

## Lecture 7 Logic Optimization

---

---

---

---

---

---

---

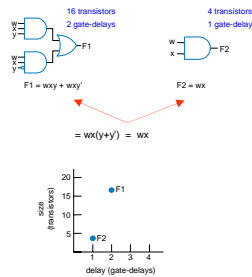
---

---

---

### Logic Optimization

- We now know how to build digital circuits
  - How can we build *better* circuits?
- Let's consider two important design criteria
  - Delay** – the time from inputs changing to new correct stable output
  - Size** – the number of transistors
- Assumption
  - Every gate has delay of "1 gate-delay"
  - Every gate *input* requires 2 transistors
  - Ignore inverters



Transforming F1 to F2 represents an **optimization**. Better in all criteria of interest

---

---

---

---

---

---

---

---

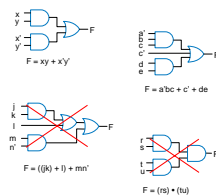
---

---

### Two-level Logic Optimization

- Two-level logic
  - Circuit with only two levels (ORed AND gates)
- Basically sum-of-products form
  - An equation written as an ORing of product terms

Are these two-level logic?



technically yes, but not what we mean in terms of logic minimization

---

---

---

---

---

---

---

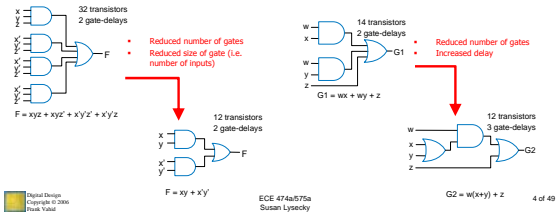
---

---

---

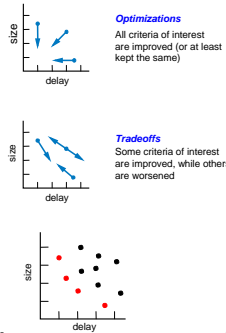
## Optimization vs. Tradeoff

- Optimization - Defined as better in all criteria of interest
  - Delay and size - we consider size minimization only (2-level logic only)
  - In reality requires a balance of many criteria metrics
    - Cost, reliability, time-to-market, etc...
- Tradeoff - Improves some, but worsens other, criteria of interest



## Pareto Points

- We obviously prefer optimizations, but often must accept tradeoffs
  - You can't build a car that is the most comfortable, and has the best fuel efficiency, and is the fastest - you have to give up something to gain other things
- Many options in solution space
  - Pareto point
    - Point in solution space in which no other point better in all metrics
    - Shown in red
    - Pareto points yield the trade-off curve



## Combinational Logic Optimization and Tradeoffs

- Two-level size optimization using algebraic methods
  - Goal: circuit with only two levels (ORed AND gates), with minimum transistors
    - Though transistors getting cheaper (Moore's Law), they still cost something
- Define problem algebraically
  - Sum-of-products yields two levels
    - $F = abc + abc'$  is sum-of-products;  $G = w(xy + z)$  is not.
  - Transform sum-of-products equation to have fewest literals and terms
    - Each literal and term translates to a gate input, each of which translates to about 2 transistors
    - Ignore inverters for simplicity

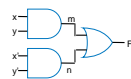
Example

$$F = xyz + xyz' + x'y'z + x'y'z'$$

$$F = xy(z + z') + x'y'(z + z')$$

$$F = xy \cdot 1 + x'y' \cdot 1$$

$$F = xy + x'y'$$



= 6 gate inputs \* 2 transistor/input  
= 12 transistors

## Boolean Algebra

- How do we use Boolean algebra to obtain fewest literals and terms?

1a. $0 \cdot 0 = 0$	10a. $x \cdot y = y \cdot x$	(Commutative)
1b. $1 + 1 = 1$	10b. $x + y = y + x$	
2a. $1 \cdot 1 = 1$	11a. $x \cdot (y \cdot z) = (x \cdot y) \cdot z$	(Associative)
2b. $0 + 0 = 0$	11b. $x + (y + z) = (x + y) + z$	
3a. $0 \cdot 1 = 1 \cdot 0 = 0$	12a. $x \cdot (y + z) = x \cdot y + x \cdot z$	(Distributive)
3b. $0 + 1 = 1 + 0 = 1$	12b. $x + (y \cdot z) = (x + y) \cdot (x + z)$	
4a. If $x = 0$ , then $x' = 1$	13a. $x + x \cdot y = x$	(Absorption)
4b. If $x = 1$ , then $x' = 0$	13b. $x \cdot (x + y) = x$	
5a. $x \cdot 0 = 0$	14a. $x \cdot y + x \cdot y' = x$	(Combining)
5b. $x + 1 = 1$	14b. $(x + y) \cdot (x + y') = x$	
6a. $x \cdot 1 = x$	15a. $(x \cdot y)' = x' + y'$	(DeMorgan's Theorem)
6b. $x + 0 = x$	15b. $(x + y)' = x' \cdot y'$	
7a. $x \cdot x = x$	16a. $x + x' \cdot y = x + y$	
7b. $x + x = x$	16b. $x \cdot (x' + y) = x \cdot y$	
8a. $x \cdot x' = 0$		
8b. $x + x' = 1$		
9. $x'' = x$		

ECE 474a/575a  
Susan Lysdecky

7 of 49

## Algebraic Two-Level Size Minimization

### Uniting Theorem

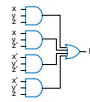
- Multiply out to sum-of-products, then apply Uniting Theorem
  - $ab + ab' = a(b + b') = a \cdot 1 = a$
  - "Combining terms to eliminate a variable"
  - (Formally called the "Uniting theorem")
- Sometimes after combining terms, can combine resulting terms

$$F = xyz + xyz' + x'y'z' + x'y'z$$

$$F = xy(z + z') + x'y'(z + z')$$

$$F = xy \cdot 1 + x'y' \cdot 1$$

$$F = xy + x'y'$$



delay = 2 gate delay  
size =  $16 \cdot 2 = 32$  transistors

$$G = xy'z' + xy'z + xyz + xyz'$$

$$G = xy'(z' + z) + xy(z + z')$$

$$G = xy' \cdot 1 + xy \cdot 1 \quad (\text{now do again})$$

$$G = x(y' + y)$$

$$G = x$$



delay = 2 gate delay  
size =  $6 \cdot 2 = 12$  transistors

Digital Design  
Copyright © 2006  
Prentice Hall

ECE 474a/575a  
Susan Lysdecky

8 of 49

## Algebraic Two-Level Size Minimization

### Duplication

- Duplicating a term sometimes helps
  - Note that doesn't change function
  - $c + d = c + d + d = c + d + d + d + \dots$

$$F = x'y'z' + x'y'z + x'yz$$

$$F = x'y'z' + x'y'z + x'y'z + x'yz$$

$$F = x'y'(z' + z) + x'z(y' + y)$$

$$F = x'y' + x'z$$

Digital Design  
Copyright © 2006  
Prentice Hall

ECE 474a/575a  
Susan Lysdecky

9 of 49







## Four-Variable K-Maple Example

- Minimize:  $H = a'b'(cd' + c'd) + ab'c'd' + ab'cd' + a'bd + a'bcd'$

- Convert to sum-of-products  
 $H = a'b'cd' + a'b'c'd' + ab'c'd' + ab'cd' + a'bd + a'bcd'$

	H <sub>cd</sub>			
ab	00	01	11	10
00				
01				
11				
10				

ECE 474a/575a  
Susan Lyshecky

19 of 49

---

---

---

---

---

---

---

---

---

---

## Four-Variable K-Maple Example - Continued

- Minimize:  $H = a'b'(cd' + c'd) + ab'c'd' + ab'cd' + a'bd + a'bcd'$

- Convert to sum-of-products  
 $H = a'b'cd' + a'b'c'd' + ab'c'd' + ab'cd' + a'bd + a'bcd'$

- Place 1s in K-map cells

	H <sub>cd</sub>			
ab	00	01	11	10
00	1	0	0	1
01	0	1	1	1
11	0	0	0	0
10	1	0	0	1

Labels for 1s:  $a'b'c'd'$ ,  $a'bd$ ,  $a'b'cd'$ ,  $a'bcd'$ ,  $ab'cd'$ ,  $ab'c'd'$

ECE 474a/575a  
Susan Lyshecky

20 of 49

---

---

---

---

---

---

---

---

---

---

## Four-Variable K-Maple Example - Continued

- Minimize:  $H = a'b'(cd' + c'd) + ab'c'd' + ab'cd' + a'bd + a'bcd'$

- Convert to sum-of-products  
 $H = a'b'cd' + a'b'c'd' + ab'c'd' + ab'cd' + a'bd + a'bcd'$

- Place 1s in K-map cells

- Cover 1s

	H <sub>cd</sub>			
ab	00	01	11	10
00	1	0	0	1
01	0	1	1	1
11	0	0	0	0
10	1	0	0	1

Labels for groups:  $a'bc$ ,  $a'bd$ ,  $b'd'$

Funny-looking circle, but remember that left/right adjacent, and top/bottom adjacent

ECE 474a/575a  
Susan Lyshecky

21 of 49

---

---

---

---

---

---

---

---

---

---









## Quine-McCluskey – Example 1

$$\text{Minimize } F = a'b'c' + a'b'c + ab'c + abc' + abc$$

### Step 1: Find all the prime implicants

- List all elements of on-set and don't care set, represented as a binary number
- Group minterms according to the number of 1's in the minterm

$a'b'c' \rightarrow (0) 000$	G0	(0) 000	} group G0 contains all minterms containing zero 1's
$a'b'c \rightarrow (1) 001$	G1	(1) 001	
$ab'c \rightarrow (5) 101$	G2	(5) 101	} group G2 contains all minterms containing two 1's
$abc' \rightarrow (6) 110$	G3	(6) 110	
$abc \rightarrow (7) 111$			} group G3 contains all minterms containing three 1's

*this grouping strategy will help us compare the minterms systematically*

ECE 474a/575a  
Susan Lyshecky

34 of 49

## Quine-McCluskey – Example 1

### Step 1: Find all the prime implicants(cont')

- Compare each entry in  $G_i$  to each entry in  $G_{i+1}$ 
  - If they differ by 1 bit, we can apply the uniting theorem and eliminate a literal
  - Add check to minterm/implicant to remind us that it is not a prime implicant (combined with another element to form a larger implicant)

G0 ✓ (0) 000	G0 (0,1) 00-	no new implicants are generated – end of step 1
G1 ✓ (1) 001	G1 (1,5) -01	
G2 ✓ (5) 101	G2 (5,7) 1-1	we have found all prime implicants (ones without check marks)
✓ (6) 110	(6,7) 11-	
G3 ✓ (7) 111		

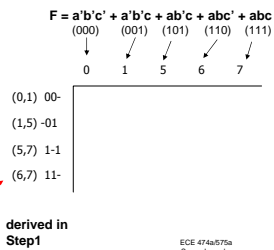
ECE 474a/575a  
Susan Lyshecky

35 of 49

## Quine-McCluskey – Example 1

### Step 2: Find all essential prime implicants

- Create prime implicant chart
  - Columns are minterm indices, rows are the prime implicants we determined



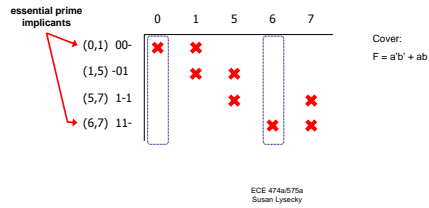
ECE 474a/575a  
Susan Lyshecky

36 of 49

## Quine-McCluskey – Example 1

### Step 2: Find all essential prime implicants (cont')

- Place "X" in a row if the prime implicant covers the minterm
- Essential prime implicants are found by looking for rows with a single "X"
  - If minterm is covered by one and only one prime implicant – it's an essential prime implicant
- Add essential prime implicants to the cover




---

---

---

---

---

---

---

---

---

---

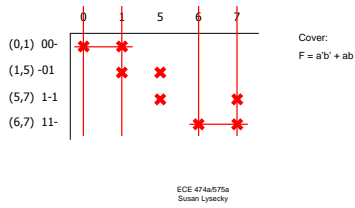
---

---

## Quine-McCluskey – Example 1

### Step 3: Select a minimal set of remaining prime implicants that covers the on set of the function

- Step 2 determined essential prime implicants, and added to cover
  - Essential prime implicants may cover other minterms, cross out all minterms covered by the prime implicants
  - Minterm only needs to be covered once




---

---

---

---

---

---

---

---

---

---

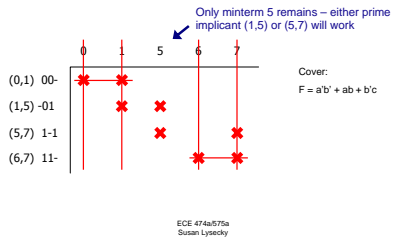
---

---

## Quine-McCluskey – Example 1

### Step 3: Select a minimal set of remaining prime implicants that covers the on set of the function (cont')

- Based on which minterms are left, add minimal set of prime implicants to cover




---

---

---

---

---

---

---

---

---

---

---

---







## Quine-McCluskey

---

- What about don't cares?
- Alternative methods to determine Minimum Cover
  - Row vs. Column Dominance

ECE 474a/574a  
Susan Lysecky

43 of 49

---

---

---

---

---

---

---

---