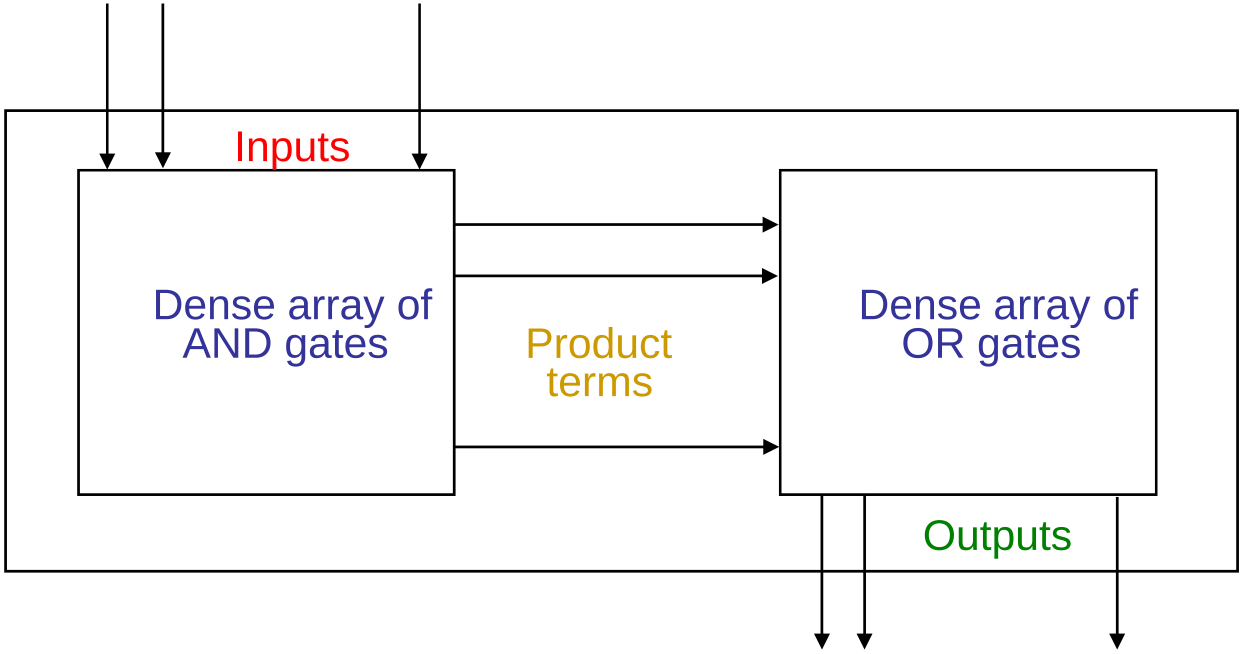
ELECTRONICS SYSTEM DESIGN

SECTION-4

MULTI INPUT SYSTEM CONTROLLER DESIGN AND PLDs

Prgrammable Logic Organization

- **Pre-fabricated building block of many AND/OR gates (or NOR, NAND)**
- **"Personalized" by making or breaking connections among the gates**



Programmable Array Block Diagram for Sum of Products Form

Basic Programmable Logic Organizations

- Depending on which of the AND/OR logic arrays is programmable, we have three basic organizations

ORGANIZATION	AND ARRAY	OR ARRAY
PAL	PROG.	FIXED
PROM	FIXED	PROG.
PLA	PROG.	PROG.

PLA Logic Implementation

Key to Success: Shared Product Terms

Equations

Example:

$$F0 = A + \overline{B}\,\overline{C}$$
$$F1 = A\,\overline{C} + A\,B$$
$$F2 = \overline{B}\,\overline{C} + A\,B$$
$$F3 = \overline{B}\,C + A$$

Personality Matrix

Product term	Inputs			Outputs			
	A	B	C	F_0	F_1	F_2	F_3
$A\,B$	1	1	-	0	①	①	0
$\overline{B}\,C$	-	0	1	0	0	0	1
$A\,\overline{C}$	1	-	0	0	1	0	0
$\overline{B}\,\overline{C}$	-	0	0	①	0	①	0
A	1	-	-	①	0	0	①

Reuse of terms

Input Side:

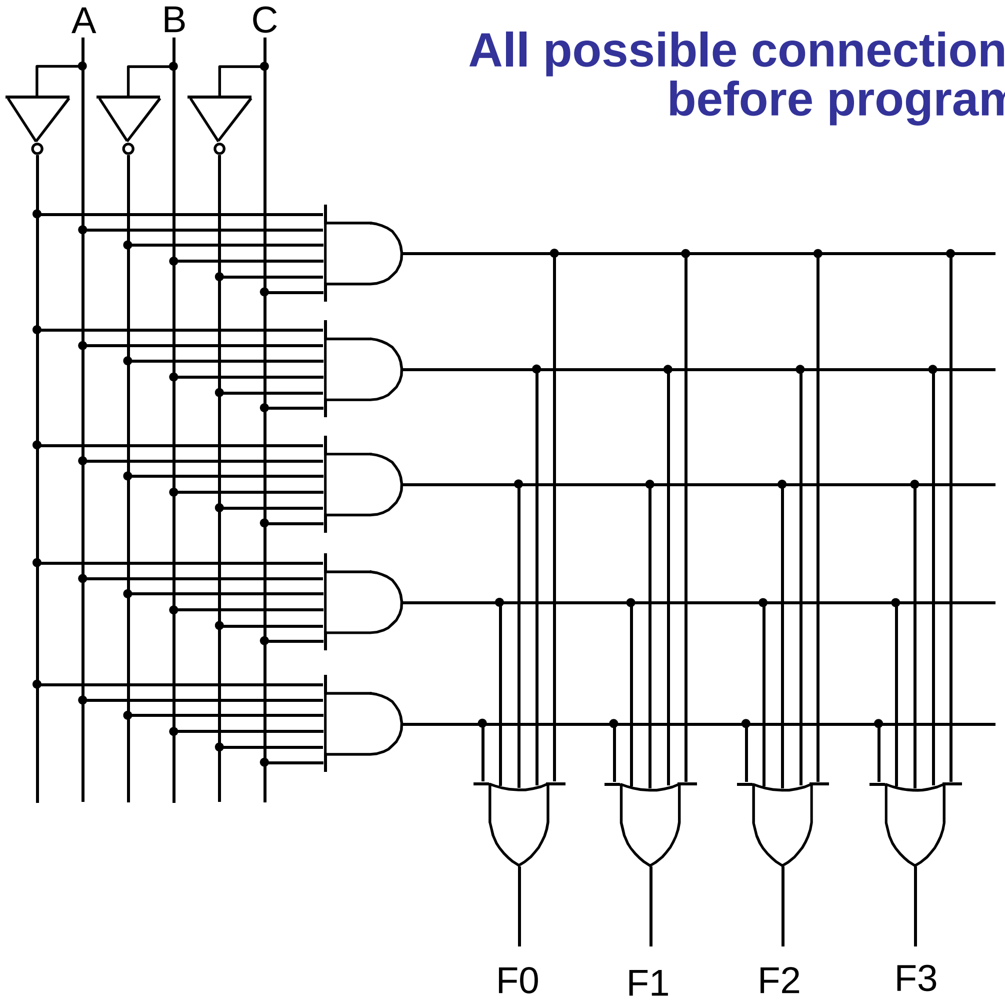
1 = asserted in term
0 = negated in term
- = does not participate

Output Side:

1 = term connected to output
0 = no connection to output

PLA Logic Implementation

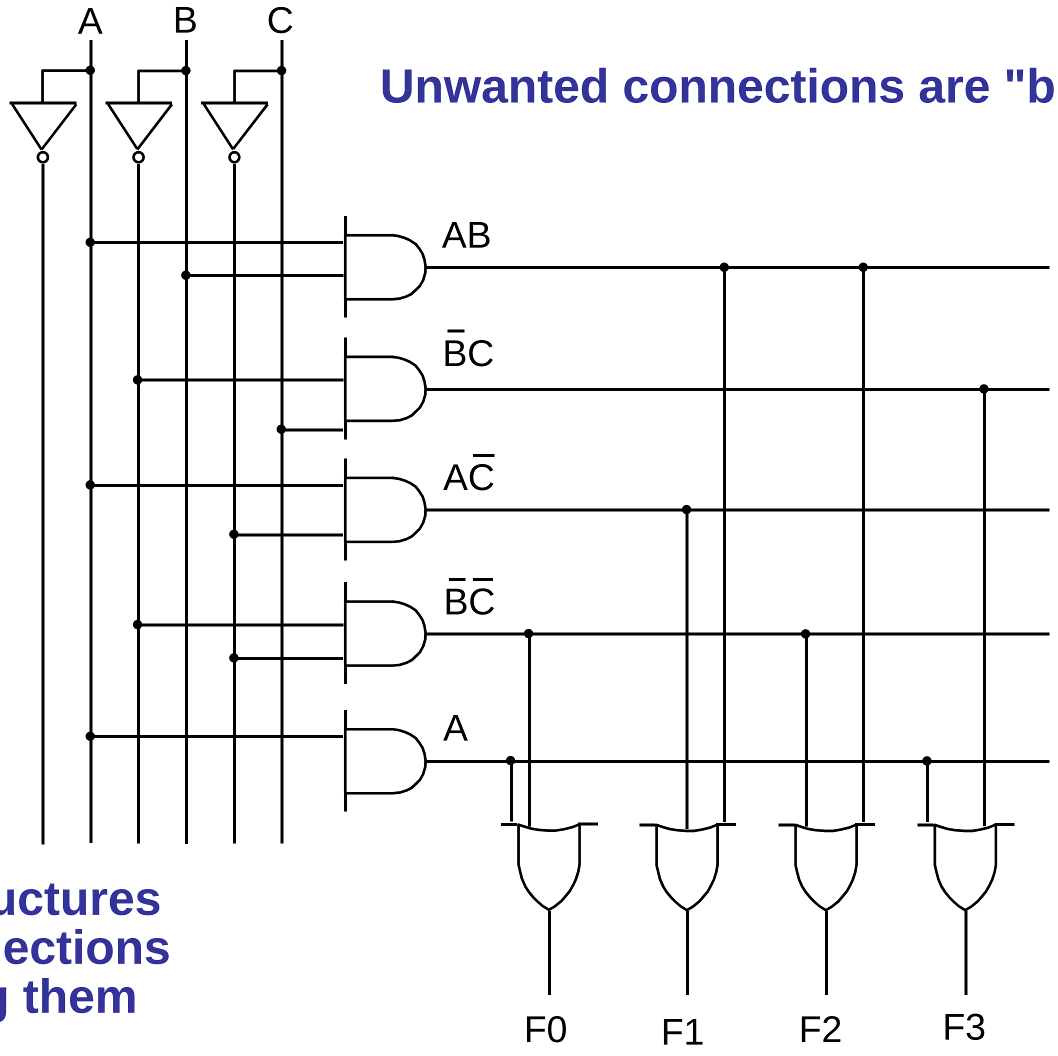
Example Continued - Unprogrammed device



All possible connections are available before programming

PLA Logic Implementation

Example Continued - Programmed part



Unwanted connections are "blown"

Note: some array structures work by making connections rather than breaking them

PLA Logic Implementation

Alternative representation

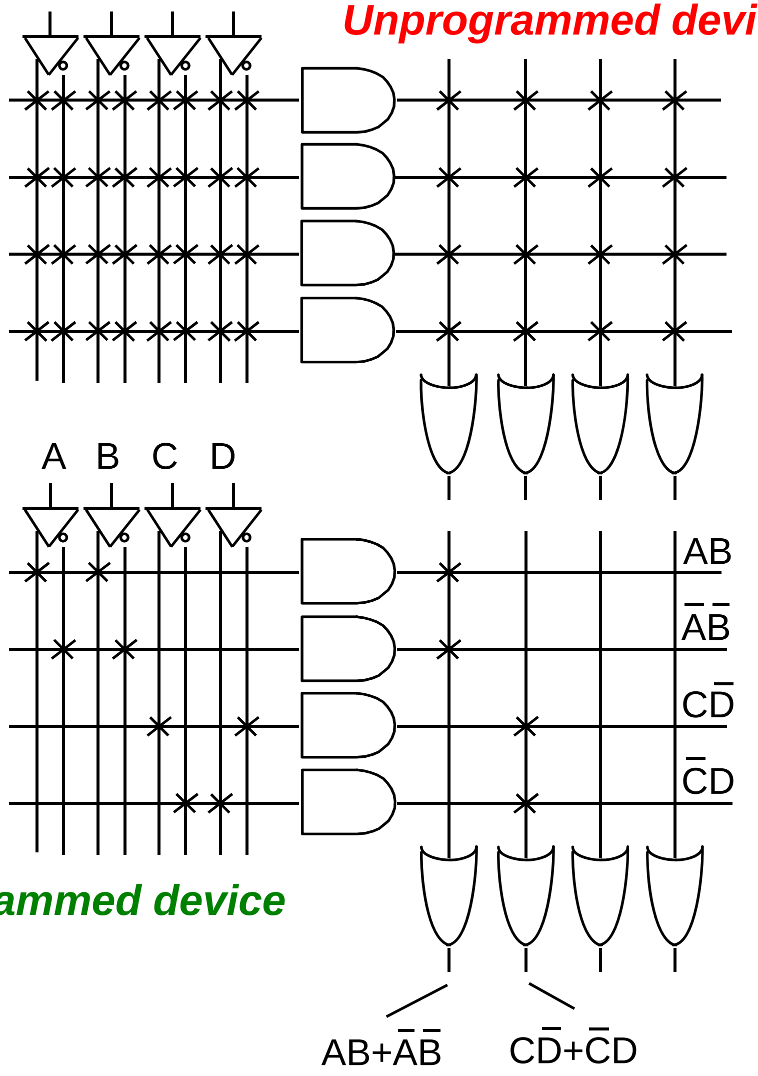
Short-hand notation
so we don't have to
draw all the wires!

X at junction indicates
a connection

Notation for implementing

$F0 = A\,B + \bar{A}\,\bar{B}$

$F1 = C\,\bar{D} + \bar{C}\,D$



Unprogrammed device

Programmed device

PLA Logic Implementation

Design Example

Multiple functions of A, B, C

$F1 = A\,B\,C$

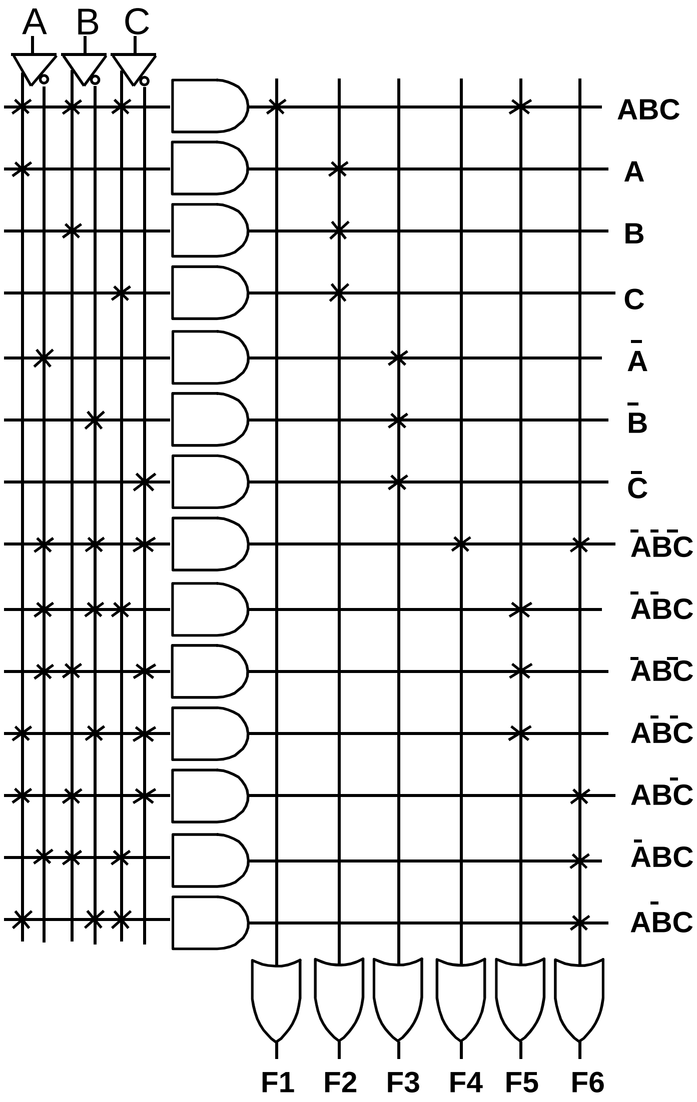
$F2 = A + B + C$

$F3 = \overline{A\,B\,C}$

$F4 = \overline{A + B + C}$

$F5 = A \oplus B \oplus C$

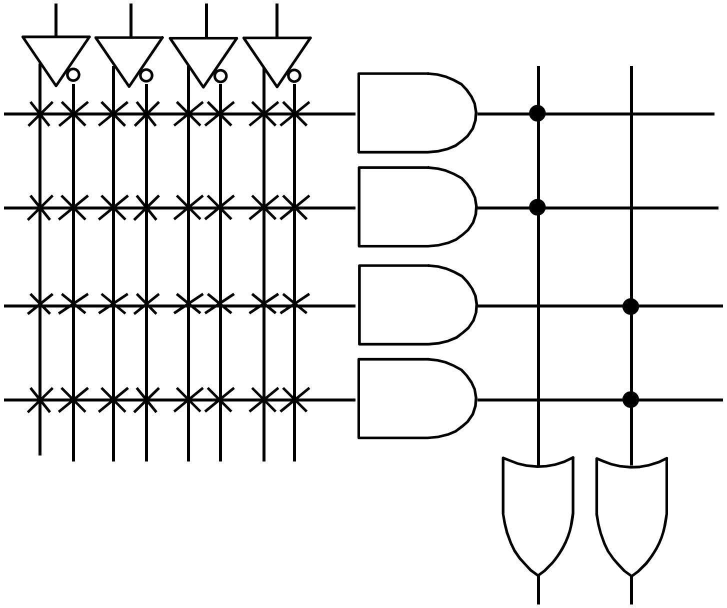
$F6 = \overline{A \oplus B \oplus C}$



PALs and PLAs

What is difference between Programmable Array Logic (PAL) and Programmable Logic Array (PLA)?

PAL concept — implemented by Monolithic Memories
AND array is programmable, OR array is fixed at fabrication



A given column of the OR array has access to only a subset of the possible product terms

PLA concept — Both AND and OR arrays are programmable

PALs and PLAs

- Of the two organizations the PLA is the most flexible
  - One PLA can implement a huge range of logic functions
  - BUT many pins; large package, higher cost
- PALs are more restricted / you trade number of OR terms vs number of outputs
  - Many device variations needed
  - Each device is cheaper than a PLA

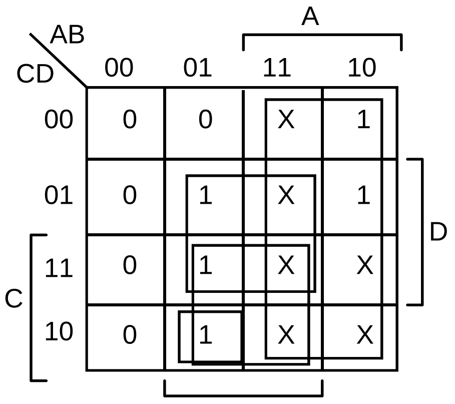
PAL Logic Implementation

Design Example: BCD to Gray Code Converter

K-maps

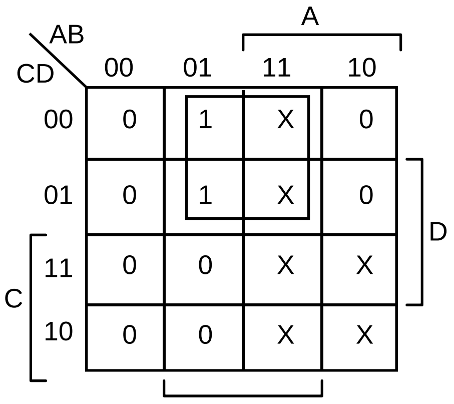
Truth Table

A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	1	1	1	0
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	1
1	0	0	0	1	0	0	1
1	0	0	1	1	0	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X



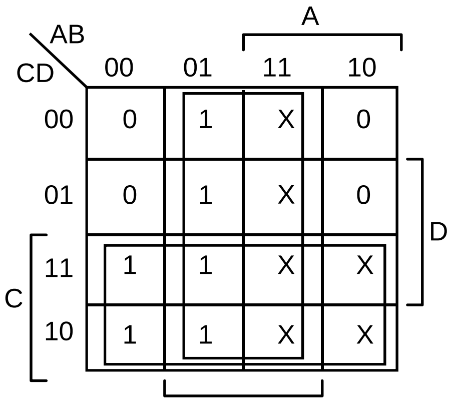
CD \ AB	00	01	11	10
00	0	0	X	1
01	0	1	X	1
11	0	1	X	X
10	0	1	X	X

K-map forW



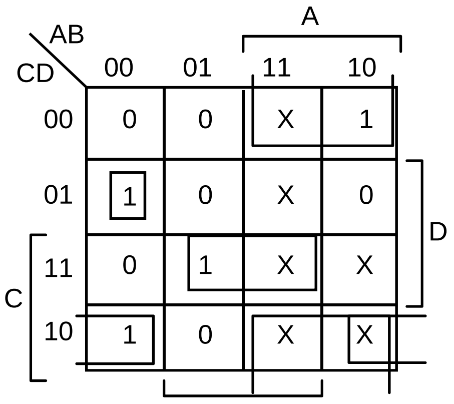
CD \ AB	00	01	11	10
00	0	1	X	0
01	0	1	X	0
11	0	0	X	X
10	0	0	X	X

K-map forX



CD \ AB	00	01	11	10
00	0	1	X	0
01	0	1	X	0
11	1	1	X	X
10	1	1	X	X

K-map forY



CD \ AB	00	01	11	10
00	0	0	X	1
01	1	0	X	0
11	0	1	X	X
10	1	0	X	X

K-map forZ

Minimized Functions:

$W = A + B\,D + B\,C$

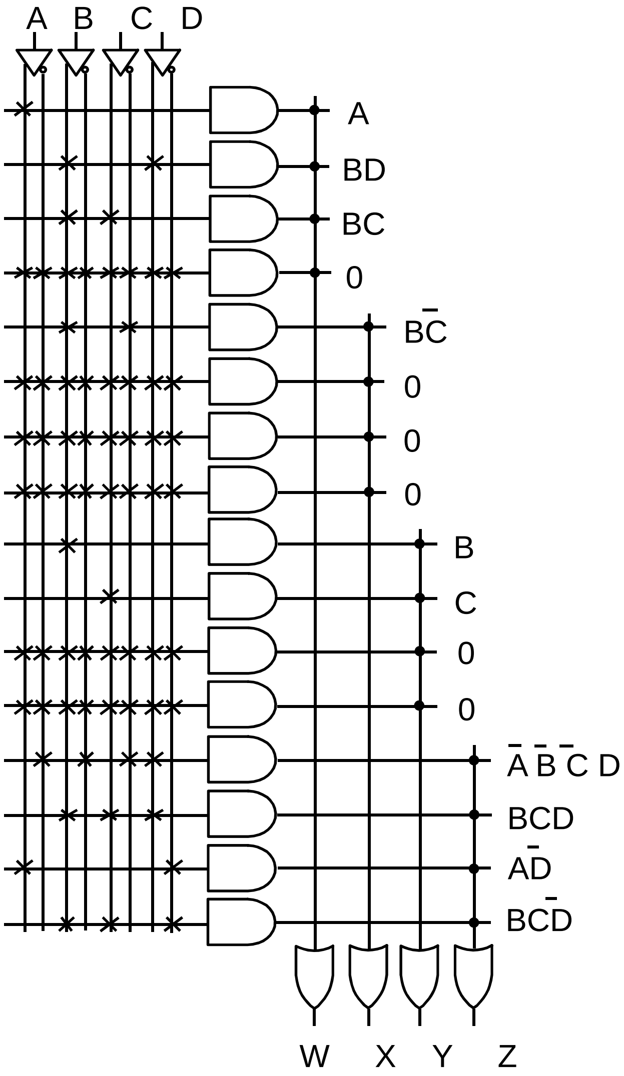
$X = B\,\bar{C}$

$Y = B + C$

$Z = \bar{A}\,\bar{B}\,\bar{C}\,D + B\,C\,D + A\,\bar{D} + \bar{B}\,C\,\bar{D}$

PAL Logic Implementation

Programmed PAL:



Minimized Functions:

$W = A + B\,D + B\,C$

$X = B\,\bar{C}$

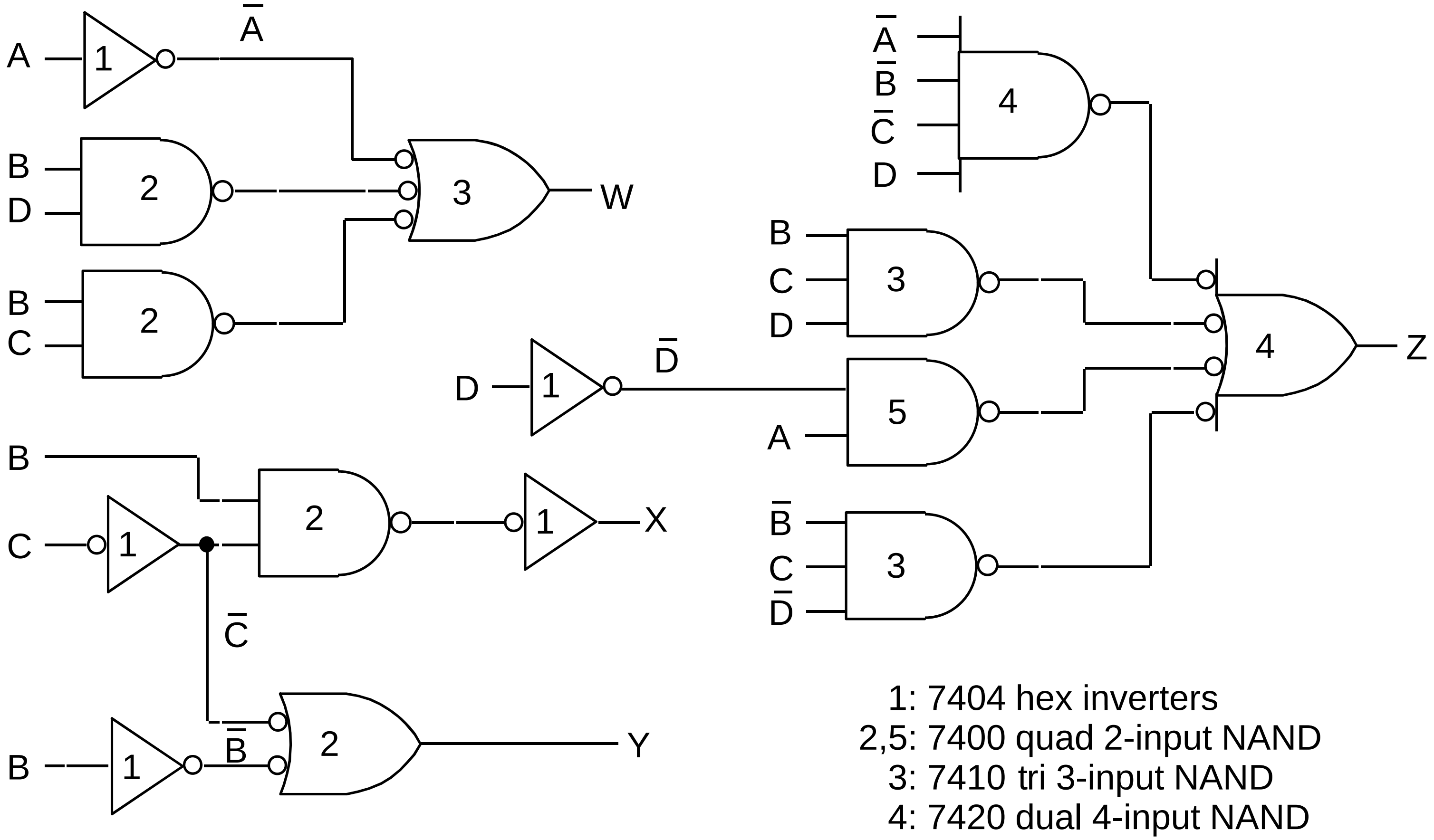
$Y = B + C$

$Z = \bar{A}\,\bar{B}\,\bar{C}\,D + B\,C\,D + A\,\bar{D} + \bar{B}\,C\,\bar{D}$

4 product terms per each OR gate

PAL Logic Implementation

Code Converter Discrete Gate Implementation



1: 7404 hex inverters
2,5: 7400 quad 2-input NAND
3: 7410 tri 3-input NAND
4: 7420 dual 4-input NAND

4 SSI Packages vs. 1 PLA/PAL Package!

PLA Logic Implementation

Another Example: Magnitude Comparator

CD \ AB	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

K-map for EQ

CD \ AB	00	01	11	10
00	0	1	1	1
01	1	0	1	1
11	1	1	0	1
10	1	1	1	0

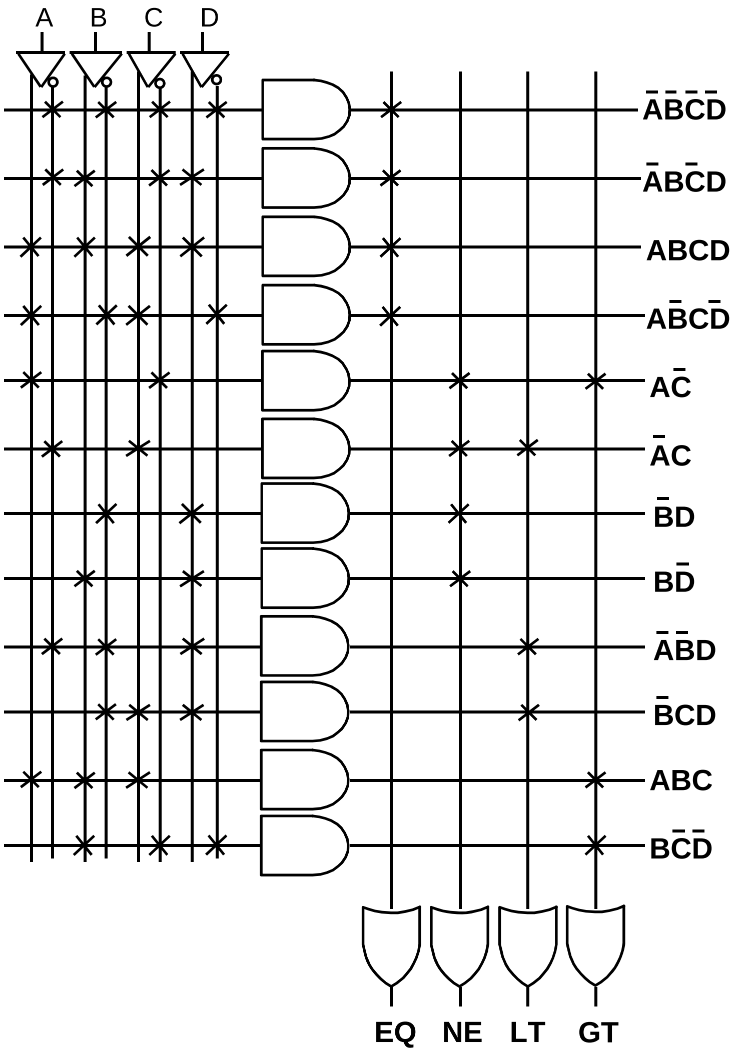
K-map for NE

CD \ AB	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

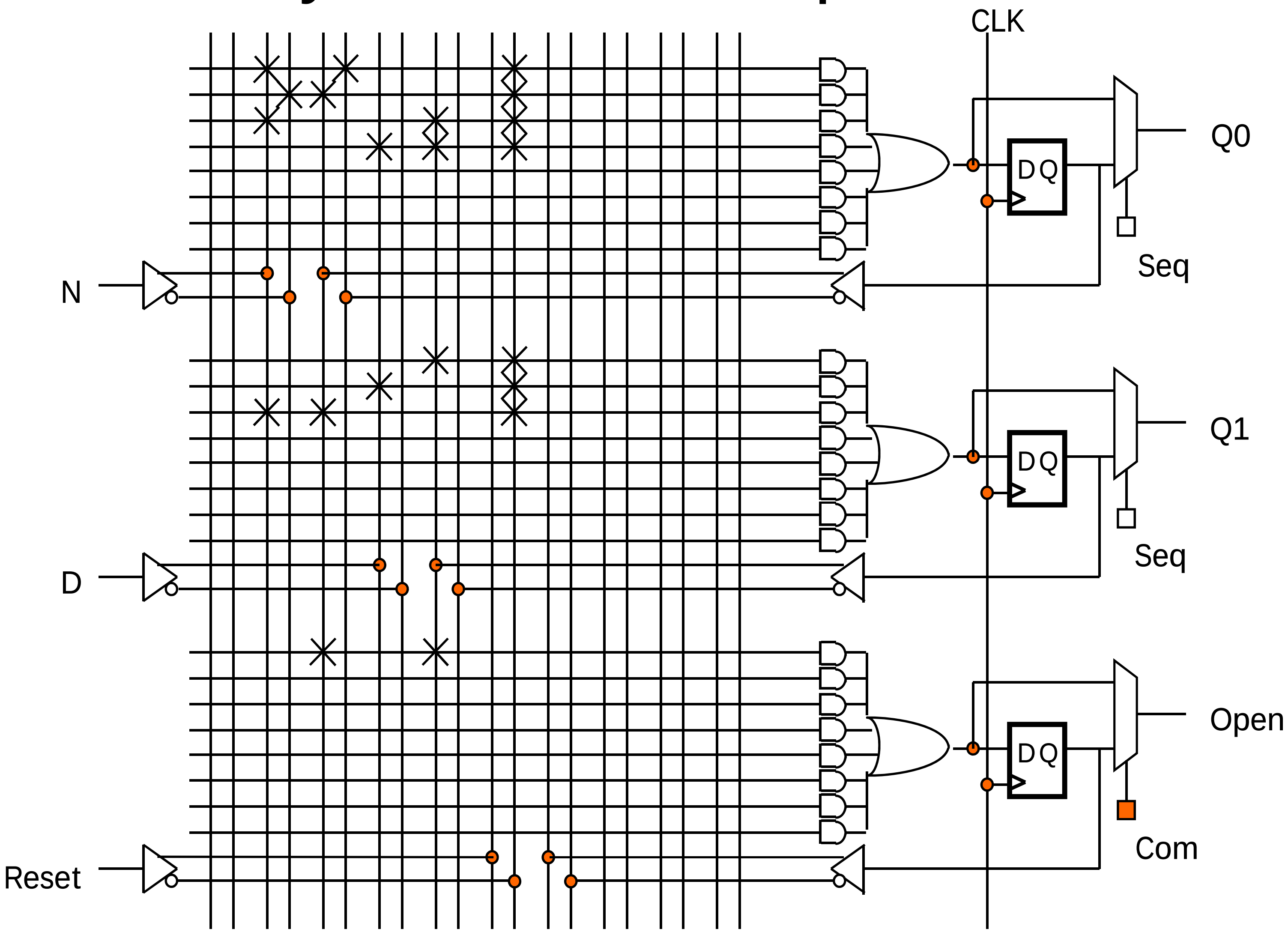
K-map for LT

CD \ AB	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
10	0	0	1	0

K-map for GT



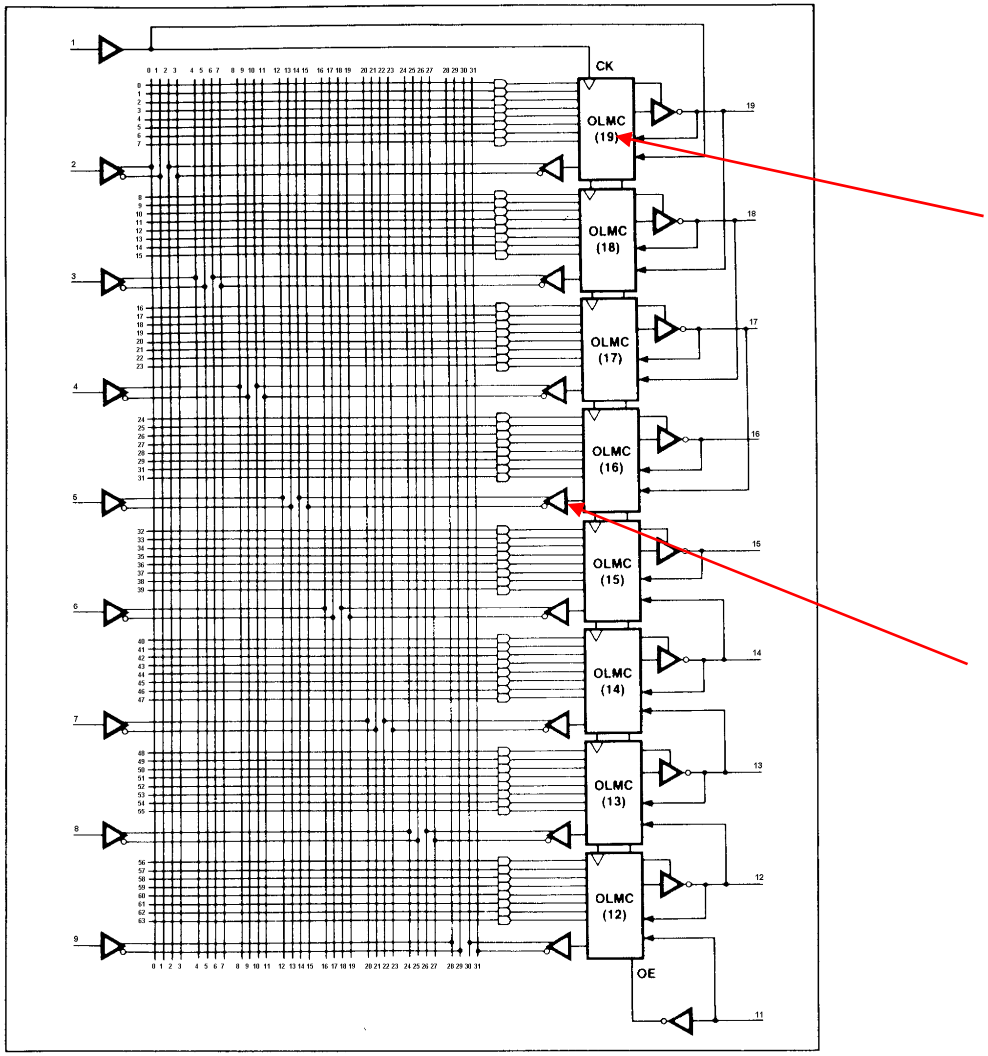
Another Variation: Synchronous vs. Asynchronous Outputs



Complex Programmable Logic Devices

- Complex PLDs typically combine PAL combinational logic with FFs
  - Organized into logic blocks
  - Fixed OR array size
  - Combinational or registered output
  - Some pins are inputs only
- Usually enough logic for simple counters, state machines, decoders, etc.
- e.g. GAL22V10, GAL16V8, etc.

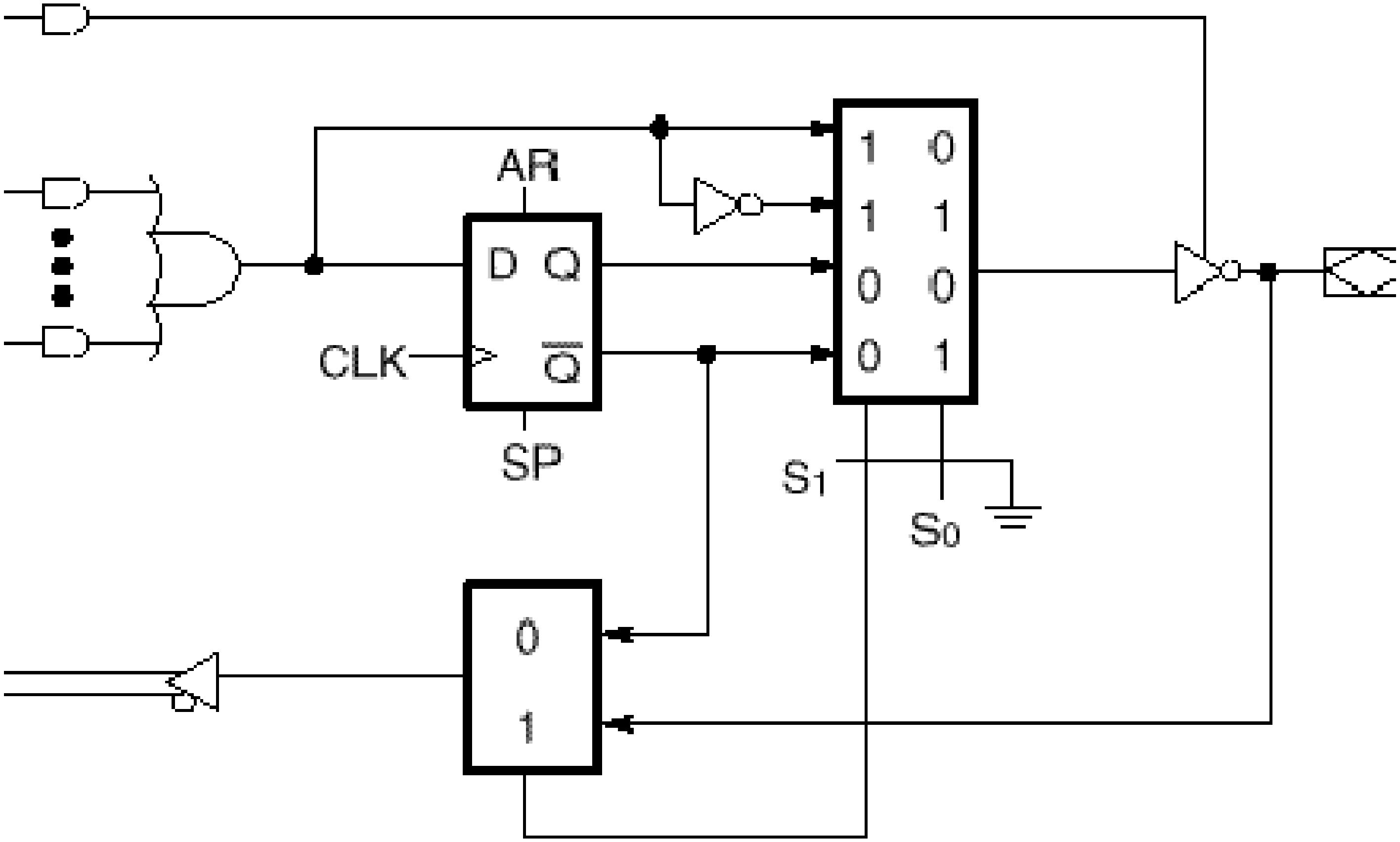
GAL CPLD



OLMC (Output Logic MacroCell) has OR, FF, output multiplexer and I/O control logic.

Note that OLMC output is fed back to input matrix for use in other OLMCs.

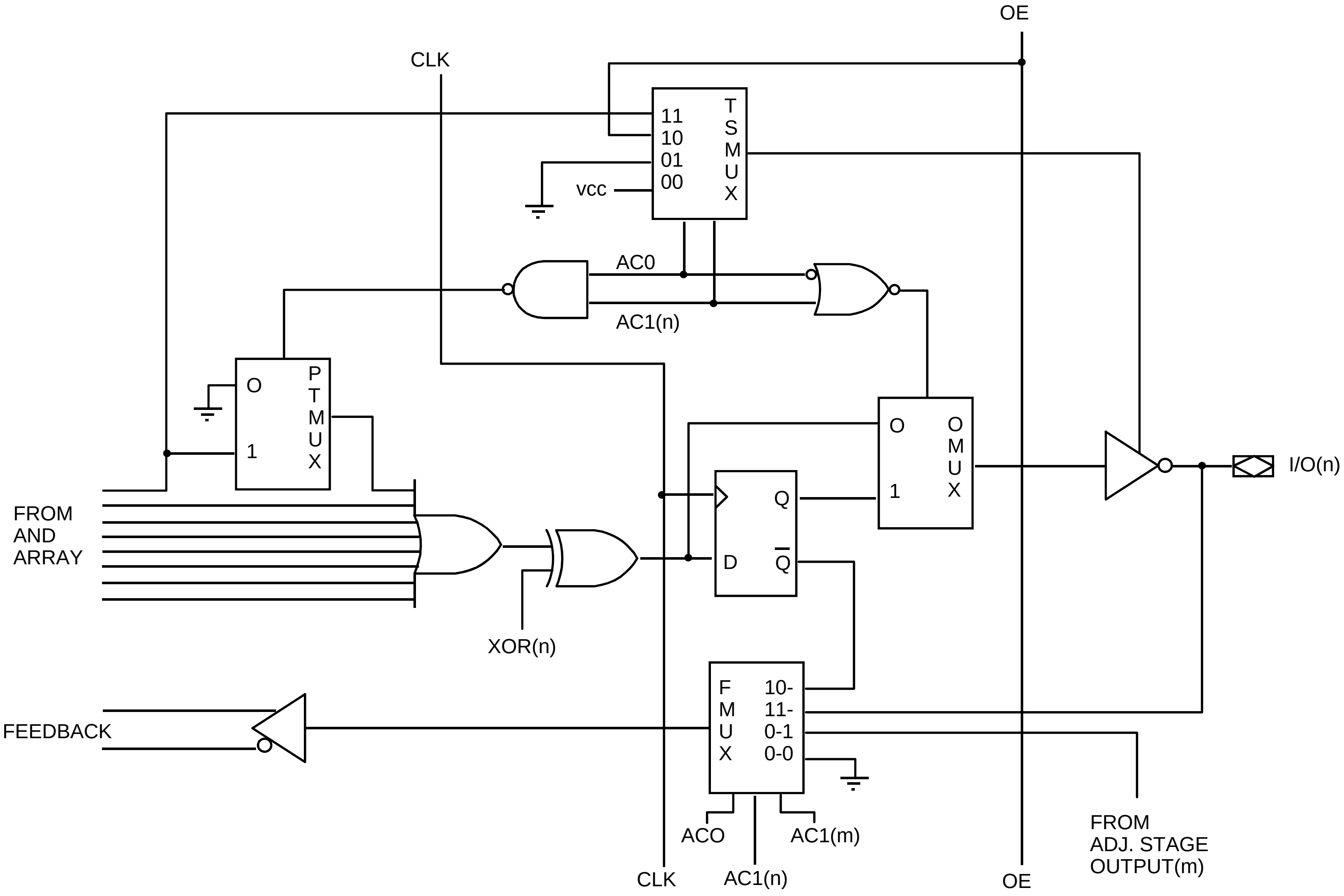
GAL22V10 OLMC Structure



PAL22V10 OLMC Configuration Modes

- Registered Mode: active (low, high): Q’ is feedback as input only, no dedicated input in this mode
- Combinational Mode: active (low, high) can feedback comb. Output (high, low) and/or dedicated input

GAL16V8 OLMC Structure



PAL16V8 OLMC Configuration Modes

Simple mode (combinational output)

combinational output

combinational output with feedback

dedicated input

Complex mode

combinational output, with feedback

combinational input from another OLMC

Registered mode: active (low, high): Q’ is feedback as input (no dedicated input)

Field Programmable Gate Arrays (FPGAs)

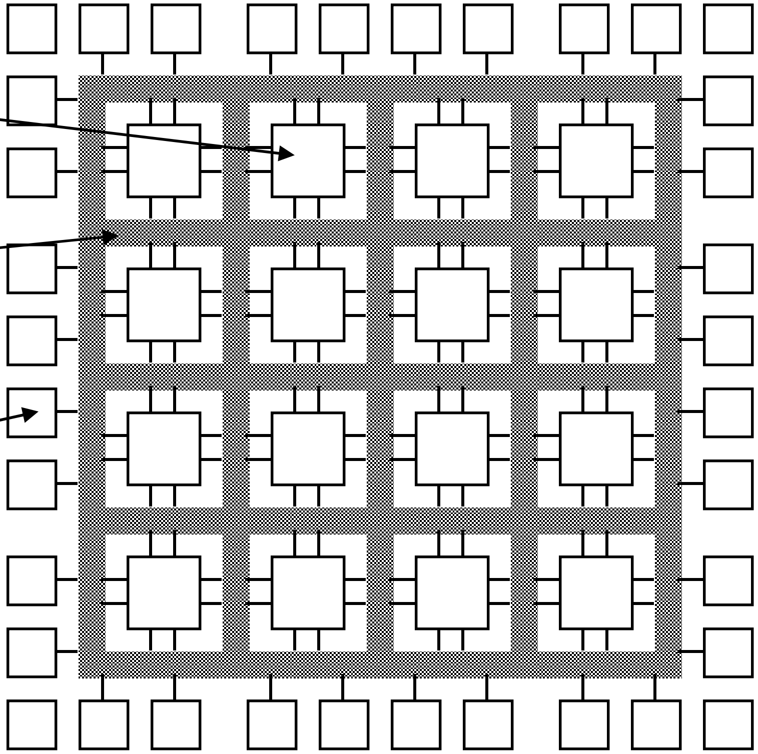
- FPGAs have much more logic than CPLDs
  - 2K to >10M equivalent gates
  - Requires different architecture
  - FPGAs can be RAM-based or Flash-based
    - RAM FPGAs must be programmed at power-on
      - External memory needed for programming data
      - May be dynamically reconfigured
    - Flash FPGAs store program data in non-volatile memory
      - Reprogramming is more difficult
      - Holds configuration when power is off

FPGA Structure

- Typical organization in 2-D array
  - Configurable logic blocks (CLBs) contain functional logic (could be similar to PAL22V10)
    - Combinational functions plus FFs
    - Complexity varies by device
  - CLB interconnect is either local or long line
    - CLBs have connections to local neighbors
    - Horizontal and vertical channels use for long distance
    - Channel intersections have switch matrix
  - IOBs (I/O logic Blocks) connect to pins
    - Usually have some additional C.L./FF in block

Field-Programmable Gate Arrays structure

- Logic blocks
  - To implement combinational and sequential logic
- Interconnect
  - Wires to connect inputs and outputs to logic blocks
- I/O blocks
  - Special logic blocks at periphery of device for external connections



- Key questions:
  - How to make logic blocks programmable?
  - How to connect the wires?
  - *After the chip has been fabbed*
  - HW problems: 7-19, 7-20, 7-22, 7-23, 7-26 due 22/9/07