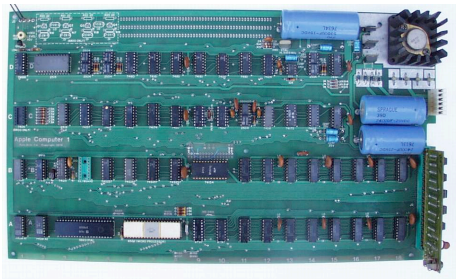
VLSI Combinational Circuit Design

Dr. Le Dung
Hanoi University of Science and Technology

A large digital logic circuit can be implemented by

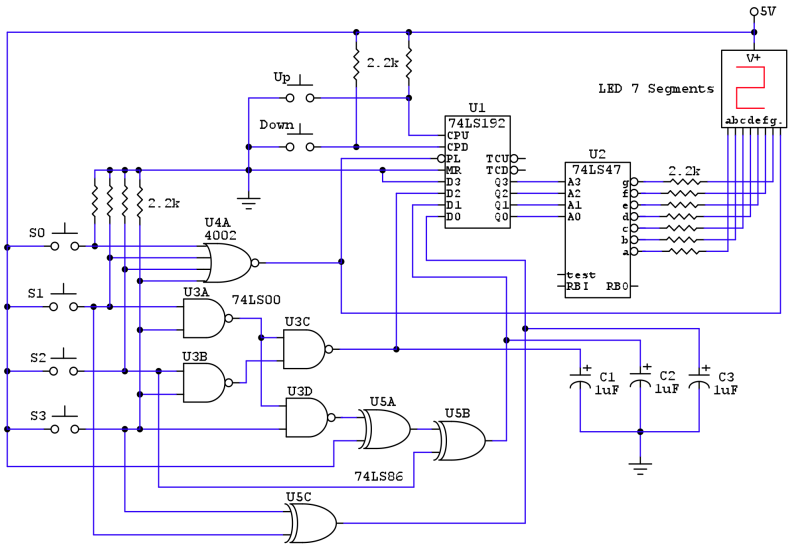
- SSI, MSI and LSI "off-the-shelf" parts
- VLSI Application-specific integrated circuit (ASIC)



Multiple ICs

Single IC

An example of the “off-the-shelf” parts design



Designing with “off-the-shelf” parts

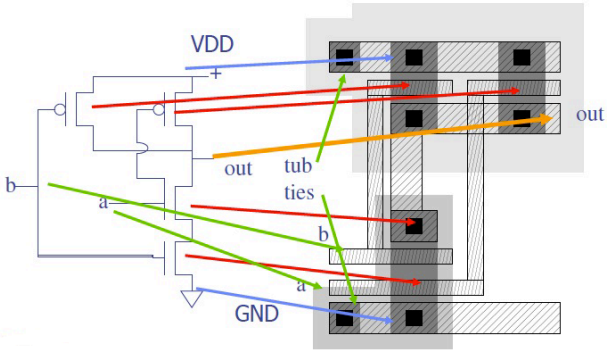
- The “off-the-shelf” parts = Commercial SSI, MSI and LSI modular logic integrated circuits (74xxx, 4xxx …)
- Quickly assembling a circuit board
- The number of parts and the cost per gate can become unacceptably large

VLSI ASIC design

- Using single VLSI IC
  - + Reduce PCB space and power requirements
  - + Reduce total cost
- Designing approaches :
  - + Full-custom design
  - + Semi-custom design
- Using hardware description language and CAD tools for designing

Full-custom design (1)

- Gate by gate designing with the physical layout of each individual transistor and the interconnections between them. Each transistor and each connection is designed individually as a set of rectangles



Full-custom design (2)

+ Both the circuit performance and the silicon area can be optimized (using ECAD tools)

- Extremely labor-intensive to implement
- Increasing manufacturing and design time
  → Time-to-market competition
- High cost of mask sets

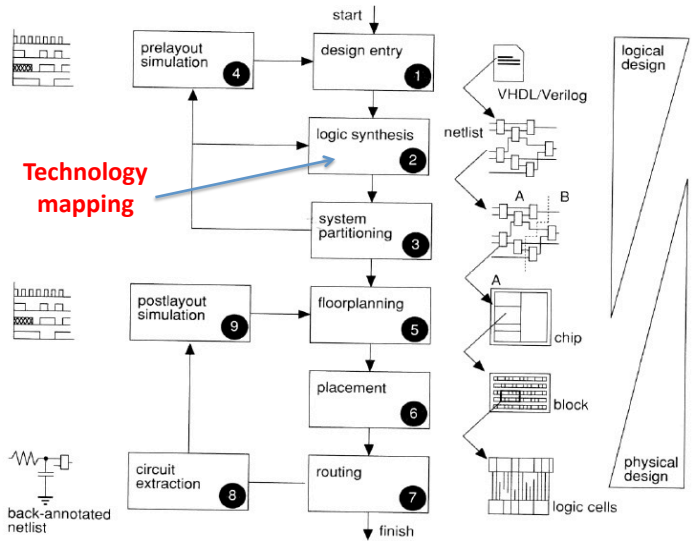
Semi-custom design

- Semi-custom device
  + has predesigned parts
- Semi-custom design approaches
  → Standard cell based design
  → Gate array based design
  → Programmable devices based design

Standard cell based design (1)

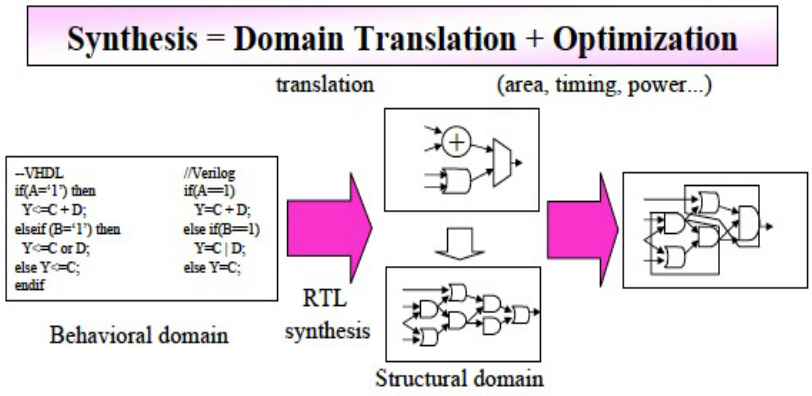
- Library of standard cells
  - + Each cell is a gate
  - + Same height, variable width, interleaved by routing channels
  - + All inputs at the top, all outputs at the bottom
- A designer selects cells from a design libarary, specifying where they should be placed on the IC and then dictating how they should be interconnected.
- Faster design of more complex building blocks
- Silicon foundries design and sell such optimized libraries for their processing technology

Basic process standard cell based design



Hardware Description Language Synthesis

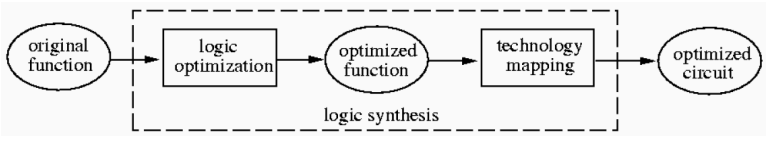
Translate HDL descriptions into logic gate networks in a particular library



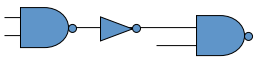
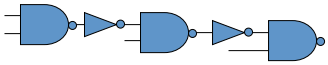
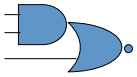
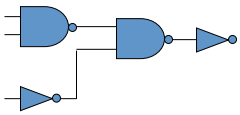
Logic Synthesis Phases

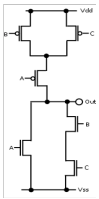
- **Logic optimization** *transforms current gate-level* network into an equivalent gate-level network more suitable for technology mapping.

- **Technology mapping** *transforms the gate-level* network into a netlist of gates (from library) which minimizes total cost.



Library of standard cell

CELLS	COST	DELAY	SYMBOL	PATTERN
INVERTER	2	1		
NAND2	3	1.4		
NAND3	4	1.8		
NAND4	5	2.2		
AOI21	4	1.8		



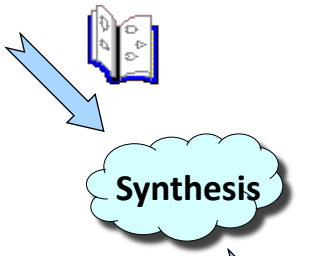
CMOS AND-OR-Invert Gate

An example of standard cell technology mapping (1)

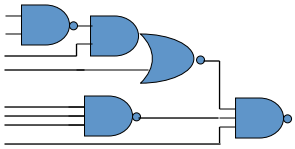
a b c d e f g h	
0 - 0 - - - - -	a'*c' +
- 0 0 - - - - -	b'*c' +
- - - 1 - - - -	d +
- - - - 1 1 1 -	e*f*g +
1 0 0 - - - - -	a*b'*c' +
- - - - - - - 0	h +
0 - - 1 - - - -	a'*d

An example of standard cell technology mapping (2)

```
a b c d e f g h
0 - 0 - - - - -
- 0 0 - - - - -
- - - 1 - - - -
- - - - 1 1 1 -
1 0 0 - - - - -
- - - - - - - 0
0 - - 1 - - - -
```



Netlist of gates (from library) which minimizes total cost.



Phases of synthesis (1/3)

1. **Independent transformations (optimization)**:

a b c d e f g h	
0 - 0 - - - - -	a'*c' +
- 0 0 - - - - -	b'*c' +
- - - 1 - - - -	d +
- - - - 1 1 1 -	e*f*g +
1 0 0 - - - - -	a*b'*c' +
- - - - - - - 0	h' +
0 - - 1 - - - -	a'*d

Phases of synthesis (1/3)

1. **Independent transformations (optimization)**:

a b c d e f g h	
0 - 0 - - - - -	a'*c' +
- 0 0 - - - - -	b'*c' +
1 0 0 - - - - -	a*b'*c' +
- - - 1 - - - -	d +
0 - - 1 - - - -	a'*d +
- - - - 1 1 1 -	e*f*g +
- - - - - - - 0	h'

Phases of synthesis (1/3)

1. **Independent transformations (optimization)**:

a b c d e f g h	
0 - 0 - - - - -	a'*c' +
- 0 0 - - - - -	b'*c' +
- - - 1 - - - -	d +
0 - - 1 - - - -	a'*d +
- - - - 1 1 1 -	e*f*g +
- - - - - - - 0	h'

Dr. Le Dung Hanoi University of Science and Technology

Phases of synthesis (1/3)

1. **Independent transformations (optimization)**:

a b c d e f g h	
0 - 0 - - - - -	a'*c' +
- 0 0 - - - - -	b'*c' +
- - - 1 - - - -	d +
- - - - 1 1 1 -	e*f*g +
- - - - - - - 0	h'

Phases of synthesis (1/3)

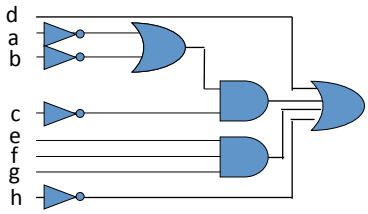
1. **Independent transformations (optimization)**:

a b c d e f g h	
0 - 0 - - - - -	(a'+b')*c' +
- 0 0 - - - - -	
- - - 1 - - - -	d +
- - - - 1 1 1 -	e*f*g +
- - - - - - - 0	h'

Phases of synthesis (1/3)

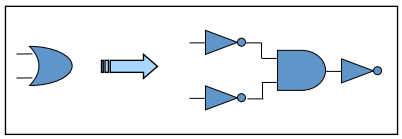
1. **Independent transformations (optimization)**:

Original netlist

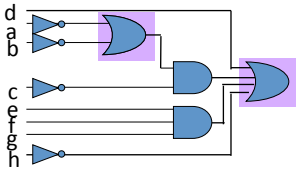


Phases of synthesis (2/3)

- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

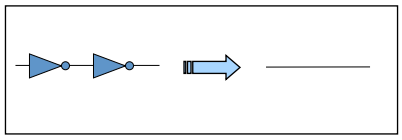


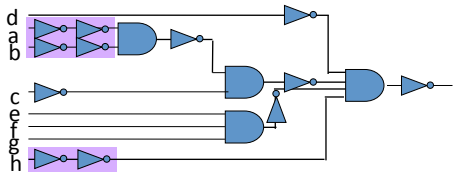
Original netlist



Phases of synthesis (2/3)

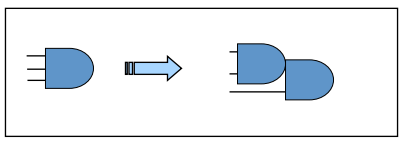
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

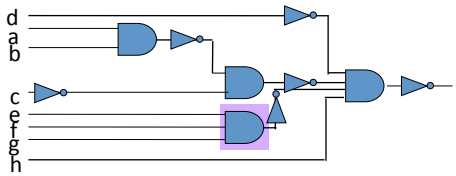




Phases of synthesis (2/3)

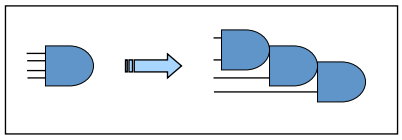
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

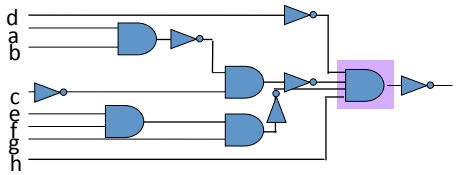




Phases of synthesis (2/3)

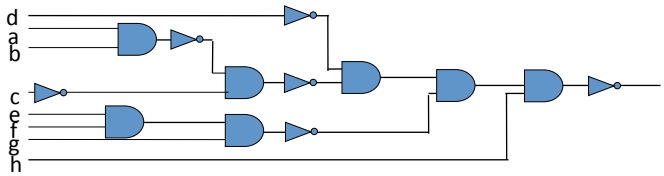
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**





Phases of synthesis (2/3)

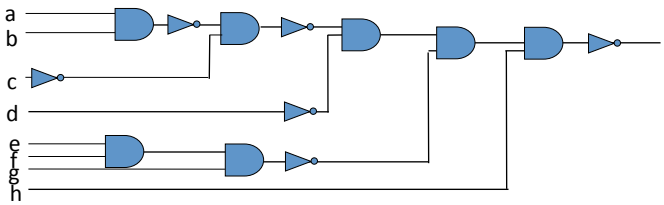
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**



Dr. Le Dung Hanoi University of Science and Technology

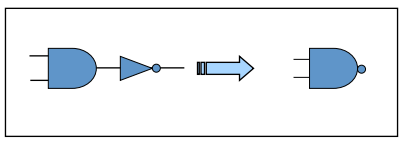
Phases of synthesis (2/3)

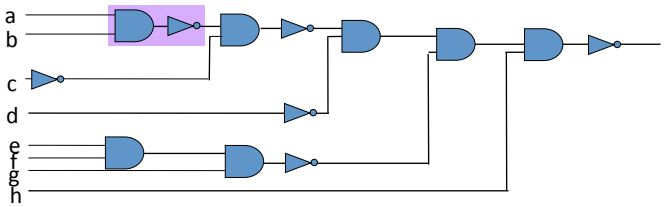
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**



Phases of synthesis (2/3)

- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

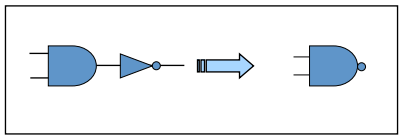


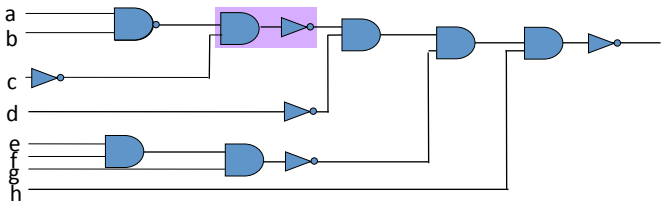


Dr. Le Dung Hanoi University of Science and Technology

Phases of synthesis (2/3)

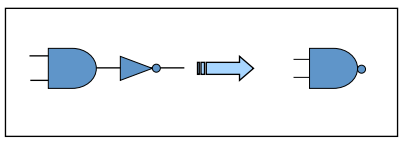
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

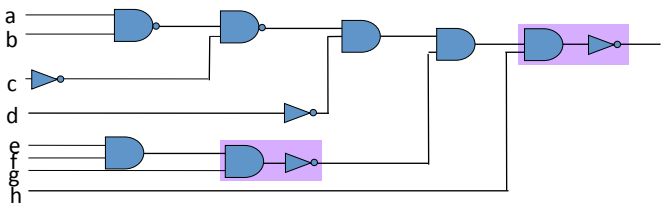




Phases of synthesis (2/3)

- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

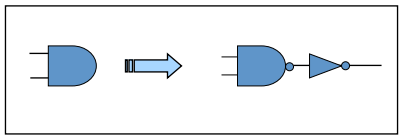


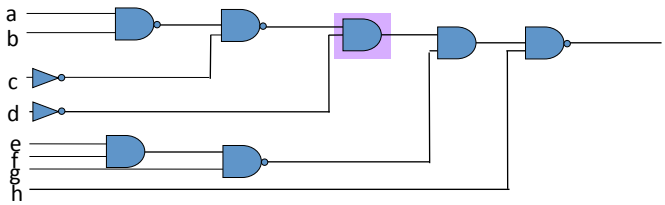


Dr. Le Dung Hanoi University of Science and Technology

Phases of synthesis (2/3)

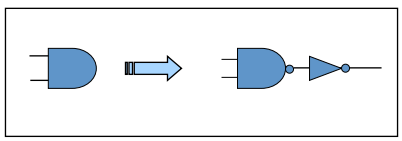
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

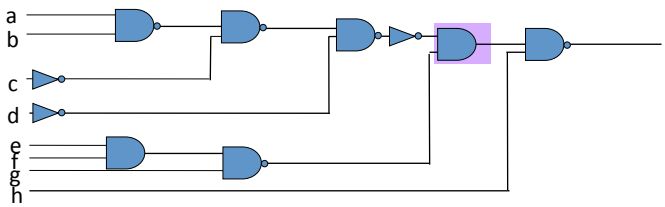




Phases of synthesis (2/3)

- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

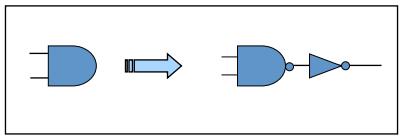


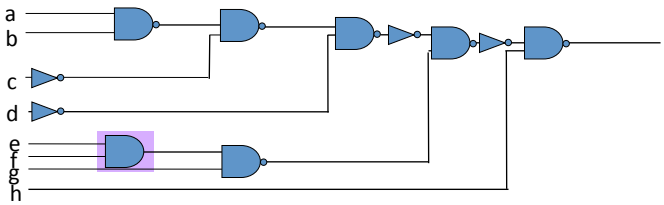


Dr. Le Dung Hanoi University of Science and Technology

Phases of synthesis (2/3)

- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

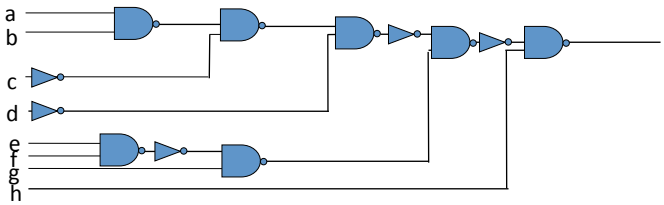




Phases of synthesis (2/3)

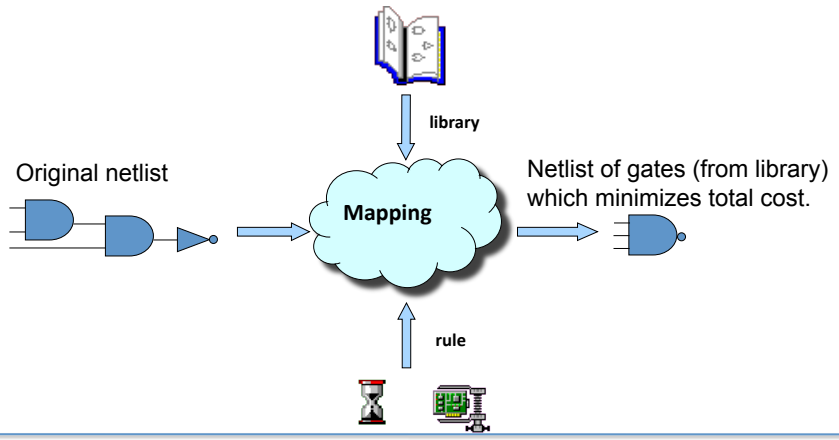
- **Decomposition using base functions**:
  - **Decompose to a network NAND2/NOT**

Subject Graph



What is technology mapping ?

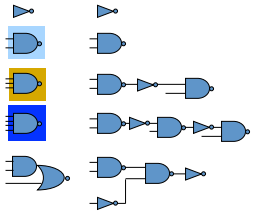
- **Technology mapping is the problem of optimising a network for area or delay, using only library cells.**

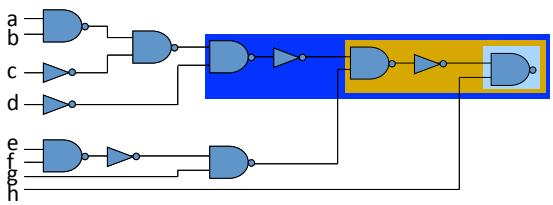


Phases of synthesis (3/3)

- **Technology mapping**:

  Greedy algorithm → Greedy search

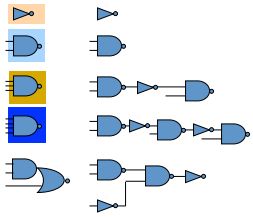


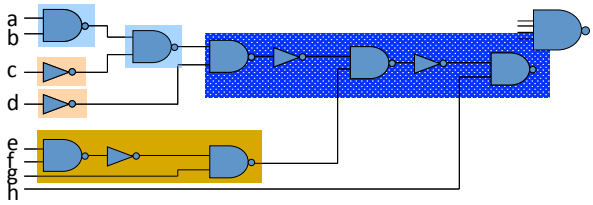


Subject Graph

Phases of synthesis (3/3)

- **Technology mapping**:
  - Greedy search

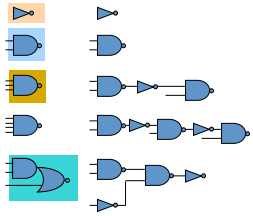


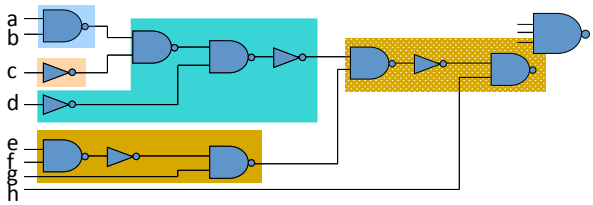


Subject Graph

Phases of synthesis (3/3)

- **Technology mapping**:
  - Greedy search

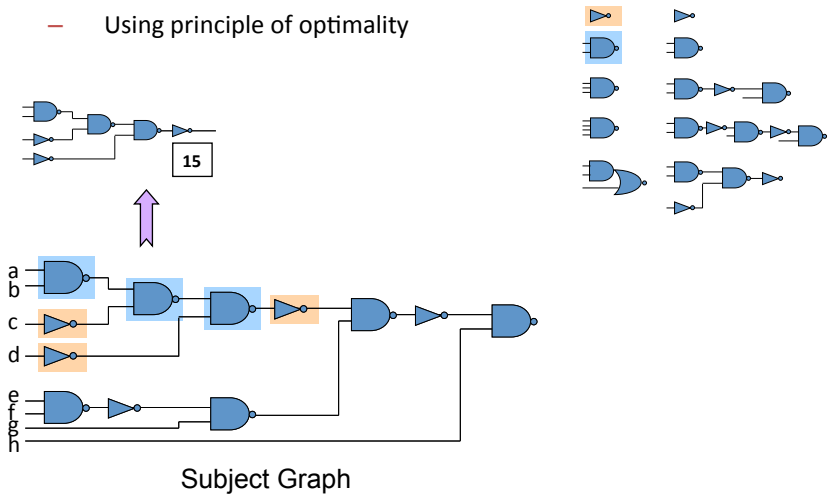




Subject Graph

Phases of synthesis (3/3)

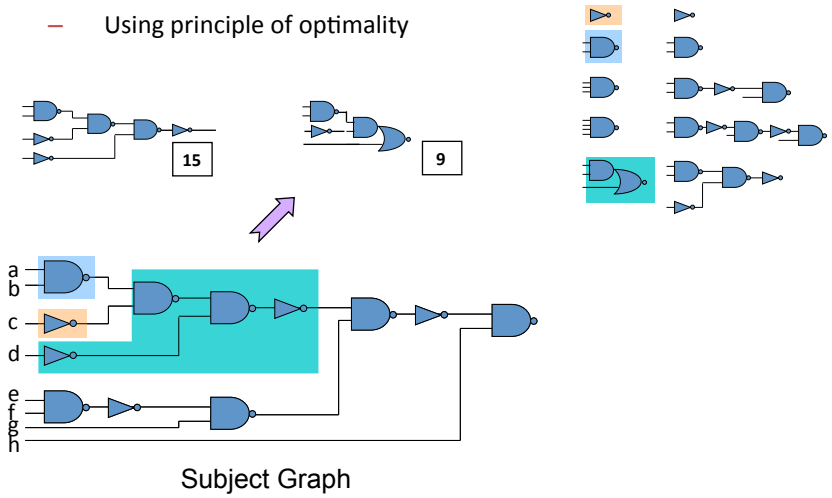
- **Technology mapping:**
  - Using principle of optimality



Subject Graph

Phases of synthesis (3/3)

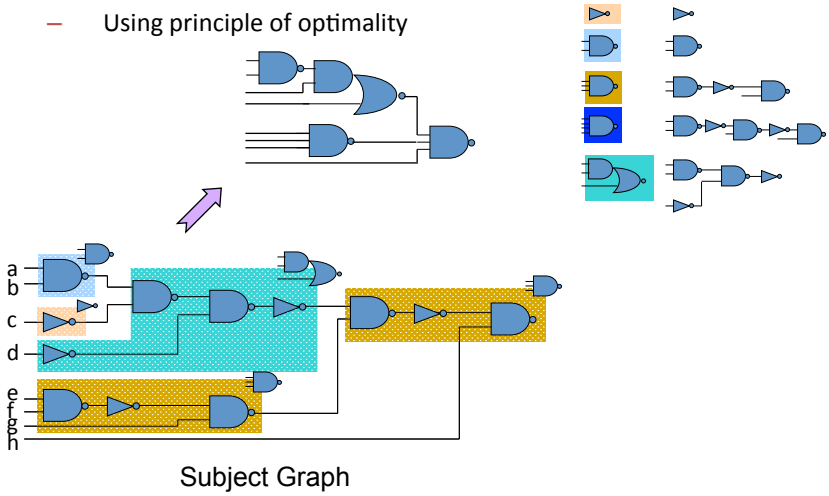
- **Technology mapping:**
  - Using principle of optimality



Subject Graph

Phases of synthesis (3/3)

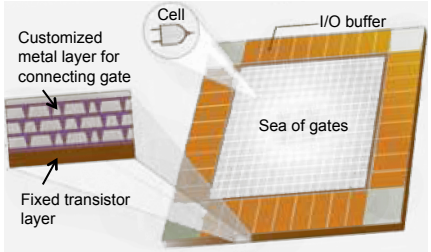
- **Technology mapping**:
  - Using principle of optimality



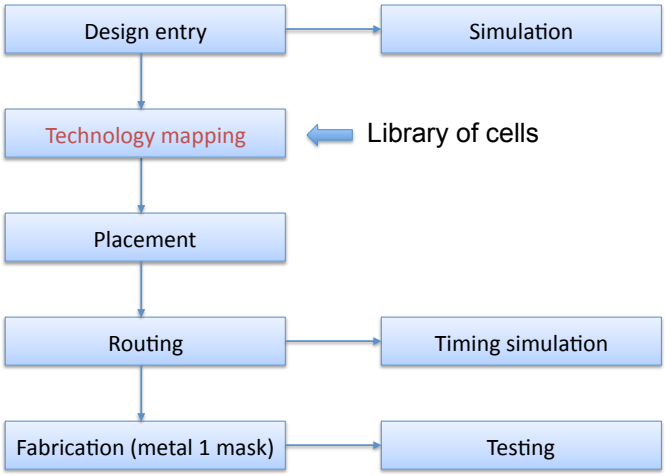
Subject Graph

Gate array based design

+ A gate array or uncommitted logic array (ULA) circuit is prefabricated with a number of unconnected logic gates (cells).
+ CMOS transistors with fixed length and width are placed at regular predefined positions and manufactured on a wafer, usually called a master slice (→ sea of gates).
+ Creation of a circuit with a specified function is accomplished by adding a final surface layer or layers of metal interconnects to the chips on the master slice late in the manufacturing process, joining these elements to allow the function of the chip to be customized as desired
  → reducing the designing time
  → reducing the mask costs
+ Disadvantages
  - slow clock speed
  - wasted chip area



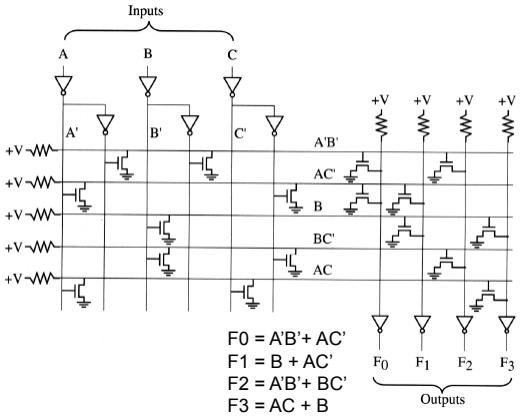
Gate array based design flow

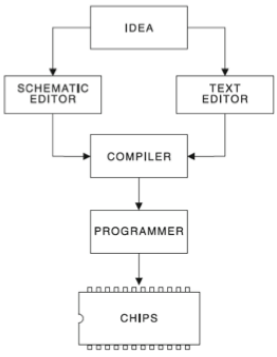


Programmable Device Based Design

Based on programmable devices:

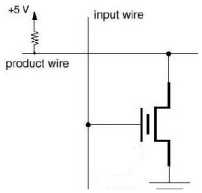
The interconnection layers are personalized by electronic means for a specific application. This work usually can be done by end-users.





Programmable Elements

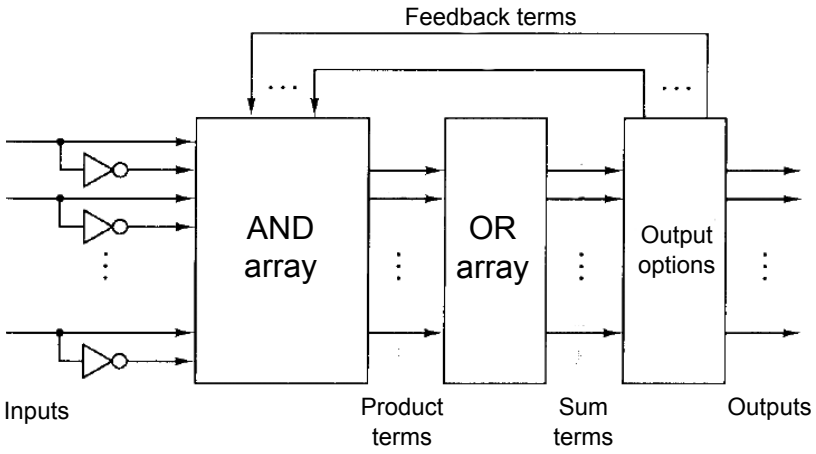
+ Fuse
+ Antifuse
+ Switch
+ Volatile
+ Non-volatile
+ One Time Programmable
+ Reprogrammable (Memory-based)



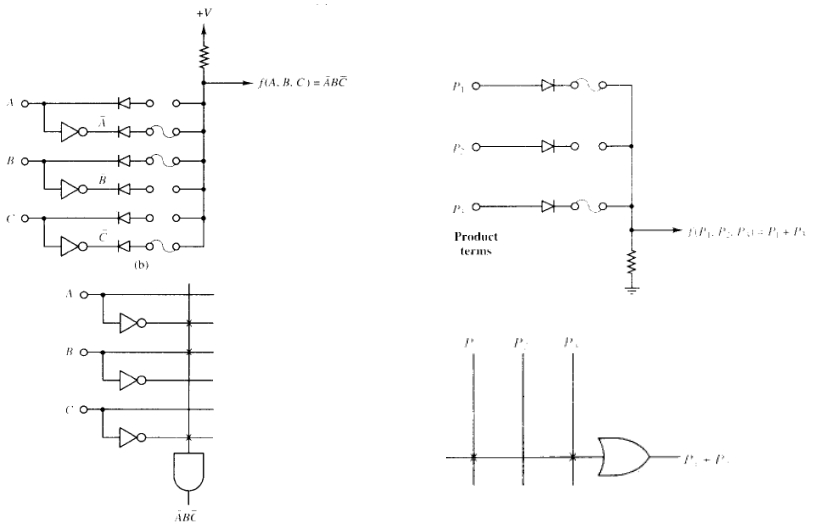
Programmable Devices

- **Simple Programmable Logic Device:**
  + Programmable read only memory (PROM)
  + Field Programmable logic array (FPLA or PLA)
  + Programmable array logic (PAL)
  + Generic array logic (GAL)
- **Complex programmable logic device (CPLD)**
- **Field programmable gate array (FPGA)**
- **Field programmable interconnect (FPIC)**

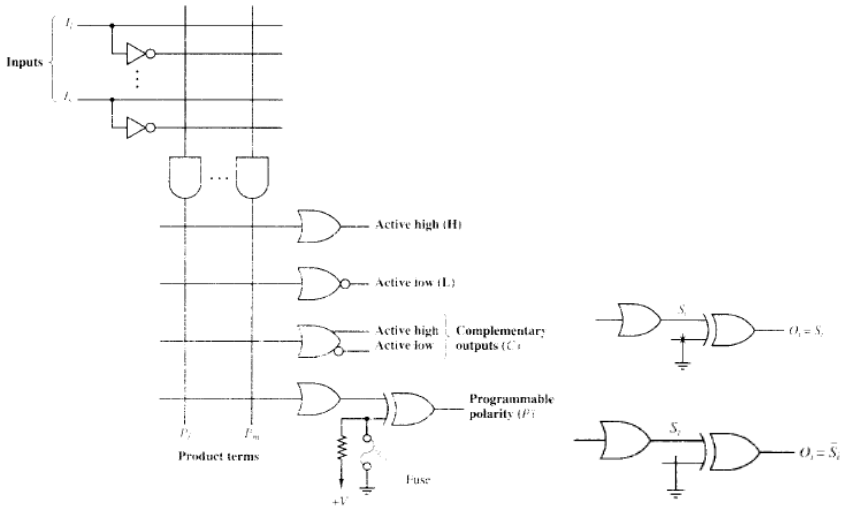
Basic SPLD organization



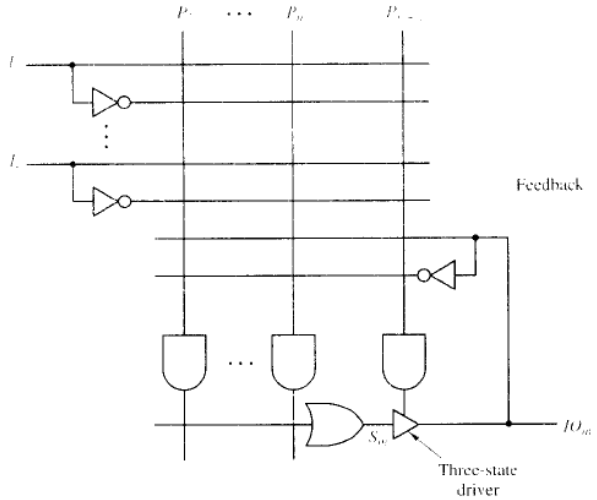
Fuse-based programmable AND – OR Array



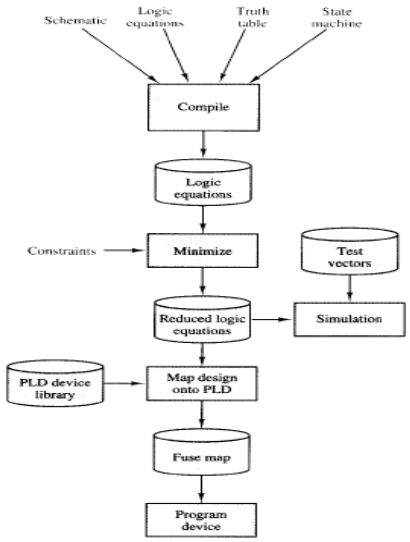
Output Polarity Options



Bidirectional Pins and Feedback line



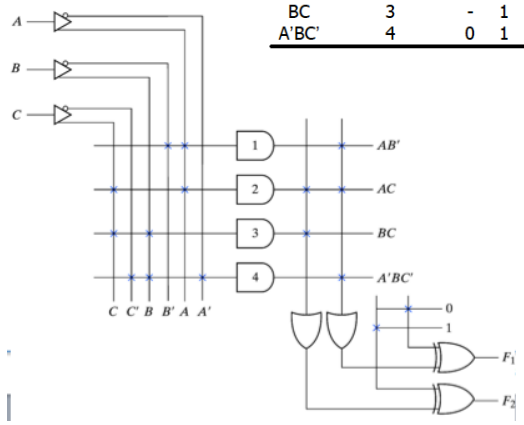
PLD Design Process



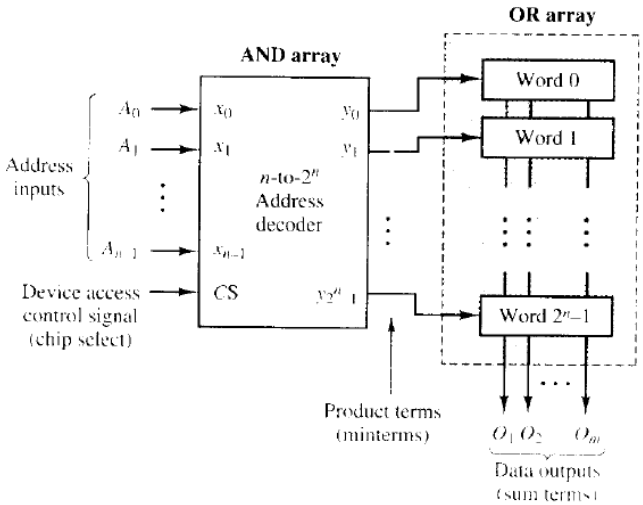
Combinational Circuit is implemented on SPLD

F1 = AB′ + AC + A′BC′
F2 = (AC +BC)′

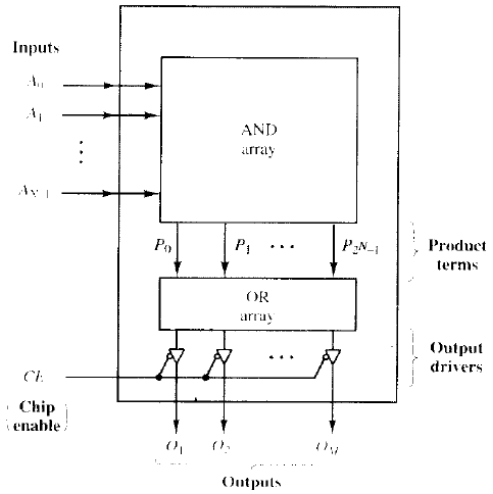
	Product Term	Inputs A	B	C	Outputs (T) (C) F_1	F_2
AB'	1	1	0	-	1	-
AC	2	1	-	1	1	1
BC	3	-	1	1	-	1
A'BC'	4	0	1	0	1	-



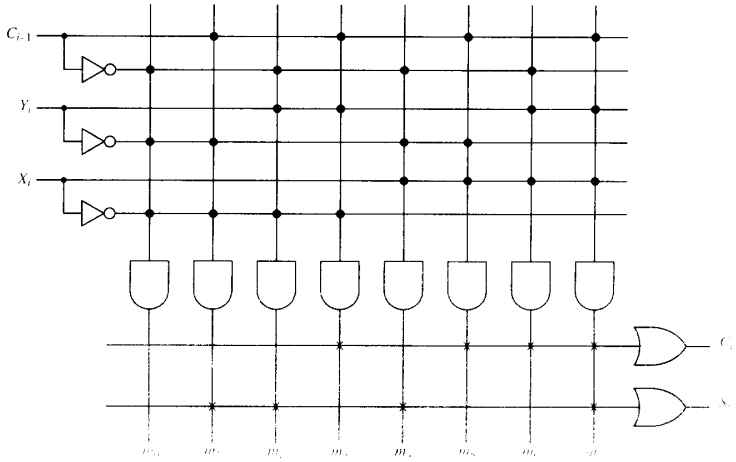
PROM = Read-Only-Memory



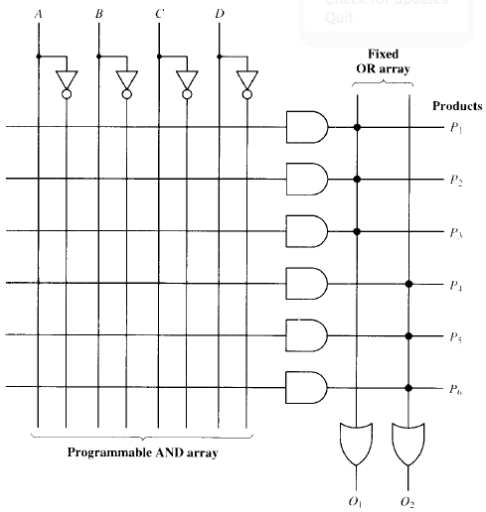
PROM = PLD with fixed AND array



Full-adder on PROM



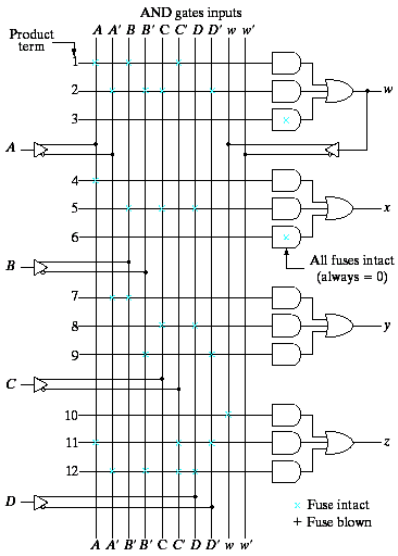
PAL



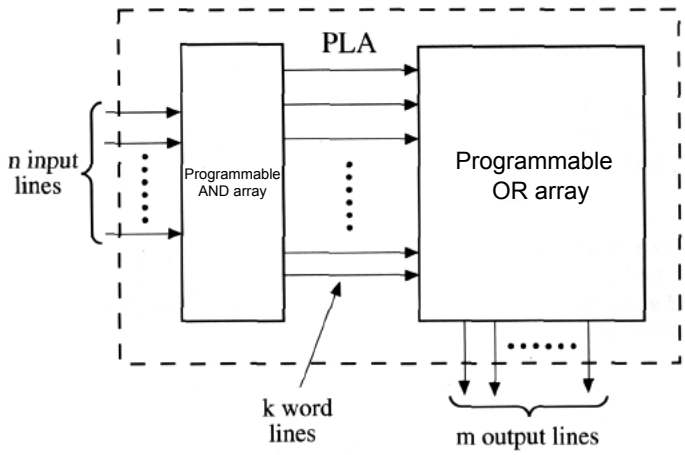
Combinational Circuit is implemented PAL

w=∑(2,12,13) x=∑(7,8,9,10,11,12,13,14,15)
y=∑(0,2,3,4,5,6,7,8,10,11,15) z=∑(1,2,8,12,13)

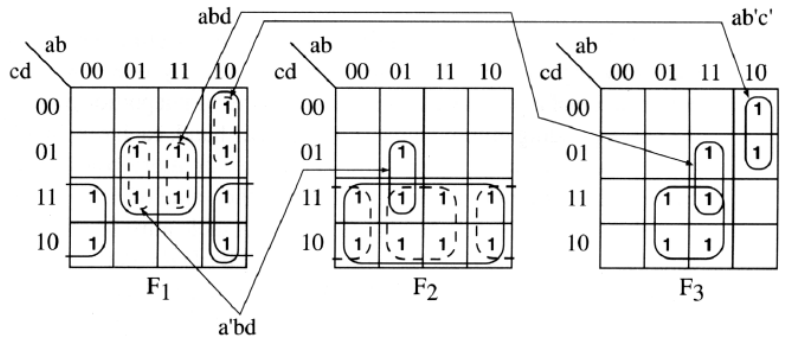
Product Term	AND Inputs A	B	C	D	W	Outputs
1	1	1	0	-	-	w = ABC'
2	0	0	1	0	-	+ A'B'CD'
3	-	-	-	-	-	
4	1	-	-	-	-	x = A
5	-	1	1	1	-	+ BCD
6	-	-	-	-	-	
7	0	1	-	-	-	y = A'B
8	-	-	1	1	-	+ CD
9	-	0	-	0	-	+ B'D'
10	-	-	-	-	1	z = w
11	1	-	0	0	-	+ AC'D'
12	0	0	0	1	-	+ A'B'C'D



FPLA



Combinational Circuit is implemented on FPLA (1)



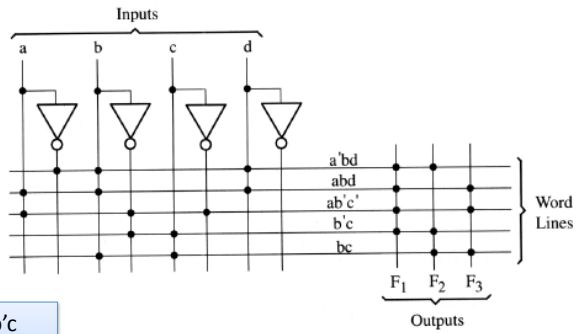
Minimize each function separately
→ 8 product terms
F1 = bd + b'c + ab'
F2 = c + a'bd
F3 = bc + ab'c'+ abd

Multiple-Output Optimization
→ 5 product terms
F1 = abd + a'bd + ab'c'+ b'c
F2 = a'bd + b'c + bc
F3 = abd + ab'c'+ bc

Combinational Circuit is implemented on FPLA (2)

Reduced PLA Table

a b c d	F_1 F_2 F_3
0 1 – 1	1 1 0
1 1 – 1	1 0 1
1 0 0 –	1 0 1
– 0 1 –	1 1 0
– 1 1 –	0 1 1



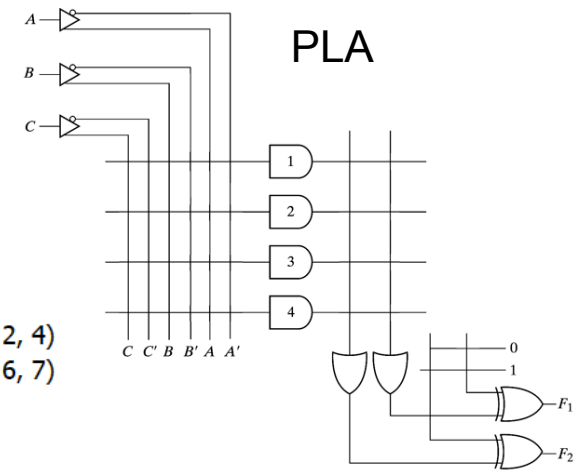
F1 = abd + a'bd + ab'c'+ b'c
F2 = a'bd + b'c + bc
F3 = abd + ab'c'+ bc

Exercise

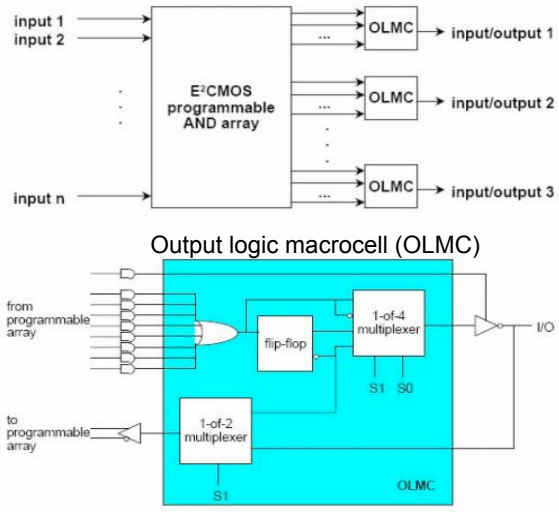
Implement the two functions with PLA

$F_1(A, B, C) = \sum (0, 1, 2, 4)$

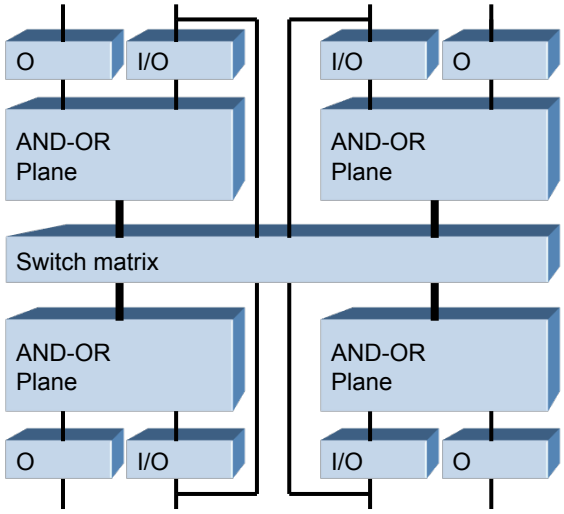
$F_2(A, B, C) = \sum (0, 5, 6, 7)$



Generic Array Logic architecture



CPLD architecture



FPGA architecture (1)

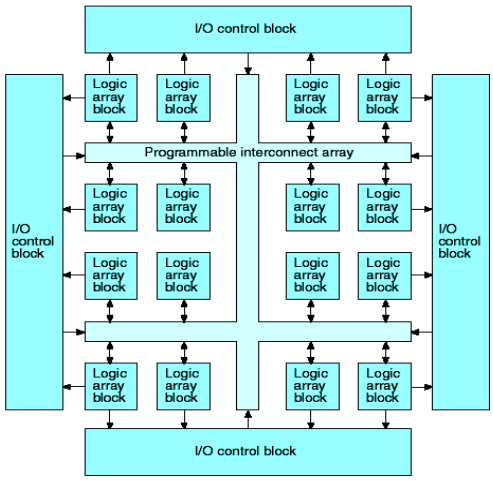
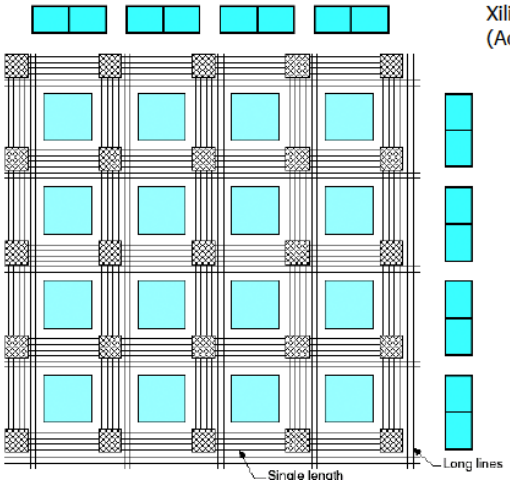


Fig. 6-28 Altera® MAX 7000™ Structure (Reprinted with Permission of Altera Corporation,© Altera Corp., 1991)

FPGA architecture (2)

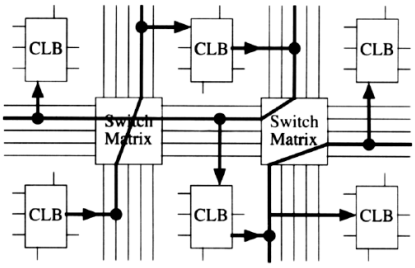


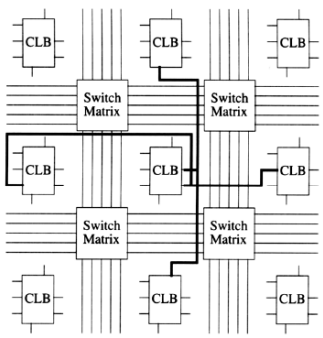
Xilinx® XC4000™ FPGA Structure
(Adapted with Permission of Xilinx, Inc.)

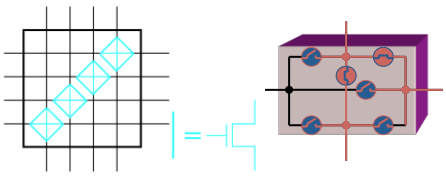
- Configurable logic block (CLB)
- Input/Output Block (IOB)
- Switch matrix

FPGA architecture (3)

Switch Matrix and interconnection





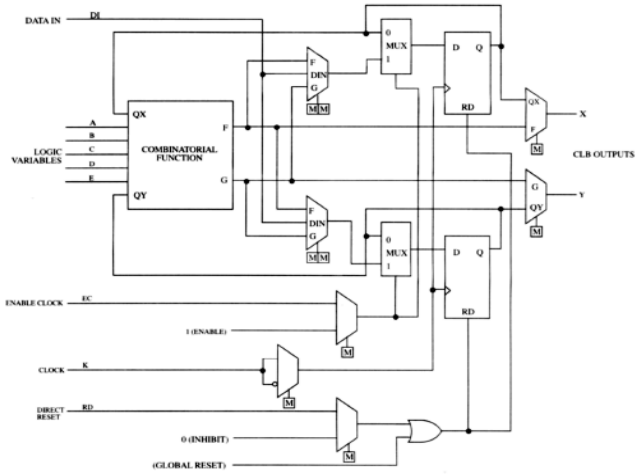


Long lines :
- Across whole chip
- High fan-out, low skew
- Suitable for global signals (CLK) and buses
- 2 tri-states per CLB for busses

FPGA architecture (4)

Configurable Logic Block (CLB)

5 logic inputs
Data input (DI)
Clock (K)
Clock enable (EC)
Direct reset (RD)
2 outputs (X,Y)

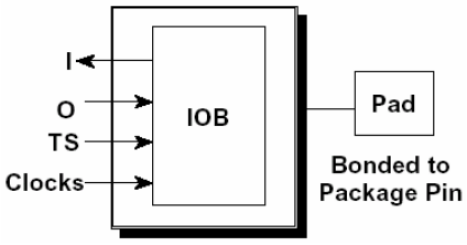


FPGA architecture (5)

I/O Block (IOB)

Periphery of identical I/O blocks

- Input, output, or bidirectional
- Registered, latched, or combinational
- Three-state output
- Programmable output slew rate



FPGA development toolkit