

CoE/EE460 Switching Theory

Lecture8

John Lockwood

Washington University

Spring 2001

<http://www.arl.wustl.edu/~lockwood/class/coe460/>

Announcements (Continued)

- Exam 1 Reminder
- Row Dominance Example
 - The dominating row is eliminated
- Computation Techniques
 - More Bitwise operations
- Data structures for the Tabular Method
 - Homework Four : Find prime Implicants
 - Organize List of sop_elements
 - Process Matrix of List headers

Row Dominance Example

- M1 dominates M2
- Eliminate M1

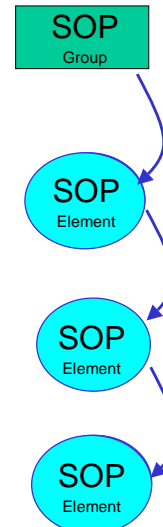
		Prime Implicants	
		$P_1=xy$	$P_2=yz$
Minterms	$M1=xyz$	1	1
	$M2=xyz'$	1	
	$M3=x'yz$		1

More Bit-wise Computational Techniques

- Bit Masking Operation Example
 - Turn on a bit in the kth position
 - In-class example
- C/C++ and 80x86 Assembly Equivalent Operations
 - $X \&= Y \rightarrow \text{AND } X, Y$
 - $X |= Y \rightarrow \text{OR } X, Y$
 - $X \sim= X \rightarrow \text{NOT } X$
- Scanning ASCII Arrays
 - If (linebuffer[0]='0') ...

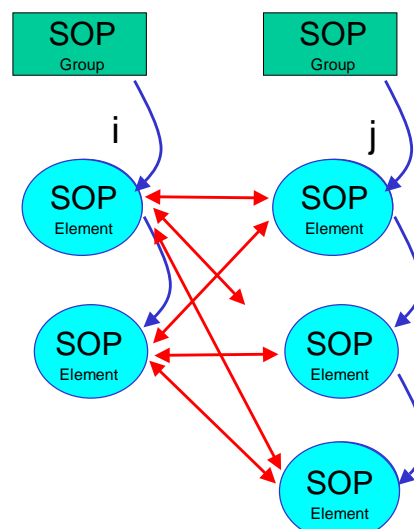
SOP Group

- SOP Group = Collection of SOP Elements
 - Allow scanning of elements
 - Allow insertion of unique members
 - Avoid duplicates
 - Allow modification of group members.



Comparing SOP Groups

- For each $SOP_i \in \text{Group 1}$
For each $SOP_j \in \text{Group 2}$
Compare SOP_i to SOP_j



SOP Group Matrix : In class example

- Consider $f(w,x,y,z)$
 - Four-variable Function

- Arrange columns by Dontcare_count

- $i = \text{one_count}$:
 - $i \in \{0 \dots n\}$
 - $n+1$ columns

- Arrange rows by One_Count

- $j = \text{dontcare_count}$:
 - $j \in \{0 \dots n\}$
 - $n+1$ rows

		DontCare Count				
		DC ₀	DC ₁	DC ₂	DC ₃	DC ₄
One Count	OC ₀	0000	000- 0-00			
	OC ₁	0001 0100	00-1 -001 -100			
	OC ₂	0011 1001 1100				
	OC ₃					
	OC ₄	1111				

SOP Group Matrix : In class example

- For each $DC \in \{0..3\}$
- For each $OC \in \{0 \dots DC-1\}$
 - Compare
 - SOP_Group (DC,OC) to SOP_Group (DC,OC+1)
 - If (Difference=1 term),
 - Insert to SOP_Group (DC+1,OC)
 - Unmark Primes

		DontCare Count				
		DC ₀	DC ₁	DC ₂	DC ₃	DC ₄
One Count	OC ₀	0000	000- 0-00			
	OC ₁	0001 0100	00-1 -001 -100			
	OC ₂	0011 1001 1100				
	OC ₃					
	OC ₄	1111				

Extending program to find Prime Implicants

```
class sop_term {  
public:  
    // Homework 3 structure  
    unsigned int value; // up to 32-bits: 0=false, 1=true  
    unsigned int mask; // up to 32-bits: 0=Care, 1=DontCare  
    // Homework 4 extensions  
    int one_count;  
    int dontcare_count;  
    int is_prime;  
    sop_term* next;  
};
```

Your structures may vary!