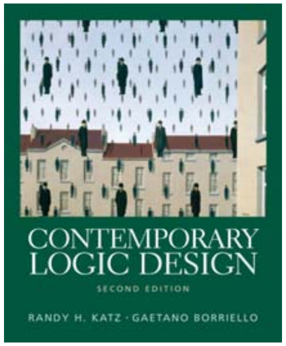
CSE 370 Spring 2006
Introduction to Digital Design

Lecture 28: Final Review



Last Lecture

- FGPAs

Today

- The Final
- Course Evaluations

Administrivia

- Turn in HW #9

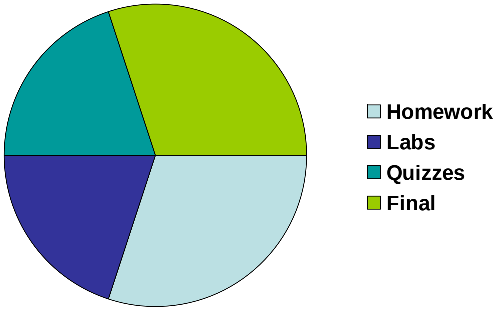
Grading

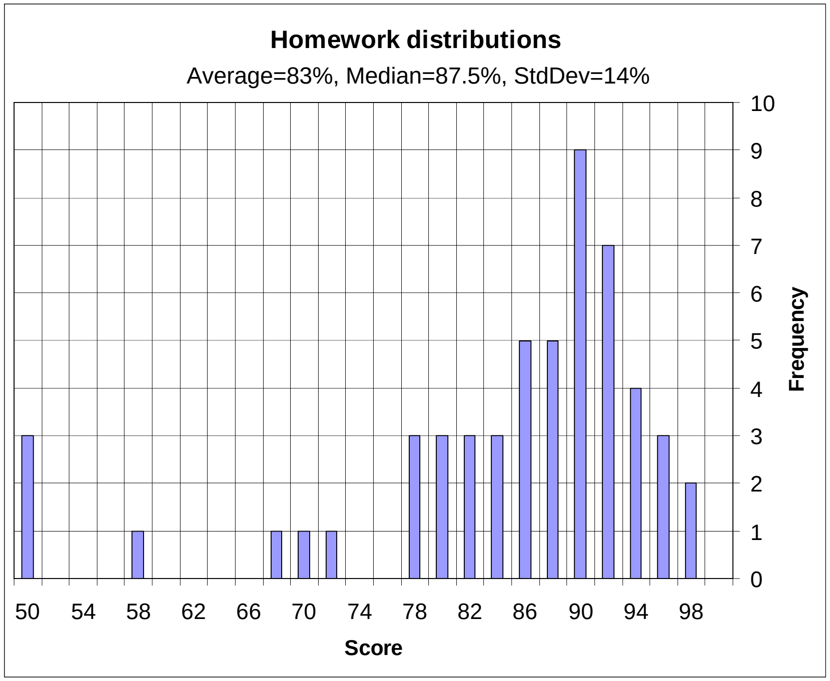
30% Homework

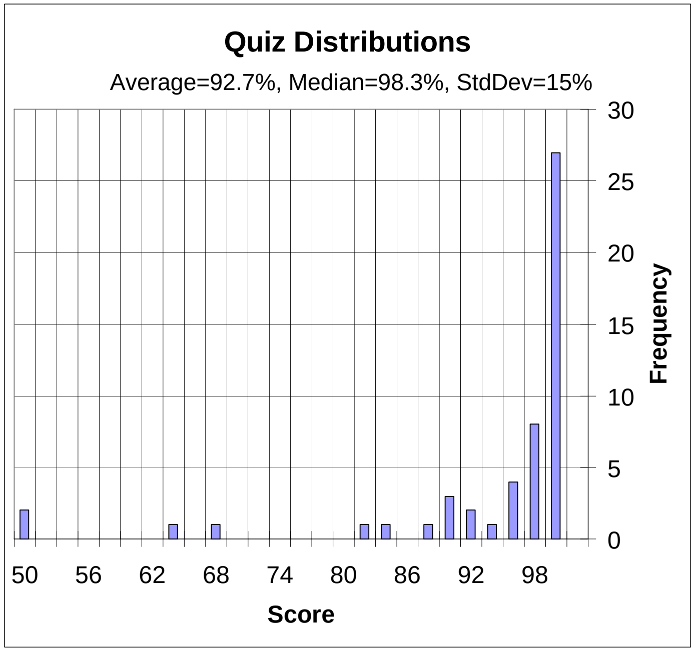
20% Labs

20% Quizzes

30% Final exam

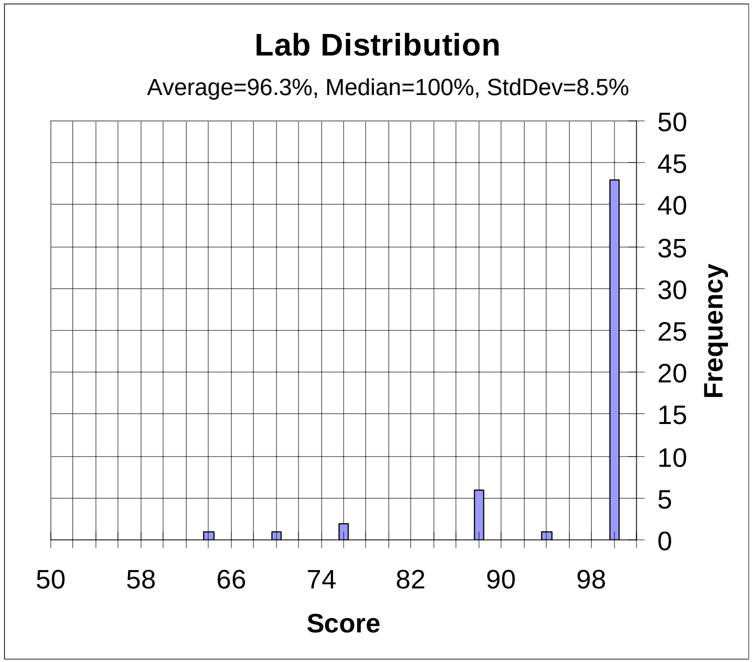






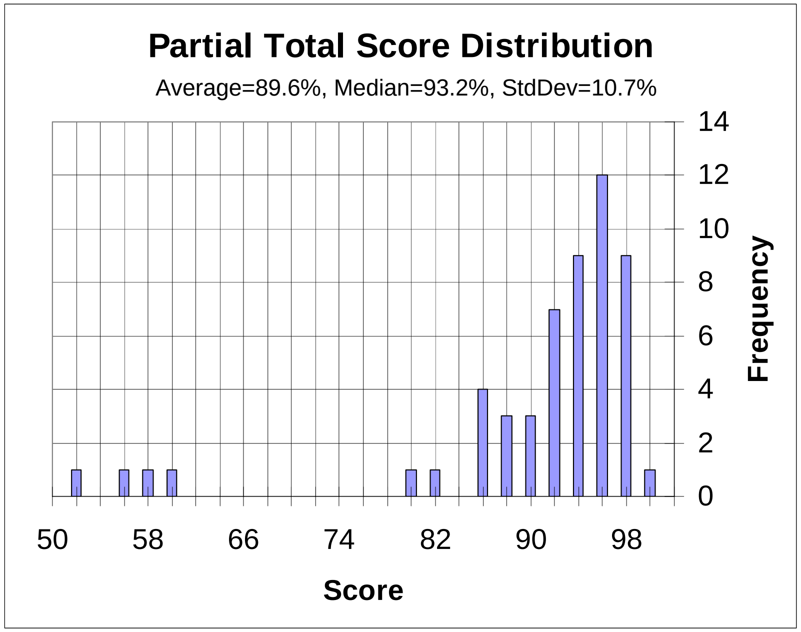
Quiz Distributions

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

Final Exam

- 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 complenet)
- Boolean logic, Boolean formulas
- Canonical forms, Karnaugh maps, minimization, NAND-NOR implementations
- Programable logic
- Adders, ALUs

Sequential Design

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

Hardware Description Lanauges

- 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} =
– 5_{10} =

Axioms and Theorems

1. Identity: X + 0 = X	Dual: X • 1 = X
2. Null: X + 1 = 1	Dual: X • 0 = 0
3. Idempotent: X + X = X	Dual: X • X = X
4. Involution: (X')' = X	
5. Complementarity: X + X' = 1	Dual: X • X' = 0
6. Commutative: X + Y = Y + X	Dual: X • Y = Y • X
7. Associative: (X+Y)+Z=X+(Y+Z)	Dual: (X•Y)•Z=X•(Y•Z)
8. Distributive: X•(Y+Z)=(X•Y)+(X•Z)	Dual: X+(Y•Z)=(X+Y)•(X+Z)
9. Uniting: X•Y+X•Y'=X	Dual: (X+Y)•(X+Y')=X
10. Absorption: X + X • Y = X	Dual: X • (X + Y) = X
11. Absorption2: (X + Y’) • Y = X • Y	Dual: (X • Y’) + Y = X + Y
12. Factoring: (X + Y) • (X’ + Z) = X • Z + X’ • Y	Dual: X • Y + X’ • Z = (X + Z) • (X’ + Y)

Axioms and Theorems

13. Concensus: (X • Y) + (Y • Z) + (X’ • Z) = X • Y + X’ • Z

Dual: (X + Y) • (Y + Z) • (X’ + Z) = (X + Y) • (X’ + Z)

14. DeMorgan’s Law: (X + Y + ...)’ = X’ • Y’ • ...

Dual: (X • Y • ...)’ = X’ + Y’ + ...

15. Generalized DeMorgan’s Laws: f’(X1,X2,...,Xn,0,1,+,•) = f(X1’,X2’,...,Xn’,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:

$f\ (X_1,X_2,...,X_n,0,1,+,\bullet) \Leftrightarrow f(X_1,X_2,...,X_n,1,0,\bullet,+)$

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 = XYZ+XY+X'YZ+X'Z

↙ *Use absorption {AB+A=A} with A=XY and B=Z*

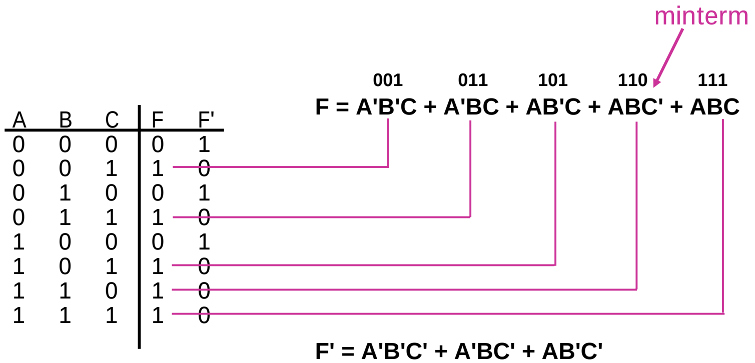
= XY+X'YZ+X'Z
Rearrange terms = XY+X'ZY+X'Z

↙ *Use absorption {AB+A=A} with A=X'Z and B=Y*

$\underline{XY+YZ+X'Z = XY+X'Z}$

Sum of Products Canonical Form

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



minterm

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

$$F = \overset{001}{A'B'C} + \overset{011}{A'BC} + \overset{101}{AB'C} + \overset{110}{ABC'} + \overset{111}{ABC}$$

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

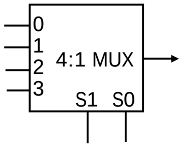
K-map Exercise

- Minimize the function F = Σ m(0, 2, 7, 8, 14, 15) + d(3, 6, 9, 12, 13)

Exercise

- Implementing the following function as a 4:1 mux

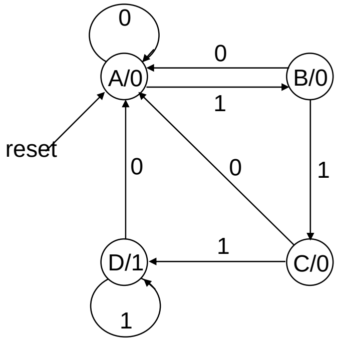
A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



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

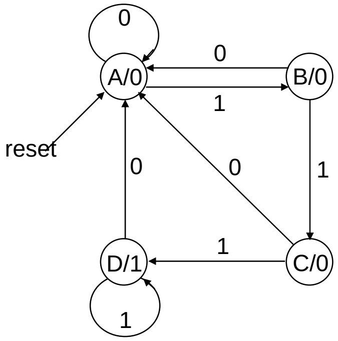
1. State diagram and state-transition table



	current		next	current
reset	state	input	state	output
1	–	–	A	0
0	A	0	A	0
0	A	1	B	0
0	B	0	A	0
0	B	1	C	0
0	C	0	A	0
0	C	1	D	0
0	D	0	A	1
0	D	1	D	1

2. State minimization & 3. State encoding

- State diagram is already minimized
- Try a binary encoding



	current		next	current
reset	state	input	state	output
1	–	–	00	0
0	00	0	00	0
0	00	1	01	0
0	01	0	00	0
0	01	1	10	0
0	10	0	00	0
0	10	1	11	0
0	11	0	00	1
0	11	1	11	1

4. Minimize next-state logic

MSB+

M

	0	0	0	0
In	0	1	1	1

L

MSB+ = LIn + MIn

LSB+

M

	0	0	0	0
In	1	0	1	1

L

LSB+ = L'In + MIn

OUT+

M

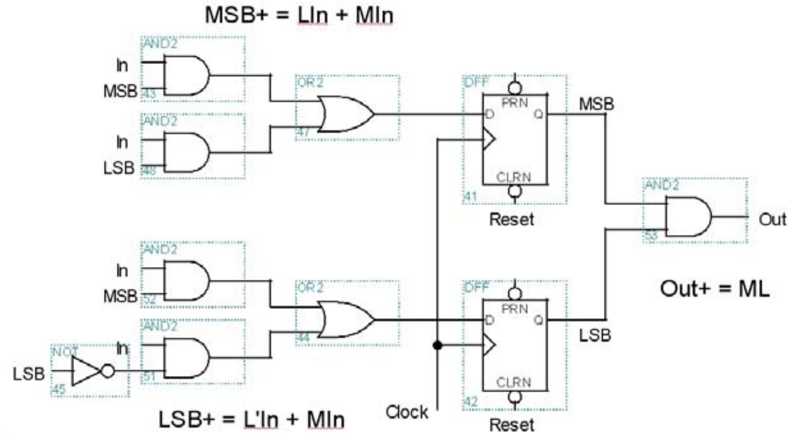
	0	0	1	0
In	0	0	1	0

L

Out+ = ML

Notation
M := MSB
L := LSB
In := Input

5. Implement the design



MSB+ = LIn + MIn
In
MSB
In
LSB
AND2
OR2
DFF
PRN
CLRN
D
Q
MSB
Reset
AND2
Out
Out+ = ML
In
MSB
In
LSB
NOT
LSB
Reset
Clock
LSB+ = L'In + MIn