# EECS150

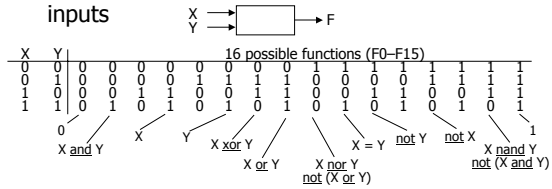
Section 1

Introduction to Combinational Logic

Fall 2001

# 2-Variable Logic Functions

- There are 16 possible functions of 2 input variables:
  - in general, there are 2**(2**n) functions of n inputs






# Boolean algebra

- Boolean algebra
  - B = {0, 1}
  - + is logical OR, • is logical AND
  - ' is logical NOT
- All algebraic axioms hold



# An algebraic structure

- An algebraic structure consists of
  - a set of elements B
  - binary operations { + , • }
  - and a unary operation { ' }
  - such that the following axioms hold:

1. set B contains at least two elements, a, b, such that $a \neq b$
2. closure: $a + b$ is in B; $a \bullet b$ is in B
3. commutativity: $a + b = b + a$; $a \bullet b = b \bullet a$
4. associativity: $a + (b + c) = (a + b) + c$; $a \bullet (b \bullet c) = (a \bullet b) \bullet c$
5. identity: $a + 0 = a$; $a \bullet 1 = a$
6. distributivity: $a + (b \bullet c) = (a + b) \bullet (a + c)$; $a \bullet (b + c) = (a \bullet b) + (a \bullet c)$
7. complementarity: $a + a' = 1$; $a \bullet a' = 0$



# Axioms and Theorems

- Identity
  1. $X + 0 = X$ — 1D. $X \bullet 1 = X$
- Null
  2. $X + 1 = 1$ — 2D. $X \bullet 0 = 0$
- Idempotency:
  3. $X + X = X$ — 3D. $X \bullet X = X$
- Involution:
  4. $(X')' = X$
- Complementarity:
  5. $X + X' = 1$ — 5D. $X \bullet X' = 0$
- Commutativity:
  6. $X + Y = Y + X$ — 6D. $X \bullet Y = Y \bullet X$
- Associativity:
  7. $(X + Y) + Z = X + (Y + Z)$ — 7D. $(X \bullet Y) \bullet Z = X \bullet (Y \bullet Z)$



# Axioms and theorems

- Distributivity:
  8. $X \bullet (Y + Z) = (X \bullet Y) + (X \bullet Z)$ — 8D. $X + (Y \bullet Z) = (X + Y) \bullet (X + Z)$
- Uniting:
  9. $X \bullet Y + X \bullet Y' = X$ — 9D. $(X + Y) \bullet (X + Y') = X$
- Absorption:
  10. $X + X \bullet Y = X$ — 10D. $X \bullet (X + Y) = X$
  11. $(X + Y') \bullet Y = X \bullet Y$ — 11D. $(X \bullet Y') + Y = X + Y$
- Factoring:
  12. $(X + Y) \bullet (X' + Z) = X \bullet Z + X' \bullet Y$ — 12D. $X \bullet Y + X' \bullet Z = (X + Z) \bullet (X' + Y)$
- Concensus:
  13. $(X \bullet Y) + (Y \bullet Z) + (X' \bullet Z) = X \bullet Y + X' \bullet Z$ — 13D. $(X + Y) \bullet (Y + Z) \bullet (X' + Z) = (X + Y) \bullet (X' + Z)$

# Axioms and theorems

- de Morgan's:
  14. (X + Y + ...)' = X' • Y' • ...  14D. (X • Y • ...)' = X' + Y' + ...
- generalized de Morgan's:
  15. f'(X1,X2,...,Xn,0,1,+,•) = f(X1',X2',...,Xn',1,0,•,+)
- establishes relationship between • and +

| A | B | F | $\overline{A}$ | $\overline{B}$ | $\overline{A}+\overline{B}$ | $\overline{F}$ | $\overline{\overline{A}+\overline{B}}$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |



# Axioms and theorems

- Duality
  - Dual of a Boolean expression is derived by replacing • by +, + by •, 0 by 1, and 1 by 0, and leaving variables unchanged
  - Any theorem that can be proven is thus also proven for its dual!
  - Meta-theorem (a theorem about theorems)
- duality:
  16. X + Y + ... ⇔ X • Y • ...
- generalized duality:
  17. f (X1,X2,...,Xn,0,1,+,•) ⇔ f(X1,X2,...,Xn,1,0,•,+)
- Different than deMorgan's Law
  - this is a statement about theorems
  - this is not a way to manipulate (re-write) expressions



# Proving theorems (rewriting)

- Using the axioms of Boolean algebra:
  - e.g., prove the theorem: X • Y + X • Y' = X

| | | | |
|---|---|---|---|
| distributivity (8) | X · Y + X · Y' | = | X · (Y + Y') |
| complementarity (5) | X · (Y + Y') | = | X · (1) |
| identity (1D) | X · (1) | = | X ✓ |

  - e.g., prove the theorem: X + X • Y = X

| | | | |
|---|---|---|---|
| identity (1D) | X + X · Y | = | X · 1 + X · Y |
| distributivity (8) | X · 1 + X · Y | = | X · (1 + Y) |
| identity (2) | X · (1 + Y) | = | X · (1) |
| identity (1D) | X · (1) | = | X ✓ |



# Proving theorems (perfect induction)

- Using perfect induction (complete truth table):
  - e.g., de Morgan's:

(X + Y)' = X' · Y'
NOR is equivalent to AND with inputs complemented

| X | Y | X' | Y' | (X + Y)' | X' · Y' |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 |

(X · Y)' = X' + Y'
NAND is equivalent to OR with inputs complemented

| X | Y | X' | Y' | (X · Y)' | X' + Y' |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 |



# Logic Functions

- Any logic function that can be expressed as a truth table can be written as an expression in Boolean algebra using the operators: ', +, and •

| X | Y | X · Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

| X | Y | X' | X' · Y |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 |

| X | Y | X' | Y' | X · Y | X' · Y' | (X · Y) + (X' · Y') |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 |

(X · Y) + (X' · Y') = X = Y

Boolean expression that is true when the variables X and Y have the same value and false, otherwise

X, Y are Boolean algebra variables



# Cost of different logic functions

- Different functions are easier or harder to implement
  - Each has a cost associated with the number of switches needed
  - 0 (F0) and 1 (F15): require 0 switches, directly connect output to low/high
  - X (F3) and Y (F5): require 0 switches, output is one of inputs
  - X' (F12) and Y' (F10): require 2 switches for "inverter" or NOT-gate
  - X nor Y (F4) and X nand Y (F14): require 4 switches
  - X or Y (F7) and X and Y (F1): require 6 switches
  - X = Y (F9) and X ⊕ Y (F6): require 16 switches
  - Because NOT, NOR, and NAND are the cheapest they are the functions we implement the most in practice

# A simple example

- 1-bit binary adder
  - inputs: A, B, Carry-in
  - outputs: Sum, Carry-out


| A | B | Cin | S | Cout |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

S = A' B' Cin + A' B Cin' + A B' Cin' + A B Cin

Cout = A' B Cin + A B' Cin + A B Cin' + A B Cin



# Apply theorems to simplify

- The theorems of Boolean algebra can simplify Boolean expressions
  - e.g., full adder's carry-out function (same rules apply to any function)

```
Cout  = A' B Cin + A B' Cin + A B Cin' + A B Cin
      = A' B Cin + A B' Cin + A B Cin' + A B Cin + A B Cin
      = A' B Cin + A B Cin + A B' Cin + A B Cin' + A B Cin
      = (A' + A) B Cin + A B' Cin + A B Cin' + A B Cin
      = (1) B Cin + A B' Cin + A B Cin' + A B Cin
      = B Cin + A B' Cin + A B Cin' + A B Cin + A B Cin
      = B Cin + A B' Cin + A B Cin + A B Cin' + A B Cin
      = B Cin + A (B' + B) Cin + A B Cin' + A B Cin
      = B Cin + A (1) Cin + A B Cin' + A B Cin
      = B Cin + A Cin + A B (Cin' + Cin)
      = B Cin + A Cin + A B (1)
      = B Cin + A Cin + A B
```

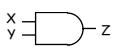


# From expressions to logic gates

- NOT X' $\overline{X}$ ~X


| X | Y |
|---|---|
| 0 | 1 |
| 1 | 0 |

- AND X • Y XY X ∧ Y




| X | Y | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

- OR X + Y X ∨ Y


| X | Y | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |



# From expressions to logic gates

- NAND


| X | Y | Z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

- NOR


| X | Y | Z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

- XOR X ⊕ Y


| X | Y | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

X <u>xor</u> Y = X Y' + X' Y
X or Y but not both
("inequality", "difference")

- XNOR X = Y

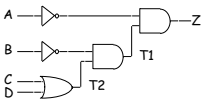

| X | Y | Z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

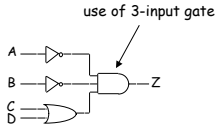
X <u>xnor</u> Y = X Y + X' Y'
X and Y are the same
("equality", "coincidence")



# From expressions to logic gates

- More than one way to map expressions to gates
  - e.g., Z = A' • B' • (C + D) = (A' • (B' • $\overbrace{(C + D)}^{T2}$))) $\overbrace{\phantom{B' \bullet (C+D)}}^{T1}$

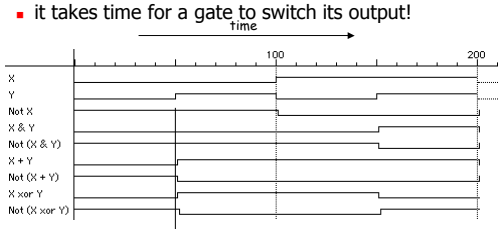





# Waveform view of logic functions

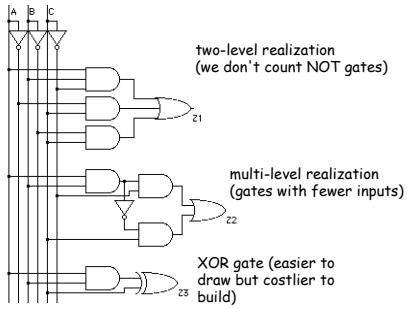
- Just a sideways truth table
  - but note how edges don't line up exactly
  - it takes time for a gate to switch its output!




change in Y takes time to "propagate" through gates

# Different realizations of a function

| A | B | C | Z |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |






# Which realization is best?

- Reduce number of inputs
  - literal: input variable (complemented or not)
    - can approximate cost of logic gate as 2 transistors per literal
    - why not count inverters?
  - Fewer literals means less transistors
    - smaller circuits
  - Fewer inputs implies faster gates
    - gates are smaller and thus also faster
  - Fan-ins (# of gate inputs) are limited in some technologies
- Reduce number of gates
  - Fewer gates (and the packages they come in) means smaller circuits
    - directly influences manufacturing costs



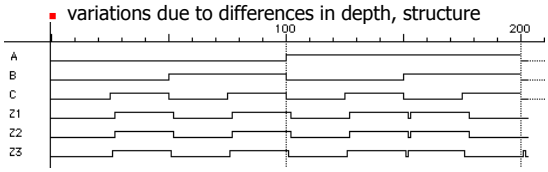
# Which is the best realization?

- Reduce number of levels of gates
  - Fewer level of gates implies reduced signal propagation delays
  - Minimum delay configuration typically requires more gates
    - wider, less deep circuits
- How do we explore tradeoffs between increased circuit delay and size?
  - Automated tools to generate different solutions
  - Logic minimization: reduce number of gates and complexity
  - Logic optimization: reduction while trading off against delay



# Are all realizations equivalent?

- Under the same input stimuli, the three alternative implementations have almost the same waveform behavior
  - delays are different
  - glitches (hazards) may arise
  - variations due to differences in depth, structure

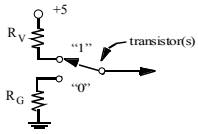





# Timing

- Two key concepts: what {0,1} and when (timing)
  - How fast can I move or process information?
  - Fundamental limitations:
    - speed of light $3 \times 108$ m/sec vacuum
      $\sim 2 \times 108$ m/sec conduction
    - switching time: capacitance & inductance

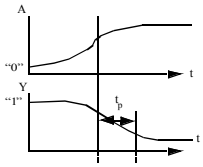
Zero$^{th}$ order model



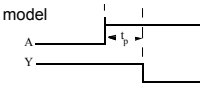



# Voltage model (more physical)

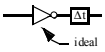

(as seen with oscilloscope)




4.3 Delay model




or



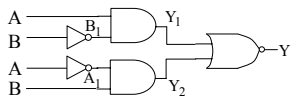
# Timing

- Why big deal?
- Dynamic output ≠ static output ("glitches")

Example: XOR function implemented from AND, OR, INV

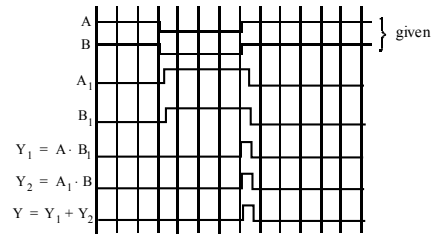
*idealization*: unit delay in each symbol

$$Y = A \cdot \overline{B} + \overline{A} \cdot B$$






# Timing Diagram




Static behavior fine $1 \oplus 1 = 0, 0 \oplus 0 = 0$ ✓

Dynamic behavior glitch → fundamental feature of switching



# Implementing Boolean functions

- Technology independent
  - Canonical forms
  - Two-level forms
  - Multi-level forms
- Technology choices
  - Packages of a few gates
  - Regular logic
  - Two-level programmable logic
  - Multi-level programmable logic



# Canonical forms

- Truth table is the unique signature of a Boolean function
- Many alternative gate realizations may have the same truth table
- Canonical forms
  - Standard forms for a Boolean expression
  - Provides a unique algebraic signature



# Sum-of-products canonical forms

- Also known as disjunctive normal form
- Also known as minterm expansion

F = *001 011 101 110 111*

F = A'B'C+ A'BC+ AB'C + ABC' + ABC

| A | B | C | F | F' |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 |

F' = A'B'C' + A'BC' + AB'C'



# Sum-of-products canonical form

- Product term (or minterm)
  - ANDed product of literals – input combination for which output is true
  - Each variable appears exactly once, in true or inverted form (but not both)

| A | B | C | minterms | |
|---|---|---|---|---|
| 0 | 0 | 0 | A'B'C' | m0 |
| 0 | 0 | 1 | A'B'C | m1 |
| 0 | 1 | 0 | A'BC' | m2 |
| 0 | 1 | 1 | A'BC | m3 |
| 1 | 0 | 0 | AB'C' | m4 |
| 1 | 0 | 1 | AB'C | m5 |
| 1 | 1 | 0 | ABC' | m6 |
| 1 | 1 | 1 | ABC | m7 |

short-hand notation for minterms of 3 variables

F in canonical form:

F(A, B, C) = Σm(1,3,5,6,7)
= m1 + m3 + m5 + m6 + m7
= A'B'C + A'BC + AB'C + ABC' + ABC

canonical form ≠ minimal form

F(A, B, C) = A'B'C + A'BC + AB'C + ABC + ABC'
= (A'B' + A'B + AB' + AB)C + ABC'
= ((A' + A)(B' + B))C + ABC'
= C + ABC'
= ABC' + C
= AB + C

# Product-of-sums canonical form

- Also known as conjunctive normal form
- Also known as maxterm expansion

F = 000 010 100
F = (A + B + C) (A + B' + C) (A' + B + C)

| A | B | C | F | F' |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 |

F' = (A + B + C') (A + B' + C') (A' + B + C') (A' + B' + C) (A' + B' + C')



# Product-of-sums canonical form

- Sum term (or maxterm)
  - ORed sum of literals – input combination for which output is false
  - each variable appears exactly once, in true or inverted form (but not both)

| A | B | C | maxterms | |
|---|---|---|---|---|
| 0 | 0 | 0 | A+B+C | M0 |
| 0 | 0 | 1 | A+B+C' | M1 |
| 0 | 1 | 0 | A+B'+C | M2 |
| 0 | 1 | 1 | A+B'+C' | M3 |
| 1 | 0 | 0 | A'+B+C | M4 |
| 1 | 0 | 1 | A'+B+C' | M5 |
| 1 | 1 | 0 | A'+B'+C | M6 |
| 1 | 1 | 1 | A'+B'+C' | M7 |

short-hand notation for maxterms of 3 variables

F in canonical form:

$F(A, B, C) = \Pi M(0,2,4)$
$= M0 \cdot M2 \cdot M4$
$= (A + B + C)(A + B' + C)(A' + B + C)$

canonical form ≠ minimal form

$F(A, B, C) = (A + B + C)(A + B' + C)(A' + B + C)$
$= (A + B + C)(A + B' + C)$
$(A + B + C)(A' + B + C)$
$= (A + C)(B + C)$

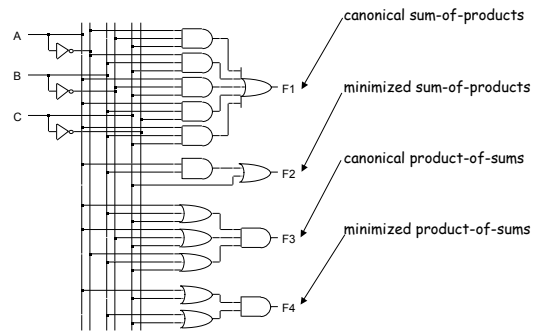


# Relating S-o-P & P-o-S

- Sum-of-products
  - F' = A'B'C' + A'BC' + AB'C'
- Apply de Morgan's
  - (F')' = (A'B'C' + A'BC' + AB'C')'
  - F = (A + B + C) (A + B' + C) (A' + B + C)
- Product-of-sums
  - F' = (A + B + C') (A + B' + C') (A' + B + C') (A' + B' + C) (A' + B' + C')
- Apply de Morgan's
  - (F')' = ( (A + B + C')(A + B' + C')(A' + B + C')(A' + B' + C)(A' + B' + C') )'
  - F = A'B'C + A'BC + AB'C + ABC' + ABC



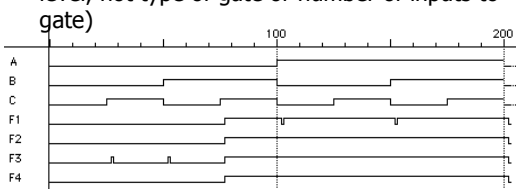
# Example: F = AB + C






# Waveforms

- Waveforms are essentially identical
  - Except for timing hazards (glitches)
  - Delays almost identical (modeled as a delay per level, not type of gate or number of inputs to gate)






# Mapping

- Minterm to maxterm conversion
  - Use maxterms whose indices do not appear in minterm expansion
  - e.g., F(A,B,C) = Σm(1,3,5,6,7) = ΠM(0,2,4)
- Maxterm to minterm conversion
  - Use minterms whose indices do not appear in maxterm expansion
  - e.g., F(A,B,C) = ΠM(0,2,4) = Σm(1,3,5,6,7)
- Minterm expansion of F to minterm expansion of F'
  - Use minterms whose indices do not appear
  - e.g., F(A,B,C) = Σm(1,3,5,6,7) F'(A,B,C) = Σm(0,2,4)
- Maxterm expansion of F to maxterm expansion of F'
  - Use maxterms whose indices do not appear
  - e.g., F(A,B,C) = ΠM(0,2,4) F'(A,B,C) = ΠM(1,3,5,6,7)

# Incompletely specified functions

- Example: binary coded decimal increment by 1
  - BCD digits encode decimal digits 0 – 9 in bit patterns 0000 – 1001

| A | B | C | D | W | X | Y | Z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | X | X | X | X |
| 1 | 0 | 1 | 1 | X | X | X | X |
| 1 | 1 | 0 | 0 | X | X | X | X |
| 1 | 1 | 0 | 1 | X | X | X | X |
| 1 | 1 | 1 | 0 | X | X | X | X |
| 1 | 1 | 1 | 1 | X | X | X | X |

off-set of W

on-set of W

don't care (DC) set of W

these inputs patterns should never be encountered in practice – "don't care" about associated output values, can be exploited in minimization



# Notation

- Don't cares and canonical forms
  - So far, only represented on-set
  - Also represent don't-care-set
  - Need two of the three sets (on-set, off-set, dc-set)
- Canonical representations of the BCD increment by 1 function:
  - Z = m0 + m2 + m4 + m6 + m8 + d10 + d11 + d12 + d13 + d14 + d15
  - Z = $\Sigma$ [ m(0,2,4,6,8) + d(10,11,12,13,14,15) ]
  - Z = M1 • M3 • M5 • M7 • M9 • D10 • D11 • D12 • D13 • D14 • D15
  - Z = $\Pi$ [ M(1,3,5,7,9) • D(10,11,12,13,14,15) ]