**CSE 370 Spring 2006**  
Introduction to Digital Design  
Lecture 28: Final Review

**Last Lecture**  
- FGPs

**Today**  
- The Final  
- Course Evaluations

**Administrivia**
- Turn in HW #9

**Grading**
- 30% Homework
- 20% Labs
- 20% Quizzes
- 30% Final exam

**Homework distributions**
- Average=83%, Median=87.5%, StdDev=14%
Quiz Distributions
Average=92.7%, Median=98.3%, StdDev=15%

Lab Distribution
Average=96.3%, Median=100%, StdDev=8.5%

Partial Total Score Distribution
Average=89.6%, Median=93.2%, StdDev=10.7%

Final Exam Format

- 1 hour, 45 minutes
- Closed book, closed notes
- Answer written on exam
- You may bring extra sheets of blank scratch paper
- Monday, 8:30-10:20 a.m. in 231 Mary Gates Hall
Main Course Outline

Combinational Design
- Number systems (sign and magnitude, one’s complement, two’s complement)
- Boolean logic, Boolean formulas
- Canonical forms, Karnaugh maps, minimization, NAND-NOR implementations
- Programmable logic
- Adders, ALUs

Sequential Design
- Latches and Flip-flops
- Registers
- Timing methodologies
- Finite state machines
- FSM state optimization, state encoding

Hardware Description Languages
- Verilog

Twos-Complement Example

test your skills convert $1_{10}$ and $-5_{10}$ to 4 bit twos-complement binary and then add them

$$1_{10} = \quad \_ \_ \_ \_$$
$$-5_{10} = \quad \_ \_ \_ \_$$

Axioms and Theorems

1. Identity: $X + 0 = X$
   Dual: $X \cdot 1 = X$

2. Null: $X + 1 = 1$
   Dual: $X \cdot 0 = 0$

3. Idempotent: $X + X = X$
   Dual: $X \cdot X = X$

4. Involution: $(X')' = X$

5. Complementarity: $X + X' = 1$
   Dual: $X \cdot X' = 0$

6. Commutative: $X + Y = Y + X$
   Dual: $X \cdot Y = Y \cdot X$

7. Associative: $(X+Y)+Z=X+(Y+Z)$
   Dual: $(X\cdot Y)\cdot Z=X\cdot(Y \cdot Z)$

8. Distributive: $X\cdot(Y+Z)=(X\cdot Y)+(X\cdot Z)$
   Dual: $X+(Y\cdot Z)=(X+Y)\cdot(X+Z)$

9. Uniting: $X\cdot Y + X\cdot Y' = X$
   Dual: $(X+Y)(X+Y') = X$

10. Absorption: $X + X \cdot Y = X$
    Dual: $X \cdot (X + Y) = X$

11. Absorption 2: $(X + Y') \cdot Y = X \cdot Y$
    Dual: $(X \cdot Y') + Y = X + Y$

12. Factoring: $(X + Y) \cdot (X' + Z) = X \cdot Z + X' \cdot Y$
    Dual: $X \cdot Y + X' \cdot Z = (X + Z) \cdot (X' + Y)$

Axioms and Theorems

13. Concensus: $(X \cdot Y) + (Y \cdot Z) + (X' \cdot Z) = X \cdot Y + X' \cdot Z$
    Dual: $(X + Y) \cdot (Y + Z) \cdot (X' + Z) = (X + Y) \cdot (X' + Z)$

14. DeMorgan’s Law: $(X + Y + ...)' = X' \cdot Y' \cdot ...$
    Dual: $(X \cdot Y \cdot ...)' = X' + Y' + ...$

15. Generalized DeMorgan’s Laws: $f'(X_1,X_2,...,X_n,0,1,+,•) = f(X_1',X_2',...,X_n',1,0,•,+)$
    Notice the DeMorgan is not Duality: Duality is not a way to rewrite an expression, it is a meta-theorem.

16. Generalized Duality:
    $f(X_1,X_2,...,X_n,0,1,+,•) \leftrightarrow f(X_1,X_2,...,X_n,1,0,•,+)$
Boolean Logic

- Example 3: Prove the consensus theorem--
  \((XY)+(YZ)+(X'Z) = XY+X'Z\)

Exercise

- Example 3: Prove the consensus theorem--
  \((XY)+(YZ)+(X'Z) = XY+X'Z\)

  Complementarity
  \(XY+YZ+X'Z = XY+(X+X')YZ + X'Z\)

  Distributive
  \(= XY+X'YZ+X'Z\)

  Use absorption \(\{AB+A=A\}\) with \(A=XY\) and \(B=Z\)

  Rearrange terms
  \(= XY+X'ZY+X'Z\)

  Use absorption \(\{AB+A=A\}\) with \(A=X'Z\) and \(B=Y\)

  \[XY+YZ+X'Z = XY+X'Z,\]

Sum of Products Canonical Form

- Also called disjunctive normal form (DNF)
  - Commonly called a minterm expansion

K-map Exercise

- Minimize the function
  \[F = \Sigma m(0, 2, 7, 8, 14, 15) + d(3, 6, 9, 12, 13)\]
Exercise

- Implementing the following function as a 4:1 mux

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Example: Sequence detector

- Design a circuit to detect 3 or more 1’s in a bit string
  - Assume Moore machine
  - Assume D flip-flops
  - Assume flip-flops have a reset

1. State diagram and state-transition table

<table>
<thead>
<tr>
<th>reset</th>
<th>state</th>
<th>input</th>
<th>next state</th>
<th>current output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>0</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>A</td>
<td>0</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>A</td>
<td>1</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>B</td>
<td>0</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>B</td>
<td>1</td>
<td>C</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>C</td>
<td>0</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>C</td>
<td>1</td>
<td>D</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>D</td>
<td>0</td>
<td>A</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>D</td>
<td>1</td>
<td>D</td>
<td>1</td>
</tr>
</tbody>
</table>

2. State minimization & 3. State encoding

- State diagram is already minimized
- Try a binary encoding
4. Minimize next-state logic

\[
\begin{array}{c|c|c}
\text{In} & \text{M} & \text{L} \\
0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 \\
\end{array}
\]
\[
\begin{array}{c|c|c}
\text{In} & \text{M} & \text{L} \\
0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 \\
\end{array}
\]
\[
\begin{array}{c|c|c}
\text{In} & \text{M} & \text{L} \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
\end{array}
\]

\[
\text{MSB}^+ = \text{LIn} + \text{MIn} \\
\text{LSB}^+ = \text{L}'\text{In} + \text{MIn} \\
\text{Out}^+ = \text{ML}
\]

Notation
\[
\begin{align*}
\text{M} & := \text{MSB} \\
\text{L} & := \text{LSB} \\
\text{In} & := \text{Input}
\end{align*}
\]

5. Implement the design

\[
\text{MSB}^+ = \text{LIn} + \text{MIn}
\]
\[
\text{LSB}^+ = \text{L}'\text{In} + \text{MIn}
\]

\[
\begin{array}{c}
\text{Out}^+ = \text{ML}
\end{array}
\]