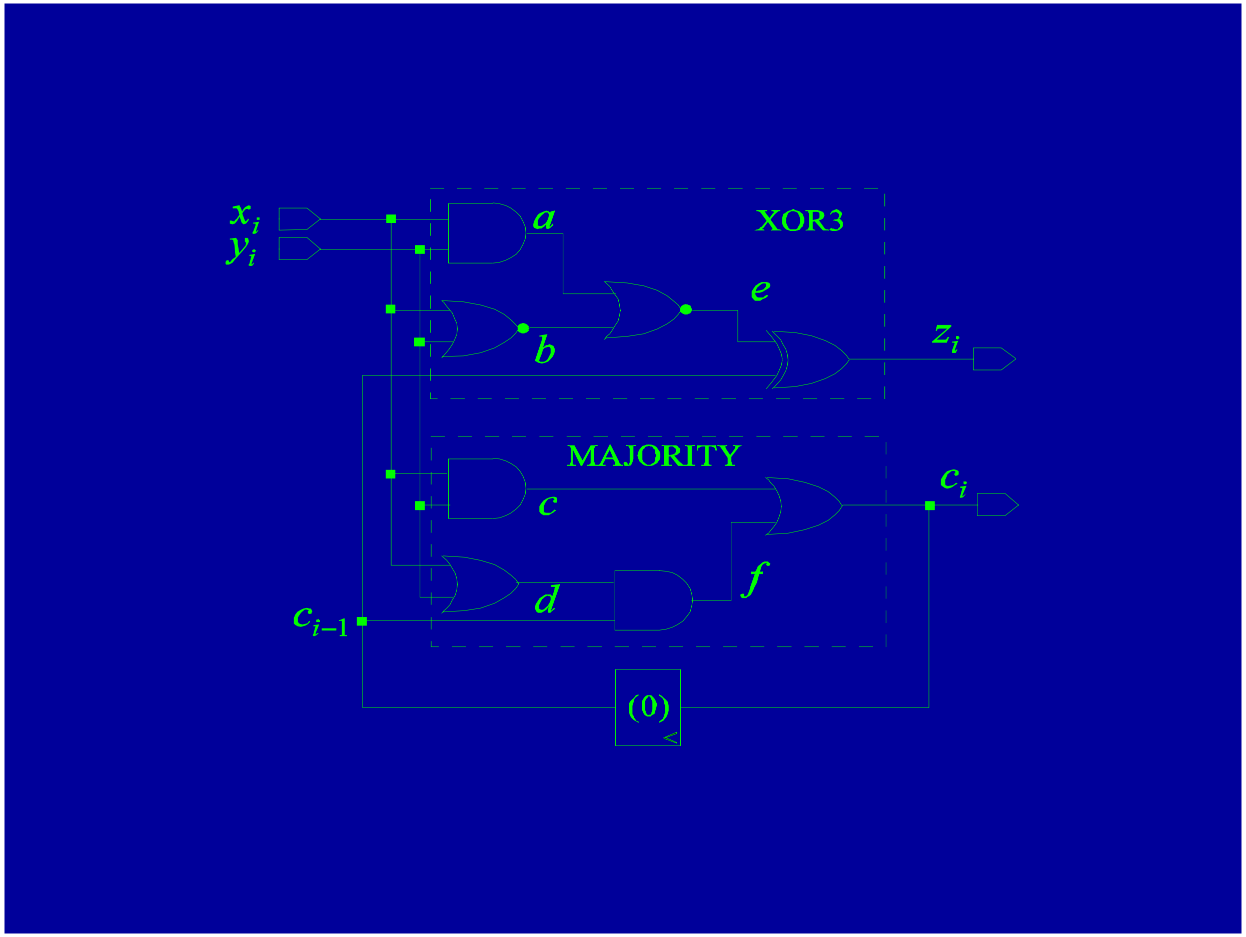
- Sometimes other factors, such as ***testability*** or power, take priority
- Typically this moves the area-delay tradeoff curve up and to the right
- Thus designs 1 and 2 are optimal

Area Optimization

- Typically performed in a *technology independent view* of the circuit
- In this view gates are regarded as logic functions
- These functions are converted to physical gates by *Technology Mapping*

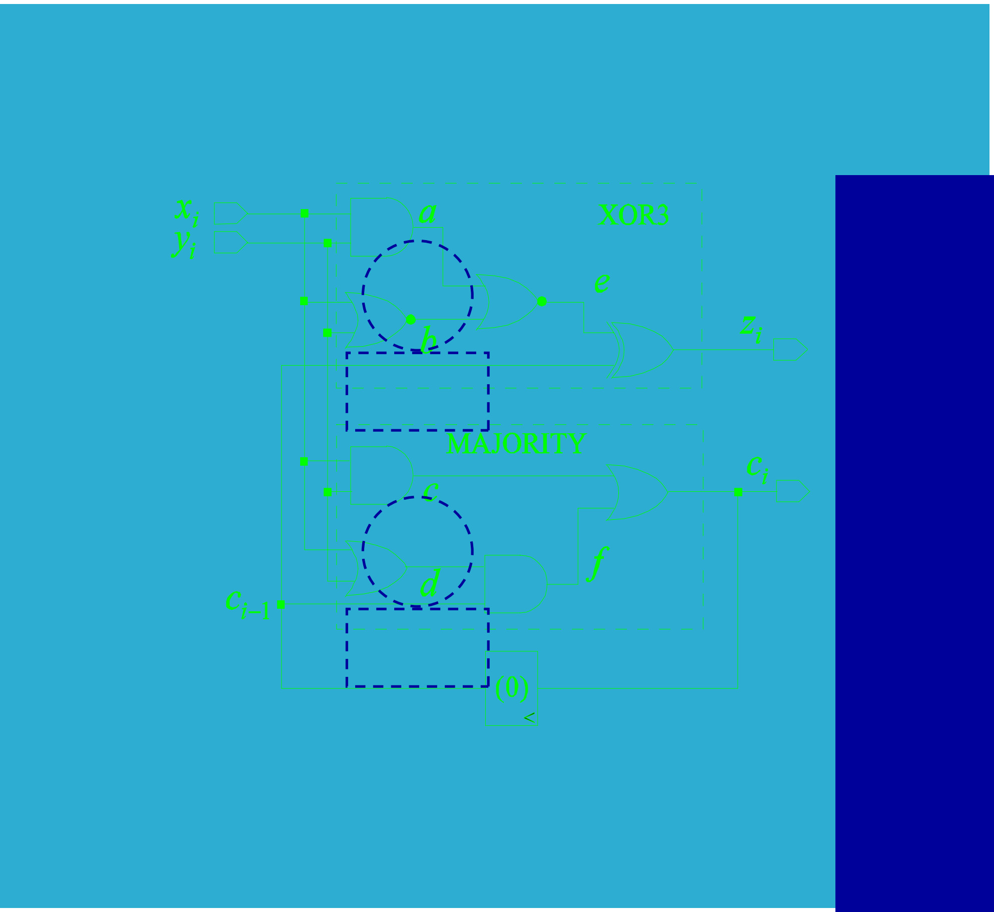
Technology Independent View

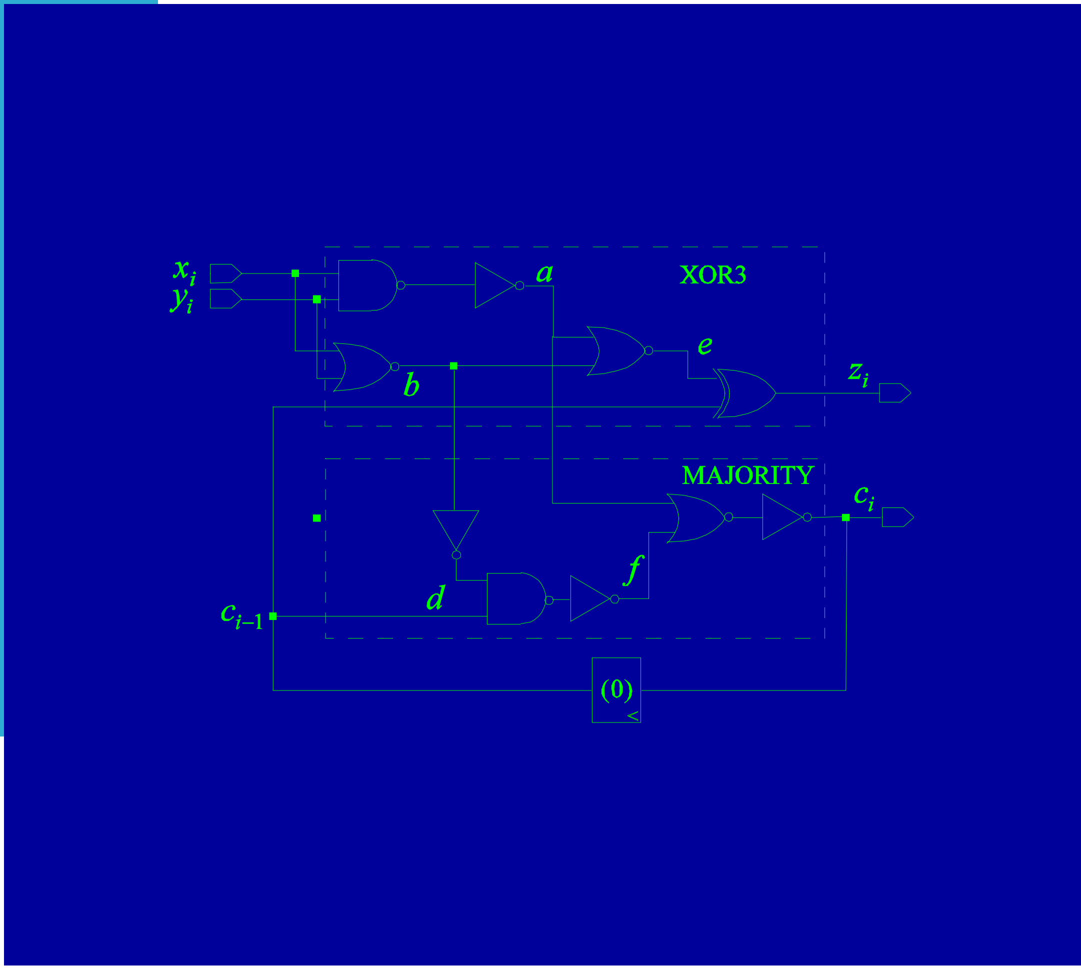
$a = xi\ yi$
$b = xi'\ yi'$
$e = a'\ b'$
$zi = ec_{i-1}{}' + e'c_{i-1}$
$c = xi\ yi$
$d = xi + y + i$
$f = dc_{i-1}$
$ci = c + f$



- In this view the gates of the full adder circuit are just logic equations

Optimization and Technology Mapping



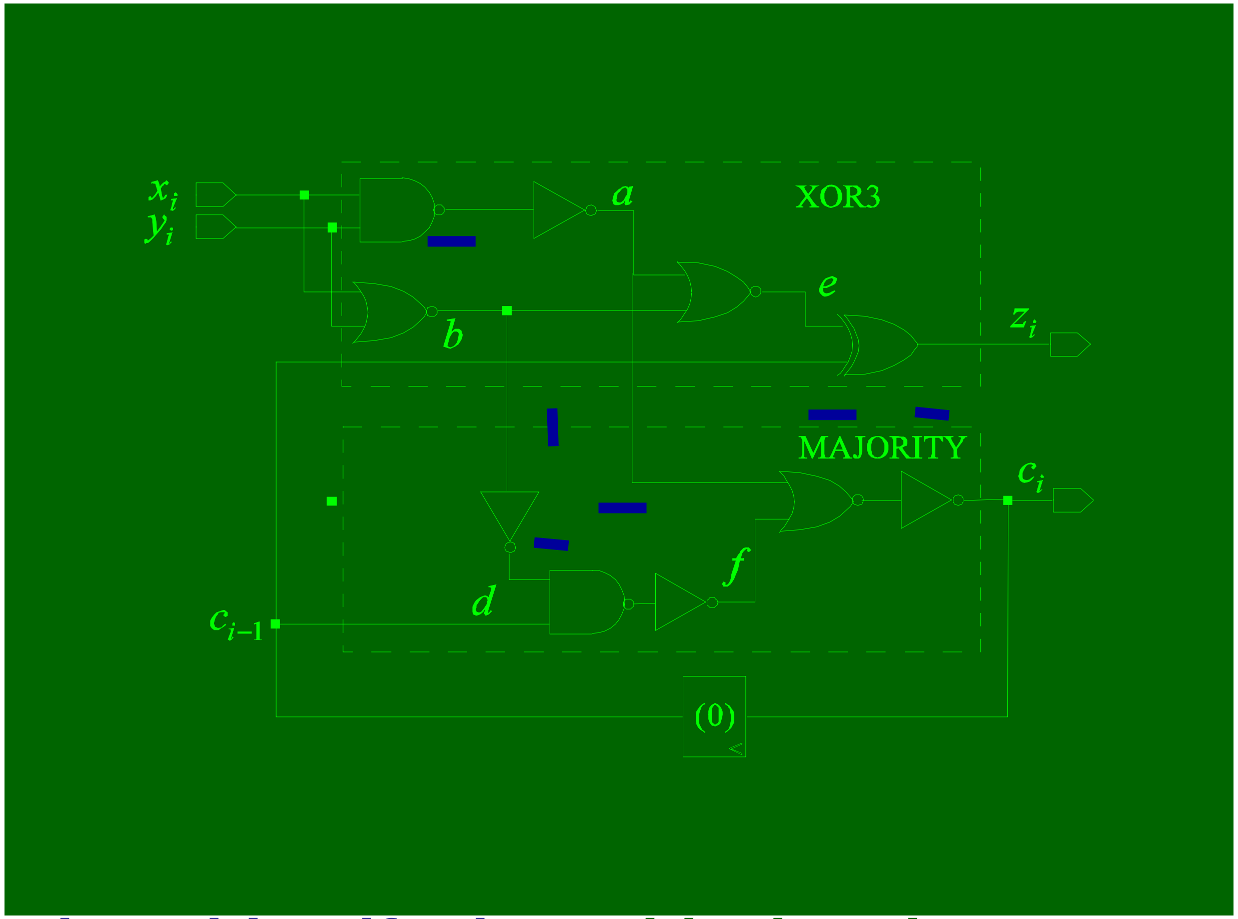


- **Common subfunctions shared**
- **Functions “Technology Mapped” to negative gates**

Testing

- Faults Models
  - Stuck-at faults
  - Delay faults
- Test vectors
  - Fault simulation
  - Automatic test pattern generation
- Diagnosis
- Testable Design

Delay Optimization



- First step is to identify the ***Critical Path***
- Simplest delay model: “number of logic levels”

Critical Path Analyzers

- Static Delay Models:
  - Levels of Logic
  - Delay function of size, load
  - Worst, best case models
- Dynamic Delay Models
  - Simplified device models
  - Full Spice analysis

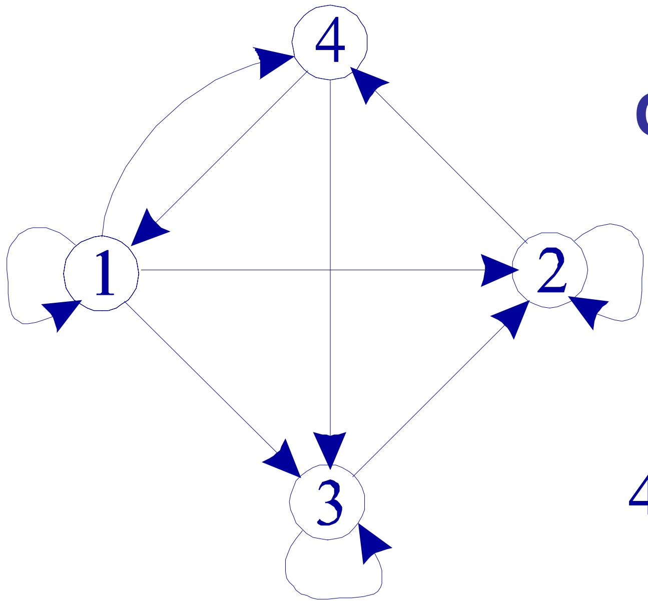
VLSI : Scalable Algorithms

- **In the 70s IBM found that 75% of all its cpu cycles went to critical path algorithms**
- **Now the bottleneck is Formal Verification**
- **Scalable algorithms required, for which cpu time and space increase:**
  - linearly in problem size *n* *(O(n))*
  - log-linearly *(O(nlog(n)))*
  - even quadratic *(O(n^2))* too expensive

Graph Models and FSM

- Graph
- Edge
- Vertex
- Undirected graph
- Digraph
  - All edges are directed
- Mixed graph
- DAG (directed acyclic graph)

Graph Terminology



Graph: ordered set of two sets

$$G = (V, E)$$

: a set of vertices or nodes

: a set of edges or arcs

$4_{\rightarrow}$**: the successors (fanouts) of node 4**

$$4_{\rightarrow} = \{1,3\}$$

${}_{\rightarrow}4$**: the predecessors (fanins) of node 4**

$$4_{\rightarrow} = \{1,2\}$$

Graph Models

- Transitive closure
- fanout
- fanin
- Source
  - v has no predecessors
- Sink
  - v has no successors

Products of Sets of Sets

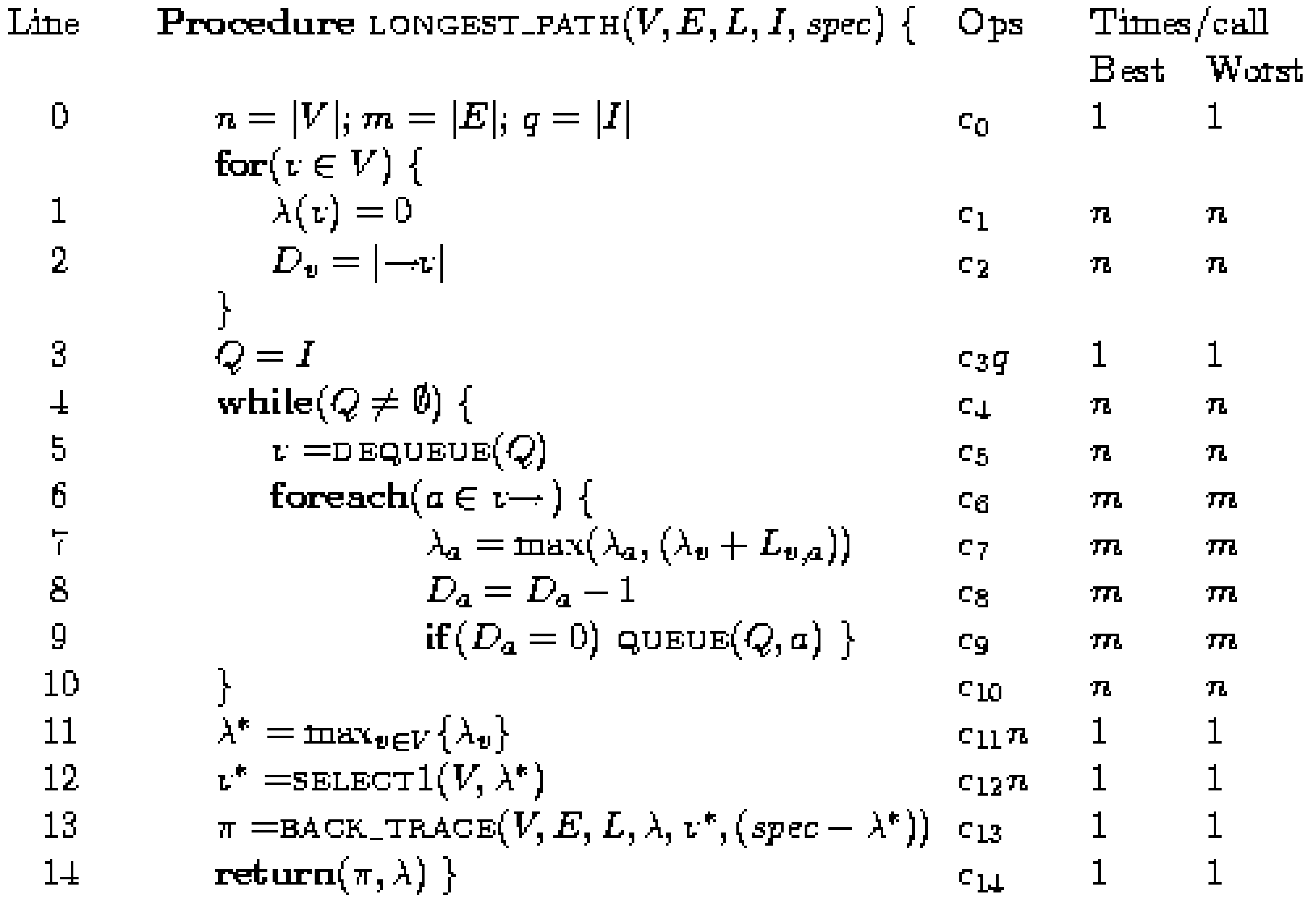
- **Intersecting 2 sets of sets**

```
Procedure SET_CARTESIAN_PRODUCT(G,H) {        Ops    Times/call
                                                     Best       Worst
  m = |G|; n= |H|;                            c1     1          1
  P = NULL                                    c2     1          1
  for (i = 1, 2, …, m) {                      c3m    1          1
        for (j = 1, 2, …, n) {                c4n    m          m
              P = P ∪ (Gi ∩ Hj)               c5     mn         mnq
        }
  }
  return(P)                                   c6     1          1
}
```

Computing Critical Path Length

- **This problem is modeled as that of finding the longest path in a DAG (Directed Acyclic Graph)**
- **Thus we digress for a while, and introduce the notions of sets and graphs**
- **Then we discuss a scalable (linear) complexity algorithm for finding the longest path in a DAG**

Longest Paths



Line	**Procedure** LONGEST_PATH$(V, E, L, I, spec)$ {	Ops	Times/call	
			Best	Worst
0	$n = \|V\|; m = \|E\|; q = \|I\|$	c_0	1	1
	for$(v \in V)$ {			
1	$\lambda(v) = 0$	c_1	n	n
2	$D_v = \|\to v\|$	c_2	n	n
	}			
3	$Q = I$	$c_3 q$	1	1
4	**while**$(Q \neq \emptyset)$ {	c_4	n	n
5	$v =$DEQUEUE(Q)	c_5	n	n
6	**foreach**$(a \in v\to)$ {	c_6	m	m
7	$\lambda_a = \max(\lambda_a, (\lambda_v + L_{v,a}))$	c_7	m	m
8	$D_a = D_a - 1$	c_8	m	m
9	**if**$(D_a = 0)$ QUEUE(Q, a) }	c_9	m	m
10	}	c_{10}	n	n
11	$\lambda^* = \max_{v \in V}\{\lambda_v\}$	$c_{11} n$	1	1
12	$v^* =$SELECT1(V, λ^*)	$c_{12} n$	1	1
13	$\pi =$BACK_TRACE$(V, E, L, \lambda, v^*, (spec - \lambda^*))$	c_{13}	1	1
14	**return**(π, λ) }	c_{14}	1	1

Backtracking

- **The slack of an edge(a,v) is the slack of v plus the difference between the length of the longest path to v and the longest path to v through (a,v)**
- **In formula, $\text{slack}_{a,v} = \text{slack}_{v} + (\lambda_v - (\lambda_a + L_{a,v}))$**
- **Here λ_v is the length of the longest path to v, and $(\lambda_a + L_{a,v})$ is the length of the longest path to v that passes through the edge (a,v)**
- **The slack of a node u is defined as the minimum of its fanout edge slacks, so**

$$\text{slack}_a = \min \text{slack}_v$$

Asymptotic Complexity

- A function $F(n)$ is in the set $O(g(n))$ if and only if there exist positive constants c_o and n_o such that
  $F(n) \leq c_o g(n)$ for all $n \geq n_o$
- This means that $F(n)$ is asymptotically bounded <u>from above</u> by a linear function of $g(n)$

- A function $F(n)$ is in the set $\Omega(g(n))$ if and only if there exist positive constants c_Ω and n_Ω such that
  $F(n) \geq c_\Omega g(n)$ for all $n \geq n_\Omega$
- This means that $F(n)$ is asymptotically bounded <u>from below</u> by a linear function of $g(n)$