Module 2

Combinational Logic: Supplemental Slides

Louis G. Johnson
School of Electrical and Computer Engineering
Oklahoma State University

Fall 2007

Cost of different logic functions

- Different functions are easier or harder to implement
  - each has a cost associated with the number of “switches” needed
  - this also corresponds to the relative electrical complexity (transistors and other electrical components) to implement a logic function
  - NOR and NAND turn out to be simpler to implement than OR and AND, so NOR and NAND (along with NOT) are used more in practice since “complexity” relates to “cost”
  - in class, we more often use OR and AND (along with NOT) because those functions are easier for most people to work with mentally

Minimal set of functions

- Can we implement all logic functions from NOT, NOR, and NAND?
  - For example, implementing *X and Y*
    is the same as implementing *not (X nand Y)*
- In fact, we can do it with only NOR or only NAND
  - NOT is just a NAND or a NOR with both inputs tied together

X	Y	X nor Y
0	0	1
1	1	0

X	Y	X nand Y
0	0	1
1	1	0

  - and NAND and NOR are "duals",
    that is, it's easy to implement one using the other

X nand Y ≡ not ((not X) nor (not Y))

X nor Y ≡ not ((not X) nand (not Y))

- NAND and NOR are sometimes referred to as *functionally complete* or just *complete* since either NAND or NOR can be used to represent an logical relationship (since either can make AND, OR and NOT)

Proving theorems (rewriting)

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

distributivity (8)	$X \bullet Y + X \bullet Y'$	=	$X \bullet (Y + Y')$
complementarity (5)	$X \bullet (Y + Y')$	=	$X \bullet (1)$
identity (1D)	$X \bullet (1)$	=	X ➽

  - e.g., prove the theorem: $X + X \bullet Y = X$

identity (1D)	$X + X \bullet Y$	=	$X \bullet 1 + X \bullet Y$
distributivity (8)	$X \bullet 1 + X \bullet Y$	=	$X \bullet (1 + Y)$
identity (2)	$X \bullet (1 + Y)$	=	$X \bullet (1)$
identity (1D)	$X \bullet (1)$	=	X ➽

Activity

- Prove the following using the laws of Boolean algebra:
  - $(X \cdot Y) + (Y \cdot Z) + (X' \cdot Z) = X \cdot Y + X' \cdot Z$

	$(X \bullet Y) + (Y \bullet Z) + (X' \bullet Z)$
identity	$(X \bullet Y) + (1) \bullet (Y \bullet Z) + (X' \bullet Z)$
complementarity	$(X \bullet Y) + (X' + X) \bullet (Y \bullet Z) + (X' \bullet Z)$
distributivity	$(X \bullet Y) + (X' \bullet Y \bullet Z) + (X \bullet Y \bullet Z) + (X' \bullet Z)$
commutativity	$(X \bullet Y) + (X \bullet Y \bullet Z) + (X' \bullet Y \bullet Z) + (X' \bullet Z)$
factoring	$(X \bullet Y) \bullet (1 + Z) + (X' \bullet Z) \bullet (1 + Y)$
null	$(X \bullet Y) \bullet (1) + (X' \bullet Z) \bullet (1)$
identity	$(X \bullet Y) + (X' \bullet Z)$ ➻

Proving theorems (perfect induction)

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

$(X + Y)' = X' \bullet Y'$
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

$(X \bullet Y)' = X' + Y'$
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

Activity

- Fill in the truth-table for a circuit that checks that a 4-bit number is divisible by 2, 3, or 5

X8	X4	X2	X1	By2	By3	By5
0	0	0	0	1	1	1
0	0	0	1	0	0	0
0	0	1	0	1	0	0
0	0	1	1	0	1	0

- Write down Boolean expressions for By2, By3, and By5

Activity

X8	X4	X2	X1	By2	By3	By5
0	0	0	0	1	1	1
0	0	0	1	0	0	0
0	0	1	0	1	0	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	1	1	0
0	1	1	1	0	0	0
1	0	0	0	1	0	0
1	0	0	1	0	1	0
1	0	1	0	1	0	1
1	0	1	1	0	0	0
1	1	0	0	1	1	0
1	1	0	1	0	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	1

By2 = X8′X4′X2′X1′ + X8′X4′X2X1′
+ X8′X4X2′X1′ + X8′X4X2X1′
+ X8X4′X2′X1′ + X8X4′X2X1′
+ X8X4X2′X1′ + X8X4X2X1′
= X1′

By3 = X8′X4′X2′X1′ + X8′X4′X2X1
+ X8′X4X2X1′ + X8X4′X2′X1
+ X8X4X2′X1′ + X8X4X2X1

By5 = X8′X4′X2′X1′ + X8′X4X2′X1
+ X8X4′X2X1′ + X8X4X2X1