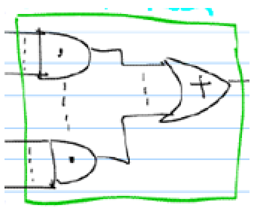
ET0901 Digital System Design

1. Programmable Logic Devices (PLDs)
2. Verilog – a hardware description language (HDL)
3. Designing Sequential Logic System (e.g. vending machine)

PLDs:

1. Simple PLDs (SPLDs)
   a. PROM (Programmable Read-Only Memory)
   b. PAL (Programmable Array of Logic)
   c. PLA (Programmable Logic Array)
   d. GAL (General Array Logic)
2. Complex PLDs (CPLDs)
3. Field Programmable Gate Array (**FPGA**s)

AND-OR array which connections are configurable (programmable)



Re-cap:

(DE1)

Sum-of-Products Form (SOP)

Example:

$$\overbrace{\underbrace{A . B}_{} + \underbrace{\bar{A} . B . \bar{C}}_{} + \underbrace{\bar{C} . \bar{D}}_{} + \underbrace{D}_{}}$$

Product of input variables (e.g. A, B, C, D) and/or their inversions.

Each product term should not have a sum within it; the following boolean expression is **not in SOP form**, for example:

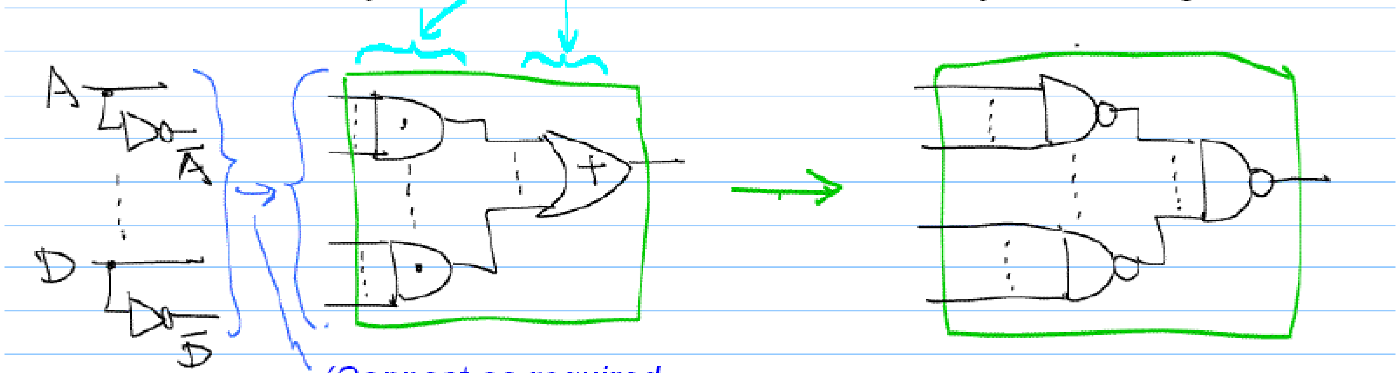
$$A . B + \bar{A} . B . \bar{C} + \bar{C} . \bar{D} + \underbrace{D . (A+B)}_{}$$

To change it to SOP form:

$$A . B + \bar{A} . B . \bar{C} + \bar{C} . \bar{D} + A.D + B.D$$

SOP expression leads to a 2-layer AND-OR circuit:

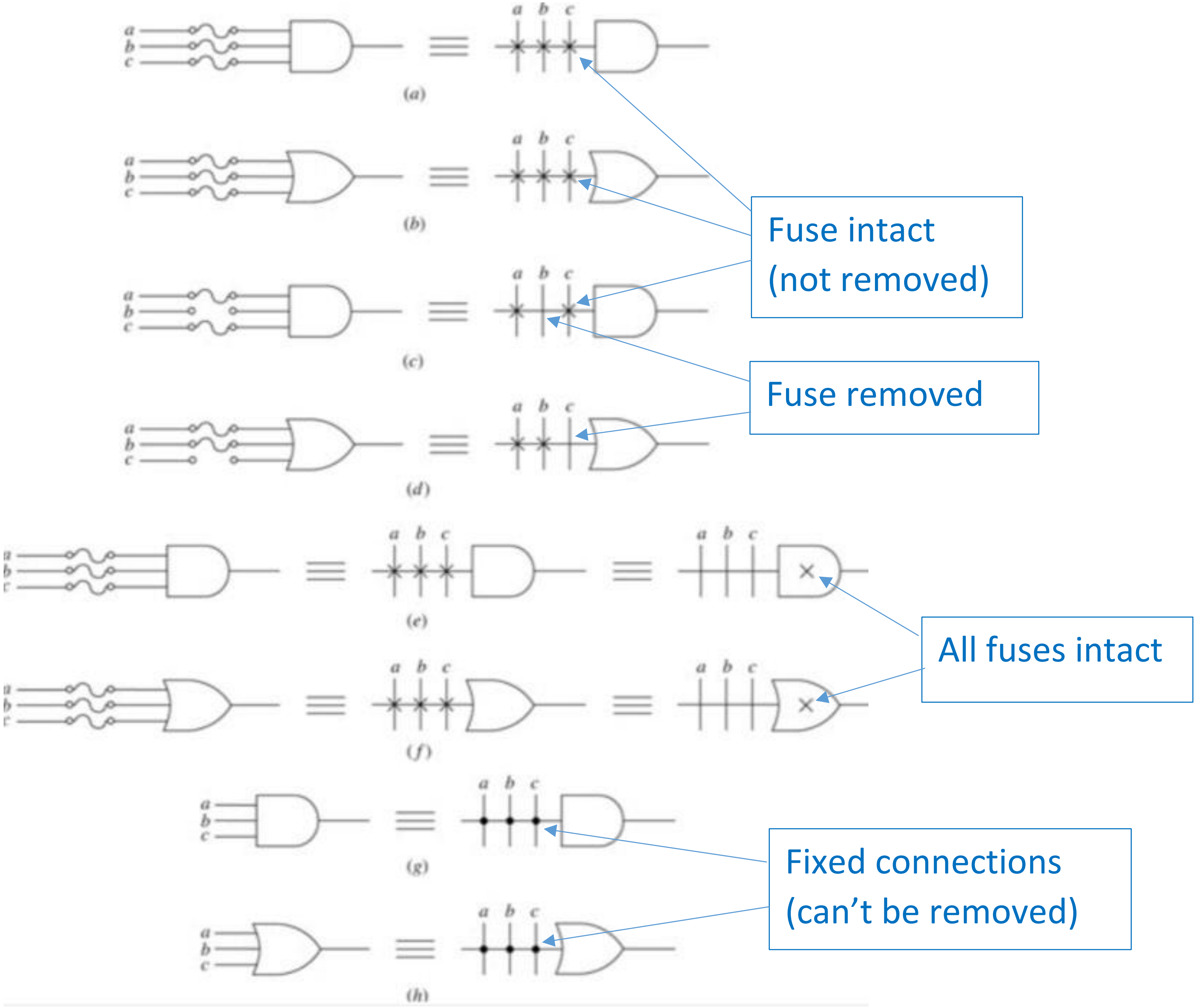
It can be converted to a 2-layer NAND gate circuit:



(Connect as required by the circuit)

PLD notation

Not Required



Fuse intact
(not removed)

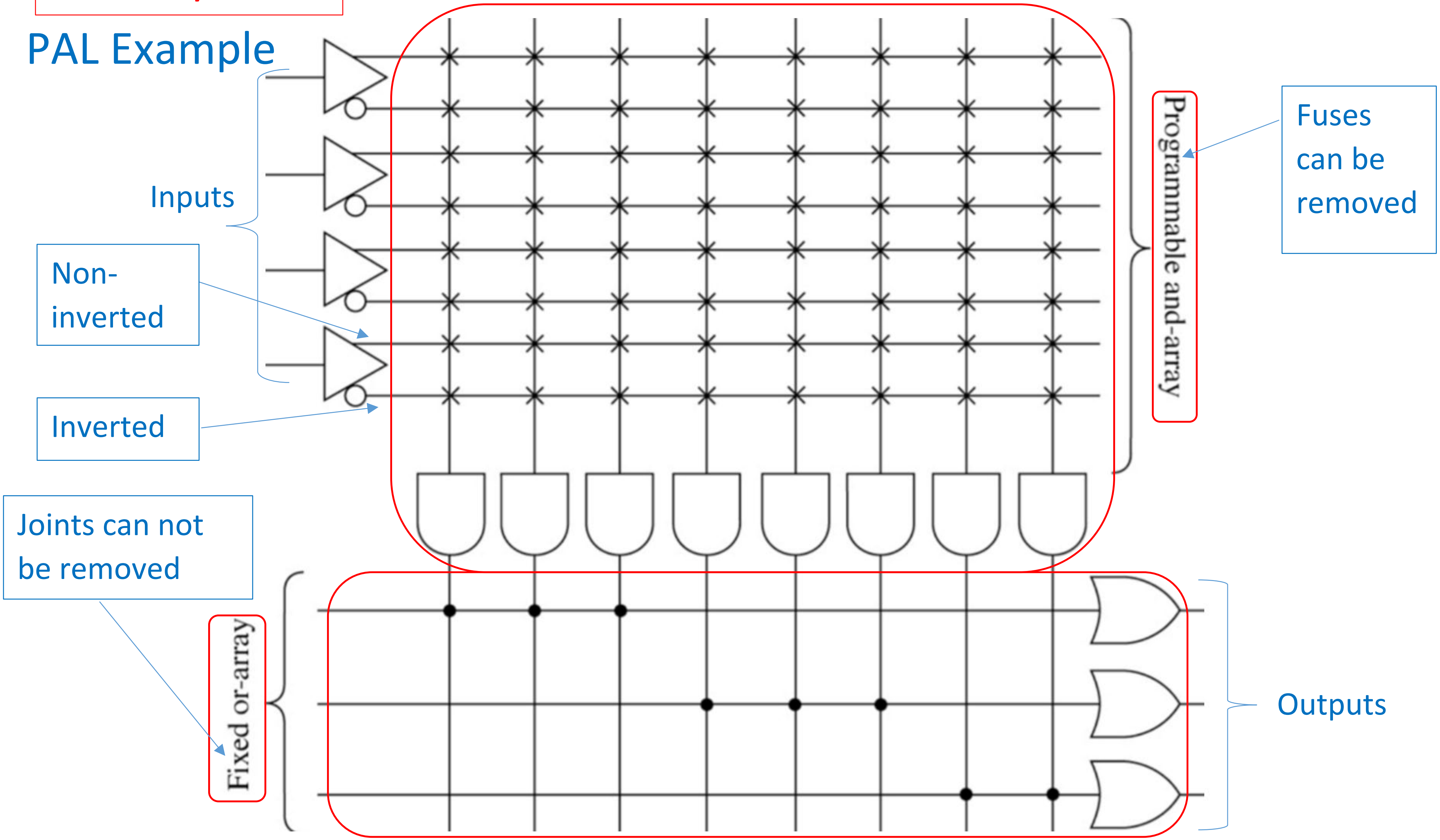
Fuse removed

All fuses intact

Fixed connections
(can’t be removed)

Not Required

PAL Example



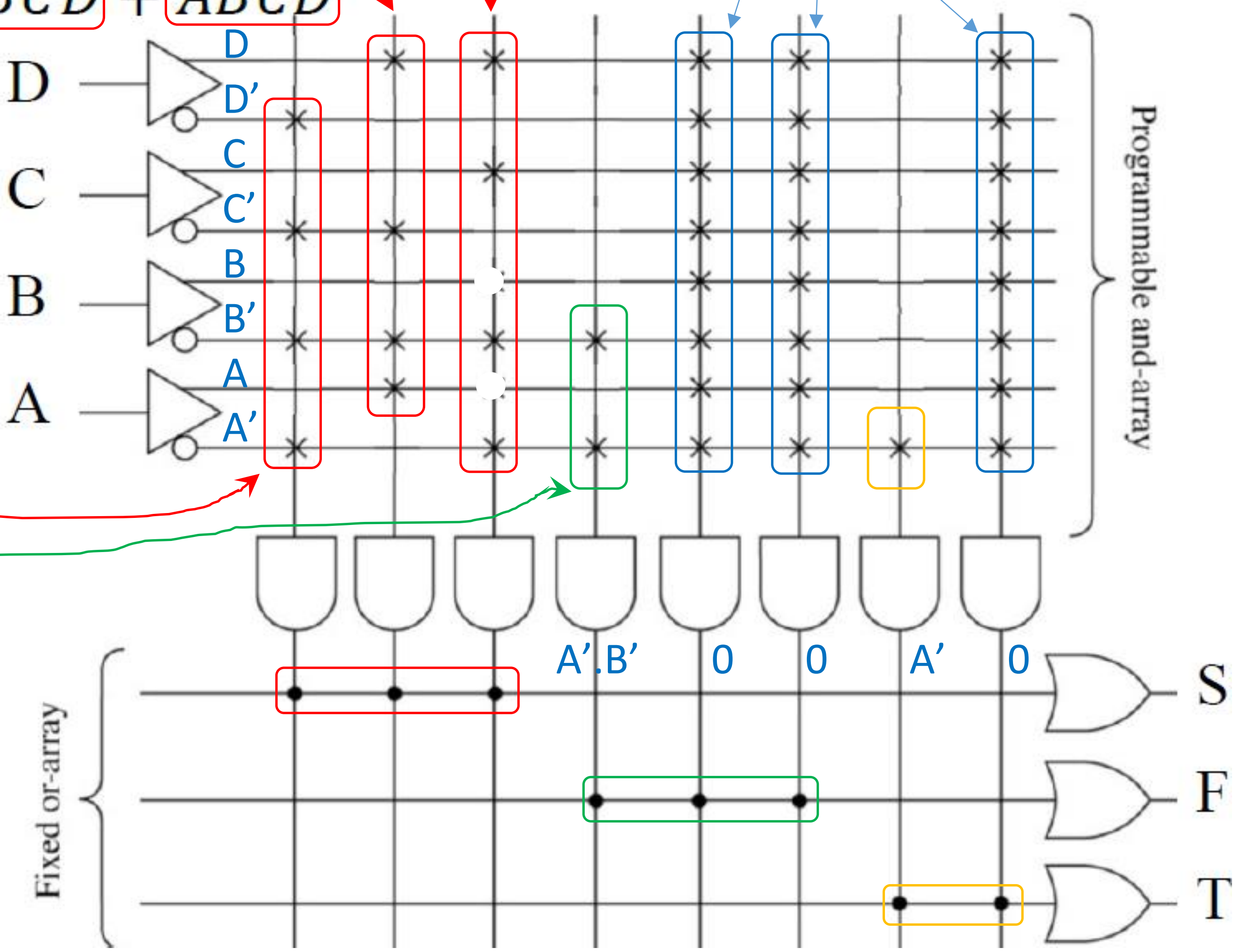
Not Required

$$S = \bar{A}\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + \bar{A}\bar{B}CD$$

$$F = \bar{A}\bar{B}$$

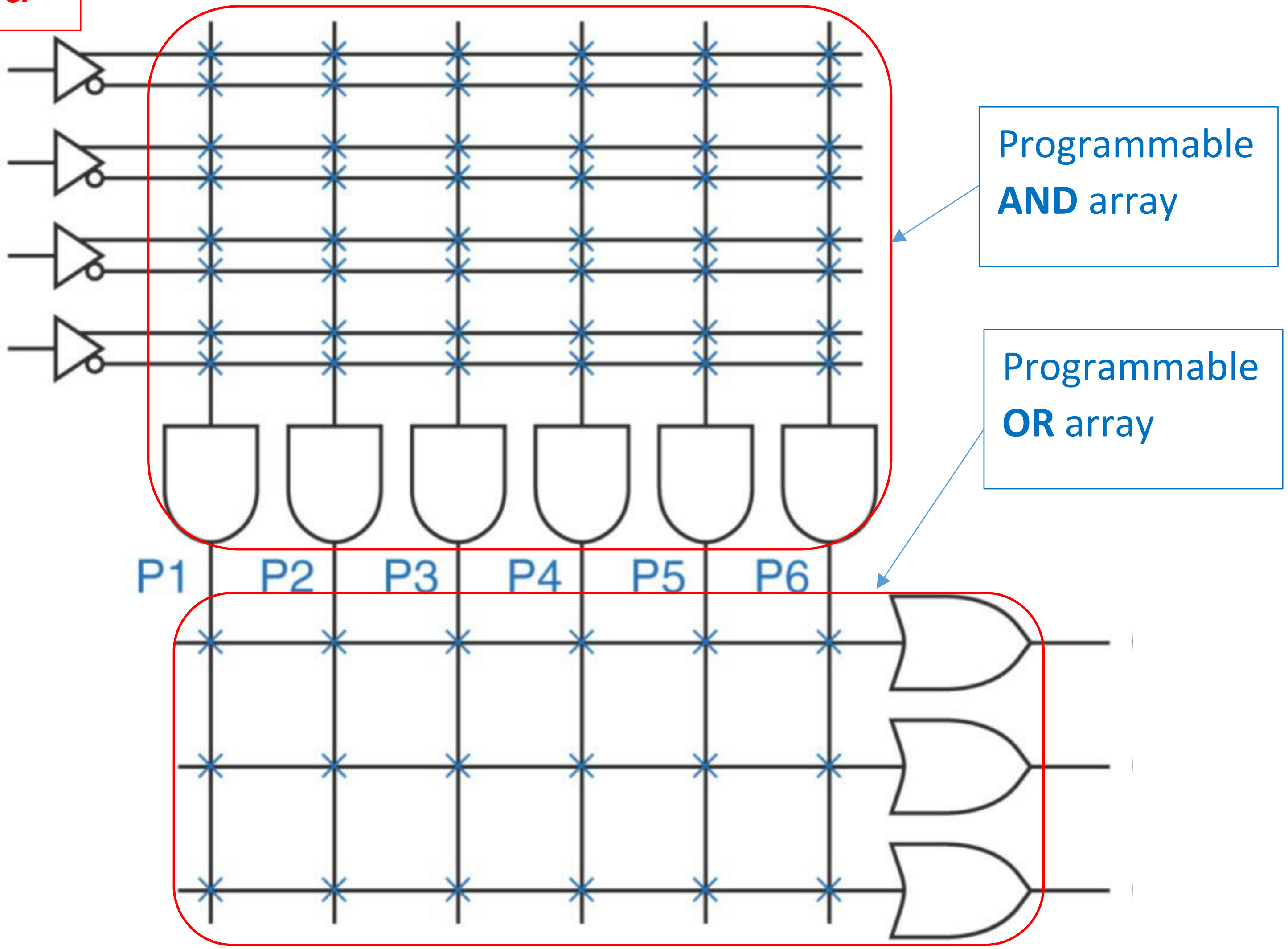
$$T = \bar{A}$$

D . D’ = 0 etc.



PLA Example

Not Required



Programmable **AND** array

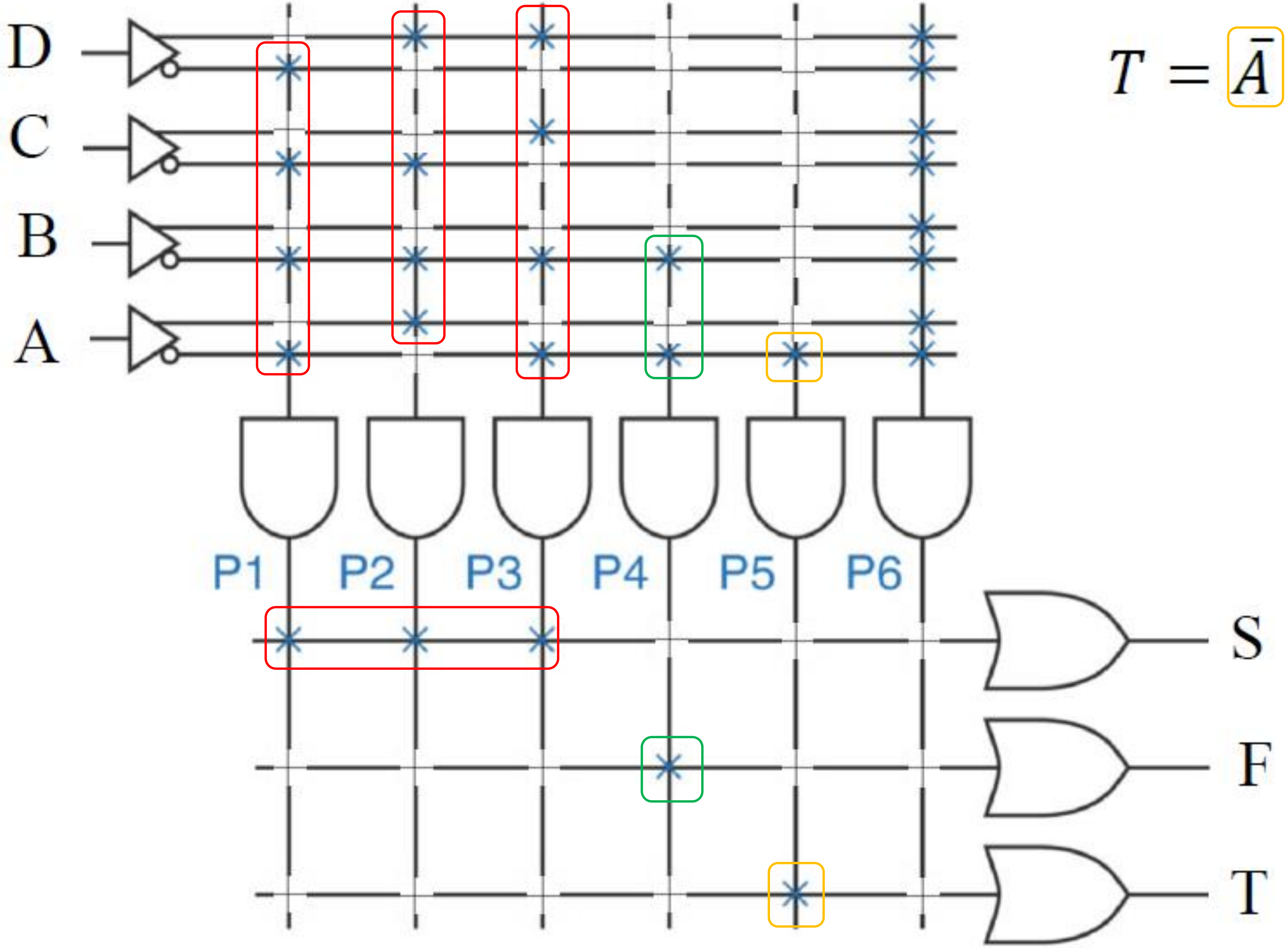
Programmable **OR** array

Not Required

$$S = \overset{(P1)}{\boxed{\bar{A}\bar{B}\bar{C}\bar{D}}} + \overset{(P2)}{\boxed{A\bar{B}\bar{C}D}} + \overset{(P3)}{\boxed{\bar{A}\bar{B}CD}}$$

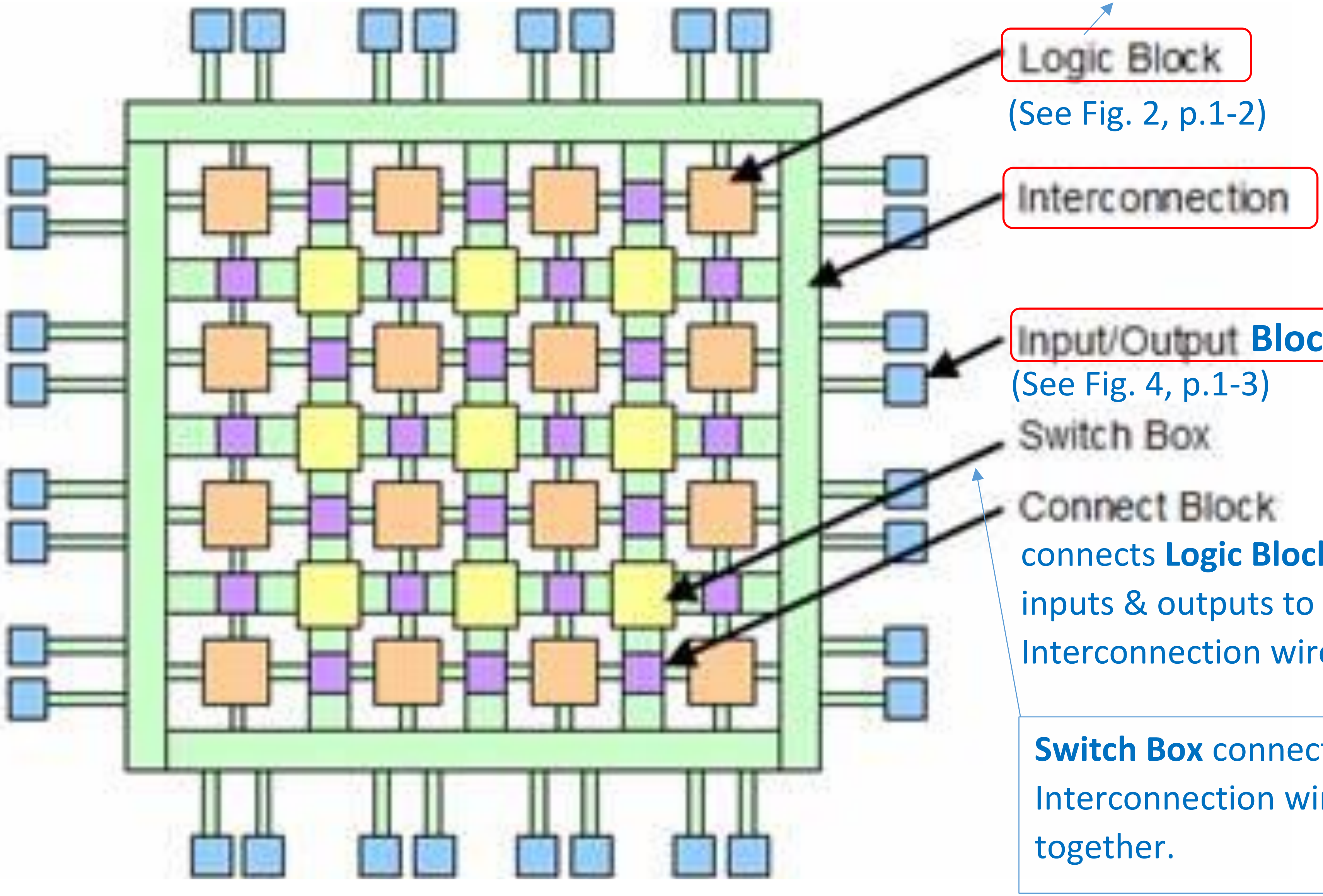
$$F = \boxed{\bar{A}\bar{B}}\ (P4)$$

$$T = \boxed{\bar{A}}\quad (P5)$$



FPGA structure (Fig. 1, p. 1-2)

Instead of AND/OR planes in SPLD & CPLD



(See Fig. 2, p.1-2)

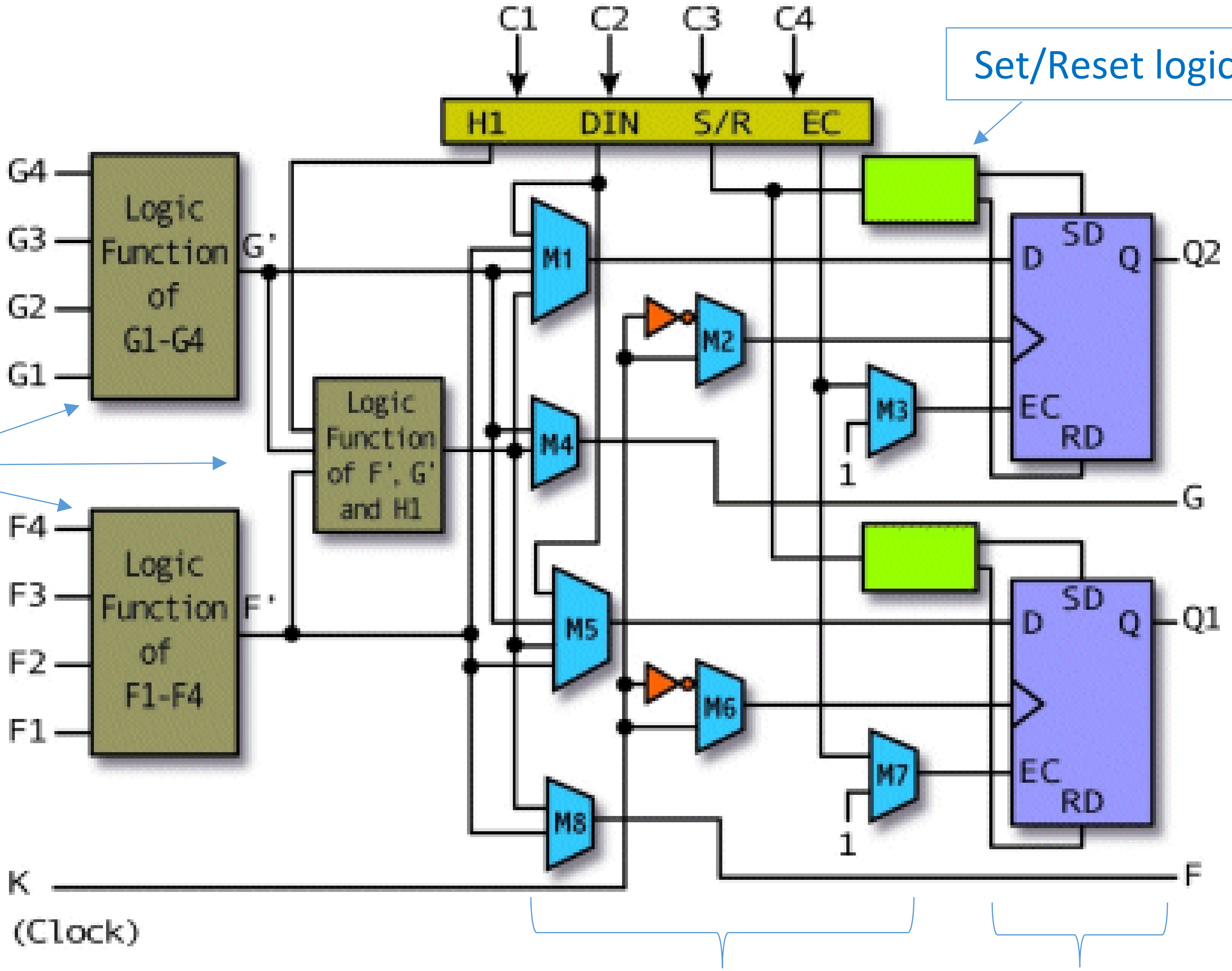
Block

(See Fig. 4, p.1-3)

connects **Logic Block**s' inputs & outputs to the Interconnection wires.

Switch Box connects Interconnection wires together.

Logic Block in FPGA (Fig. 2, p. 1-2)



Set/Reset logic

Look-Up Tables (LUT) (See Fig. 3)

All combinational logics are implemented with LUTs.

All flip-flops are driven by this common clock signal

MUXs (for inter-connection)

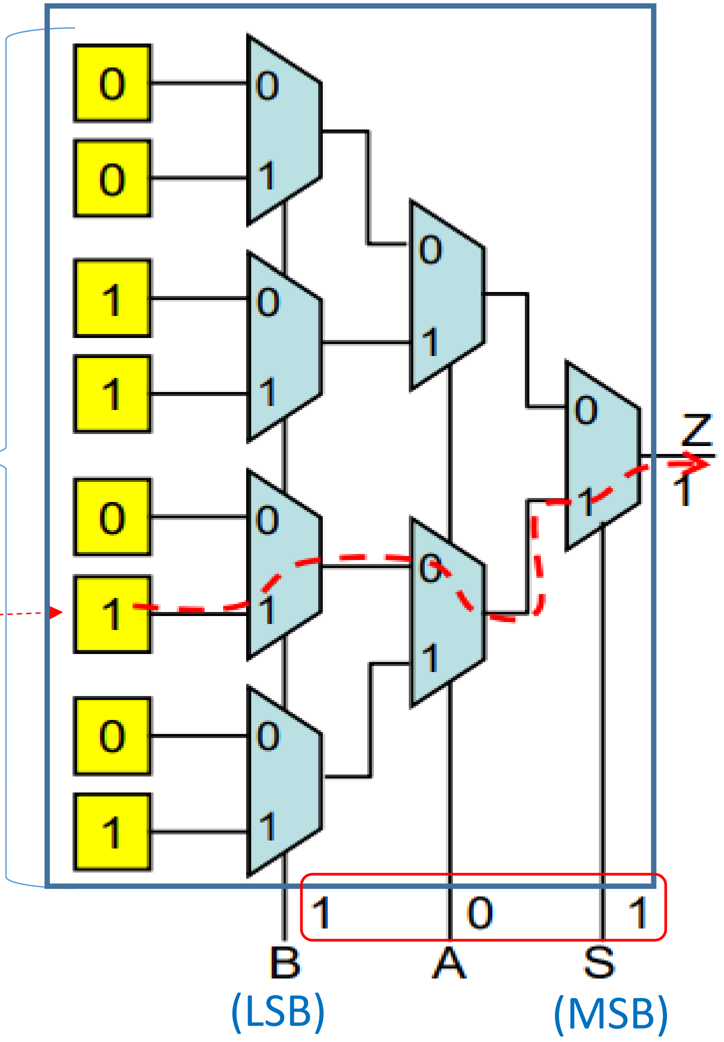
D flip-flops

Example: Implementing a 3- variable truth-table with a 3-variable Look-Up Table (LUT3).

(A 8:1 MUX can be formed from seven 2:1 MUXs.)

Truth table

S A B	Z
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

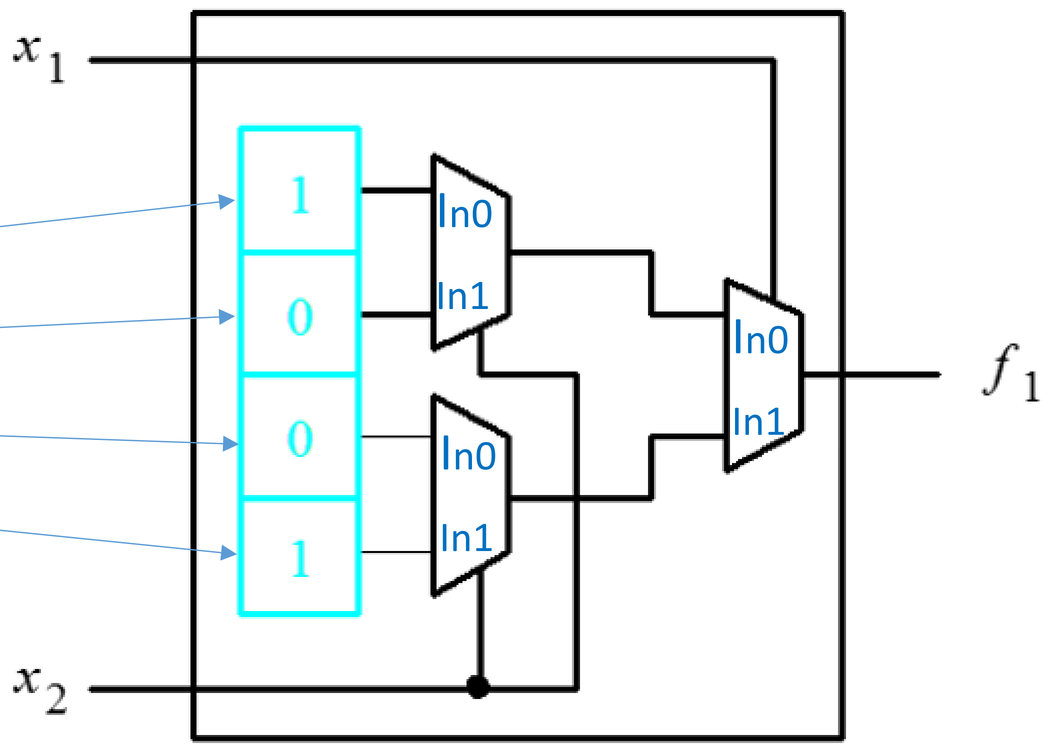


Implement a logic function in **LUT** (Fig. 1.13, Ch.1 p. 10)

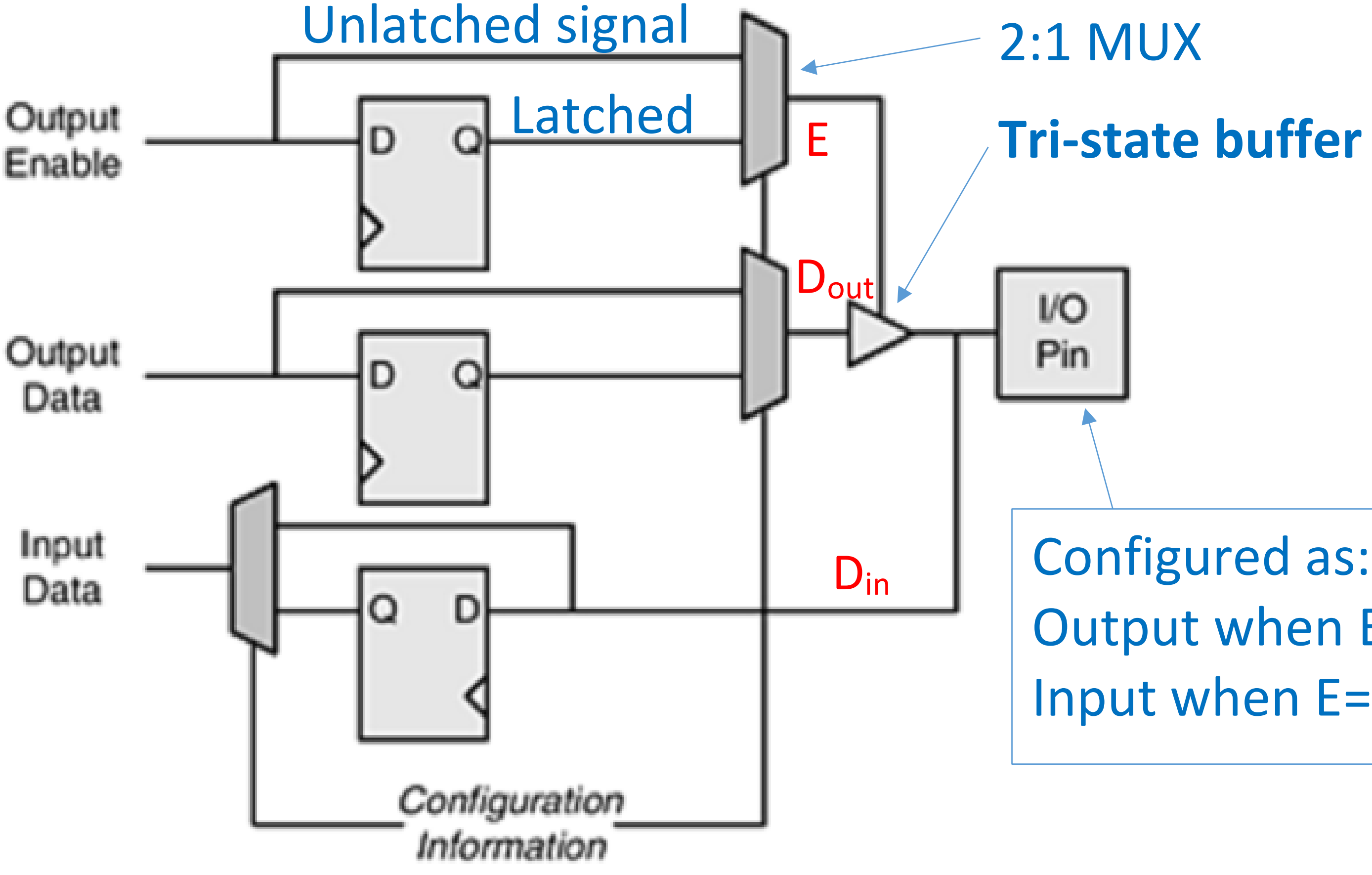
x_1	x_2	f_1
0	0	1
0	1	0
1	0	0
1	1	1

(b) $f_1 = \bar{x}_1\bar{x}_2 + x_1x_2$

2-variable LUT



Input / Output Block in FPGA (Fig. 1.14, Ch.1 p. 11)



Unlatched signal

Latched

2:1 MUX

E

Tri-state buffer

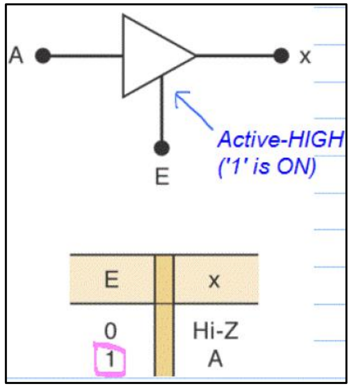
D_{out}

D_{in}

Configured as:
Output when E=1
Input when E=0.

0: select unlatched signal
1: select latched signal

Re-cap on
Tristate buffer:



E	x
0	Hi-Z
1	A