

# CS137: Electronic Design Automation

Day 4: January 14, 2004  
Two-Level Logic-Synthesis



CALTECH CS137 Winter2004 -- DeHon

## Today

- Two-Level Logic Optimization
  - Problem
  - Definitions
  - Basic Algorithm: Quine-McClusky
  - Improvements

CALTECH CS137 Winter2004 -- DeHon

## Problem

- **Given:** Expression in combinational logic
- **Find:** Minimum (cost) sum-of-products expression
- Ex.
  - $Y = a*b*c + a*b*/c + a*/b*c$
  - $Y = a*b + a*c$

CALTECH CS137 Winter2004 -- DeHon

## EDA Use

- Minimum size PLA, PAL, ...
  - Programmable Logic Array
  - Programmable Array Logic
- Minimum number of gates for two-level implementation
- Starting point for multi-level optimization

CALTECH CS137 Winter2004 -- DeHon

# Programmable Array Logic (PLAs)

CALTECH CS137 Winter2004 -- DeHon

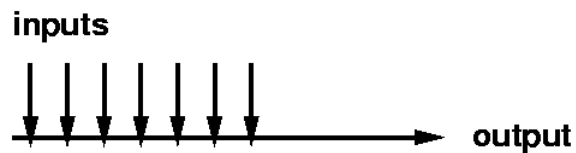
## PLA

- Directly implement flat (two-level) logic
  - $O = a*b*c*d + !a*b!*d + b!*c*d$
- Exploit substrate properties allow wired-OR

CALTECH CS137 Winter2004 -- DeHon

# Wired-or

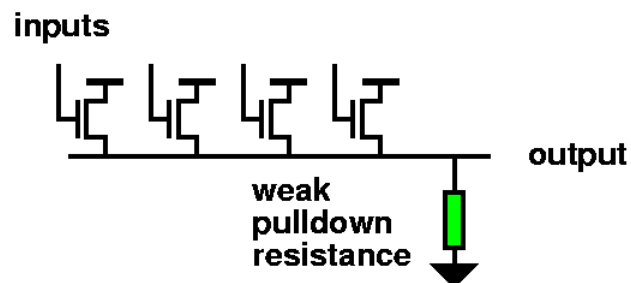
- Connect series of inputs to wire
- Any of the inputs can drive the wire high



CALTECH CS137 Winter2004 -- DeHon

# Wired-or

- Obvious Implementation with Transistors

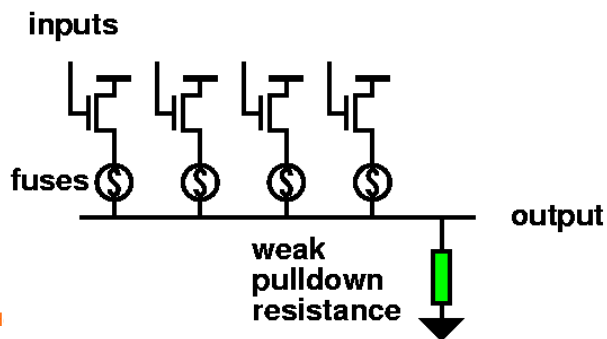


CALTECH CS137 Winter2004 -- DeHon

# Programmable Wired-or

- Use some memory function to programmable connect (disconnect) wires to OR

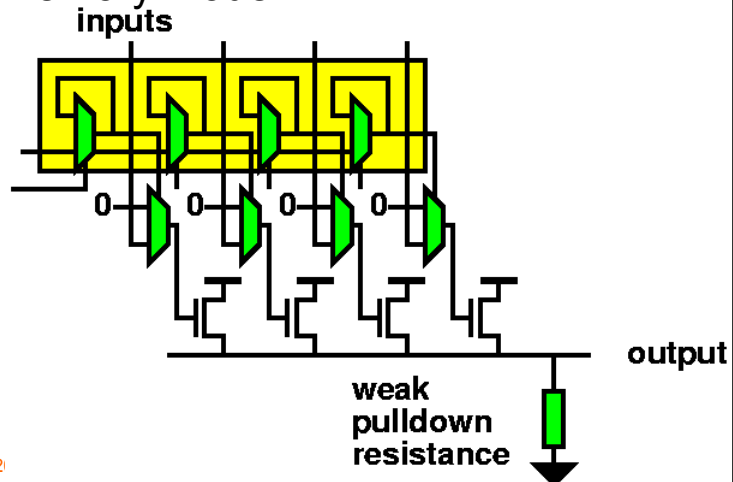
- Fuse:



CALTECH CS137 Winter2004 -- DeH

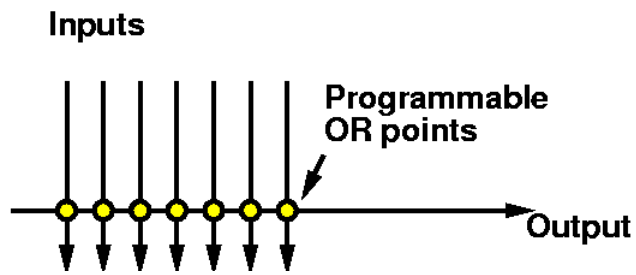
# Programmable Wired-or

- Gate-memory model



CALTECH CS137 Winter2

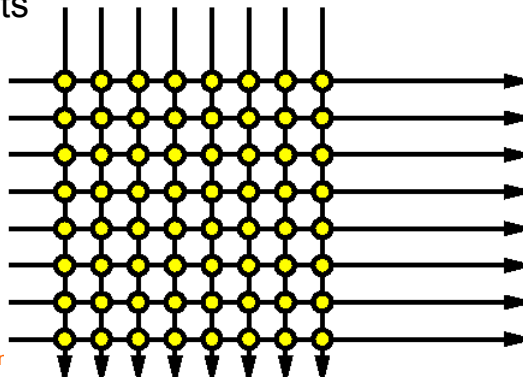
## Diagram Wired-or



CALTECH CS137 Winter2004 -- DeHon

## Wired-or array

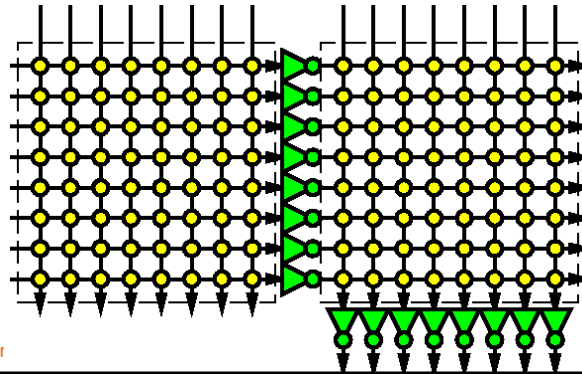
- Build into array
  - Compute many different **or** functions from set of inputs



CALTECH CS137 Winter2004 -- DeHon

# Combined **or**-arrays to PLA

- Combine two **or** (**nor**) arrays to produce PLA (**or-and** / **and-or** array)

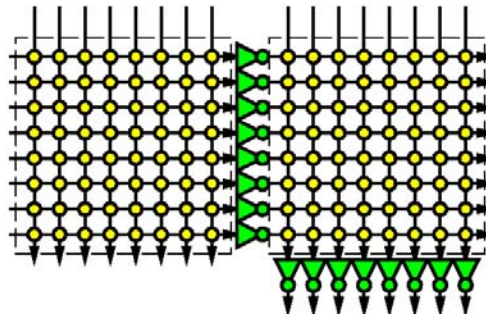


CALTECH CS137 Winter

# PLA

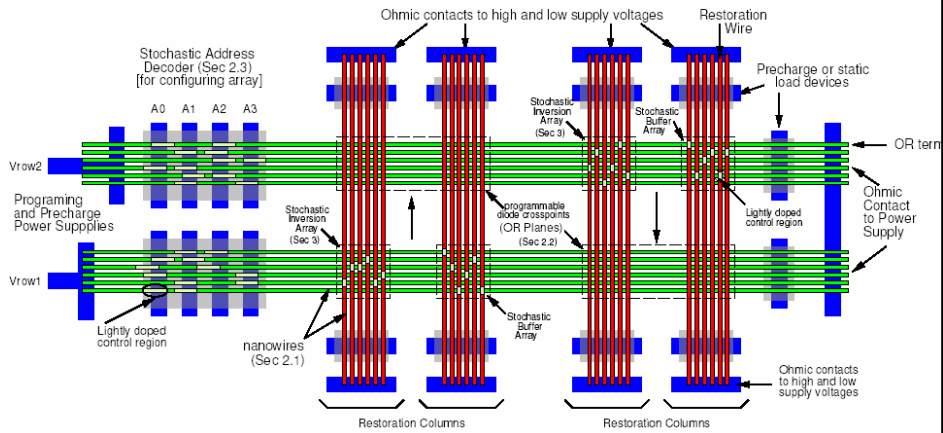
- Can implement each **and** on single line in first array
- Can implement each **or** on single line in second array

Strictly speaking:  
**or** in first term **and**  
in second,  
but with both polarities  
of inputs, can invert so  
is **and-or**.



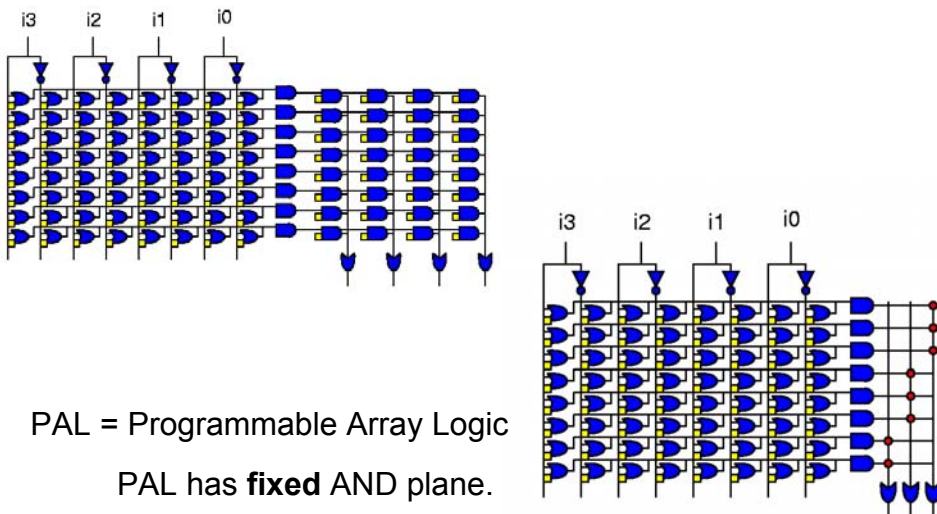
CALTECH CS137 Winter2004 -- DeHon

# Nanowire PLA



CALTECH CS137 Winter2004 -- DeHon

# PLA and PAL



CALTECH CS137 Winter2004 -- DeHon

...back to optimization...

CALTECH CS137 Winter2004 -- DeHon

## EDA Use for 2-level Logic Min.

- Minimum size PAL, PLA, ...
  - Programmable Logic Array
  - Programmable Array Logic
- Minimum number of gates for two-level implementation
- Starting point for multi-level optimization

CALTECH CS137 Winter2004 -- DeHon

# Complexity

- Set covering problem
  - NP-hard

CALTECH CS137 Winter2004 -- DeHon

# Cost

- PLA/PAL - first order
  - number of product terms
- Abstract (mis, sis)
  - {multilevel, sequential} interactive synthesis
  - number of literals
    - $\text{cost}(y=a*b+a*/c)=4$
- General (simple, multi-level)
  - $\sum \text{cost}(\text{product-term})$ 
    - e.g.  $\text{nand}2=4, \text{nand}3=5, \text{nand}4=6\dots$

CALTECH CS137 Winter2004 -- DeHon

## Terminology (1)

- Literals --  $a$ ,  $/a$ ,  $b$ ,  $/b$ , ....
  - Qualified, single inputs
- Minterms --
  - full set of literals covering one input case
  - in  $y=a*b+a*c$ 
    - $a*b*c$
    - $a*/b*c$

CALTECH CS137 Winter2004 -- DeHon

## Terminology (2)

- Cube:
  - product covering one or more minterms
  - $Y=a*b+a*c$
  - cubes:
    - $a*b*c$      $abc$
    - $a*b$         $ab$
    - $a*c$         $ac$

CALTECH CS137 Winter2004 -- DeHon

## Terminology (3)

- Cover:
  - set of cubes
  - sum products
  - {abc, a/bc, ab/c}
  - {ab,ac}

CALTECH CS137 Winter2004 -- DeHon

## Truth Table

- Also represent function

a	b	c	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Specify on-set only

a	b	c	y
1	0	1	1
1	1	0	1
1	1	1	1

CALTECH CS137 Winter2004 -- DeHon

# Cube/Logic Specification

- Canonical order for variables
- Use {0,1,-} to indicate input appearance in cube

- 0  $\equiv$  inverted
- 1  $\equiv$  not inverted
- -  $\equiv$  not present

abc 111

a/bc 101

ac 1-1

a b c y

1 0 1 1

1 1 0 1

1 1 1 1

1 0 1 1

1 1 0 1

1 1 1 1

1 - 1 1

1 1 - 1

1 - 1

1 1 -

CALTECH CS137 Winter2004 -- DeHon

# In General

- Three sets:
  - on-set (must be set to one by cover)
  - off-set (must be set to zero by cover)
  - don't care set (can be zero or one)
- Don't Cares
  - allow freedom in covering (reduce cost)
  - arise from cases where value doesn't matter
    - e.g. outputs in non-existent FSM state
    - data bus value when not driving bus

CALTECH CS137 Winter2004 -- DeHon

# Multiple Outputs

Truth Table:

a	b	y	x
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	1

Convert to  
single-output  
problem

a	b	y	x	o
0	0	1	-	1
0	0	-	1	1
0	1	0	-	1
0	1	-	0	1
1	0	0	-	1
1	0	-	0	1
1	1	0	-	1
1	1	-	1	1

On-set  
for result

001-
00-1
010-
01-0
100-
10-0
110-
11-1

CALTECH CS137 Winter2004 -- DeHon

# Multiple Outputs

- Can reduce to single output case
  - write equations on inputs and each output
    - with onset for relation being true
  - after cover
    - remove literals associated with outputs

CALTECH CS137 Winter2004 -- DeHon

## Multiple Outputs

- Could Optimize separately
- By optimizing together
  - Maximize sharing of cubes/product-terms

CALTECH CS137 Winter2004 -- DeHon

## Multiple Outputs

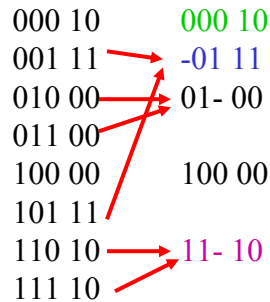
- Consider:
  - $X = a/b + ab + ac$
  - $Y = bc$
- Trivial solution has 4 product terms

000 10		000 10
001 11	→	-01 11
010 00	→	01- 00
011 00	→	
100 00		100 00
101 11		
110 10	→	11- 10
111 10	→	

CALTECH CS137 Winter2004 -- DeHon

## Multiple Outputs

- Consider:
  - $X = \bar{a}/b + ab + ac$
  - $Y = \bar{b}/c$
- Now read off cover:
  - $Y = \bar{b}/c$
  - $A = \bar{a}/b/c + \bar{b}/c + ab$
  - $= \bar{a}/b + \bar{b}/c + ab$



Only need 3 product terms  
(versus 4 w/ no sharing)

CALTECH CS137 Winter2004 -- DeHon

## Prime Implicants

- Implicant -- cube in on-set
  - (not entirely in don't-case set)
- Prime Implicant -- implicant, not contained in any other cube
  - for  $y = a*b + a*c$ 
    - $a*b$  is a prime implicant
    - $a*b*c$  is not a prime implicant (contained in  $ab$ ,  $ac$ )
  - *i.e.* largest cube still in on-set (on+dc-sets)

CALTECH CS137 Winter2004 -- DeHon

# Prime Implicants

- Minimum cover will be made up of primes
  - less products if cover more
  - less literals in prime than contained cubes
- Necessary but not sufficient that minimum cover contain only primes
  - $y=ab+ac+b/c$
  - $y=ac+b/c$
- Number of PI's can be exponential in input size
  - more than minterms, even!
  - Not all PI's will be in optimum cover

CALTECH CS137 Winter2004 -- DeHon

# Restate Goal

- Goal in terms of PIs
  - Find minimum size set of PIs which cover the on-set.

CALTECH CS137 Winter2004 -- DeHon

## Essential Prime Implicants

- Prime Implicant which contains a minterm not covered by any other PI
  - Essential PI **must** occur in any cover
  - $y=ab+ac+b/c$
  - $ab$  11- 110 111
  - $ac$  1-1 101 111 \* essential (only 101)
  - $b/c$  -10 110 010 \* essential (only 010)

CALTECH CS137 Winter2004 -- DeHon

## Computing Primes

- Start with minterms
  - for on-set and dc-set
- merge pairs (distance one apart)
- for each pair merged,
  - mark source cubes as covered
- repeat merging for resulting cube set
  - until no more merging possible
- retain all unmarked cubes which aren't entirely in dc-set

CALTECH CS137 Winter2004 -- DeHon

## Compute Prime Example

0 0000  
5 0101  
7 0111  
8 1000  
9 1001  
10 1010  
11 1011  
14 1110  
15 1111

CALTECH CS137 Winter2004 -- DeHon

## Compute Prime Example

0 0000	0, 8 -000
5 0101	5, 7 01-1
7 0111	7,15 -111
8 1000	8, 9 100-
9 1001	8,10 10-0
10 1010	9,11 10-1
11 1011	10,11 101-
14 1110	10,14 1-10
15 1111	11,15 1-11
	14,15 111-

CALTECH CS137 Winter2004 -- DeHon

## Compute Prime Example

0 0000	0, 8 -000	0, 8 -000	/b/c/d
5 0101	5, 7 01-1	5, 7 01-1	/abd
7 0111	7,15 -111	7,15 -111	bcd
8 1000	8, 9 100-	8, 9 100- * 8, 9,10,11 10--	a/b
9 1001	8,10 10-0	8,10 10-0 *	ac
10 1010	9,11 10-1	9,11 10-1 *	
11 1011	10,11 101-	10,11 101- * 10,11,14,15 1-1-	
14 1110	10,14 1-10	10,14 1-10 *	
15 1111	11,15 1-11	11,15 1-11 *	
	14,15 111-	14,15 111- *	

CALTECH CS137 Winter2004 -- DeHon

## Covering Matrix

- Minterms  $\times$  Prime Implicants

Goal:  
minimum  
cover

	/b/c/d	/abd	bcd	a/b	ac
0000	X				
0101		X			
0111		X	X		
1000	X			X	
1001				X	
1010				X	X
1011				X	X
1110					X
1111			X		X

CALTECH CS137 Winter2004 -- DeHon

# Essential Reduction

- Must pick essential PI
  - pick and eliminate row and column

	/b/c/d	/abd	bcd	a/b	ac
0000	X				
0101		X			
0111		X	X		
1000	X			X	
1001				X	
1010				X	X
1011				X	X
1110					X
1111			X		X

CALTECH CS137 Winter2004 -- DeHon

# Essential Reduction

- This case:
  - Cover determined by essentials
- General case:
  - Reduces size of problem
  - These are easy...

CALTECH CS137 Winter2004 -- DeHon

## Dominators: Column

- If a column (PI) covers the same or strictly more than another column
  - can remove **dominated** column

	B	C	D	E	F	G	H	
0101	X	X						<b>C dominates B</b>
0111		X	X					<b>G dominates H</b>
1000						X	X	
1010					X	X		
1110				X	X			
1111			X	X				

CALTECH CS137 Winter2004 -

## Dominators: Column

- If a column (PI) covers the same or strictly more than another column
  - can remove **dominated** column

	B	C	D	E	F	G	H	
0101	X	X						
0111		X	X					
1000						X	X	
1010					X	X		
1110				X	X			
1111			X	X				

CALTECH CS137 Winter2004 -- DeHon

# New Essentials

- Dominance reduction may yield new Essential PIs

	C	D	E	F	G
0101	X				
0111	X	X			
1000					X
1010				X	X
1110			X	X	
1111	X	X			

C,G now essential

	C	D	E	F	G
1110			X	X	
1111	X	X			

E dominates D and F

Cover = {C,E,G}

CALTECH CS137 Winter2004 -- DeHon

# Dominators: Row

- If a row has the same (or strictly more) PIs than another row, the larger row dominates
  - we can remove the **dominating** row
  - (NOTE OPPOSITE OF COLUMN CASE)

	C	D	E	F	G
0101	X				
0111	X	X			
1000					X
1010				X	X
1110			X	X	
1111	X	X			

0111 dominates 0101  
remove 0111

1010 dominates 1000  
remove 1010

CALTECH CS137 Winter2004 -- DeHon

# Cyclic Core

- After applying reductions
  - essential
  - column dominators
  - row dominators
- May still have a non-trivial covering matrix
- How do we move forward from here?

CALTECH CS137 Winter2004 -- DeHon

# Example

	A	B	C	D	E	F	G	H
0000	X							X
0001	X	X						
0101		X	X					
0111			X	X				
1000							X	X
1010						X	X	
1110					X	X		
1111				X	X			

CALTECH CS137 Winter2004 -- DeHon

# Cyclic Core

- Cannot select (e.g. essential) or exclude (e.g. dominated) a PI definitively.
- Make a guess
  - A in cover
  - A not in cover
- Proceed from there

CALTECH CS137 Winter2004 -- DeHon

# Example

	A	B	C	D	E	F	G	H					
0000	X							X	A in Cover:				
0001	X	X											
0101		X	X						0101	X	X		
0111			X	X					0111		X	X	
1000							X	X	1000			X	X
1010						X	X		1010			X	X
1110				X	X				1110		X	X	
1111			X	X					1111	X	X		

CALTECH CS137 Winter2004 -- DeHon

# Example

	A	B	C	D	E	F	G	H
0000	X							X
0001	X	X						
0101		X	X					
0111			X	X				
1000							X	X
1010						X	X	
1110					X	X		
1111			X	X				

A in Cover:

	B	C	D	E	F	G	H
0101	X	X					
0111		X	X				
1000						X	X
1010					X	X	
1110				X	X		
1111			X	X			

C dominates B  
G dominates H

# Example

	A	B	C	D	E	F	G	H
0000	X							X
0001	X	X						
0101		X	X					
0111			X	X				
1000							X	X
1010						X	X	
1110					X	X		
1111			X	X				

A not in Cover:

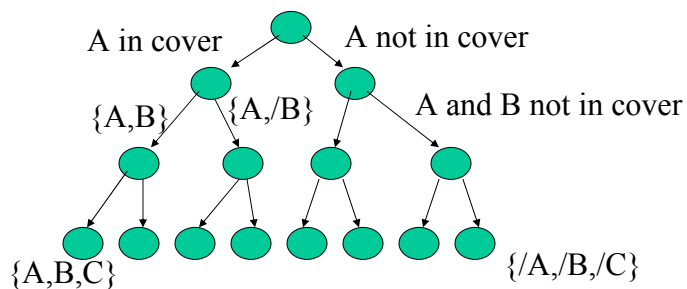
	B	C	D	E	F	G	H
0000							X
0001	X						
0101	X	X					
0111		X	X				
1000						X	X
1010					X	X	
1110				X	X		
1111			X	X			

# Basic Two-Level Minimization

- Generate Prime Implicants
- Reduce (essential, dominators)
- If not done,
  - pick a cube
  - branch (back to reduce) on selected/not
    - *i.e.* search tree ... branch and bound
- Save smallest

CALTECH CS137 Winter2004 -- DeHon

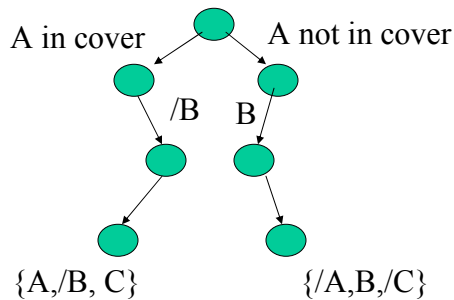
# Branching Search



For N primes, how large?

CALTECH CS137 Winter2004 -- DeHon

## Branching Search w/ Implications



Implications Prune Tree

Only exponential in decision  
where must branch

CALTECH CS137 Winter2004 -- DeHon

## Optimization

- Summarize Minterms (signature cubes)
  - rows represent collection of minterms with same primes
- Avoid generating full set of PIs
  - pre-combining dominators during generation
- Branch-and-bound pruning
  - get lower bound on remaining cost of a cover by computing independent set of primes
    - (not necessarily maximal, that would be NP-hard)

CALTECH CS137 Winter2004 -- DeHon

## Heuristic

- Don't backtrack when select prime for inclusion/exclusion
  - pick cover large set of minterms/signatures
  - weight to select “hard” to cover signatures
- Generate reduced set of PIs
- Iterative improvement

CALTECH CS137 Winter2004 -- DeHon

## Canonical Form

- Can start with *any* form of logical expression
- Get unique truth-table/minterms
- Problem not sensitive to input statement
  - compare covering (decomposition)
  - compare sequential programming languages
- **Cost:** potentially exponential explosion in minterms/PIs

CALTECH CS137 Winter2004 -- DeHon

# Summary

- Formulate as covering problem
- Solution space restricted to PIs
- Essentials must be in solution
- Use dominators to further reduce space
- Then branching/pruning to explore rest of PIs
- Ways to reduce work
  - group minterms/PIs together early
  - mostly fall into this general scheme

CALTECH CS137 Winter2004 -- DeHon

# Admin

- Homework #1 Due Friday
- **TA:** Michael Wrighton  
<wrighton@cs.caltech.edu>
- Monday (19<sup>th</sup>) is a holiday

CALTECH CS137 Winter2004 -- DeHon

# Big Ideas

- Canonical Form
  - eliminate bias of input specification
- Technique:
  - branch-and-bound
  - dominators
  - use structure of problem to derive reduction between branching selection