Digital Logic Design

Week 3

Gate-Level Minimization

Outline

- The Map Method
- 2,3,4 variable maps
- 5 and 6 variable maps (very briefly)
- Product of sums simplification
- Don't Care conditions
- NAND and NOR implementation
- Other 2-level implementation
- Hardware Description language (HDL)

The Map Method

- After constructing the map. We mark the squares whose minterms.
- Any two adjacent squares in the map differ by only one variable, primes in one square and unprimed in the other.
- The sum of the elements in these 2 squares, can be simplified to an and gates that does not contain that literal.
- The more adjacent squares we combine them together, the simple the term will be.

2-variable map

Y X	0	1
0	m_0	m_1
1	m_2	m_3

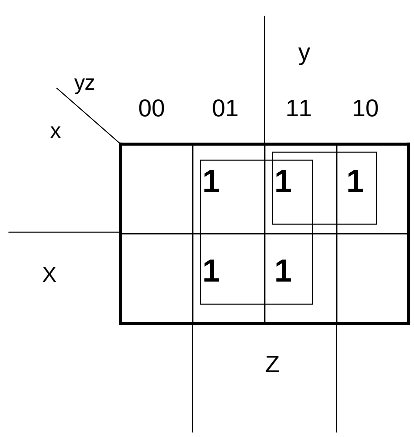
3-variable map

x \ yz	00	01	11	10
	x'y'z' m_0	x'y'z m_1	x'yz m_3	x'yz' m_2
	xy'z' m_4	xy'z m_5	xyz m_7	xyz' m_6

y (columns 11, 10); Z (columns 01, 11)

3-variable map

F=X'Z+X'Y+XY'Z+YZ



F=Z+X'Y

4-variable map

Y

wx \ yz	00	01	11	10
00	w’x’y’z’ m_0	w’x’y’z m_1	w’x’yz m_3	w’x’yz’ m_2
01	w’xy’z’ m_4	w’xy’z m_5	w’xyz m_7	w’xyz’ m_6
11	wxy’z’ m_{12}	wxy’z m_{13}	wxyz’ m_{15}	wxyz’ m_{14}
10	wx’y’z’ m_8	wx’y’z m_9	wx’yz m_{11}	wx’yz’ m_{10}

X

W

Z

4-variable map

Y

wx \ yz	00	01	11	10
00	**1**	**1**		**1**
01	**1**	**1**		**1**
11	**1**	**1**		**1**
10	**1**	**1**		

X

W

Z

5-Variable Map

- Mention an example for 5 and 6 very briefly,
- Too complicated

Prime Implicants

- **prime Implicant**: is a product term obtained by combining together the maximum possible number of adjacent squares in the map.
- **Essential prime implicant**: if a minterm in the map is covered by only one prime implicant, that prime implicant is called an essential prime implicant.

Prime Implicant

- The procedure of finding the simplified expression from the map is as follows:

1. First, determine all the essential prime implicants.
2. The simplified expression is obtained by combining all the essential prime implicants
3. After that add other prime implicants that may be needed to cover any remaining minterms that was not covered by essential prime implicants.

Example

F=Σ(0,2,3,5,7,8,9,10,11,13,15)

Y

wx \ yz	00	01	11	10
00	**1**		1	**1**
01		**1**	**1**	
11		**1**	**1**	
10	**1**	1	1	**1**

X

W

Z

Example

F=Σ(0,2,3,5,7,8,9,10,11,13,15)

wx \ yz	00	01	11	10
00	1		1	1
01		1	1	
11		1	1	
10	1	1	1	1

Y, X, W, Z

Product of Sum Simplification

- F=A+DB+C’A
- Using maxterm
- F=(A+D)(A+B)
- F=A+AB+DA+DB
- O.K.

Product of Sums Simplification

- Simplify as PoS

wx \ yz	00	01	11	10
00	**1**	**1**	0	**1**
01	**0**	**1**	**0**	**0**
11	**0**	**0**	**0**	**0**
10	**1**	1	0	**1**

Y X W Z

PoS

- $F(A,B,C)=\Sigma(0,1,2,5,8,9,10)$
- F=B'D' + B'C' + A'C'D
- F'=AB + CD + BD'
- Taking the complement
- F=(AB)' (CD)' (BD')'
- F = (A' + B')(C' + D')((B' +D)

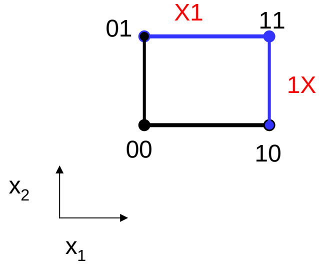
SoP and PoS

Don’t Care Conditions

- Used as 1 or zero to simplify the design

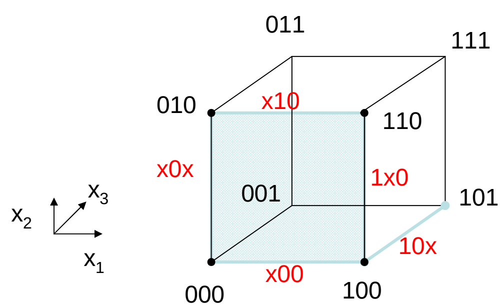
Cubical Representation

- 2-D cube



x_1	x_2	f
0	0	0
0	1	1
1	0	1
1	1	1

Cubical Representation



Generation pf prime implicants

- We use $x_i\ x_j + x_i\ \overline{x}_j = x_i$
- Larger cubes can be formed only from minterms that differ in just one variables.
- We have to compare every 2 minterms.
- In order to reduce the number of comparisons, group the minterms and order them.

Generation pf prime implicants

- Place the minterms in groups such that each group has the same number of 1's
- Sort the groups by the number of 1's
- Combine together any 2 minterms that differ by only 1 position (that position becomes x).
- Every cube that included in a larger cube is checked.
- Repeat until no more inclusion
- The unchecked groups are the prime implicants

Generation pf prime implicants