**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
  - 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

---

**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)

---

**Midterm 1 Topics Covered (continued)**

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

---

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

Minimized functions:

\[
\begin{align*}
W &= A + BC + BD \\
X &= BC' \\
Y &= B + C \\
Z &= AB'CD + BCD + AD' + B'CD'
\end{align*}
\]

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

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

---

**Recall: Wiring a PAL**

Minimized functions:

\[
\begin{align*}
W &= A + BC + BD \\
X &= BC' \\
Y &= B + C \\
Z &= AB'CD + BCD + AD' + B'CD'
\end{align*}
\]

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

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

---

**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
Read-only memories (ROMs)

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

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

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

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 + A'B'C + AB'C
    - F1 = A'B'C + A'B'C + ABC
    - F2 = A'B'C + A'B'C + A'BC

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)
Formalize the problem

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

Not all rows of the truth table are listed separately

Sum-of-products implementation

- **15 unique product terms if we minimize individually**
- **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 ⇒ only 9 unique product terms
  - Each term not necessarily minimized

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**
  - Has smaller delays (faster circuits)
  - More gates and more wires (more circuit area)
- **Multilevel logic**
  - 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
  - Factored form
  - One 3-input OR, two 2-input OR’s, one 3-input AND; 11 wires

2-level circuit

\( X = (A+B+C)(D+E)F + G \)
Multilevel NAND/NAND conversion

\[ F = A(B+CD) + BC' \]

original AND-OR network
introduce bubbles (conserve inversions)

Level 1 Level 2 Level 3 Level 4

F

A
C
D
B
B
C'

Multilevel NOR/NOR conversion

\[ F = A(B+CD) + BC' \]

original AND-OR network
introduce bubbles (conserve inversions)

Level 1 Level 2 Level 3 Level 4

F

A
C
D
B
B
C'

Generic multilevel conversion

\[ F = ABC + BC + D = AX + X + D \]

(a) original circuit
(b) add double bubbles at inputs
(c) distribute bubbles some mismatches
(d) insert inverters to fix mismatches

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

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