LEC2&3

CPLD&FPGA

CPLD(Complex Programmable device)

- Introduction
- PLD
- Expansion Theorm
- CPLD
- FPGA

Introduction

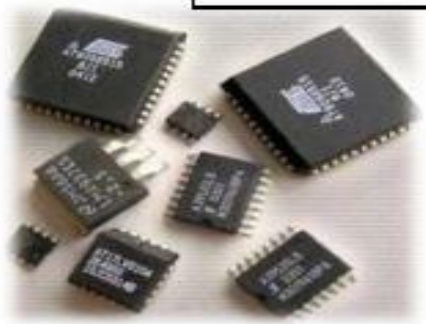
Ideally, though, the hardware designer wanted something that gave him or her the flexibility and complexity of an ASIC but with the shorter turn-around time of a programmable device. The solution came in the form of two new devices - the Complex Programmable Logic Device (CPLD) and the Field Programmable Gate Array. As can be seen in Figure 4, CPLDs and FPGAs bridge the gap between PALs and Gate Arrays. CPLDs are as fast as PALs but more complex. FPGAs approach the complexity of Gate Arrays but are still programmable

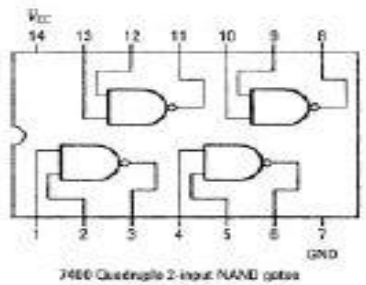
Introduction

The early digital electronic devices were rather simple and consisted only of a handful of logic gates. However, over time, the complexity of digital circuits increased thus, programmability became an important feature of modern digital control devices. Two different classes of digital devices emerged to provide programmability. The first class consisted of fixed hardware design with reprogrammable software. Examples of such devices include microcontrollers and microprocessors. The second class of digital devices featured reconfigurable hardware to achieve flexible logic circuit design. Examples of such devices include FPGAs, SPLDs, and CPLDs. Integrated circuits meant for digital control and signal processing typically consist of processor, logic circuit, and memory. Each of these modules can be realized using different technologies.

Type of Circuits

Type of circuits	Number of gates
Small-scale integration (SSI)	1-10
Medium-scale integration (MSI)	10-100
Large-scale integration (LSI)	100-1,000
Very-large-scale integration (VLSI)	1,000 up
Ultra-large-scale integration (ULSI)	1,000,000





7400 Quadruple 2-input NAND gates

Different Types of PLDs

Basically, PLDs can be categorized into three types. They are:

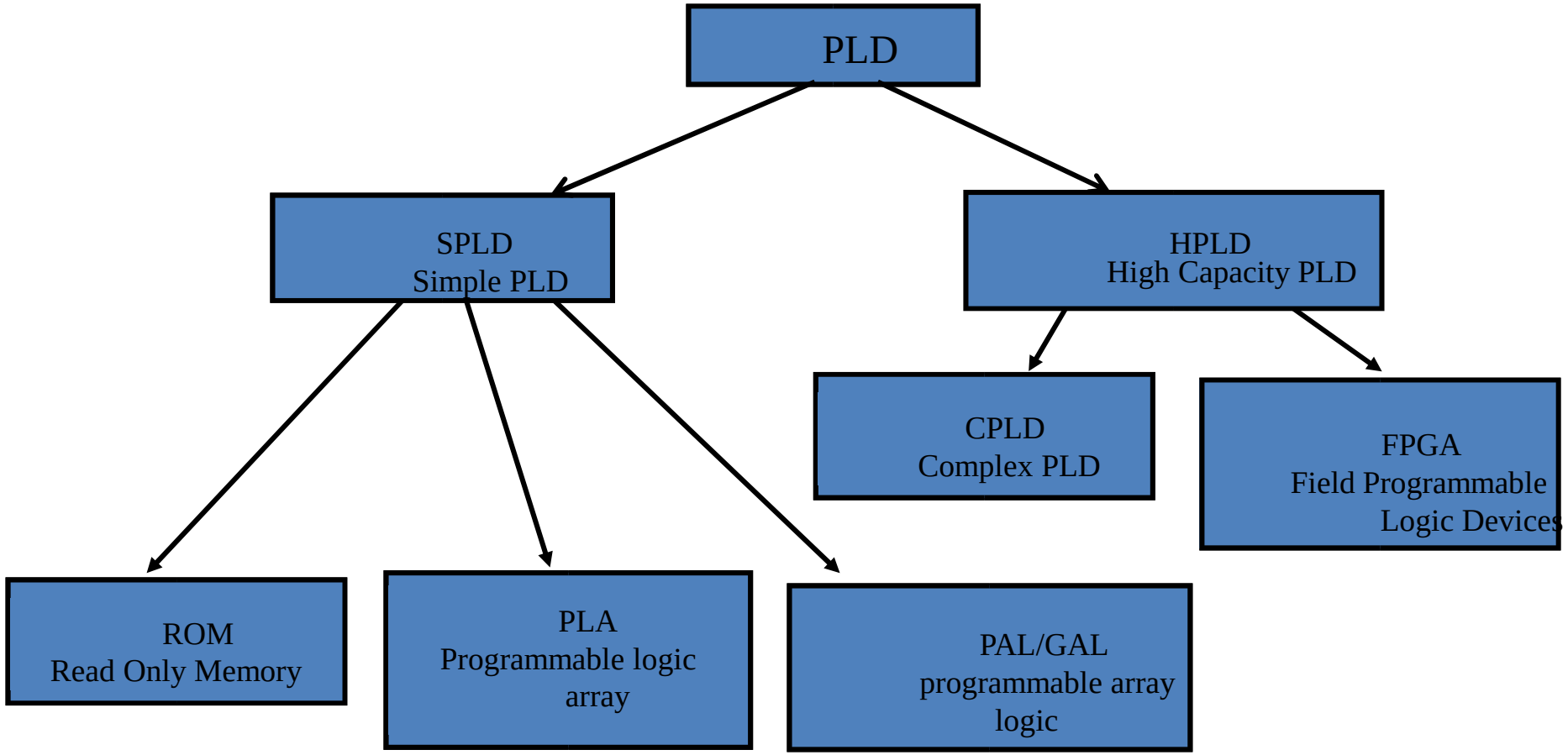
- Simple Programmable Logic Devices (SPLD)
- Complex Programmable Logic Devices (CPLD)
- Field Programmable Gate Arrays (FPGA)

The Simple Programmable Logic Devices are further divided into:

- Programmable Logic Array (PLA)
- Programmable Array Logic (PAL)
- Generic Array Logic (GAL)

Programmable Logic Devices – Programmable Technologies

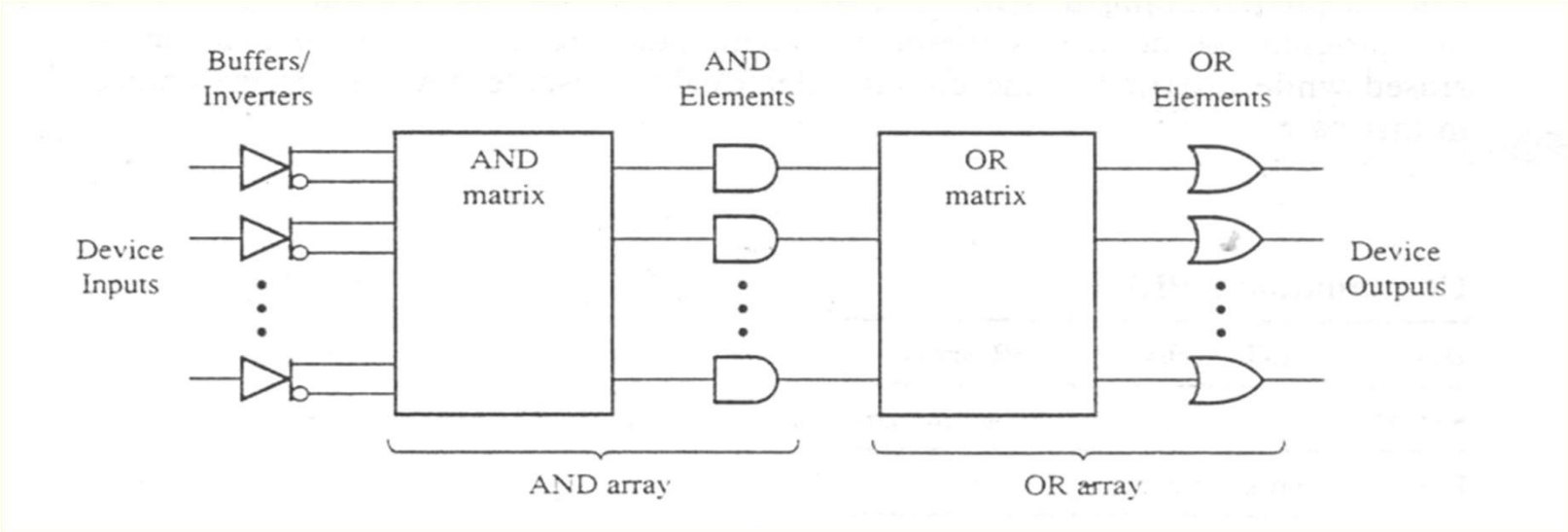
Fig(1)



Programmable Logic Devices – Programmable Technologies.

Internal Structures of PLD

Fig(2)



Shannon's Expansion Theorem

T11a:

$$F(X1,\ldots,Xn) = F(0,X2,\ldots,Xn) \bullet \overline{X1} + F(1,X2,\ldots,Xn) \bullet X1$$

T11b:

$$F(X1,\ldots,Xn) = [F(1,X2,\ldots,Xn) + \overline{X1}] \bullet [F(0,X2,\ldots,Xn) + X1]$$

Shannon's Expansion Theorem

Let X1=0 in T11a

$$F(X1,\ldots,Xn) = F(0,X2,\ldots,Xn)\bullet\bar{0} + F(1,X2,\ldots,Xn)\bullet 0$$
$$F(X1,\ldots,Xn) = F(0,X2,\ldots,Xn)\bullet 1 + F(1,X2,\ldots,Xn)\bullet 0$$
$$F(X1,\ldots,Xn) = F(0,X2,\ldots,Xn)$$

Let X1=1 in T11a

$$F(X1,\ldots,Xn) = F(0,X2,\ldots,Xn)\bullet\bar{1} + F(1,X2,\ldots,Xn)\bullet 1$$
$$F(X1,\ldots,Xn) = F(0,X2,\ldots,Xn)\bullet 0 + F(1,X2,\ldots,Xn)\bullet 1$$
$$F(X1,\ldots,Xn) = F(1,X2,\ldots,Xn)$$

Shannon's Expansion Theorem

$$
\begin{aligned}
F(X1,\ldots,Xn) &= F(0,0,X3,\ldots,Xn)\bullet\overline{X1}\bullet\overline{X2}+F(0,1,X3,\ldots,Xn)\bullet\overline{X1}\bullet X2 \\
&\quad +F(1,0,X3,\ldots,Xn)\bullet X1\bullet\overline{X2}+F(1,1,X3,\ldots,Xn)\bullet X1\bullet X2 \\
&= I0\bullet\overline{X1}\bullet\overline{X2}+I1\bullet\overline{X1}\bullet X2+I2\bullet X1\bullet\overline{X2}+I3\bullet X1\bullet X2 \\
&= I0\bullet m0+I1\bullet m1+I2\bullet m2+I3\bullet m3 \\
&= \sum_{k=0}^{2^n-1} Ki\bullet m_i \qquad \text{Where } m_i = m_i(X1,X2)
\end{aligned}
$$

Example 7-1/p333

- Design a circuit using MUX to implement the following function by applying Shannon's Expansion Theorem T11a with respect to A and B

$$F(A,B,C) = A + B.\overline{C}$$

Solution

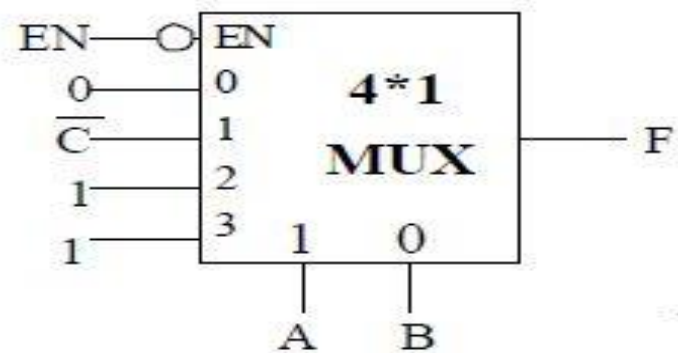
$$F(A,B,C) = F(0,0,C).\overline{A}.\overline{B} + F(0,1,C).\overline{A}.B + F(1,0,C).A.\overline{B} + F(1,1,C).A.B$$

$$F(0,0,C) = 0 + 0.\overline{C} = 0$$

$$F(0,1,C) = 0 + 1.\overline{C} = \overline{C}$$

$$F(1,0,C) = 1 + 0.\overline{C} = 1$$

$$F(1,1,C) = 1 + 1.\overline{C} = 1$$



Analyzing a Multiplexer Design

$$F(A,B,C)=\sum_{k=0}^{2^n-1} Ki \bullet m_i \qquad \text{Where } m_i = m_i(A,B)$$

$$= I0 \bullet m0 + I1 \bullet m1 + I2 \bullet m2 + I3 \bullet m3$$

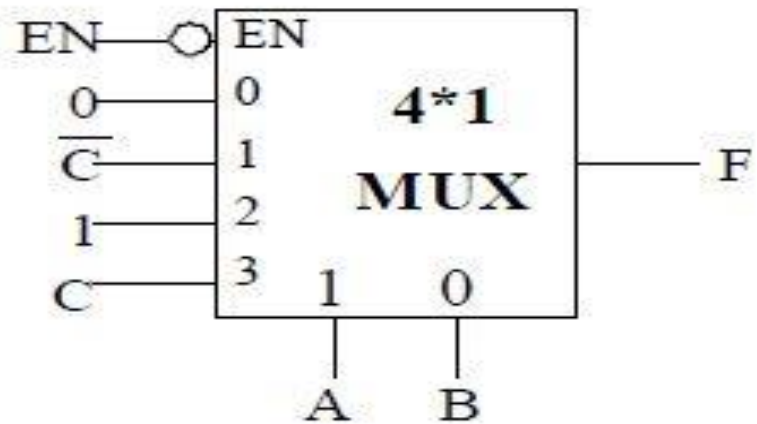
$$= 0 \bullet m0 + \overline{C} \bullet m1 + 1 \bullet m2 + C \bullet m3$$

$$= 0.\overline{A}.\overline{B} + \overline{C}.\overline{A}.B + 1.A.\overline{B} + C.A.B$$

$$= \overline{A}.B.\overline{C} + A.\overline{B}.(C + \overline{C}) + A.B.C$$

$$= m_2 + m_4 + m_5 + m_7$$

$$= \sum m(2,4,5,7)$$



CPLD

- CPLD : is a logic device with completely programmable AND/OR arrays and macrocells. It consists of a few thousand logic gates. In terms of complexity, CPLD (complex programmable logic device) lies in between SPLD (simple programmable logic device) and FPGA and thus, inherits features from both these devices. CPLDs are more complex than SPLDs but less complex than FPGAs.
- The **CPLD Architectures**

The diagram in Figure 3 shows the internal architecture of a typical CPLD. While each manufacturer has a different variation, in general they are all similar in that they consist of function blocks, input/output block, and an interconnect matrix. The devices are programmed using programmable elements that, depending on the technology of the manufacturer, can be EPROM cells, EEPROM cells, or Flash EPROM cells

CPLD

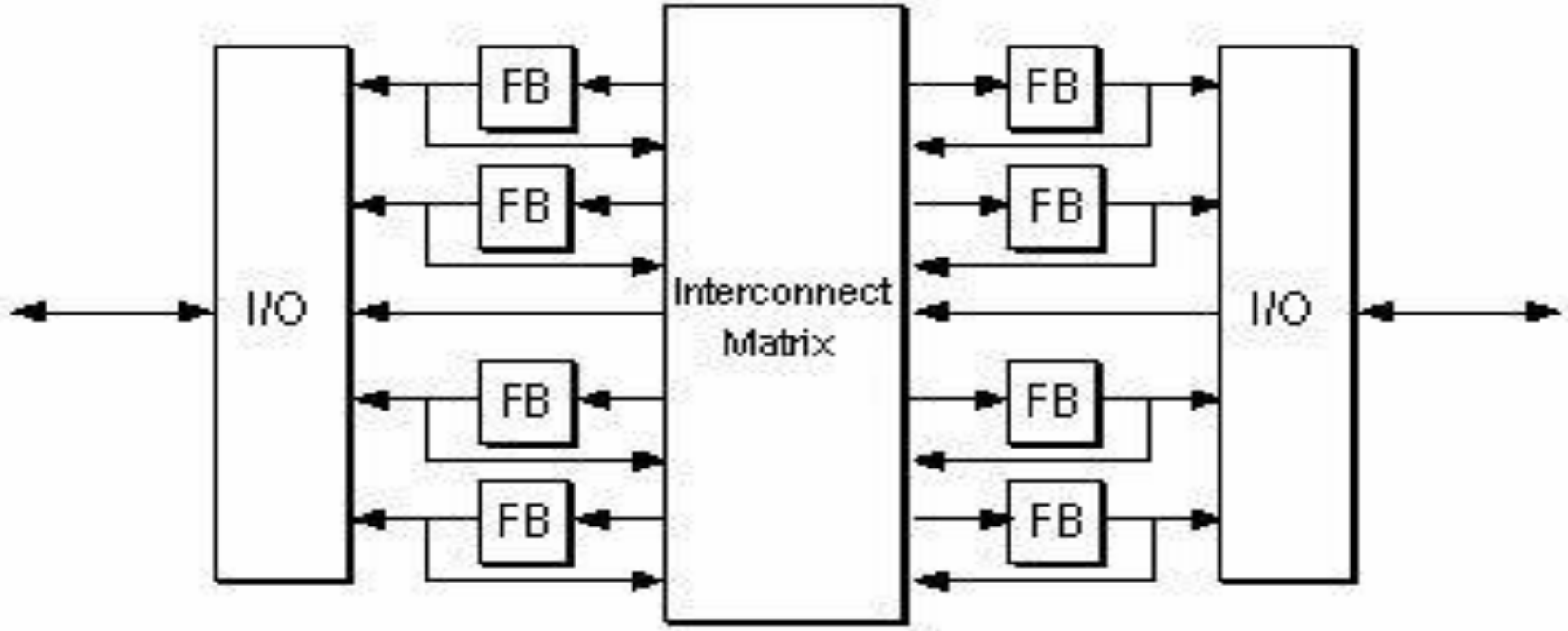
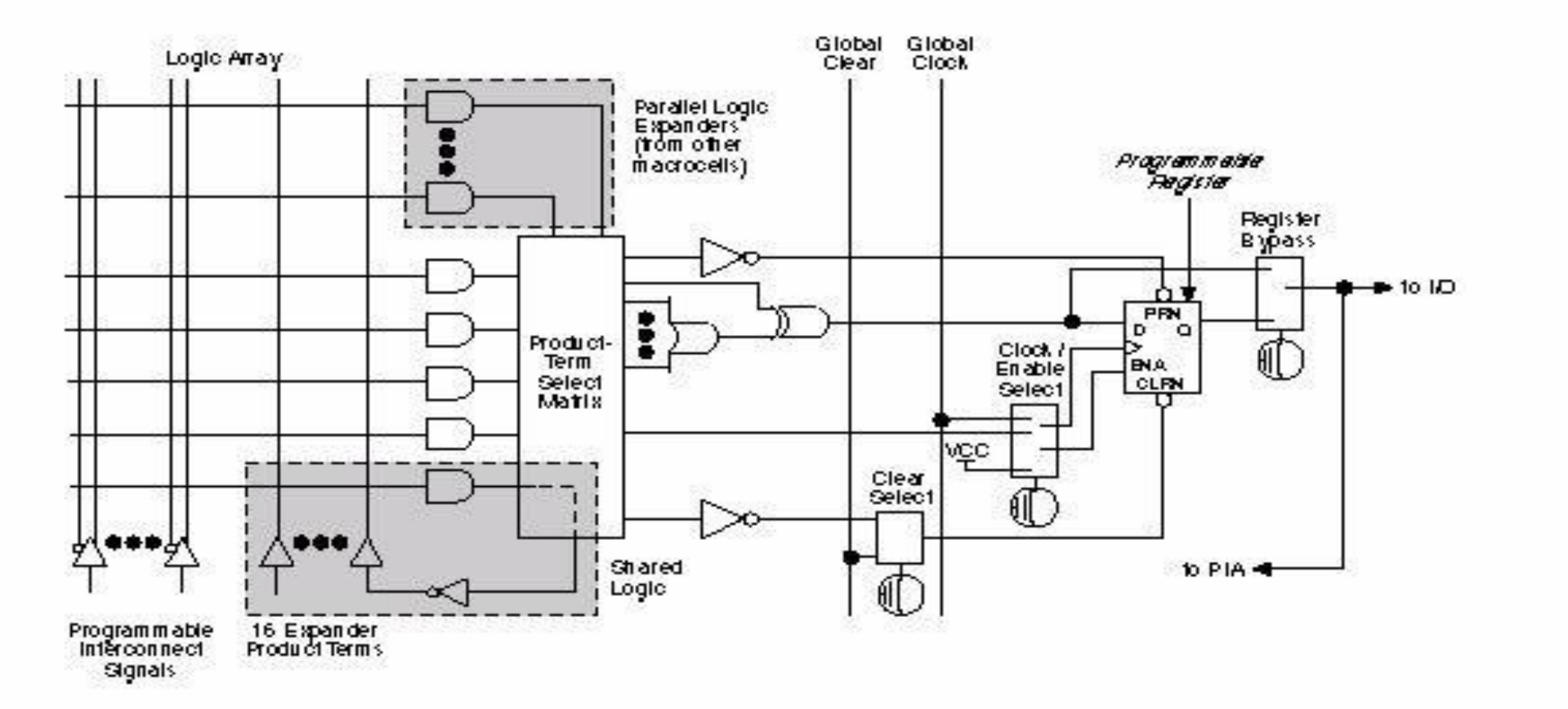


Figure 3 CPLD Architecture

Function Blocks

A typical function block is shown in Figure 4. The AND plane still exists as shown by the crossing wires. The AND plane can accept inputs from the I/O blocks, other function blocks, or feedback from the same function block. The terms and then OR together using a fixed number of OR gates, and terms are selected via a large multiplexer. The outputs of the mux can then be sent straight out of the block, or through a clocked flip-flop. This particular block includes additional logic such as a selectable exclusive OR and a master reset signal, in addition to being able to program the polarity at different stages. Usually, the function blocks are designed to be similar to existing PAL architectures, such as the 22V10, so that the designer can use familiar tools or even older designs without changing them.

Function Block



Figure(4)

I/O Blocks

Figure 2. shows a typical I/O block of a CPLD. The I/O block is used to drive signals to the pins of the CPLD device at the appropriate voltage levels with the appropriate current. Usually, a flip-flop is included, as shown in the figure. This is done on outputs so that clocked signals can be output directly to the pins without encountering significant delay. It is done for inputs so that there is not much delay on a signal before reaching a flip-flop which would increase the device hold time requirement. Also, some small amount of logic is included in the I/O block simply to add some more resources to the device.

I/O Blocks

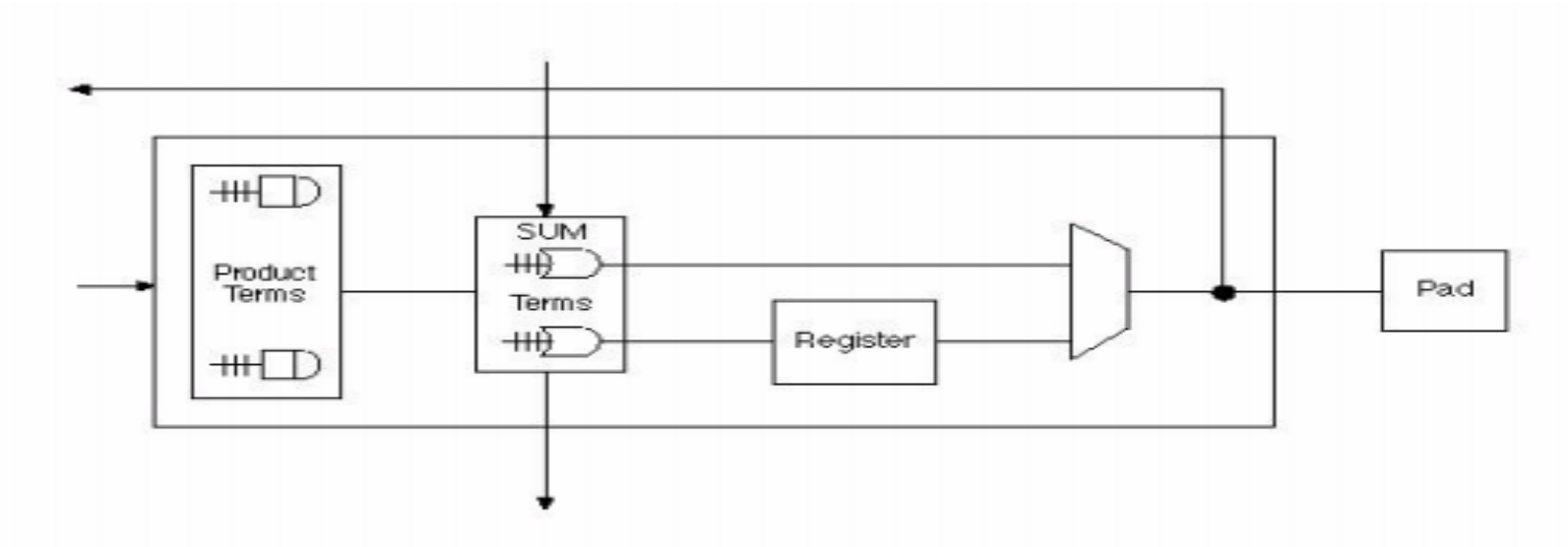


Figure 7 CPLD Input/Output Block

Interconnect and Programmable Elements

- **Interconnect** :The CPLD interconnect is a very large programmable switch matrix that allows signals from all parts of the device go to all other parts of the device. While no switch can connect all internal function blocks to all other function blocks, there is enough flexibility to allow many combinations of connections.
- **Programmable Elements** :Different manufacturers use different technologies to implement the programmable elements of a CPLD. The common technologies are Electrically . Programmable Read Only Memory (EPROM), Electrically Erasable PROM (EEPROM) and Flash EPROM. These technologies are similar to, or next generation versions of, the technologies that were used for the simplest programmable devices, PROMs.

CPLD Architecture Issues

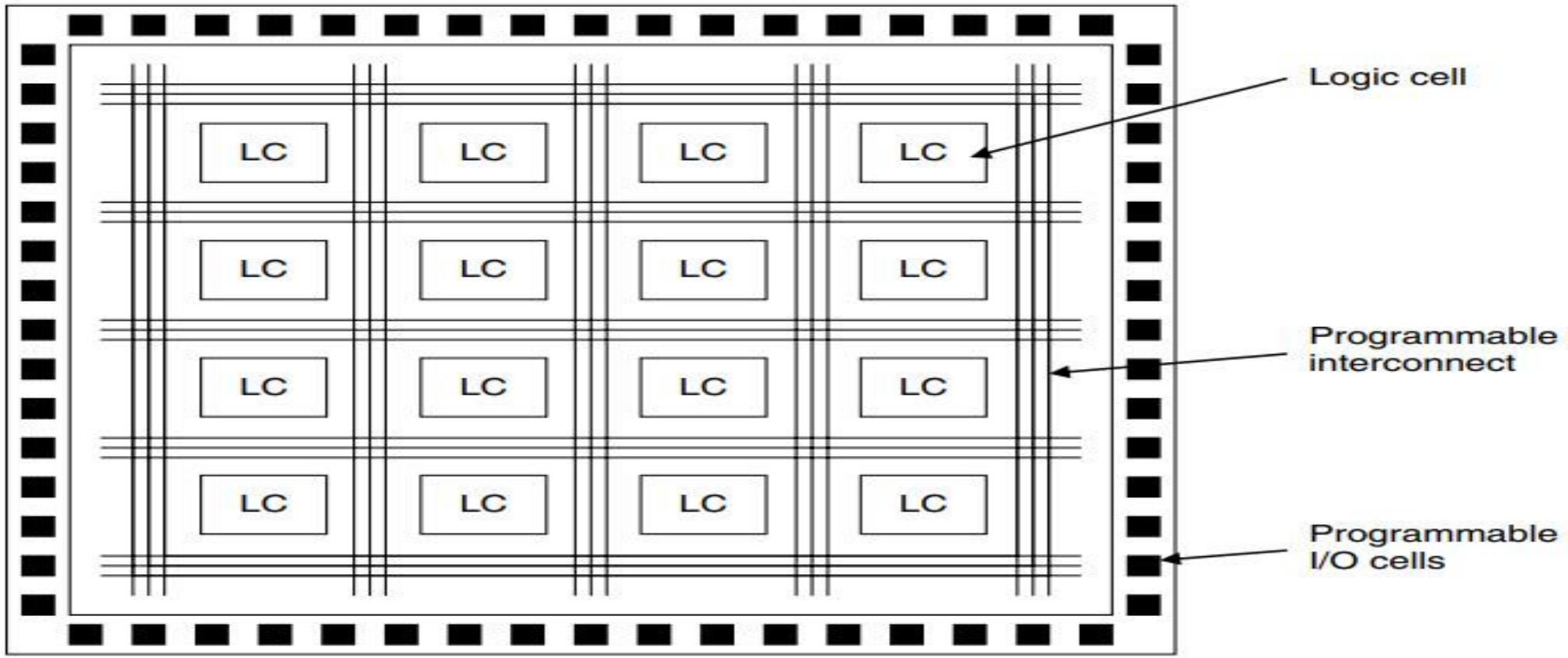
- When considering a CPLD for use in a design, the following issues should be taken into account:
- The programming technology
  - EPROM, EEPROM, or Flash EPROM? This will determine the equipment needed to program the devices and whether they came be programmed only once or many times.
- The function block capability
  - Function blocks are there in the device
  - how many product and sum terms can be used
  - The are the minimum and maximum delays through the logic
  - The additional logic resources are there such as XNORs, ALUs, etc.
  - The kind of register controls are available (e.g., clock enable, reset, preset, polarity control)
- The I/O capability
  - How many I/O are independent, used for any function, and how many are dedicated for clock input, master reset, etc.?
  - What is the output drive capability in terms of voltage levels and current?

Example CPLD Families

Some CPLD families from different vendors are listed below:

- Altera MAX 7000 and MAX 9000 families
- Atmel ATF and ATV families
- Lattice ispLSI family
- Lattice (Vantis) MACH family
- Xilinx XC9500 family

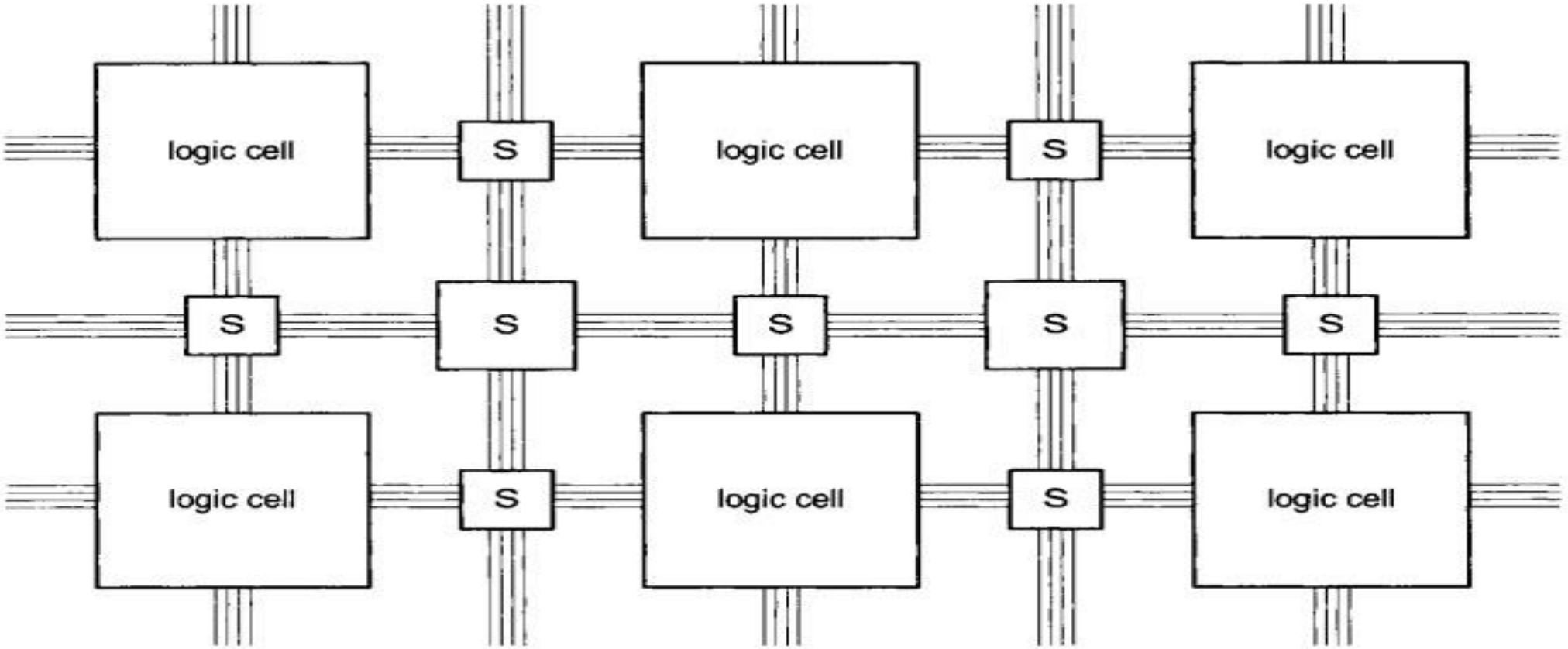
FPGA: is made up of configurable logic cells and programmable interconnections. These are surrounded by a number of programmable IO blocks, which are used to talk to the external world



What is FPGA

- Field Programmable Gate Arrays or FPGAs in short are pre-fabricated Silicon devices that consists of a matrix of reconfigurable logic circuitry and programmable interconnects arranged in a two-dimensional array. The programmable Logic Cells can be configured to perform any digital function and the programmable interconnects (or switches) provide the connections among different logic cells. Using an FPGA, you can implement any custom design by specifying the logic or function of each logic block and setting the connection of each programmable switch. Since this process of designing a custom circuit is done in the field rather than in a fab, the device is known as “Field Programmable”.
- The following image shows a typical internal structure of an FPGA in a very broad sense

FPGA

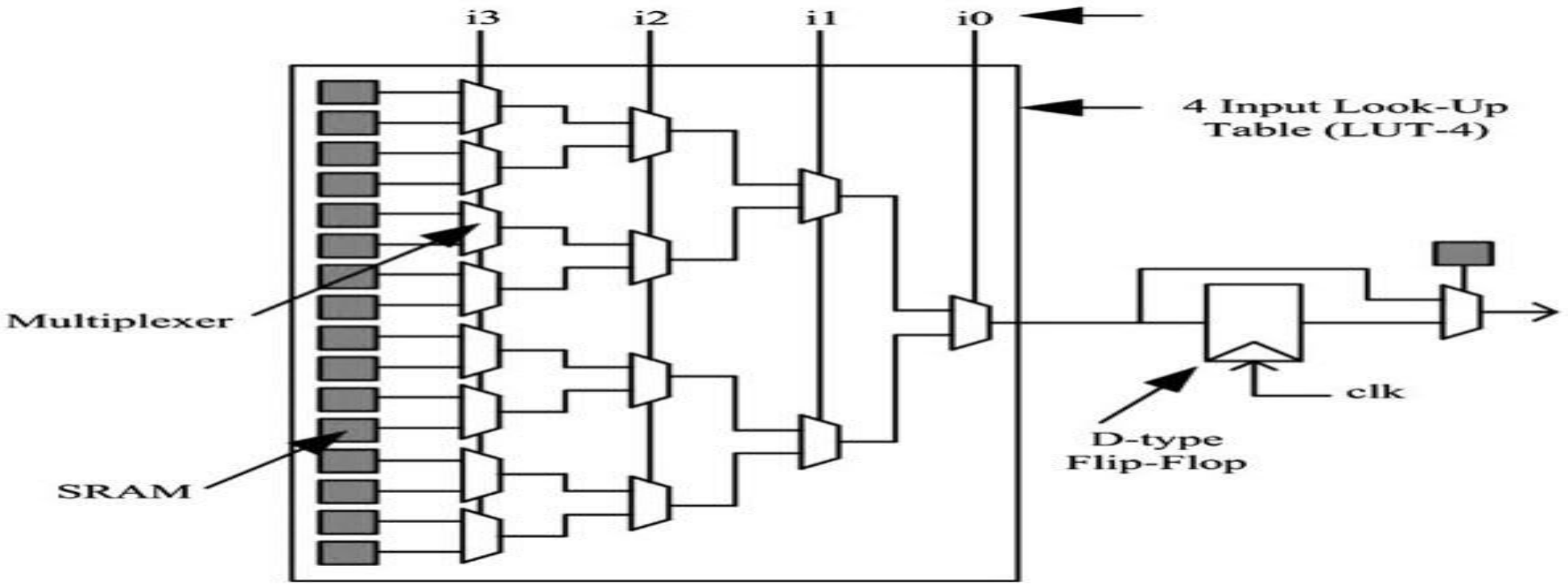

logic cell
S
logic cell
S
logic cell
S
S
S
S
S
logic cell
S
logic cell
S
logic cell

Components of an FPGA

Typically, an FPGA consists of three basic components. They are:

- Programmable Logic Cells (or Logic Blocks) – responsible for implementing the core logic functions.
- Programmable Routing – responsible for connecting the Logic Blocks.
- IO Blocks – which are connected to the Logic Blocks through the routing and help to make external connections

A Logic Block can be made up of a single Basic Logic Element or a set of interconnected Basic Logic Elements, where a Basic Logic Element is a combination of a Look-up table (which is in turn made up of SRAM and Multiplexors) and a Flip-flop



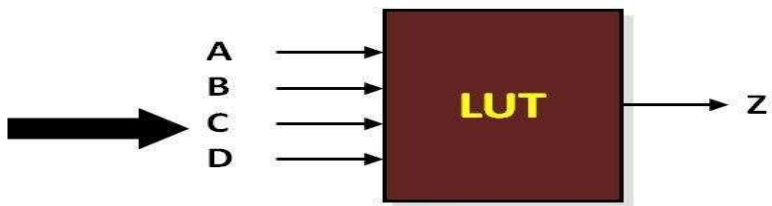
A LUT with ‘n’ inputs consists of 2^n configuration bits, which are implemented by SRAM Cells. Using these 2^n SRAM Bits, the LUT can be configured to implement any logical function.

Look-Up Tables (LUT)

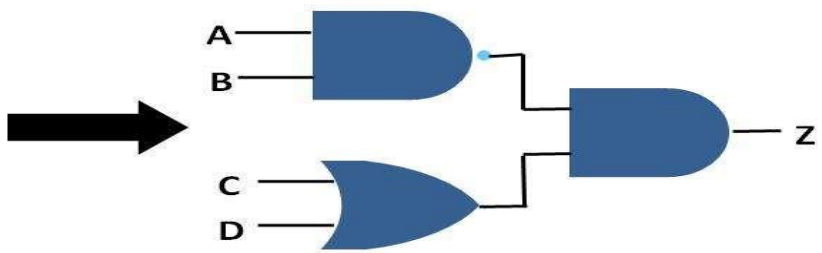
- Look-up table with N-inputs can be used to implement any combinatorial function of N inputs
- LUT is programmed with the truth-table

A	B	C	D	Z
0	0	0	0	**0**
0	0	0	1	**1**
0	0	1	0	**1**
0	0	1	1	**1**
0	1	0	0	**0**
0	1	0	1	**1**
0	1	1	0	**1**
0	1	1	1	**1**
1	0	0	0	**0**
1	0	0	1	**1**
1	0	1	0	**1**
1	0	1	1	**1**
1	1	0	0	**0**
1	1	0	1	**0**
1	1	1	0	**0**

Truth-table



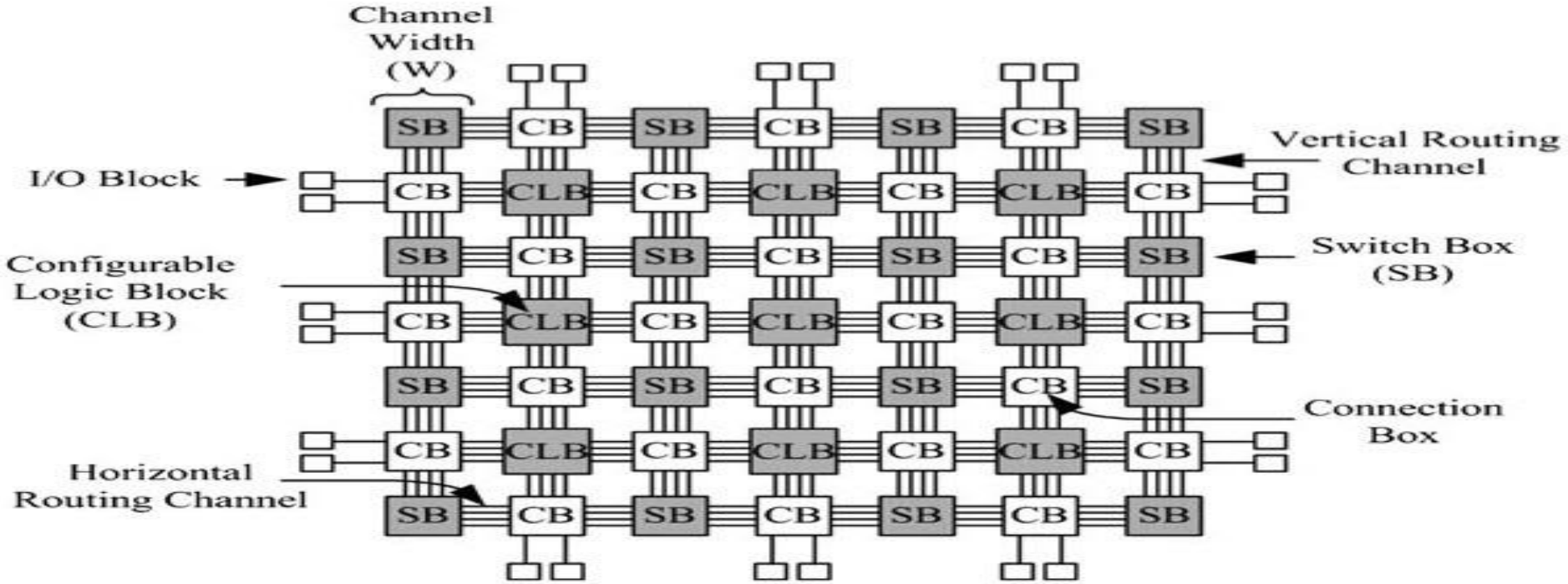
LUT implementation



Gate implementation

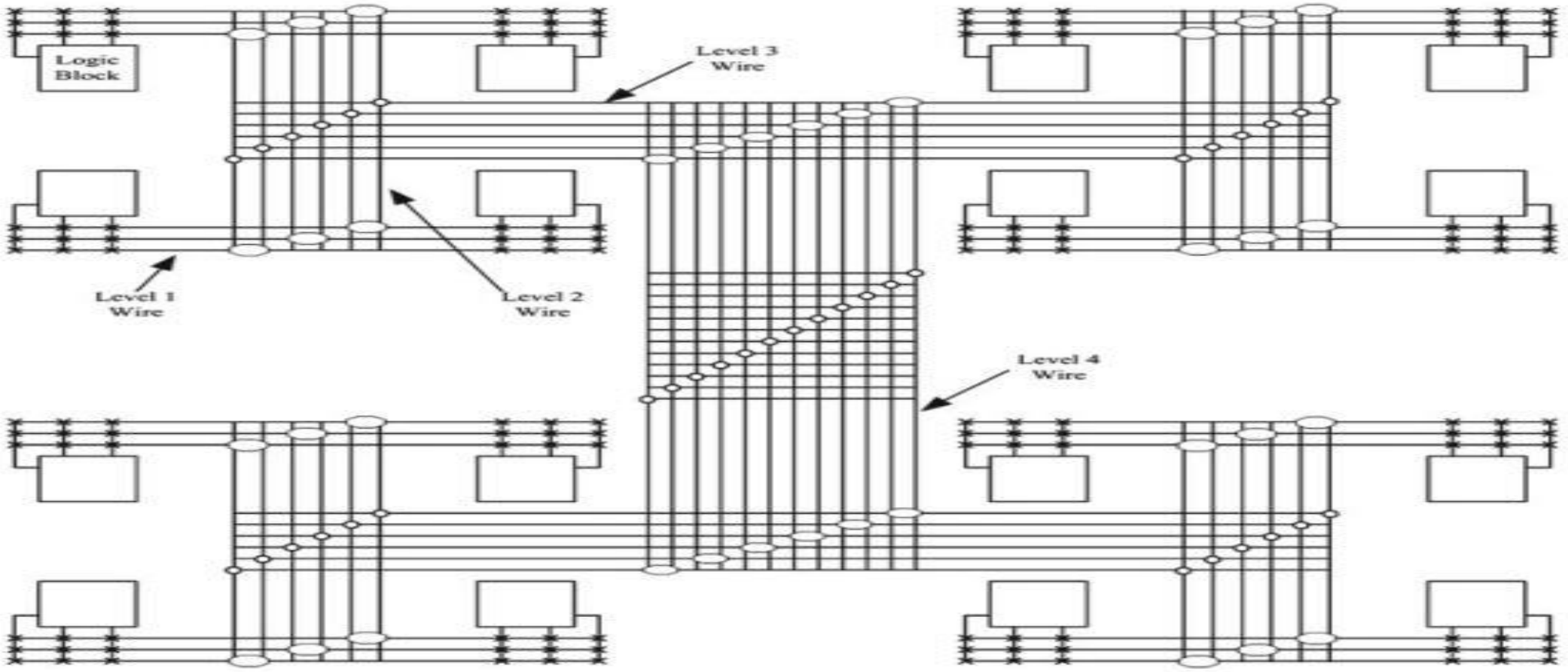
Routing

- If the computational functionality is provided by the Logic Blocks, then the programmable routing network is responsible for interconnection these logic blocks. The Routing Network provides interconnections between one logic block to other as well as between the logic block and the IO Block to completely implement a custom circuit.
- Basically, the routing network consists of connecting wires with programmable switches, which can be configured using any of the programming technologies. There are basically two types of routing architectures. They are:
- Island Style Routing (also known as Mesh Routing)
- Hierarchical Routing
- In island style routing architecture, the logic blocks are arranged in a two-dimensional array and are interconnected using a programmable routing network. This type of routing is widely used in commercial FPGAs



Many logic blocks are confined to a local set of connections and hierarchical routing architecture makes use of this feature by dividing the logic blocks into several groups or clusters. If the logic blocks are residing in the same cluster, then the hierarchical routing connects them in a low level of hierarchy.

If the logic blocks are residing in different clusters, then wiring is done over a higher level of hierarchy.



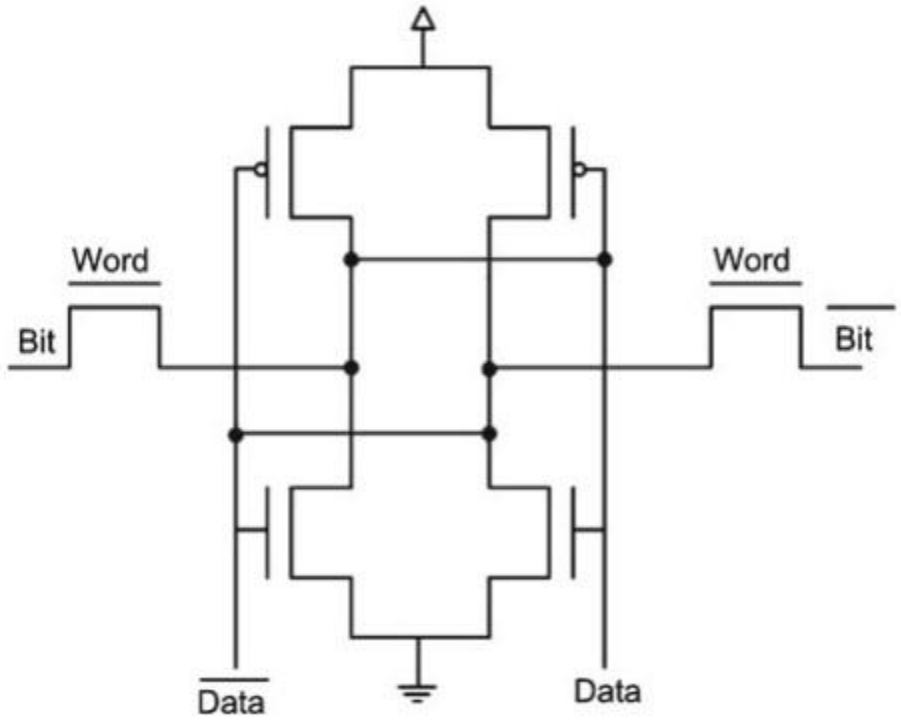
FPGA Programming Technologies

The following are three of the well-known programming technologies used in FPGAs.

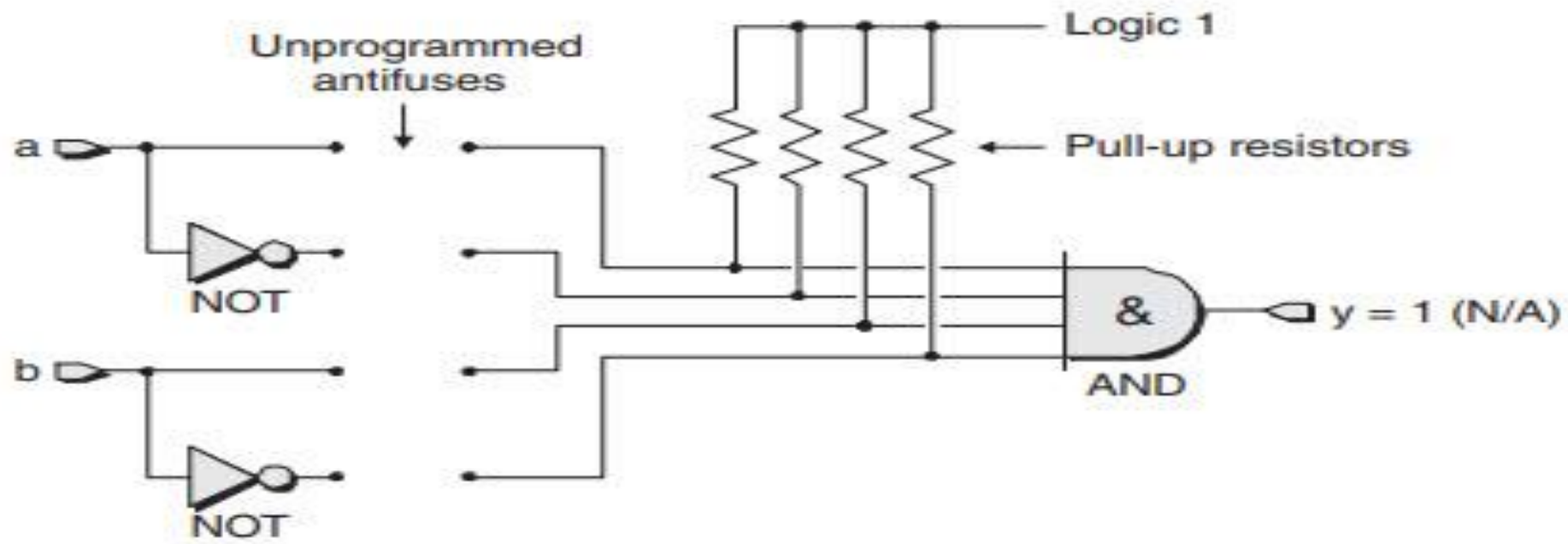
- SRAM
- EEPROM / Flash
- Anti-Fuse

SRAM:

A typical 6 transistor SRAM Cell to store 1 bit is shown in the following image.



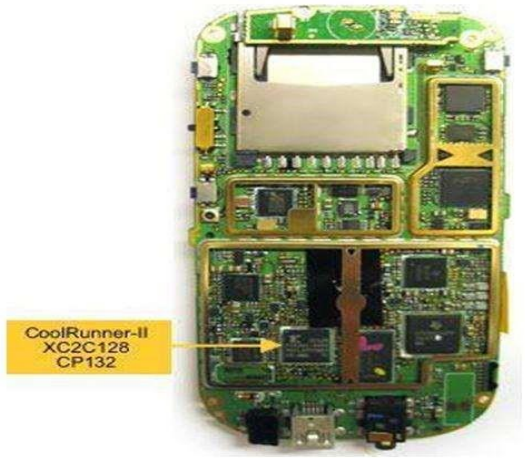
Anti-Fuse


Unprogrammed
antifuses
Logic 1
Pull-up resistors
a
NOT
b
NOT
&
AND
y = 1 (N/A)

Application of FPGA

CPLDs - Applications

➡ GSM Phone



- Keypad scanner
- Logic consolidation

➡ Printer

- Controller and interface conversion
- Interface expansion
- Simple logic

Refrence

- https://www.electronicshub.org/
- Dr.YasirAmerAbbas /2020/lecture
- **Introduction to CPLD and FPGA Design/By Bob Zeidman**