

## Lecture 10

- Logistics
  - HW3 due now
    - Solutions will be available at the midterm review session tomorrow (and at the end of class today)
  - HW4 handed out today
    - Due next week
  - Midterm 1 Friday in class. Closed book. Closed notes. No calculators.
    - Sample midterm on the web
    - Review session, Thursday 4:30 here (EEB 037)
      - Bring your questions!
- Last lecture
  - Demultiplexers
  - PLDs
    - PLAs
    - PALS
- Today
  - PLDs
    - ROMs
  - Multilevel Logic

CSE370, Lecture 10

1

## Midterm 1 Topics Covered

- ♦ Combinational logic basics
  - Binary/hex/decimal numbers
  - Ones and twos complement arithmetic
  - Truth tables
  - Boolean algebra
  - Basic logic gates
  - Schematic diagrams
  - de Morgan's theorem
  - AND/OR to NAND/NOR logic conversion
  - K-maps (up to 4 variables), logic minimization, don't cares
  - SOP, POS
  - Minterm and maxterm expansions (canonical, minimized)

CSE370, Lecture 10

2

## Midterm 1 Topics Covered (continued)

- ♦ Combinational logic applications
  - Combinational design
    - ↳ Input/output encoding
    - ↳ Truth table
    - ↳ K-map
    - ↳ Boolean equations
    - ↳ Schematics
  - Multiplexers

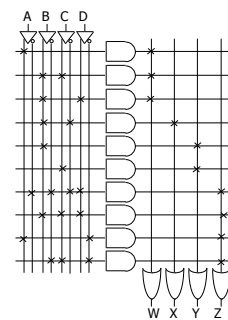
CSE370, Lecture 10

3

## Recall example: BCD to Gray --- Wiring of a PLA

### Minimized functions:

$$\begin{aligned} W &= A + BC + BD \\ X &= BC' \\ Y &= B + C \\ Z &= A'B'C'D + BCD \\ &\quad + AD' + B'CD' \end{aligned}$$



CSE370, Lecture 10

4

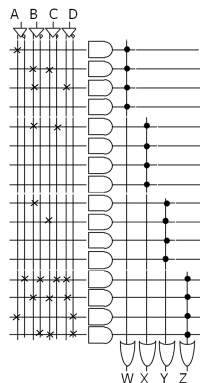
## Recall: Wiring a PAL

### Minimized functions:

$$\begin{aligned} W &= A + BC + BD \\ X &= BC' \\ Y &= B + C \\ Z &= A'B'C'D + BCD \\ &\quad + AD' + B'CD' \end{aligned}$$

Fine example for the use of PAL  
(because no shared AND terms)

Many AND gates wasted, but  
still faster and cheaper than PLA



CSE370, Lecture 10

5

## Compare implementations for this example

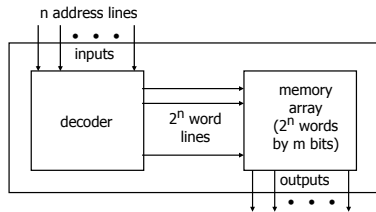
- PLA:
  - No shared logic terms in this example
  - 10 decoded functions (10 AND gates)
- PAL:
  - Z requires 4 product terms
    - 16 decoded functions (16 AND gates)
    - 6 unused AND gates
- This decoder is a good candidate for PALs
  - 10 of 16 possible inputs are decoded
  - No sharing among AND terms
- Another option?
  - Yes — a ROM

CSE370, Lecture 10

6

## Read-only memories (ROMs)

- Two dimensional array of stored 1s and 0s
  - Input is an address  $\Rightarrow$  ROM decodes all possible input addresses
  - Stored row entry is called a "word"
  - ROM output is the decoded word



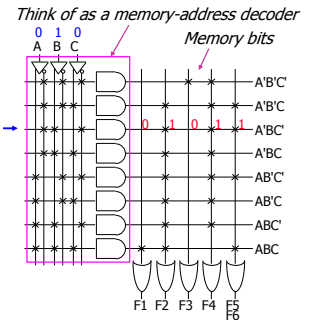
CSE370, Lecture 10

7

## Like this special PLA example: only more efficient

- $F1 = ABC$
- $F2 = A + B + C$
- $F3 = A' B' C'$
- $F4 = A' + B' + C'$
- $F5 = A \text{ xor } B \text{ xor } C$
- $F6 = A \text{ xnor } B \text{ xnor } C$

A	B	C	F1	F2	F3	F4	F5	F6
0	0	0	0	0	1	1	0	0
0	0	1	0	1	0	1	1	1
0	1	0	0	1	0	1	1	1
0	1	1	0	1	0	1	0	0
1	0	0	1	0	1	1	1	1
1	0	1	0	1	0	1	0	0
1	1	0	1	0	1	0	0	0
1	1	1	1	1	0	0	1	1

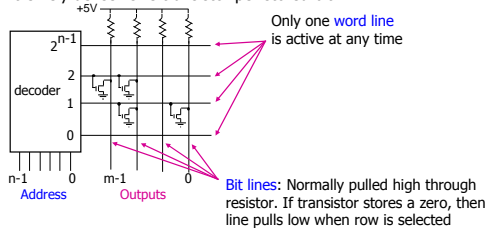


CSE370, Lecture 10

8

## ROM details

- Similar to a PLA but with a fully decoded and fixed AND array
- Completely flexible OR array (unlike a PAL)
- Extremely dense: One transistor per stored bit



CSE370, Lecture 10

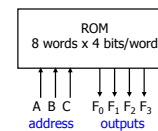
9

## Two-level combinational logic using a ROM

- Use a ROM to directly store a truth table

- No need to minimize logic
- Example:
  - $F0 = A'B'C' + AB'C' + AB'C$
  - $F1 = A'B'C + A'BC' + ABC$
  - $F2 = A'B'C' + A'BC' + AB'C'$
  - $F3 = A'BC + AB'C' + ABC'$

A	B	C	F0	F1	F2	F3
0	0	0	0	0	1	0
0	0	1	1	1	1	0
0	1	0	0	1	0	0
0	1	1	0	0	0	1
1	0	0	1	0	1	1
1	0	1	1	0	0	0
1	1	0	0	0	0	1
1	1	1	0	1	0	0



You specify whether to store 1 or 0 in each location in the ROM

CSE370, Lecture 10

10

## ROMs versus PLAs/PALs

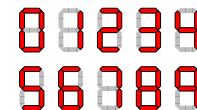
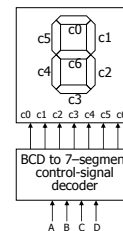
- ROMs
  - Benefits
    - Quick to design, simple, dense
  - Limitations
    - Size doubles for each additional input
    - Can't exploit don't cares
- PLAs/PALs
  - Benefits
    - Logic minimization reduces size
    - PALs faster/cheaper than PLAs
  - Limitations
    - PAL OR-plane has hard-wired fan-in
- Another alternative: Field programmable gate arrays
  - Learn a bit more later in this course

CSE370, Lecture 10

11

## Example: BCD to 7-segment display controller

- The problem
  - Input is a 4-bit BCD digit (A, B, C, D)
  - Need signals to drive a display (7 outputs C0 - C6)



CSE370, Lecture 10

12

## Formalize the problem

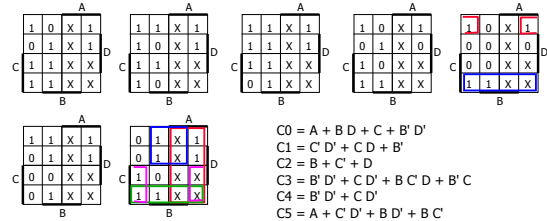
- Truth table
  - Many don't cares
- Choose implementation target
  - If ROM, we are done
  - Don't cares imply PAL/PLA may be good choice
- Implement design
  - Minimize the logic
  - Map into PAL/PLA

A	B	C	D	C0	C1	C2	C3	C4	C5	C6
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X
1	1	X	X	X	X	X	X	X	X	X

Not all rows of the truth table are listed separately

## Sum-of-products implementation

- 15 unique product terms if we minimize individually



$$\begin{aligned}
 C0 &= A + BD + C + B'D' \\
 C1 &= C'D' + CD + B' \\
 C2 &= B + C + D \\
 C3 &= B'D' + CD' + B'C'D + B'C \\
 C4 &= B'D' + CD' \\
 C5 &= A + C'D' + B'D' + B'C' \\
 C6 &= A + C'D' + B'C' + B'C
 \end{aligned}$$

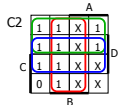
4 input, 7 output

PLA: 15 AND gates

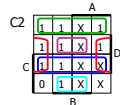
PAL: 4 product terms per output (28 AND gates)

## If choosing PLA: better SOP implementation

- Can do better than 15 product terms
  - Share terms among outputs  $\Rightarrow$  only 9 unique product terms
    - Each term not necessarily minimized



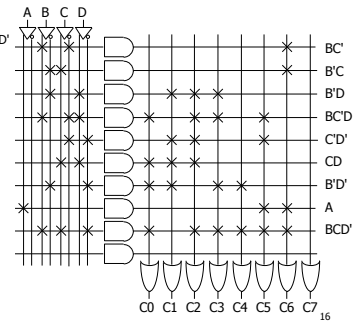
$$\begin{aligned}
 C0 &= A + BD + C + B'D' \\
 C1 &= C'D' + CD + B' \\
 C2 &= B + C + D \\
 C3 &= B'D' + CD' + B'C'D + B'C \\
 C4 &= B'D' + CD' \\
 C5 &= A + C'D' + B'D' + B'C' \\
 C6 &= A + C'D' + B'C' + B'C
 \end{aligned}$$



$$\begin{aligned}
 C0 &= BC'D + CD + B'D' + BCD' + A \\
 C1 &= B'D + C'D' + CD + B'D' \\
 C2 &= B'D + BCD + C'D' + CD + BCD' \\
 C3 &= BCD + B'D + B'D' + BCD' \\
 C4 &= B'D' + BCD' \\
 C5 &= BCD + C'D' + A + BCD' \\
 C6 &= B'C + B'C' + BCD' + A
 \end{aligned}$$

## PLA implementation

$$\begin{aligned}
 C0 &= BC'D + CD + B'D' + BCD' + A \\
 C1 &= B'D + C'D' + CD + B'D' \\
 C2 &= B'D + BCD + C'D' + CD + BCD' \\
 C3 &= BCD + B'D + B'D' + BCD' \\
 C4 &= B'D' + BCD' \\
 C5 &= BCD + C'D' + A + BCD' \\
 C6 &= B'C + B'C' + BCD' + A
 \end{aligned}$$

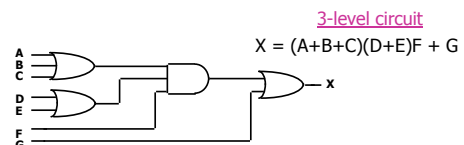


## Multilevel logic

- Basic idea: Simplify logic using >2 gate levels
  - Time-space (speed versus gate count) tradeoff
    - Will talk about the speed issue with timing diagram
- Two-level logic *usually*
  - Has smaller delays (faster circuits)
  - more gates and more wires (more circuit area)
- Multilevel logic *usually*
  - Has fewer gates (smaller circuits)
  - more gate delays (slower circuits)

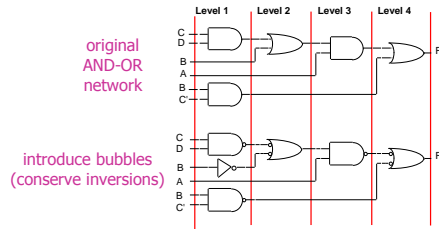
## Multilevel logic example

- Function X
  - SOP:  $X = ADF + AEF + BDF + BEF + CDF + CEF + G$ 
    - X is minimized!
    - Six 3-input ANDs; one 7-input OR; 26 wires
  - Multilevel:  $X = (A+B+C)(D+E)F + G$ 
    - Factored form
    - One 3-input OR, two 2-input OR's, one 3-input AND; 11 wires



## Multilevel NAND/NAND conversion

$$F = A(B+CD) + BC'$$

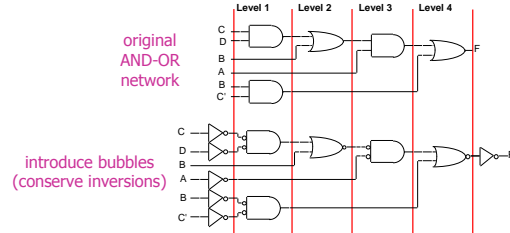


CSE370, Lecture 13

19

## Multilevel NOR/NOR conversion

$$F = A(B+CD) + BC'$$

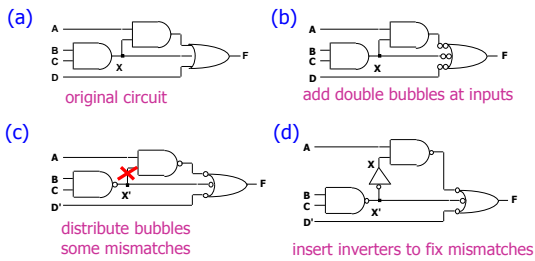


CSE370, Lecture 13

20

## Generic multilevel conversion

$$F = ABC + BC + D = AX + X + D$$



CSE370, Lecture 13

21

## Issues with multilevel design

- ◆ No global definition of "optimal" multilevel circuit
  - Optimality depends on user-defined goals
- ◆ Synthesis requires CAD-tool help
  - No simple hand methods like K-maps
  - CAD tools manipulate Boolean expressions
  - Covered in more detail in CSE467

CSE370, Lecture 13

22

## Multilevel logic summary

- ◆ Advantages over 2-level logic
  - Smaller circuits
  - Reduced fan-in
  - Less wires
- ◆ Disadvantages w.r.t 2-level logic
  - More difficult design
  - Less powerful optimizing tools
- ◆ What you should know for CSE370
  - The basic multilevel idea
  - Multilevel NAND/NAND and NOR/NOR conversion

CSE370, Lecture 13

23