Physical Implementation

물적 구현

Yoonjin Kim

Full Professor

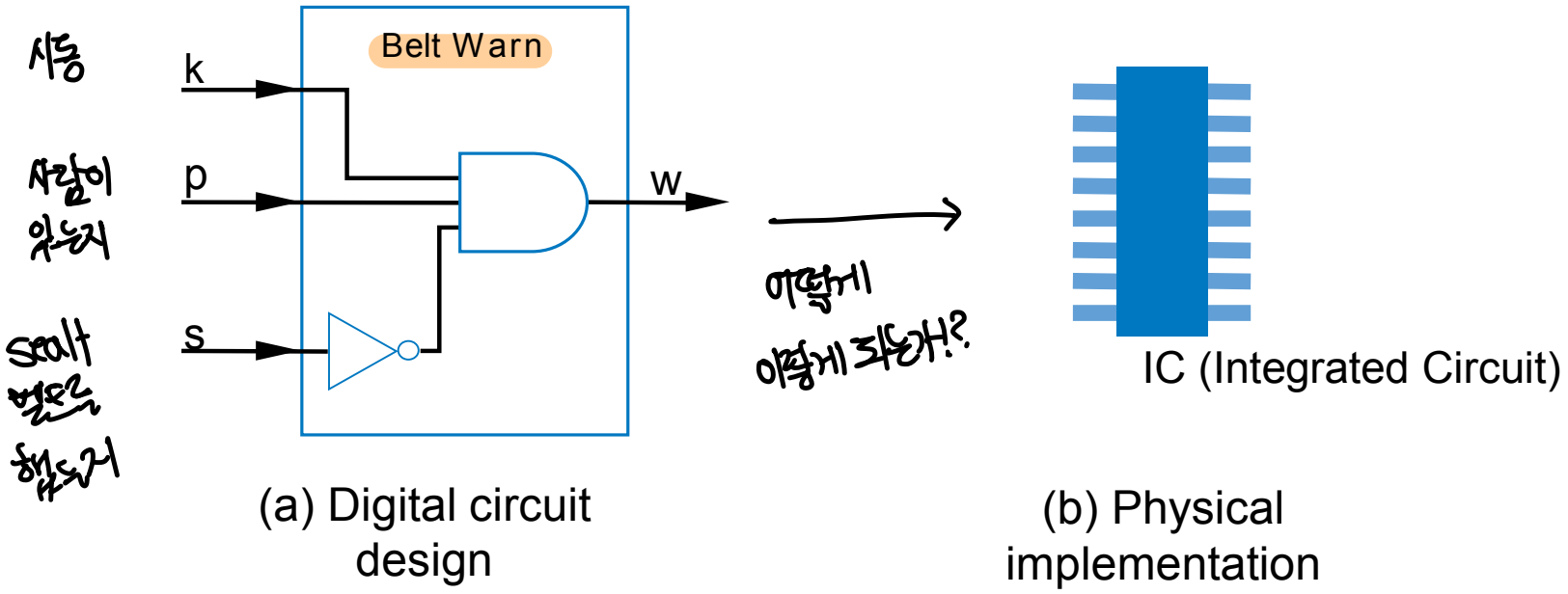
Division Computer Science
Sookmyung Women's University

Outline

- Introduction
- SSI(Small Scale Integration) technologies
- VLSI (Very Large Scale Integration) technologies
- Determining clock frequency
- Programmable/reconfigurable IC technology
  - PLD (Programmable Logic Device)
  - FPGA (Field Programmable Gate Array)
- Technology comparison
- Entire IC design flow

Introduction

- We eventually need to implement the circuit on a physical device from just design.
  - How do we get from (a) to (b)?

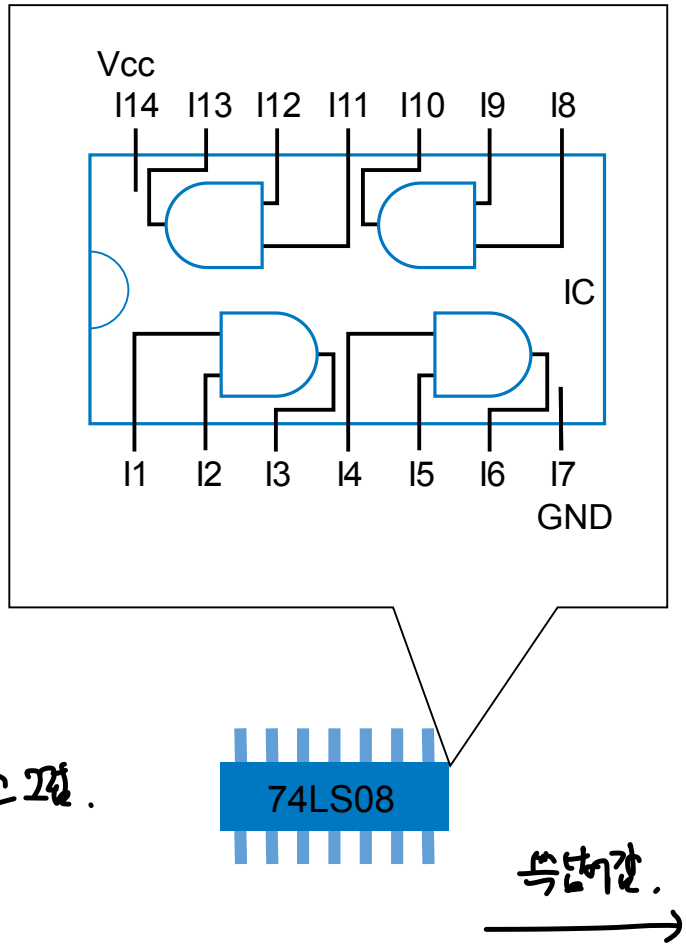


(a) Digital circuit design

(b) Physical implementation

SSI(Small Scale Integration) Technologies

- Off-the-shelf logic IC
  - Logic IC has a few gates, connected to IC's pins
    - Known as Small Scale Integration (SSI)
  - Popular logic IC series: 7400
    - Originally developed 1960s
      - At that time, each IC cost $1000
      - Today, costs just tens of cents



SSI(Small Scale Integration) Technologies

7400-Series Logic ICs

TABLE 7.1: Commonly used 7400-series ICs.

Part	Description	Pins
74LS00	Four 2-input NAND	14
74LS02	Four 2-input NOR	14
74LS04	Six inverters	14
74LS08	Four 2-input AND	14
74LS10	Three 3-input NAND	14
74LS11	Three 3-input AND	14
74LS14	Six inverters (Schmitt trigger)	14
74LS20	Two 4-input NAND	14
74LS27	Three 3-input NOR	14
74LS30	One 8-input NAND	14
74LS32	Four 2-input OR	14
74LS74	Two D flip-flop, positive edge triggered, with preset and reset	14
74LS83	4-bit binary full-adder	16
74LS85	4-bit magnitude comparator	16

Source: www.digikey.com

SSI(Small Scale Integration) Technologies

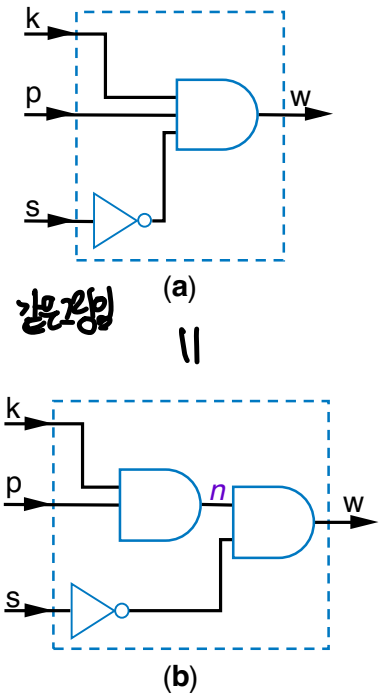
- Example: Seat belt warning light using off-the-shelf 7400 ICs
  - Option 1: Use one 74LS08 IC having 2-input AND gates, and one 74LS04 IC having inverters

TABLE 7.1: Commonly used 7400-series ICs.

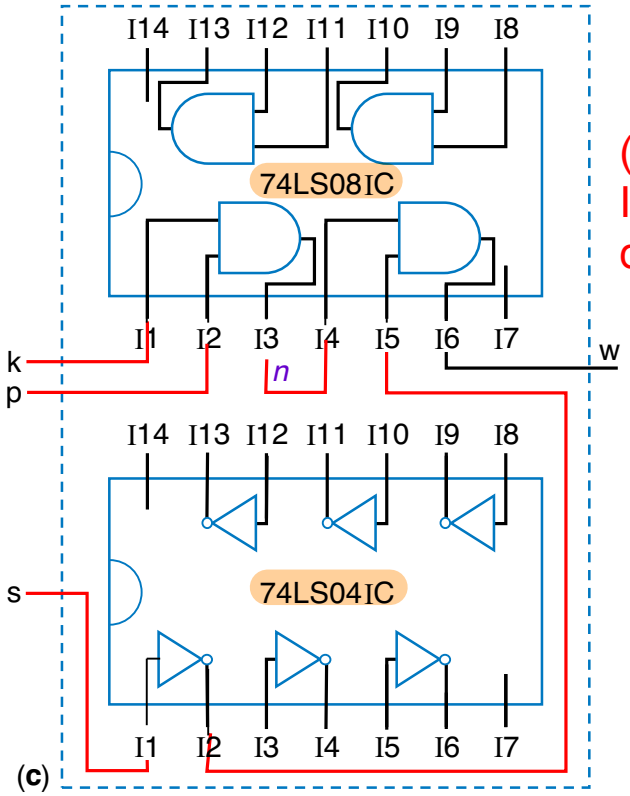
Part	Description	Pins
74LS00	Four 2-input NAND	14
74LS02	Four 2-input NOR	14
74LS04	Six inverters	14
74LS08	Four 2-input AND	14
74LS10	Three 3-input NAND	14
74LS11	Three 3-input AND	14
74LS14	Six inverters (Schmitt trigger)	14
74LS20	Two 4-input NAND	14
74LS27	Three 3-input NOR	14
74LS30	One 8-input NAND	14
74LS32	Four 2-input OR	14
74LS74	Two D flip-flop, positive edge triggered, with preset and reset	14
74LS83	4-bit binary full-adder	16
74LS85	4-bit magnitude comparator	16

Source: www.digikey.com

(a) Desired circuit



(b) Decompose into 2-input AND gates



(c) Connect ICs to create desired circuit

VLSI (Very Large Scale Integration) Technologies

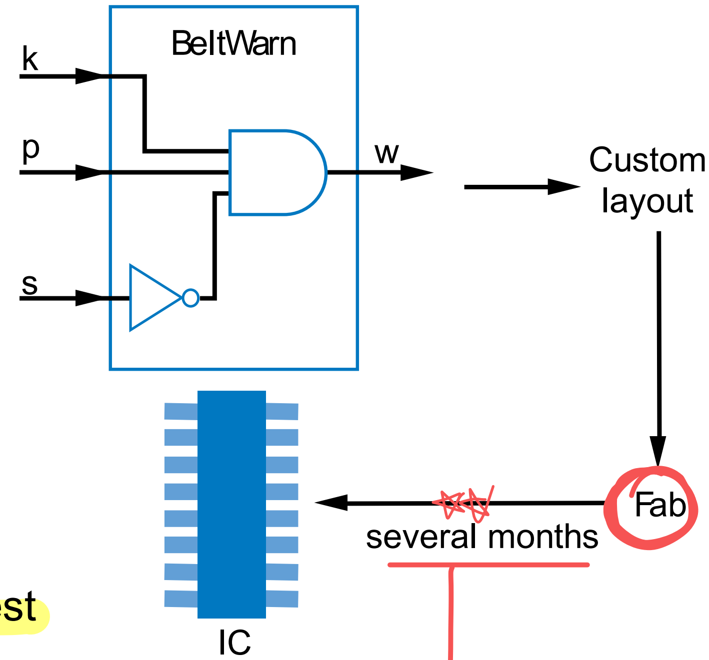
2가지 방식 A. Full-custom / B. semicustom from HDL

→ 자신만의 IC 제작가능

- We can manufacture our own IC
  - Long time and millions of dollars
  - A. Full-custom or B. semicustom from HDL
    - HDL: Hardware Description Language

→ 설계자가 일일히 고쳐서 만드는 것 → H.W의 코딩.

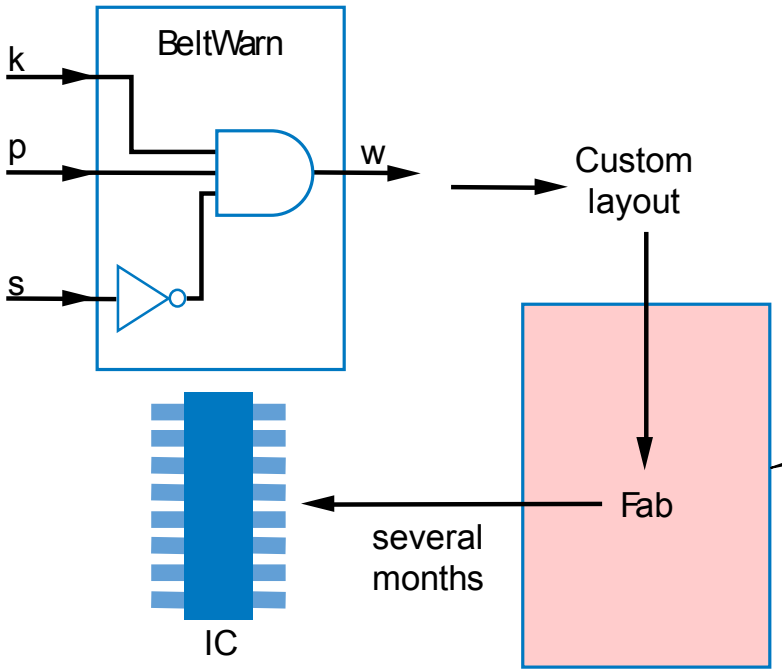
- A. Full-custom IC
  - Making a *full custom layout*
    - They are not synthesized circuit from HDL.
    - Layout describes the location and size of every transistor and wire
    - By using CAD (Computer Aided Design) tools
  - Reserved for special ICs that demand the very best performance or the very smallest size/power.
  - However, Fabrication setup costs, process time, difficulty are very high.



(실제 CPU ⇒ Full-custom IC 70%, semicustom from 30%) 고성능은 Full-custom으로 만듦...

VLSI (Very Large Scale Integration) Technologies

- A. Full-custom IC



Fab 이란!?

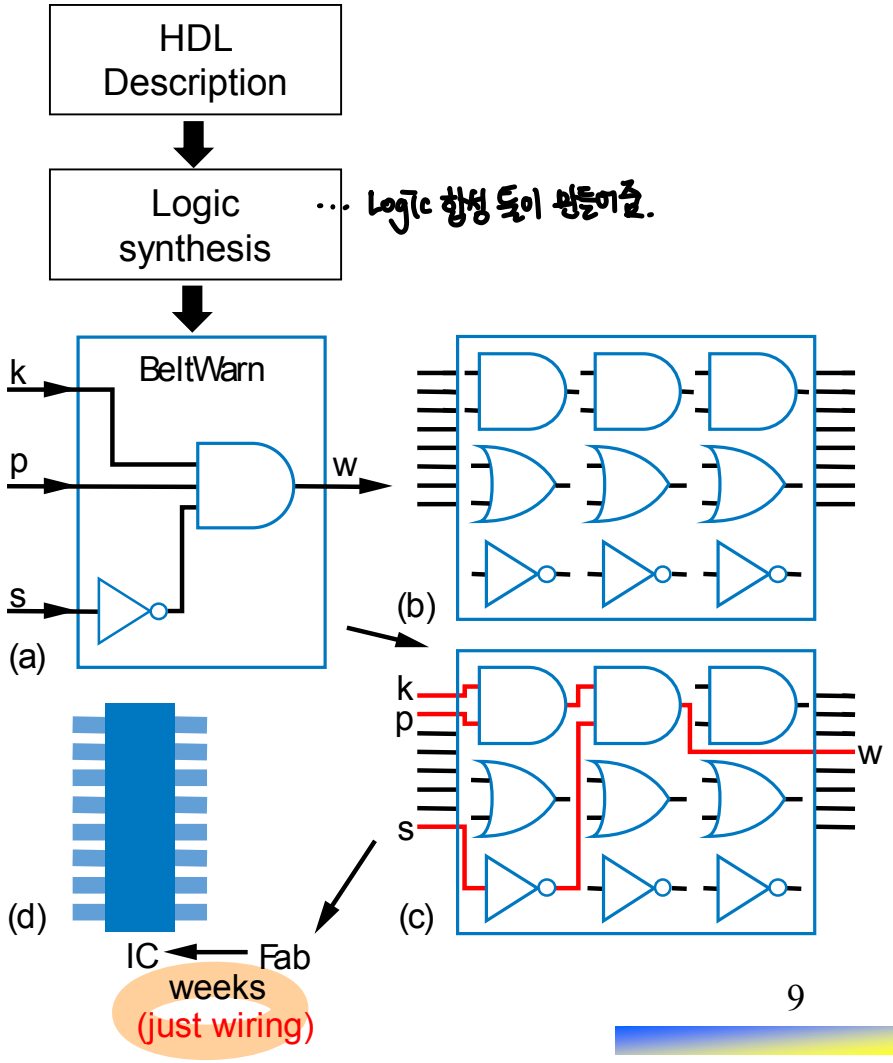
IC manufacturers (foundry companies) fabricate ICs based on their own process technologies (currently, MOSFET technology).

ex) Samsung 7nm process,
SK Hynix 10nm process,
Intel 14nm process,
TSMC12nm process,
etc.

VLSI (Very Large Scale Integration) Technologies – Gate Array

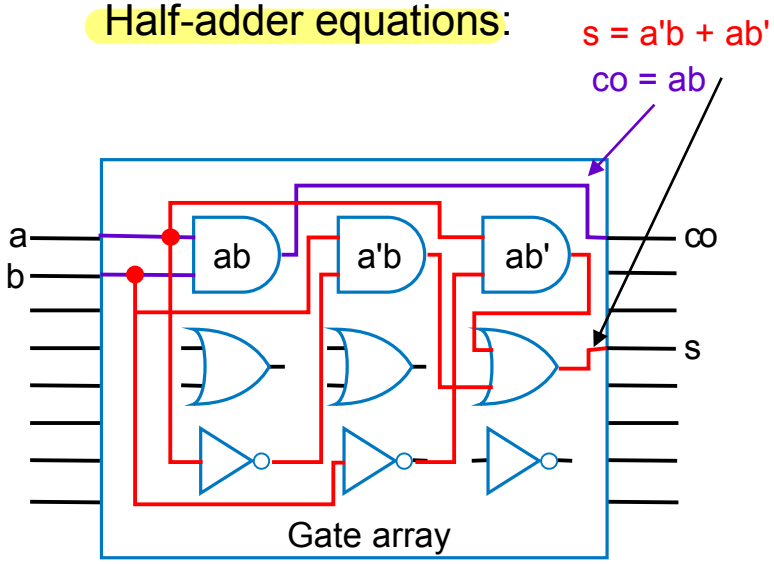
- B. Semicustom IC from Hardware Description Language (HDL)
  - 1. Gate array or 2. standard cell
  - They are implemented with gates synthesized from HDL.

- B.1. Gate array
  - Series of gates already layout on chip ⇒ 칩에 gate layout은 이미 되어있음.
  - We just wire them together ⇒ wire만 하는 것.
    - Using CAD tools
  - Vs. full-custom
    - Cheaper and quicker to design.
    - But worse performance, size, power
  - Very popular



VLSI (Very Large Scale Integration) Technologies – Gate Array

- B.1. Gate array
  - Example: Mapping a half-adder to a gate array



↳ 안쪽은 inverter, OR gate 존재.

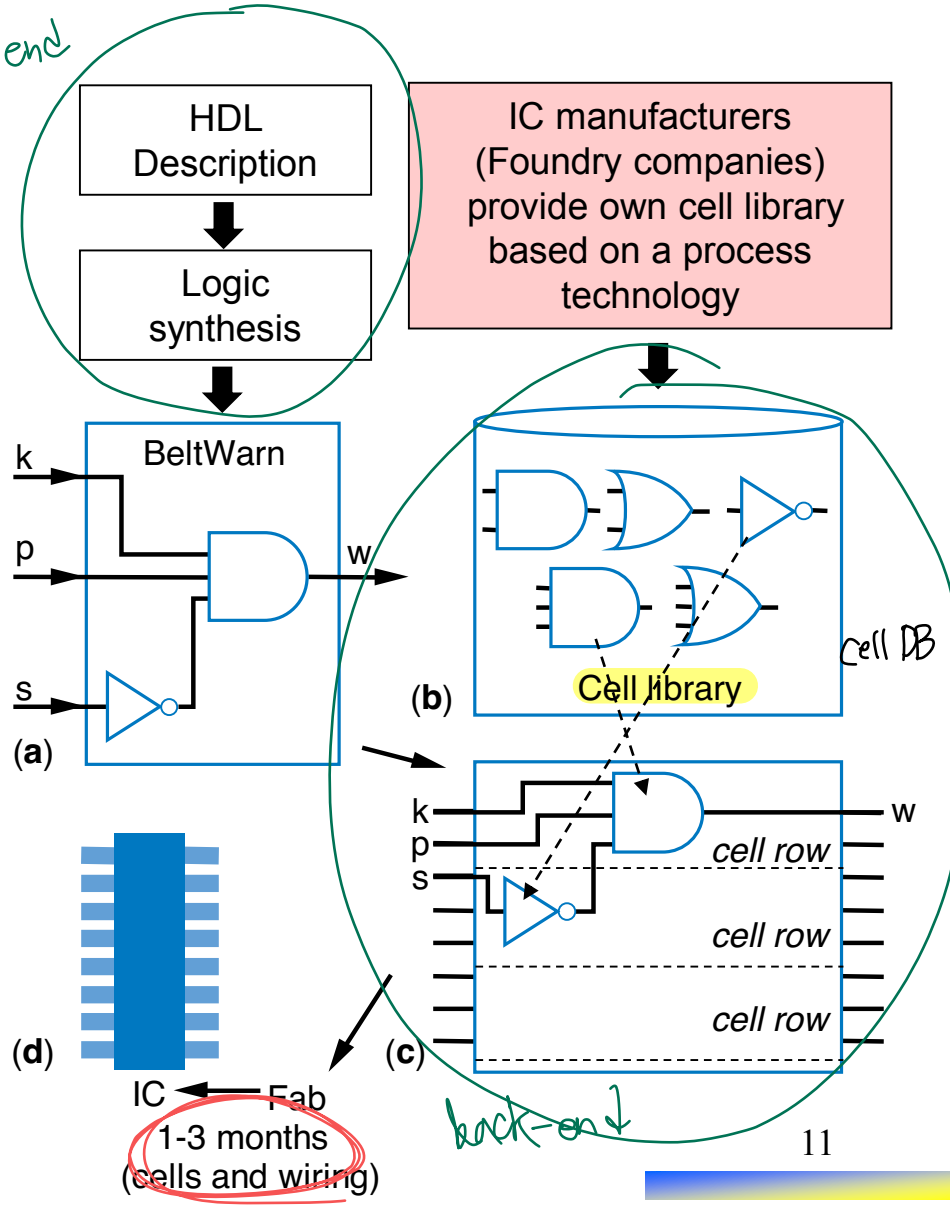
↳ but, Gate array 에선 변수 없음.

VLSI (Very Large Scale Integration) Technologies – Standard Cell

- B. Semicustom IC
  - 1. Gate array or 2. standard cell

- B.2. Standard cell (표준 cell 방식)
  - Already layout "cells" exist in library, not on chip. ⇒ layout이 설계엔 라이브러리의 cell에 존재함.
  - Designer instantiates cells into pre-defined rows, and connects.
  - Vs. gate array
    - Better performance/power/size
    - A bit harder to design
  - Vs. full custom
    - Not as good of circuit, but still far easier to design.

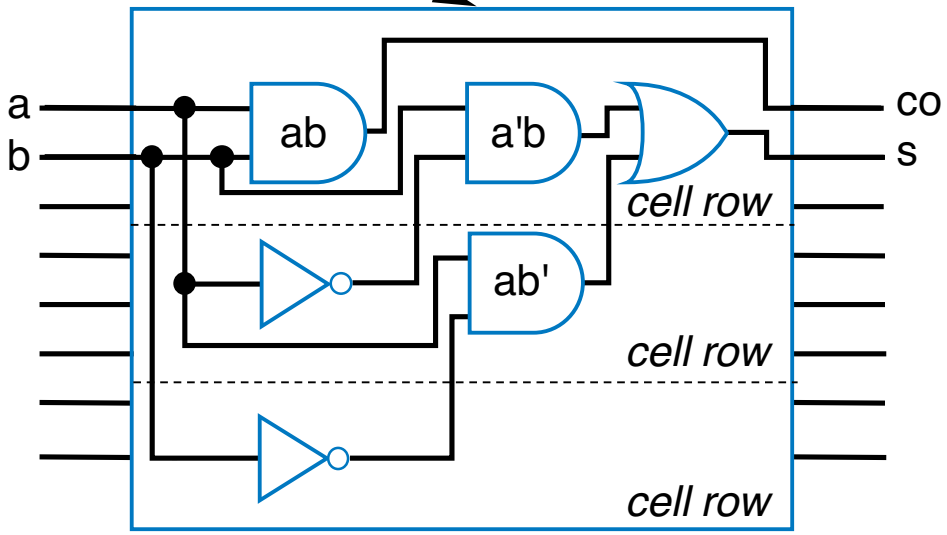
주어진 행에만 gate를 배치할 수 있음 (⇒ cell row를 맞춰야함)

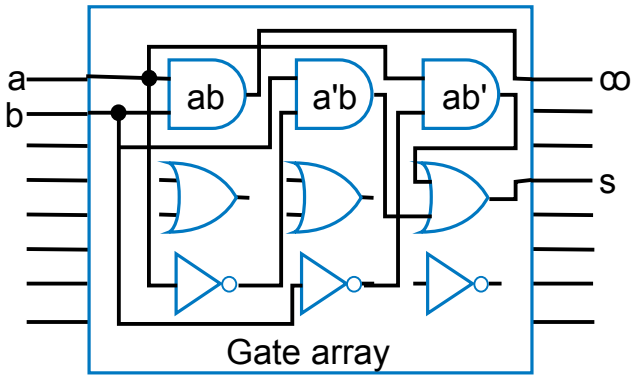


VLSI (Very Large Scale Integration) Technologies – Standard Cell

- B.2. Standard cell
  - Example: Mapping a half-adder to standard cells

co = ab
s = a'b + ab'





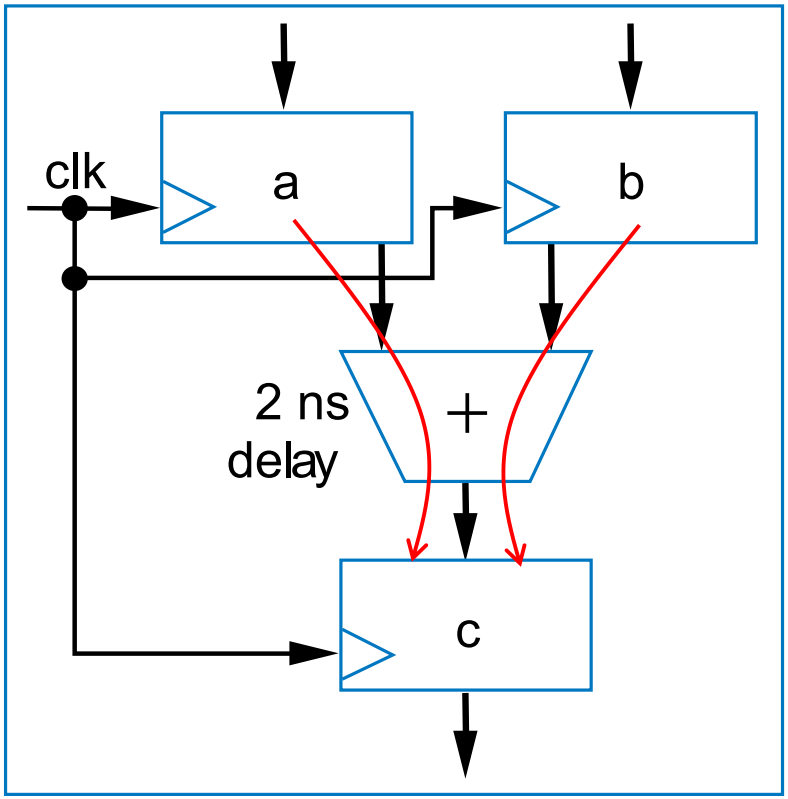
Notice fewer gates and shorter wires for standard cells versus gate array, but at cost of more design effort

⇒ critical path에 대한 것이

Determining Clock Frequency

- Designers of digital circuits often want fastest performance.
  - Means want high clock frequency

- Frequency limited by ***longest register-to-register delay***
  - Known as ***critical path***
  - If clock is any faster, incorrect data may be stored into register.
  - Longest path on right is 2 ns
    - Ignoring wire delays, and register setup time/internal delays for simplicity.

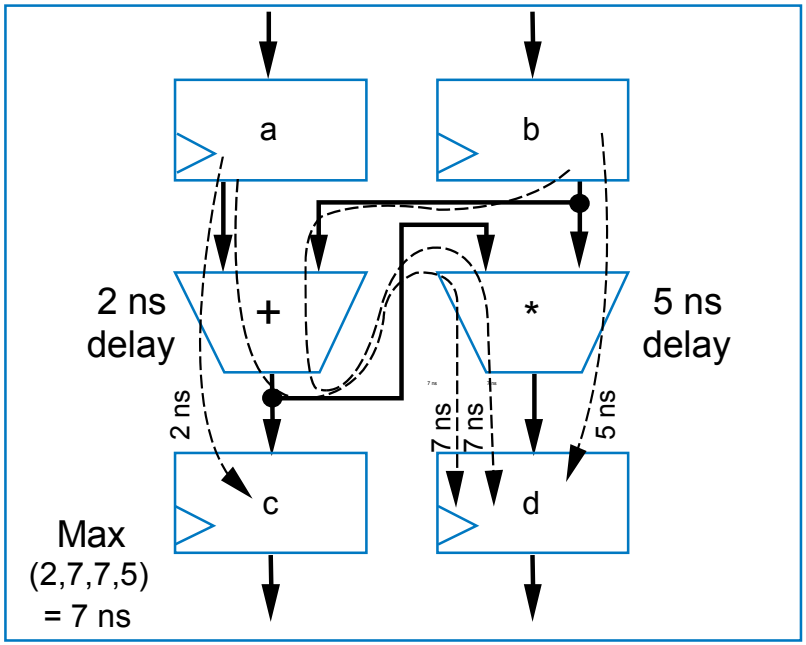


나머지 무시된 상태임.

operating $CLK = \frac{1}{\text{critical path}}$

Determining Clock Frequency

- Example shows four paths
  - a to c through +: 2 ns
  - b to c through +: 2 ns
  - a to d through + and *: 7 ns
  - b to d through + and *: 7 ns
  - b to d through *: 5 ns
- Longest path is thus 7 ns
- Fastest frequency
  - 1 / 7 ns = 142 MHz



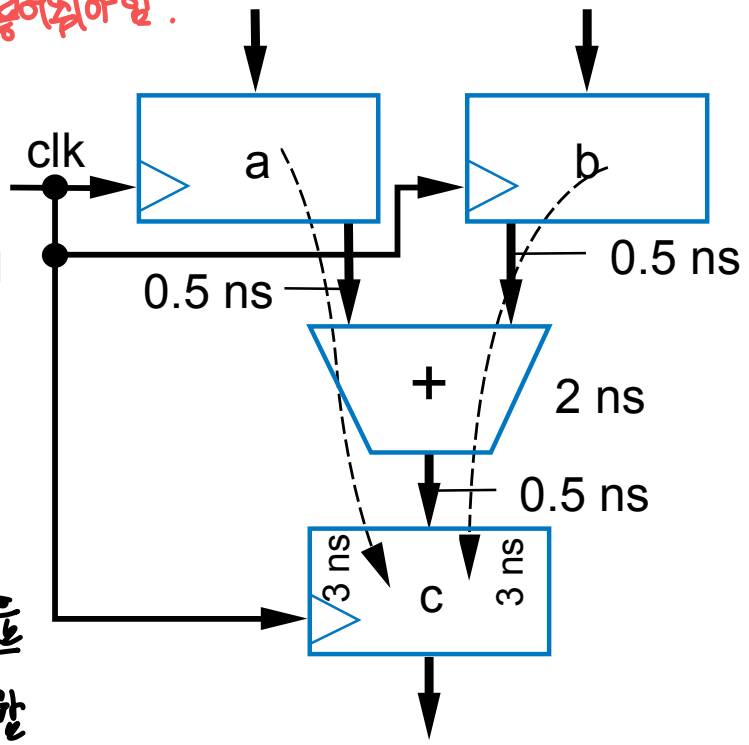
↳ 이것보다 크면 안돌아감. catch 불가

↳ 이런것은 컴퓨터 tool 이 대신 해줌.

Determining Clock Frequency

Critical Path Considering Wire Delays

- Real wires have delay too.
  - Must include in critical path.
- Example shows two paths
  - Each is 0.5 + 2 + 0.5 = 3 ns
- Trend
  - 1980s/1990s: Wire delays were tiny compared to logic delays
  - But wire delays not shrinking as fast as logic delays
    - Wire delays may even be greater than logic delays!.



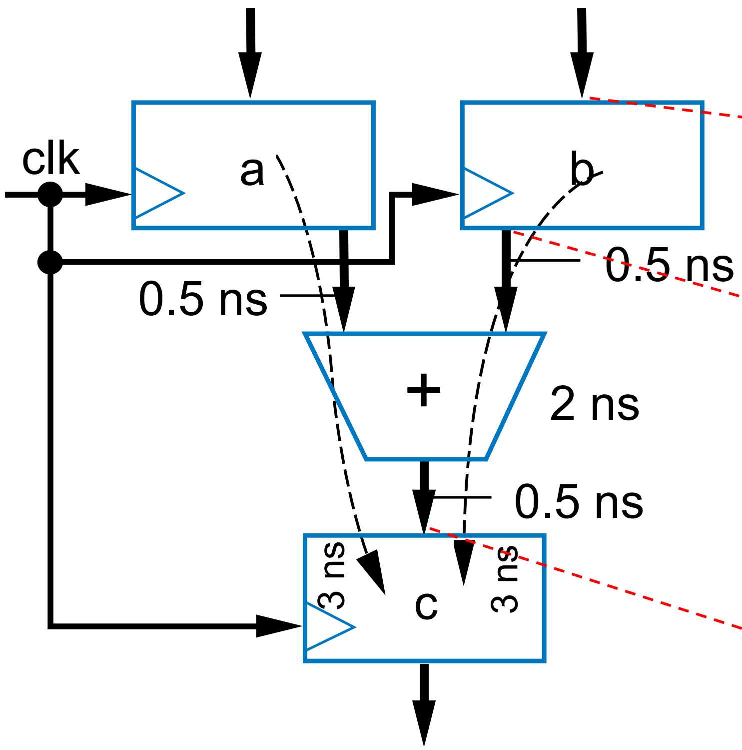
critical delay에서 또 고려해야할 것.

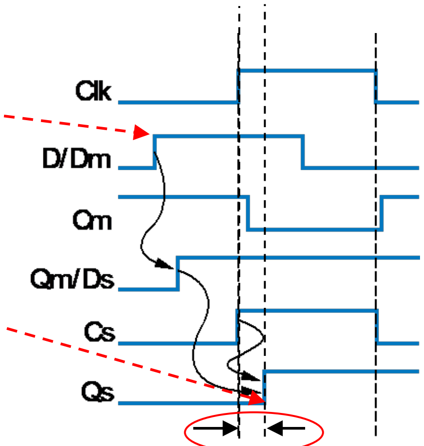
Determining Clock Frequency

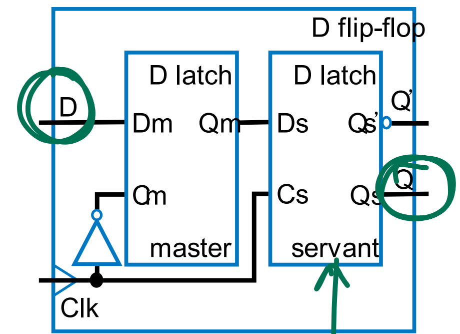
Critical Path Considering Setup Time/Internal Delays

↳ register가 가지고 있는 내부 delay.

- We must also consider register setup time/internal delay, also add to path.

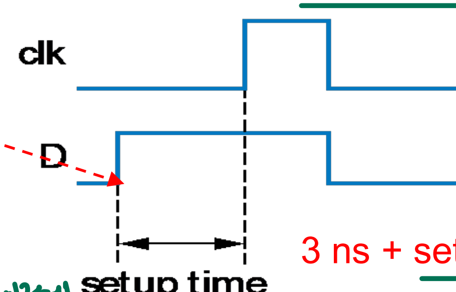






여기에도 delay존재

Internal delay + 3ns required!



3 ns + setup time required!

레지스터들 clk rising이전에 충분한 set up time이 필요. 그 없으면 값이 적용이 안됨.

Determining Clock Frequency

Critical Path Considering All of Them

- Then add some time to the computed path, just to be safe.
  - For the previous example, let's assume that register setup time is 0.1ns and internal delay is 0.05 ns.
  - Critical path = wire_delay (1ns) + logic delay (2 ns) + setup time (0.1ns) + internal delay (0.05 ns) = 3.15 ns.
  - Therefore, clock period of 3.5 ns ~ 4 ns is safe.

They depend on IC process technologies (currently, MOSFET technology).

ex) Samsung 7nm process,
SK Hynix 10nm process,
Intel 14nm process,
TSMC12nm process,
etc.

A Circuit May Have Numerous Paths

- Paths can exist
  - In the datapath
  - In the controller
  - Between the controller and datapath
  - May be hundreds or thousands of paths

- Timing analysis tools that evaluate all possible paths automatically very helpful.

↳ path의 수로 최대동작 성능을 안수있음.

1:33:48

Programmable/Reconfigurable IC Technology

- Manufactured IC technologies require weeks to months to fabricate.
  - And have large (hundred thousand to million dollar) initial costs

- Programmable/Reconfigurable ICs are pre-manufactured.
  - Can implement circuit *today*
  - Just download bits into device
  - Slower/bigger/more-power than manufactured ICs
    - But get it today, and no fabrication costs.

↳ 하드웨어를 설계 해놨다고 하면 바로 구현이 가능함.

bit stream ⇒ bit file

단점

장점

- Two representative Programmable/Reconfigurable ICs
  1. PLD (Progammable Logic Device) → logic 기반 … (작은 칩)
  2. FPGA (Field Programmable Gate Array) → 메모리 기반

⇒ 중간에는 존재하지 않아요.

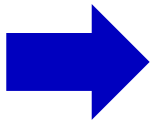
2가지 type의 Programmable/Reconfiguable ICs

1) PLD

2) FPGA

Programmable/Reconfigurable IC Technology – PLD

- Programmable Logic Device (PLD)
  - Developed 1970s (pre-dates FPGAs)
  - Prefabricated IC with large AND-OR structure
    - because of canonical form for representing boolean equations: sum of minterms



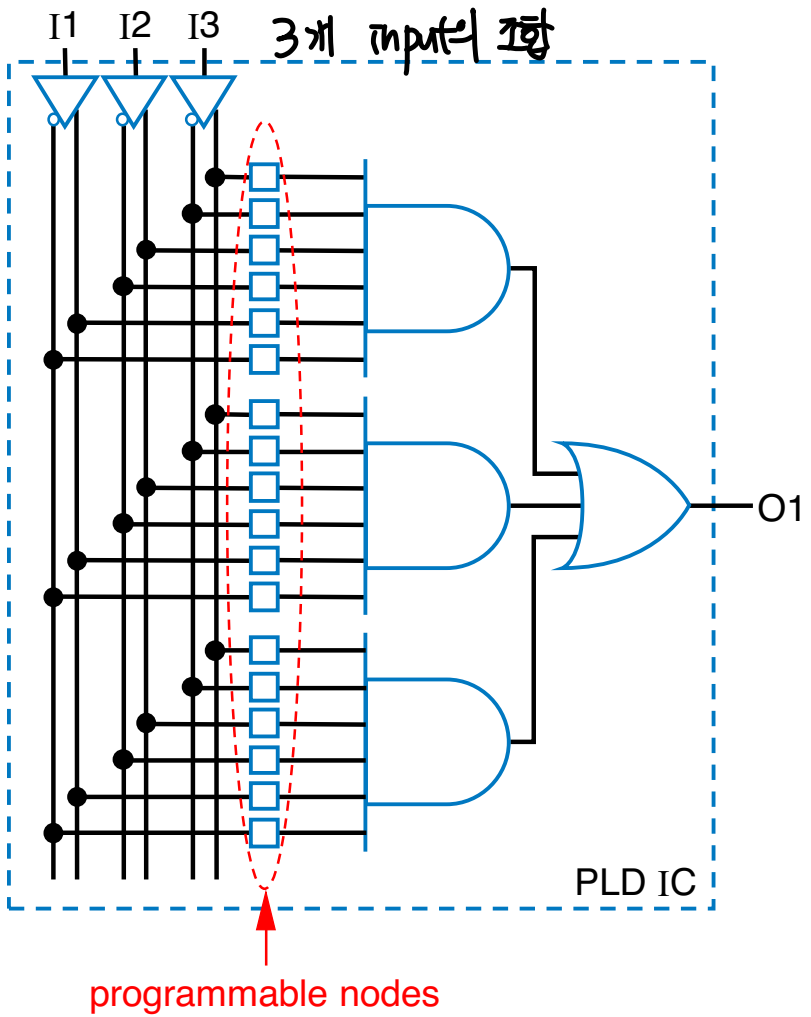
ALTERA® now part of Intel

intel ARRIA inside
intel STRATIX inside
intel AGILEX inside
intel CYCLONE inside
intel MAX inside

Programmable/Reconfigurable IC Technology – PLD

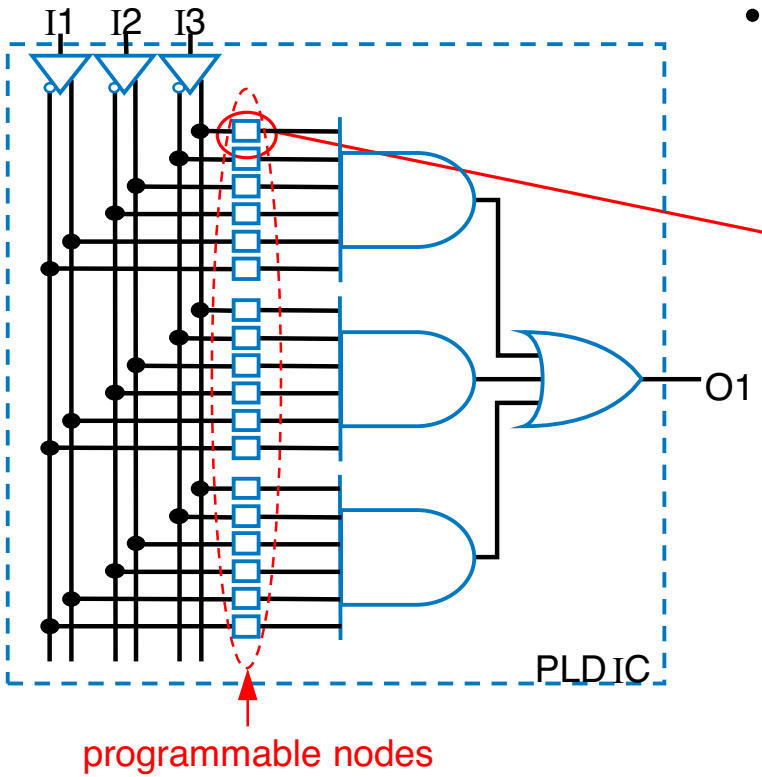
- Programmable Logic Device (PLD)
  - Connections can be "programmed" to create custom circuit
    - Circuit shown can implement any 3-input function of up to 3 terms
      - e.g., F = abc + a'c'

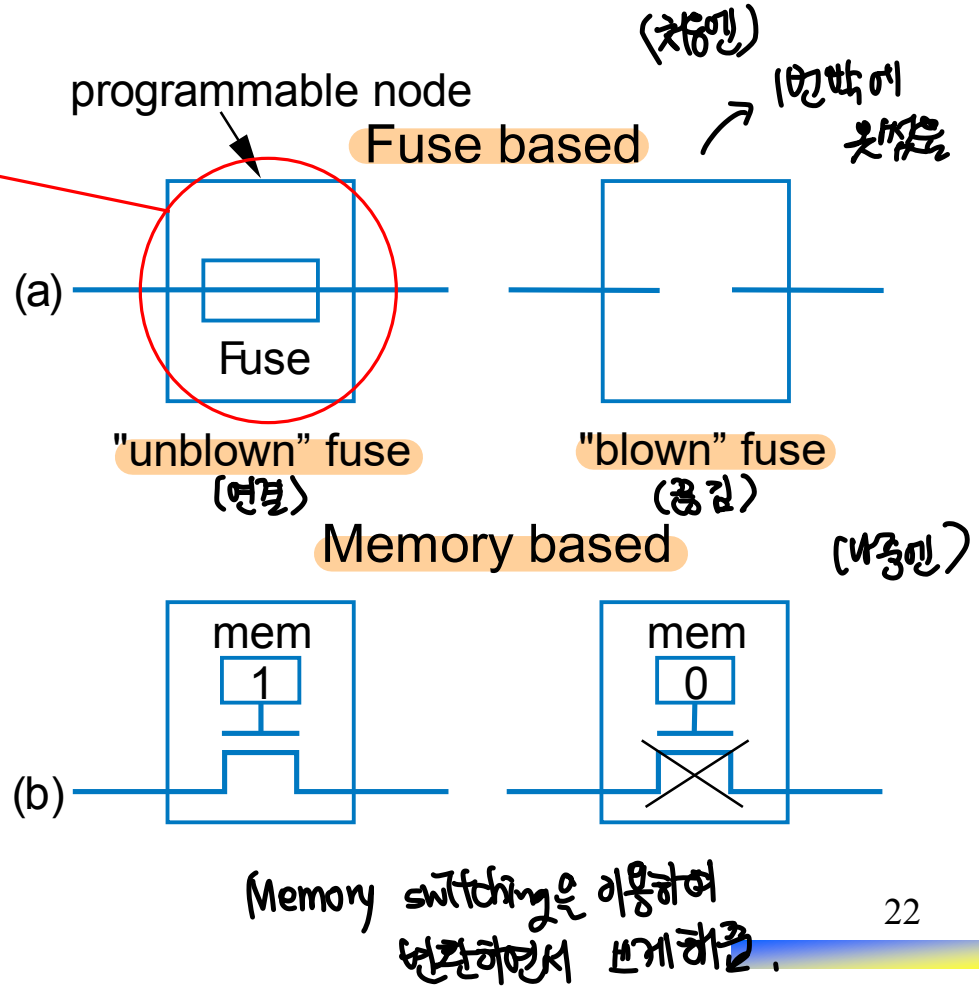
AND 1개 = 1 term



Programmable Nodes in an PLD

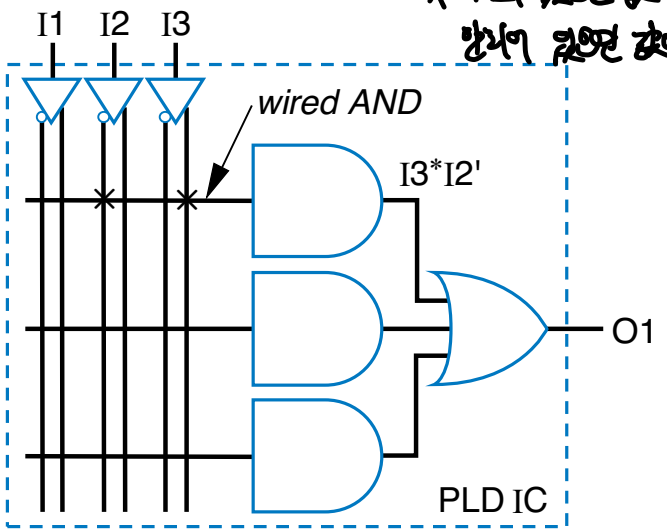
- Fuse based – "blown" fuse removes connection
- Memory based – 1 creates connection



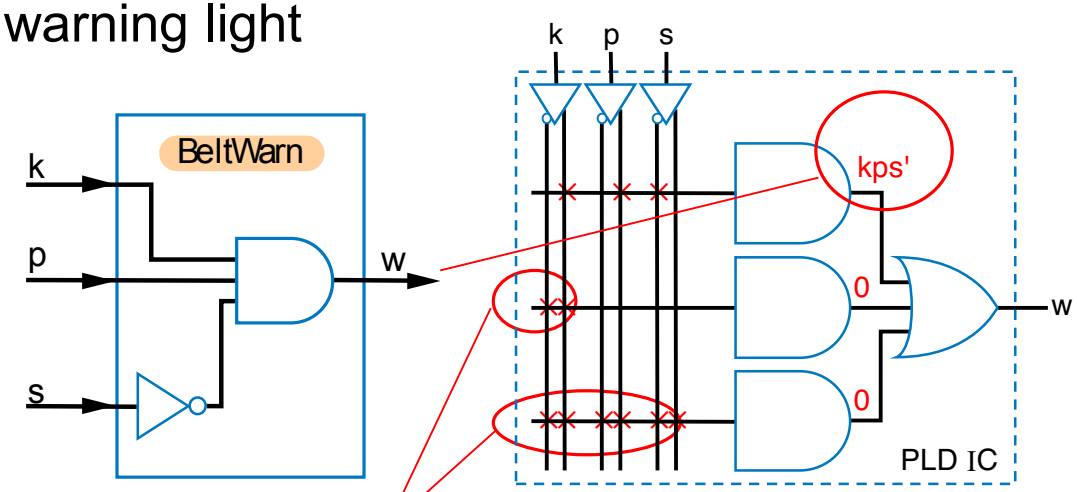


PLD Implementation Example

- Common way of drawing PLD connections:
  - Uses one wire to represent all inputs of an AND
  - Uses "x" to represent connection
    - Crossing wires are not connected unless "x" is present.

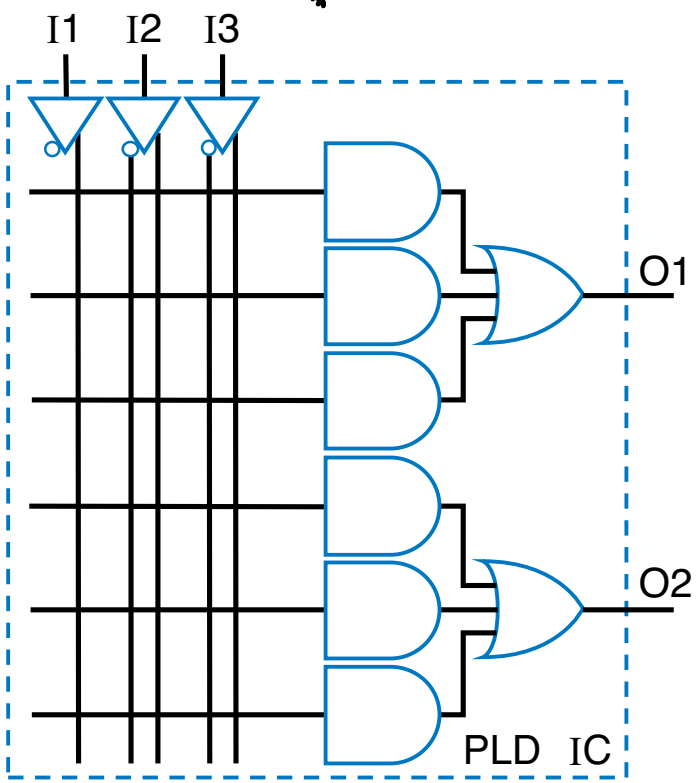


- Example: Seat belt warning light using SPLD



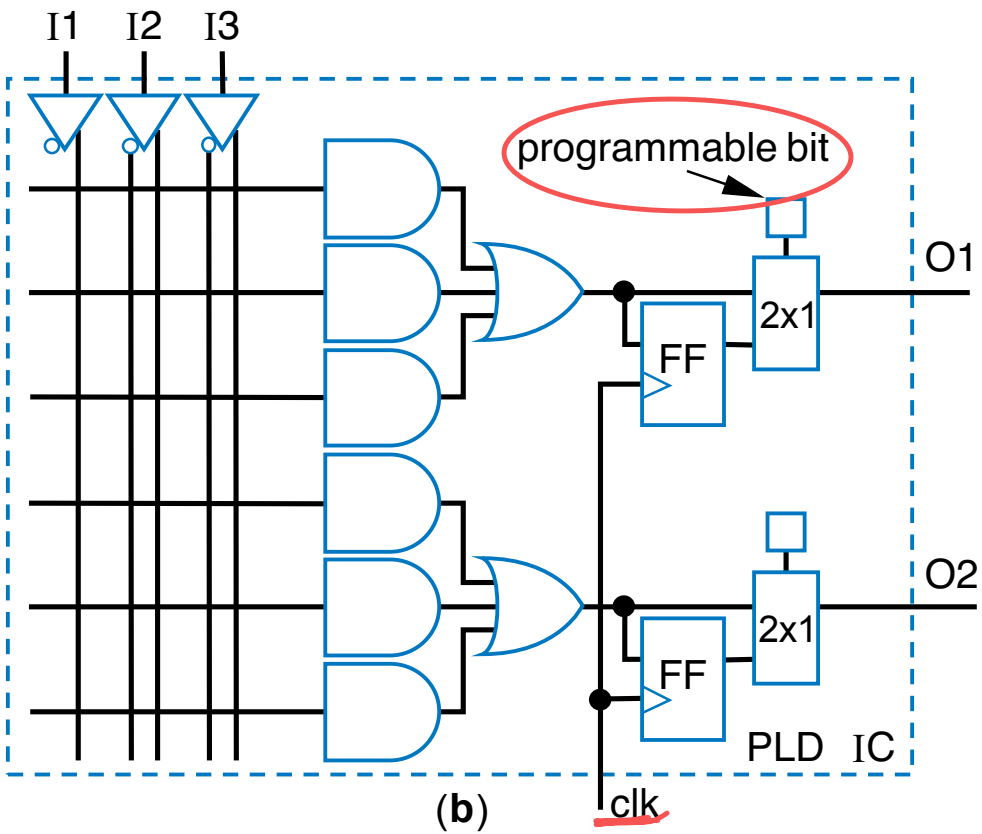
Two ways to generate a 0 term

PLD Extensions (⇒ PLD 확장)



(a)

Two-output PLD



(b)

PLD with programmable registered outputs

More on PLDs

…> 용어가 정의가 잘안되어있음.

역사

- History of PLD
  - Originally (1970s) known as Programmable Logic Array – *PLA*
    - Had programmable AND and OR arrays (fuse based programmable nodes).
  - AMD created "Programmable Array Logic" – "*PAL*" (trademark)
    - Only AND array was programmable (fuse based programmable nodes).

Fuse based

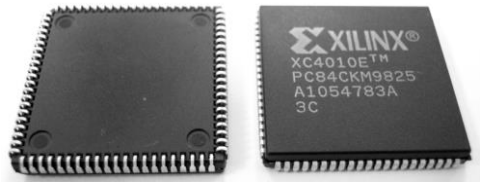
Memory based

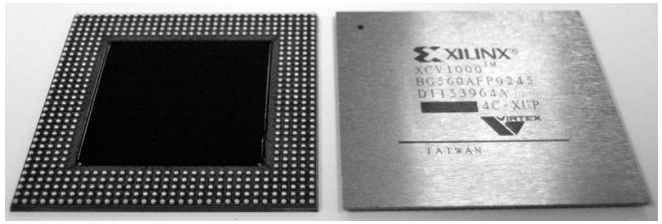
  - Lattice Semiconductor Corp. created "Generic Array Logic – "GAL" (trademark)
    - Memory based programmable nodes.
  - As IC capacities increased, companies put multiple PLD structures on one chip, interconnecting them
    - Become known as Complex PLDs (CPLD), and older PLDs became known as Simple PLDs (SPLD).

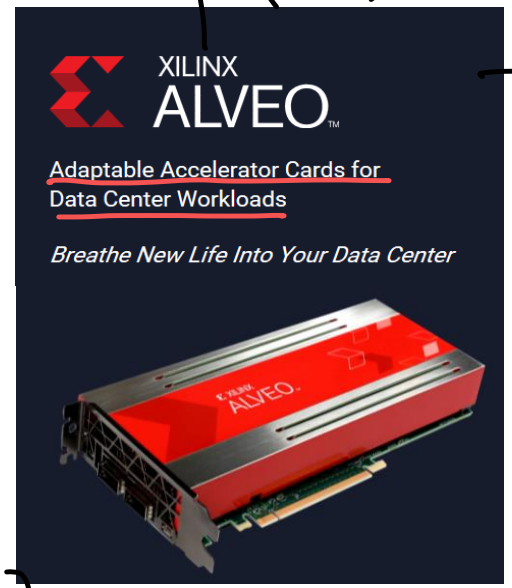
↳ 단순한 PLD가 아닌 FPGA까지도 가지고 있음.

Programmable/Reconfigurable IC Technology – FPGA

- Today, the most popular programmable IC – FPGA
  - "Field-programmable gate array"
    - Developed late 1980s
    - Though no "gate array" inside
      - Named when gate arrays were very popular in the 1980s
    - Programmable in seconds.

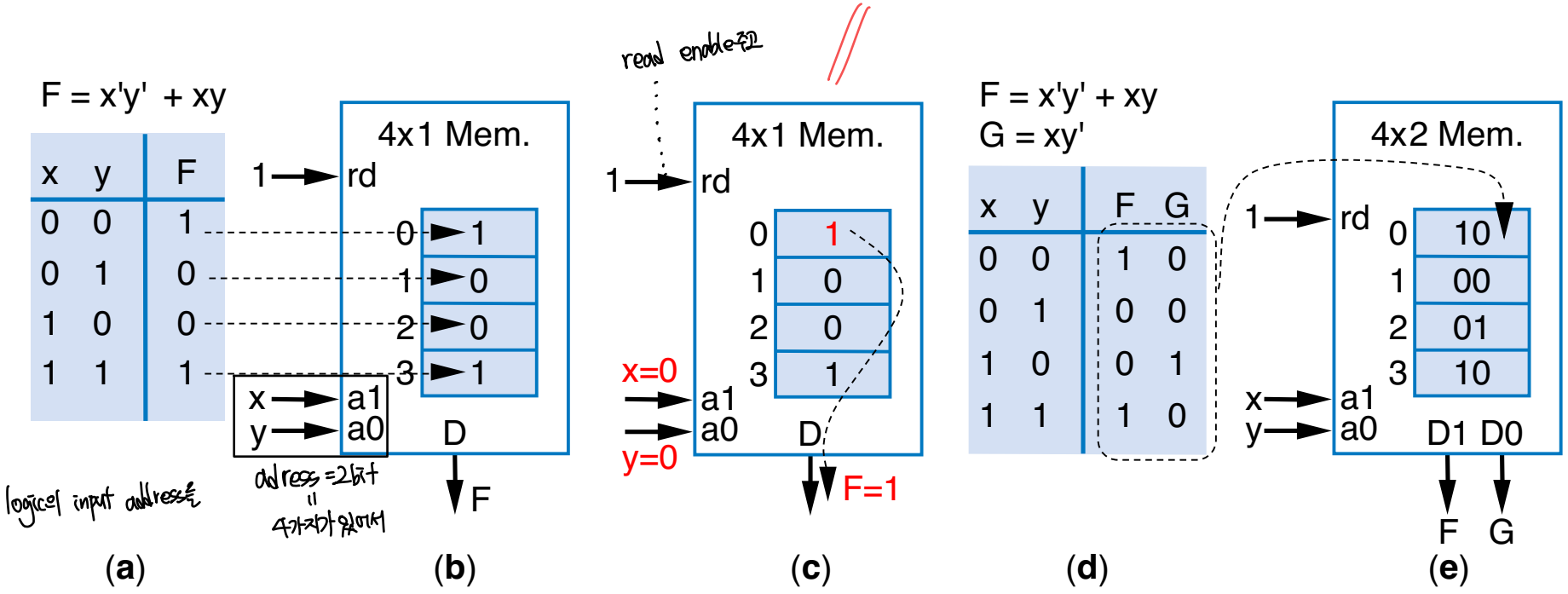






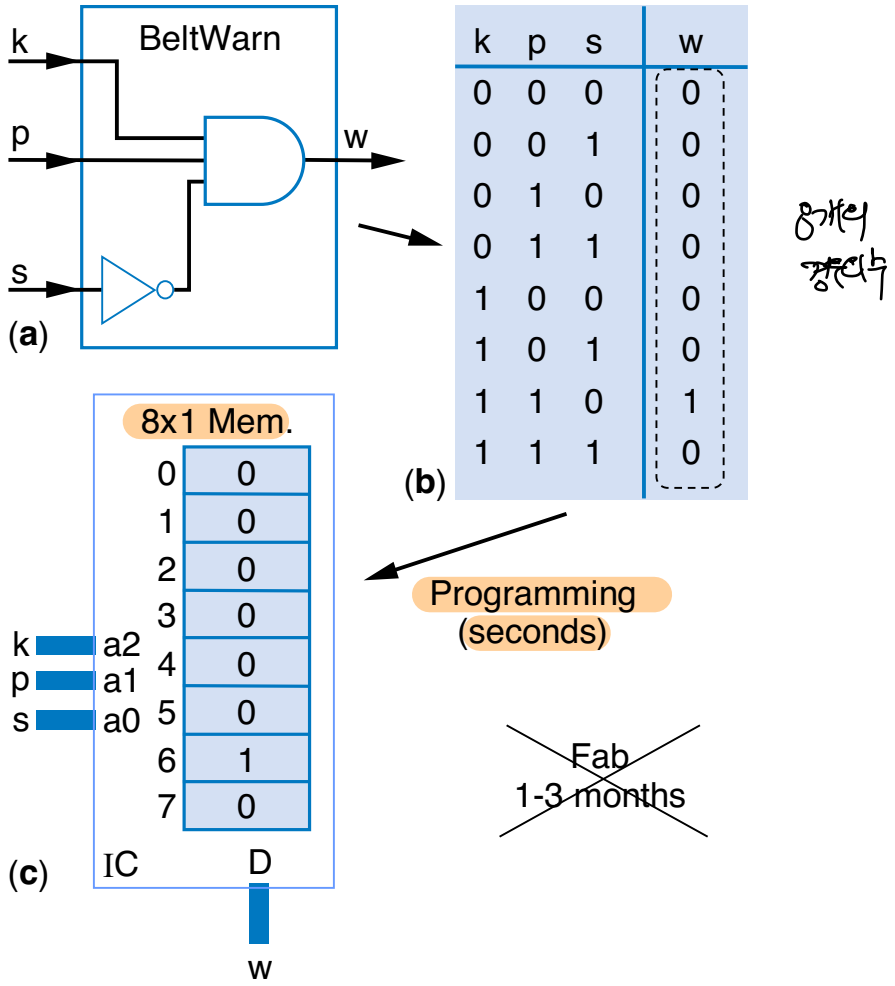
FPGA Internals: Lookup Tables (LUTs)

- Basic idea: Asynchronous SRAM can implement combinational logic
  - e.g., 2-address memory can implement 2-input logic
  - 1-bit wide memory – 1 function; 2-bits wide – 2 functions
- Such memory in FPGA known as Lookup Table (LUT)



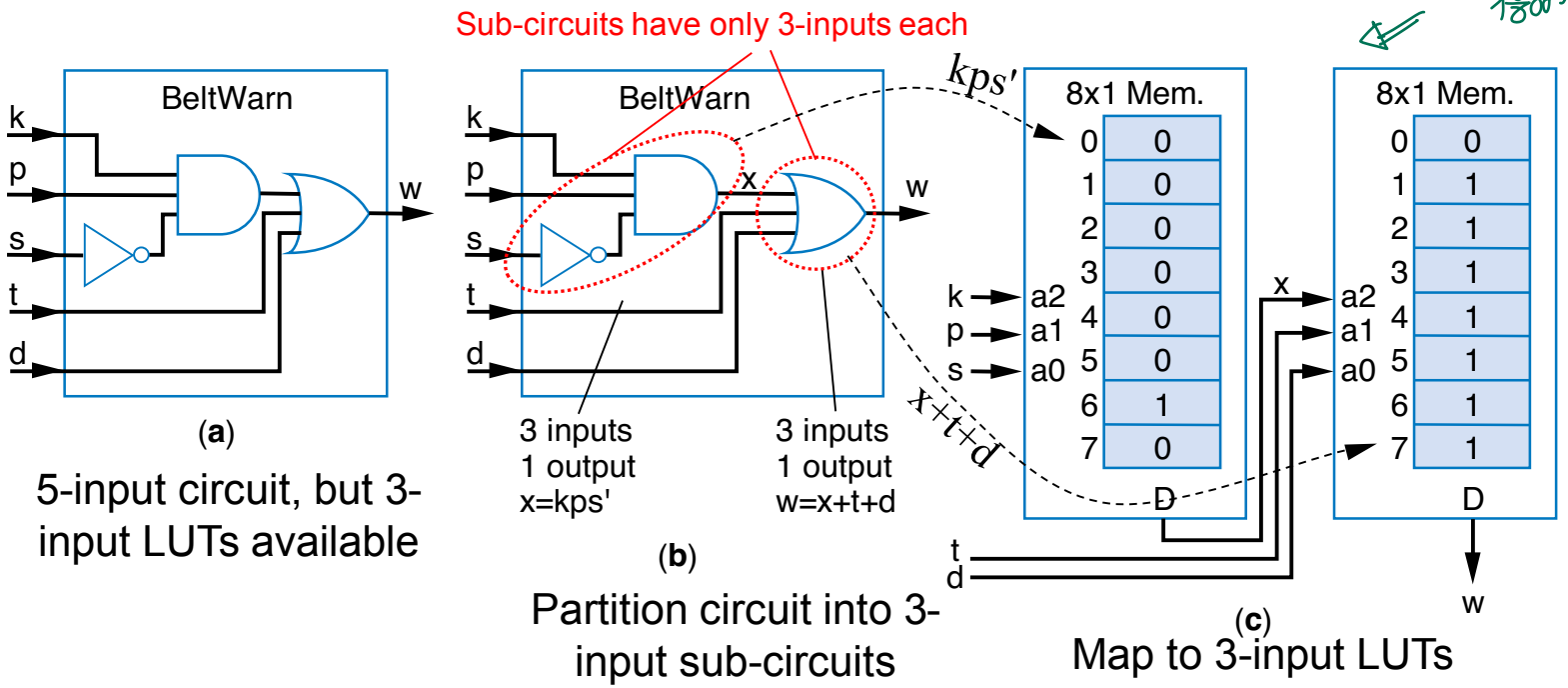
FPGA Internals: Lookup Tables (LUTs)

- Example: Seat-belt warning light (again)



FPGA Internals: Lookup Tables (LUTs)

- Lookup tables become inefficient for more inputs
  - 3 inputs → only 8 words
  - 8 inputs → 256 words; 16 inputs → 65,536 words!
- FPGAs thus have numerous small (3, 4, 5, or even 6-input) LUTs
  - If circuit has more inputs, must partition circuit among LUTs
  - Example: Extended seat-belt warning light system:



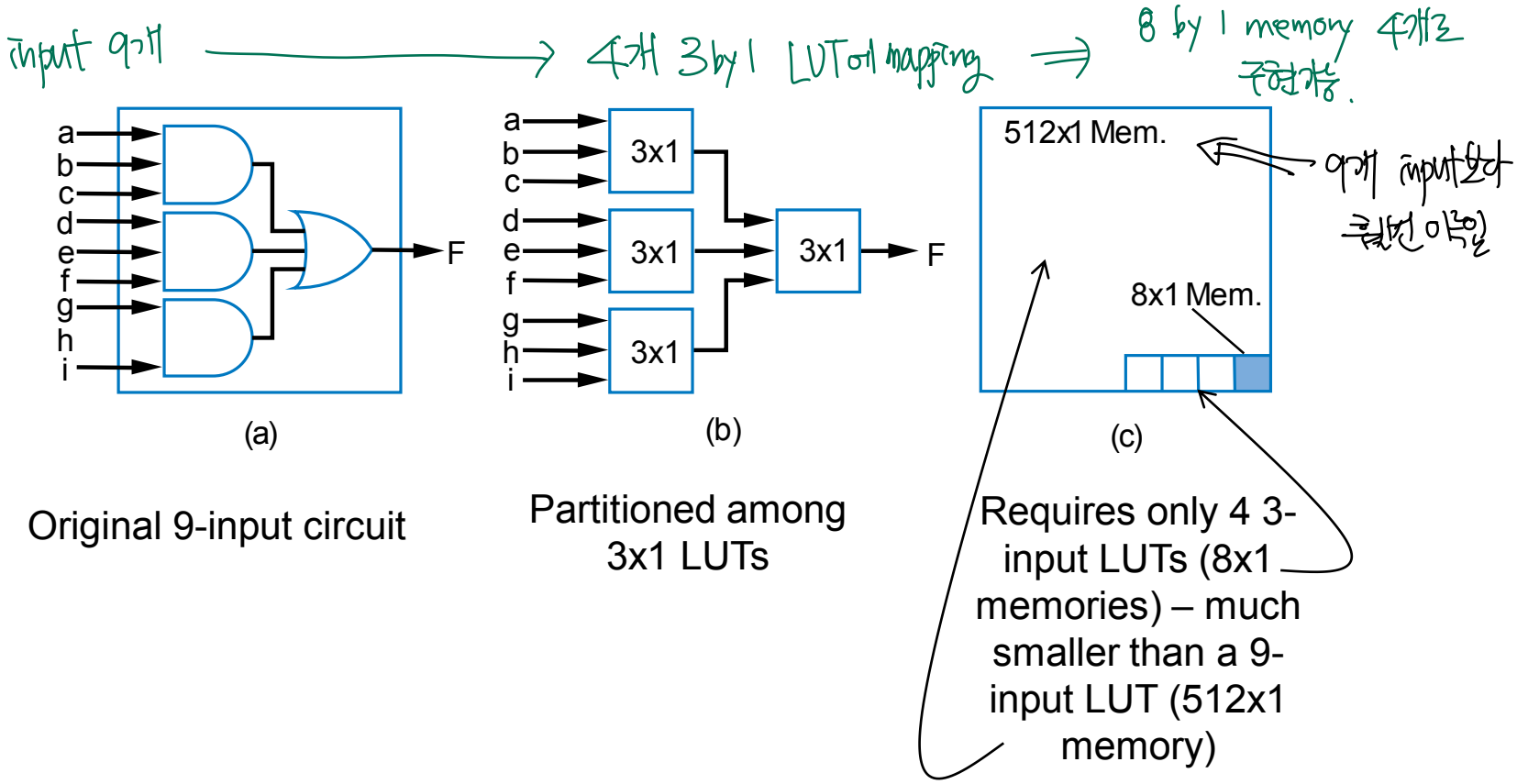
5-input circuit, but 3-input LUTs available

Partition circuit into 3-input sub-circuits

Map to 3-input LUTs

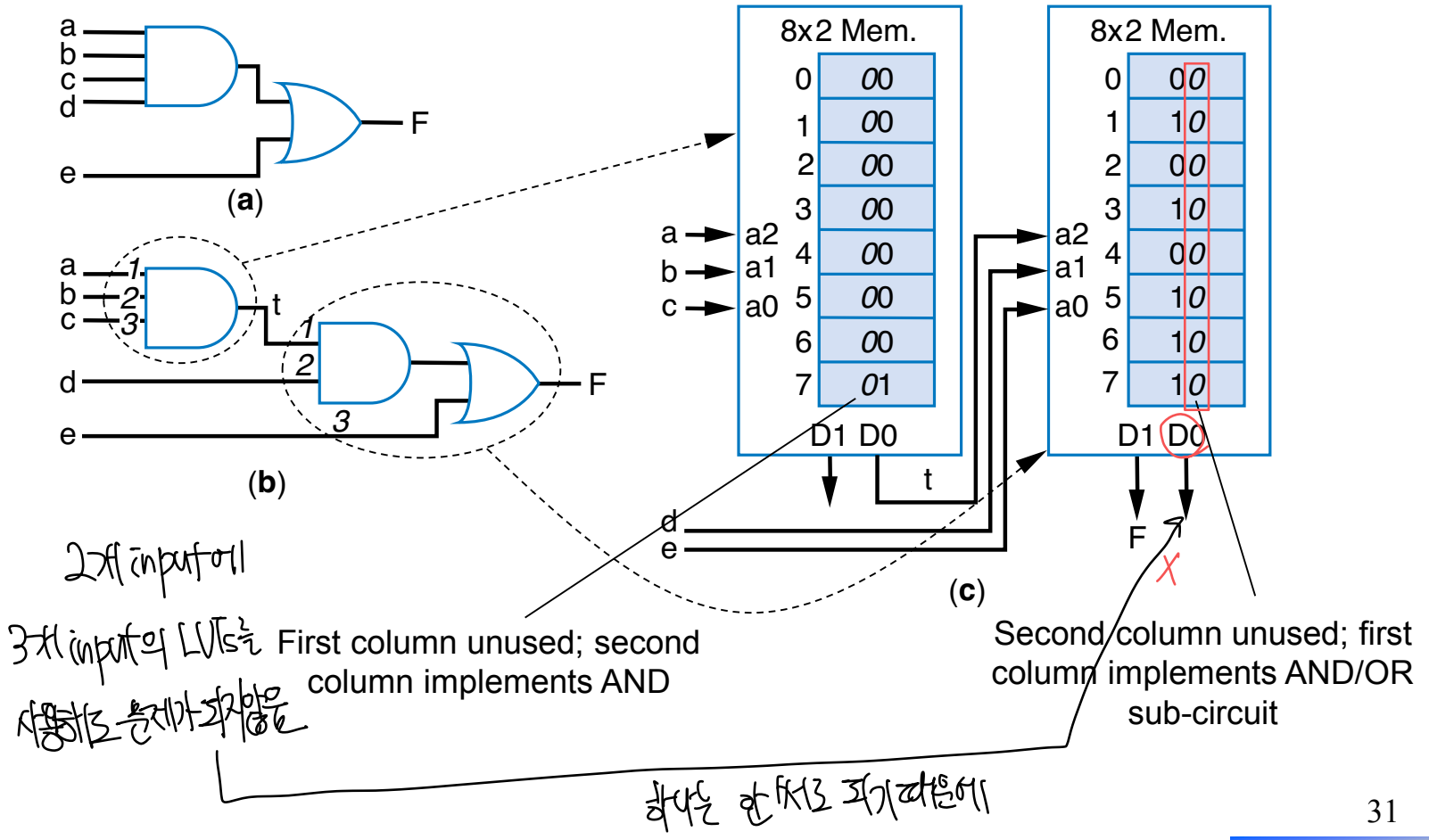
FPGA Internals: Lookup Tables (LUTs)

- Partitioning among smaller LUTs is more size efficient.
  - Example: 9-input circuit



FPGA Internals: Lookup Tables (LUTs)

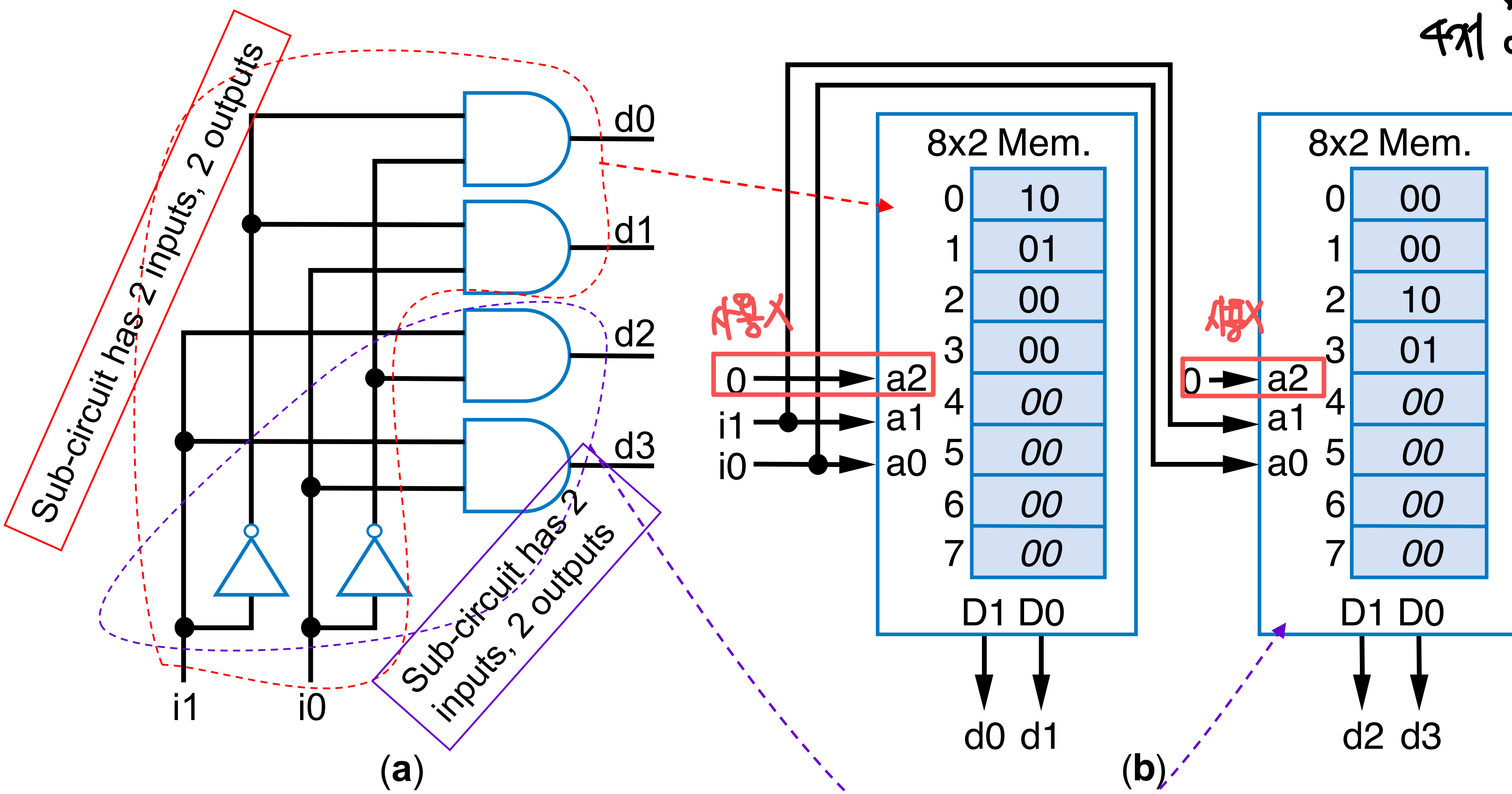
- LUT typically has 2 (or more) outputs, not just one
- Example: Partitioning a circuit among 3-input *2-output* lookup tables



FPGA Internals: Lookup Tables (LUTs)

- Example: Mapping a 2x4 decoder to 3-input 2-output LUTs

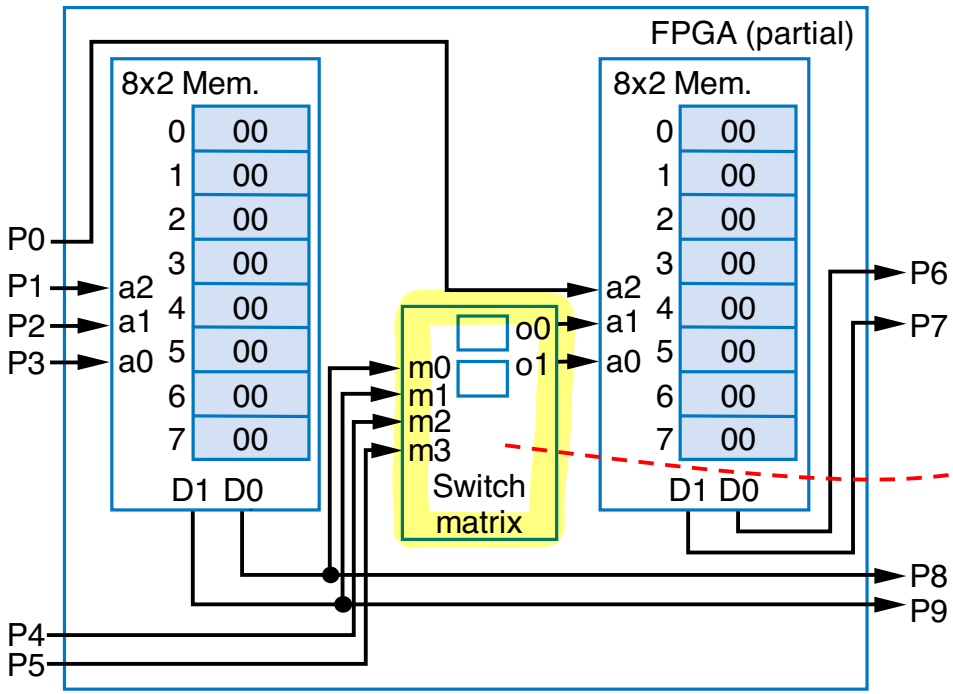
2개 input ⇓ 4개 output



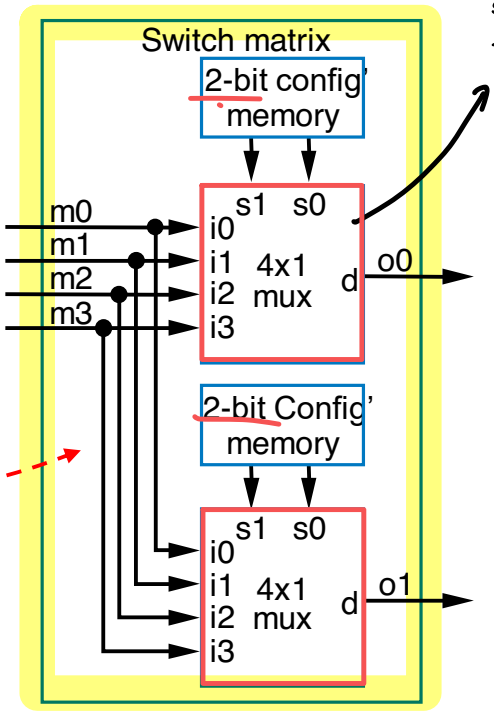
⇒ 보내주는 경우 다른 logic LUTs를 쓰면됨 . ....> 사용안해도 되는게 있어도 됨 .

FPGA Internals: Switch Matrices

- Previous slides had hardwired connections between LUTs.
- Instead, want to program the connections too.
- Use switch matrices (also known as programmable interconnect)
  - Simple mux-based version – each output can be set to any of the four inputs just by programming its 2-bit **configuration memory**



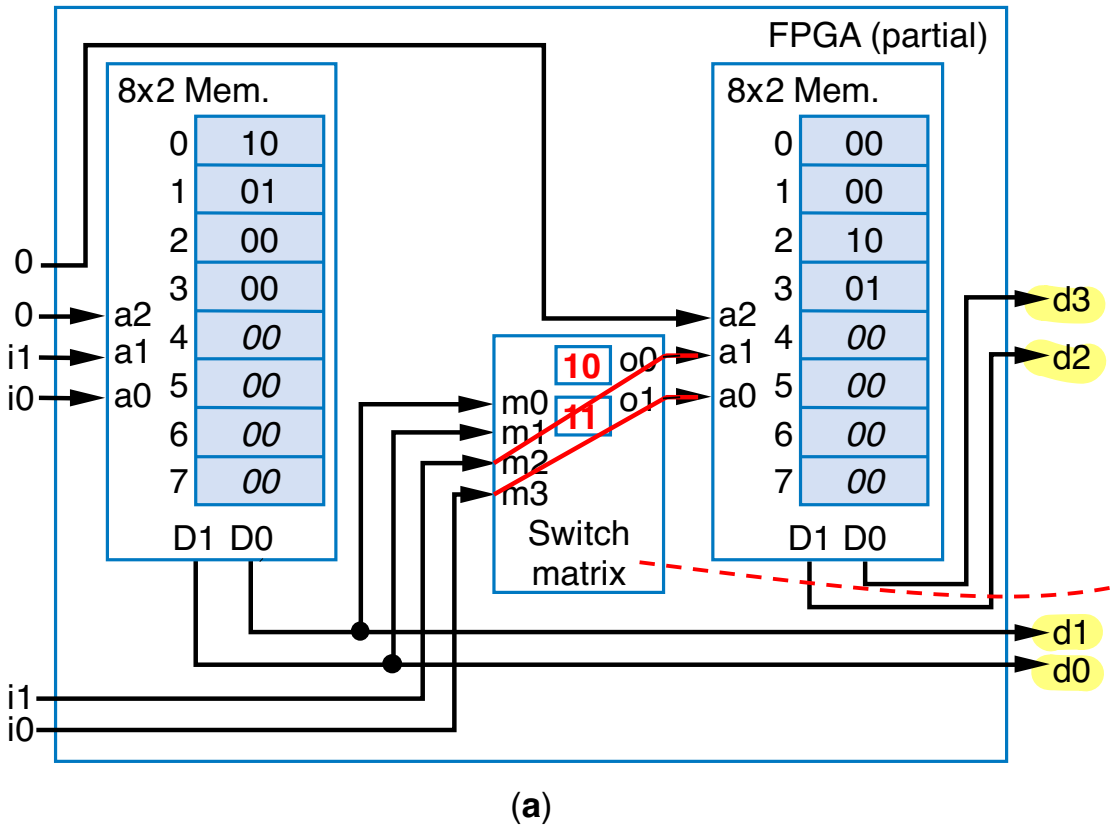
(a)

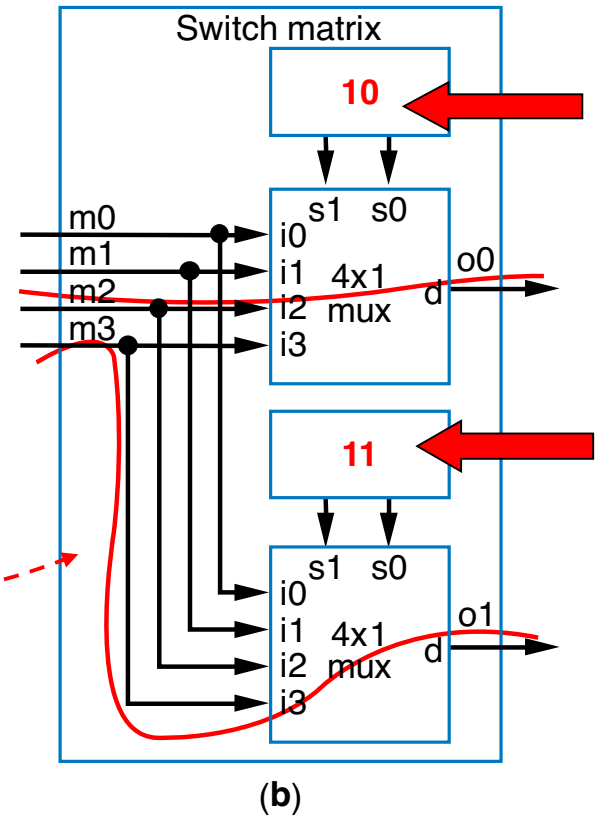


(b)

FPGA Internals: Switch Matrices

- Mapping a 2x4 decoder onto an FPGA with a switch matrix





These bits establish the desired connections

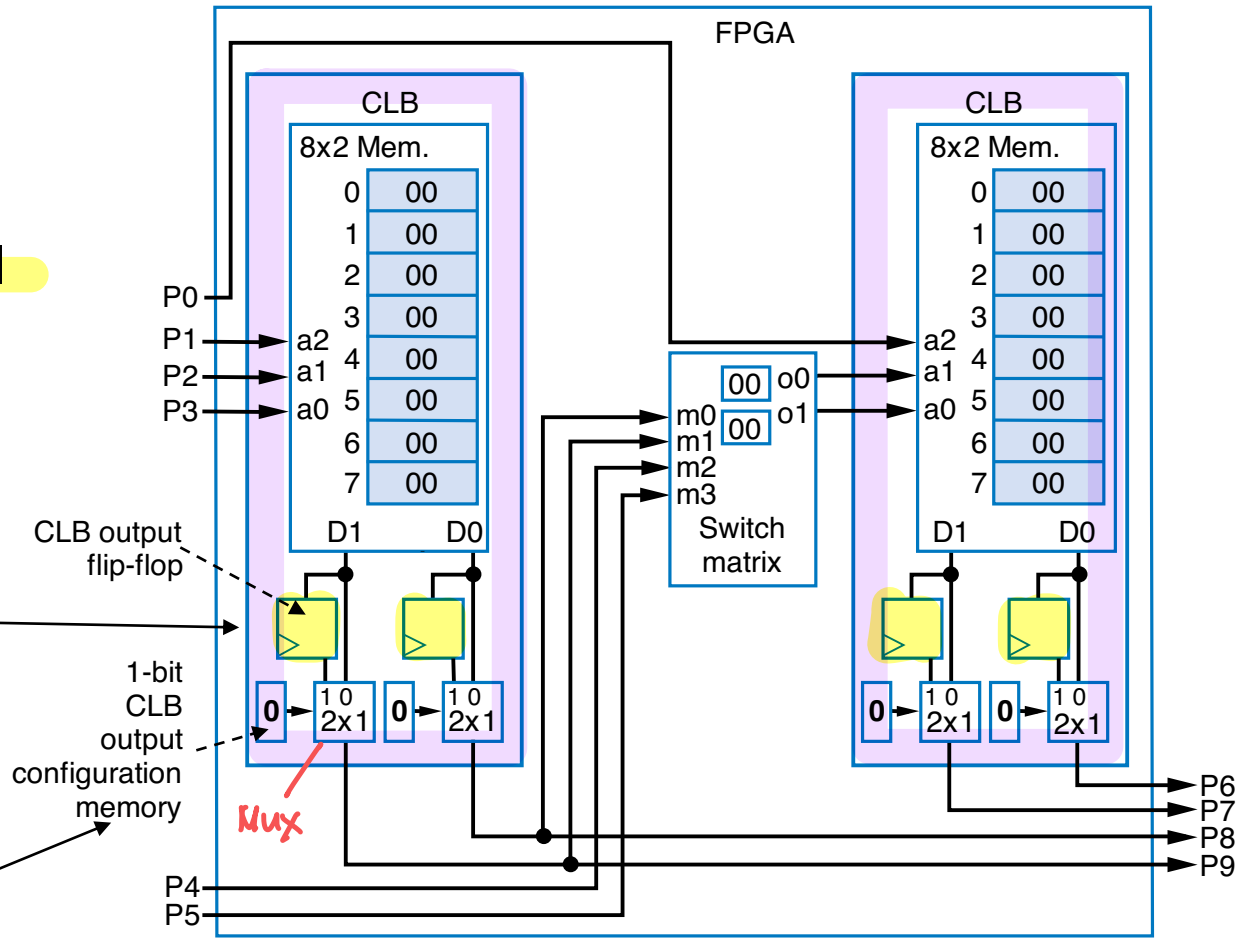
여기까지는 CLK이 없는 FPGA

FPGA에 CLK이 생기게 해주면

FPGA Internals: Configurable Logic Blocks (CLBs)

- LUTs can only implement combinational logic.
- Need flip-flops to implement sequential logic
- Add flip-flop to each LUT output
  - Configurable Logic Block (CLB)
    - LUT + flip-flops
  - Can program CLB outputs to come from flip-flops or from LUTs directly

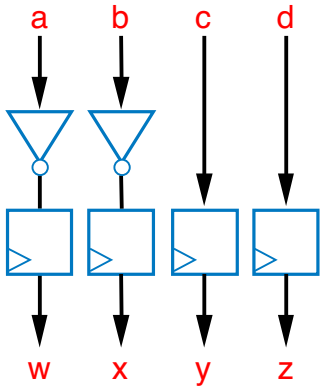
CLB =



CLB ⇒ 조합회로, 순차회로 둘다 만들 수 있다.

flip-flop base ⇒ register base 회로 만들 수 있다.

FPGA Internals: Sequential Circuit Example using CLBs



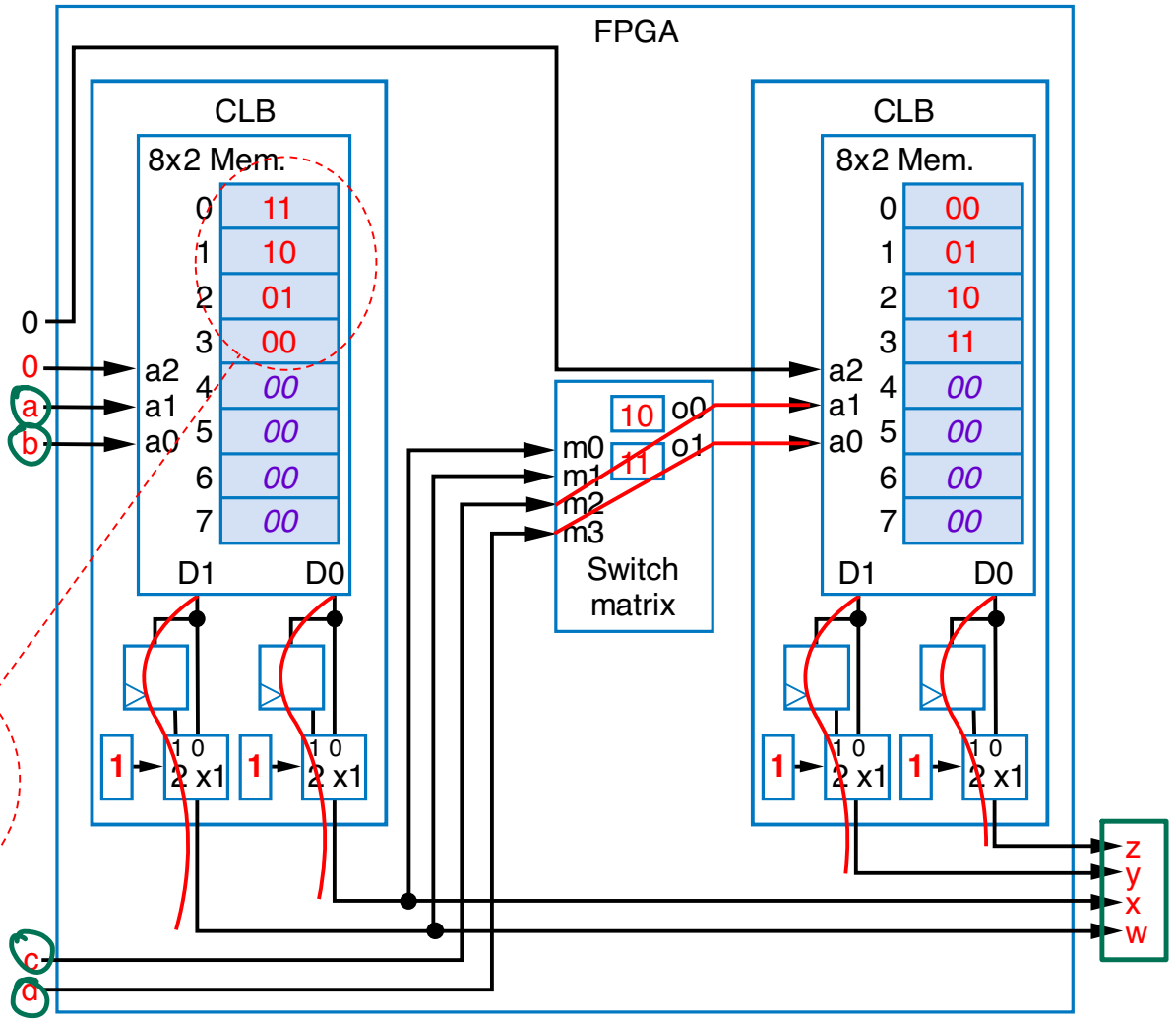
(a)

Left lookup table

a2	a1	a0	D1	D0
0	a	b	w=a'	x=b'
0	0	0	1	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	0

unused

(b)

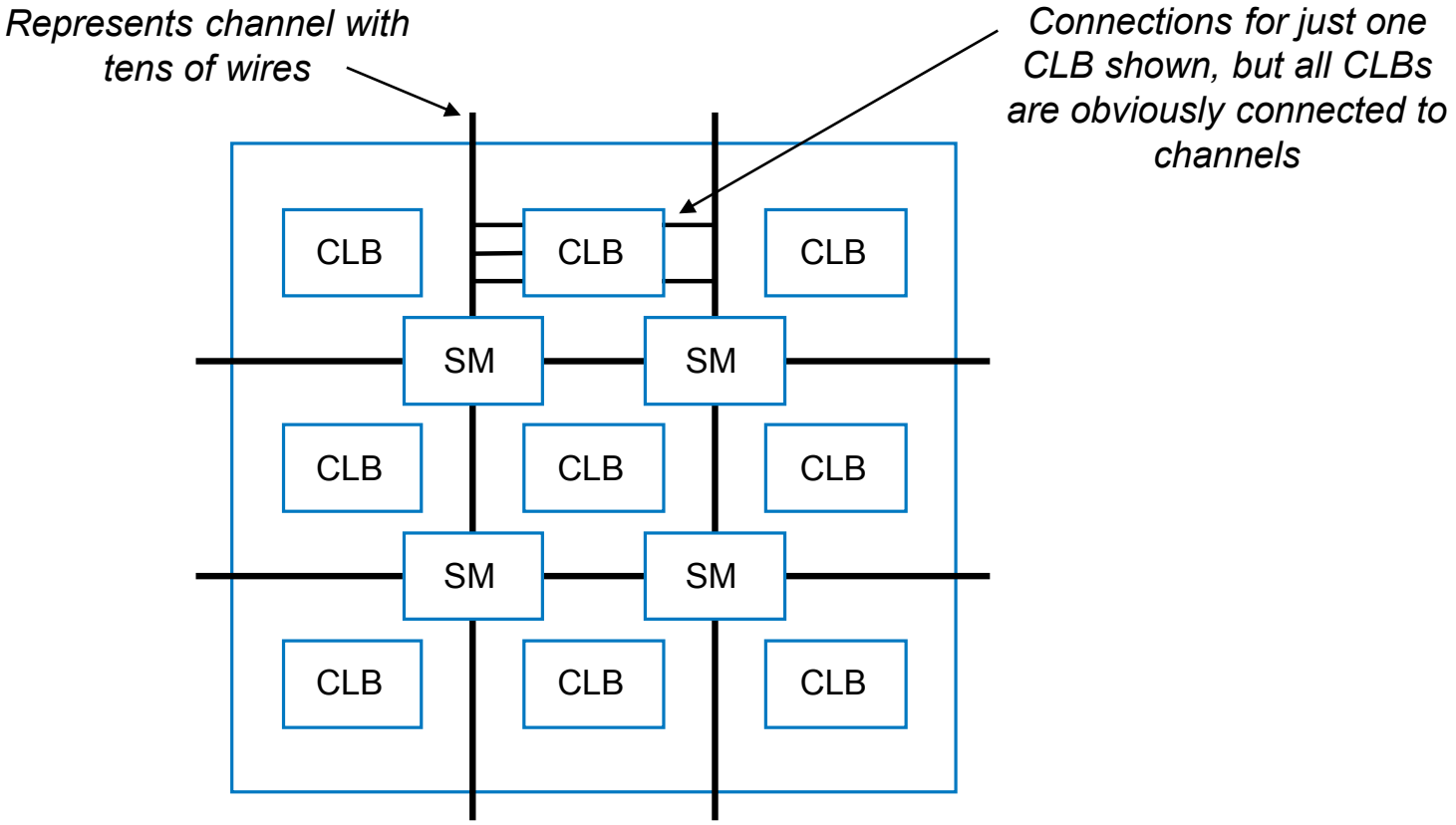


(c)

↳ 수많은 (CLB + SM)으로 구성되어있음.

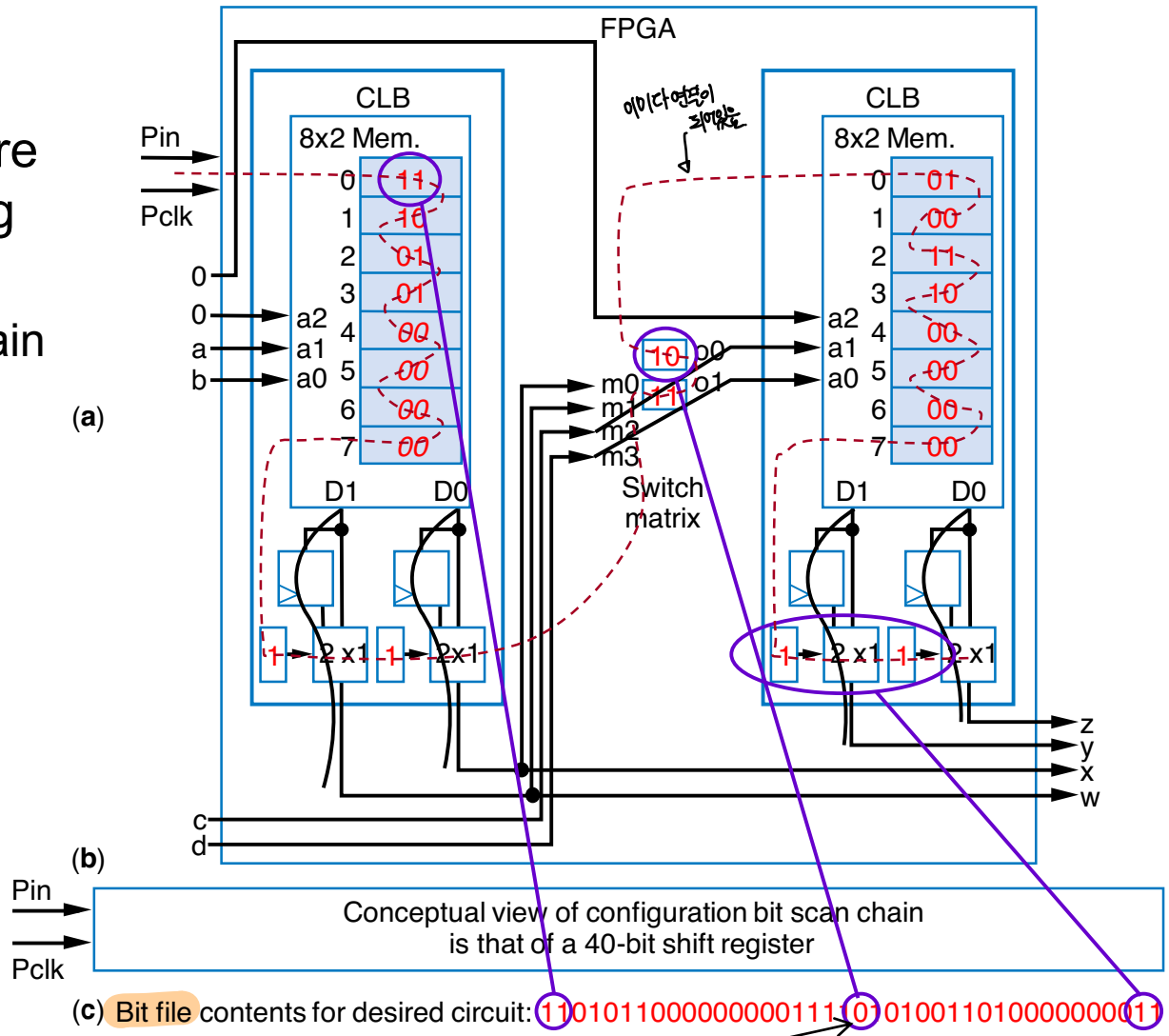
FPGA Internals: Overall Architecture

- Consists of hundreds or thousands of CLBs and switch matrices (SMs) arranged in regular pattern on a chip

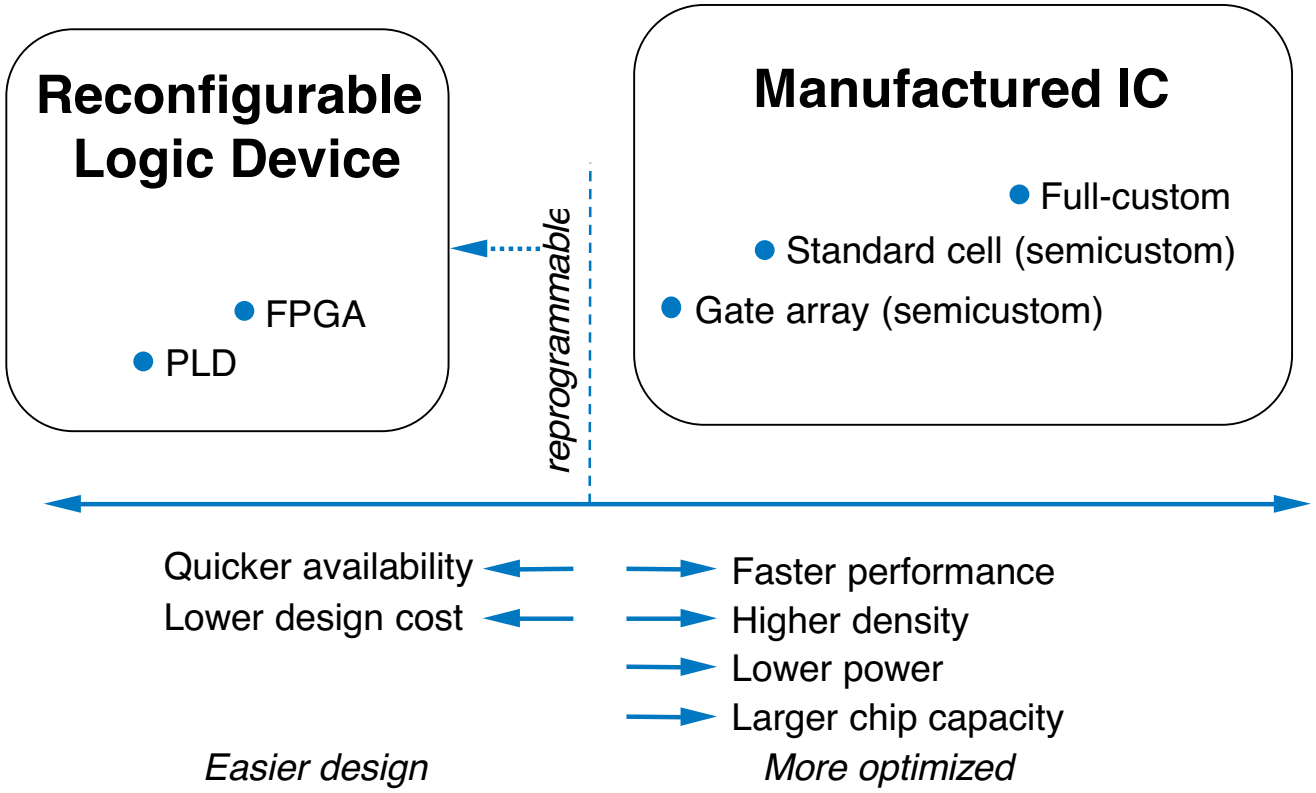


FPGA Internals: Programming an FPGA

- All configuration memory and LUTs are connected as one big shift register
  - Known as scan chain
- Shift in "bit file" of desired circuit



Technology Comparisons



Full custom 제외

Entire IC Design Flow based on HDL