Introduction to Electronic Devices

- Whenever electronic devices has been developed, 3 things play a pivotal role: memory, microprocessors, and logic.
- **Memory** devices store random information such as the contents of a spreadsheet or database.
- **Microprocessors** execute software instructions to perform a wide variety of tasks such as running a word processing program or video game.
- **Logic** devices provide specific functions, including device-to-device interfacing, data communication, signal processing, data display, timing and control operations, and almost every other function a system must perform.

Programmable Logic Devices

- **What is a Programmable Logic?**
- It is a logic element whose function is *not restricted to a particular function*. It may be programmed at different points of the life cycle. A **programmable logic device** (**PLD**) is an electronic component used to build reconfigurable digital circuits.
- **What are Programmable Logic Devices (PLDs)?**
- PLDs are ICs with a large number of gates and flip flops that can be configured with basic software to perform a specific logic function or to perform the logic for a complex circuit. Unlike a logic gate, which has a fixed function, a PLD has an undefined function at the time of manufacture. Before the PLD can be used in a circuit it must be programmed, that is, reconfigured.

General Structure of PLD

- Inputs to the PLD are applied to a set of buffer/inverters. These devices have both the true value of the input as well as the complemented value of the input as its outputs.
- Outputs from these devices are the inputs to an array of and-gates. The AND array generates a set of p product terms.
- The product terms are inputs to an array of or-gates to realize a set of m sum-of-product expressions.

General Structure of PLD

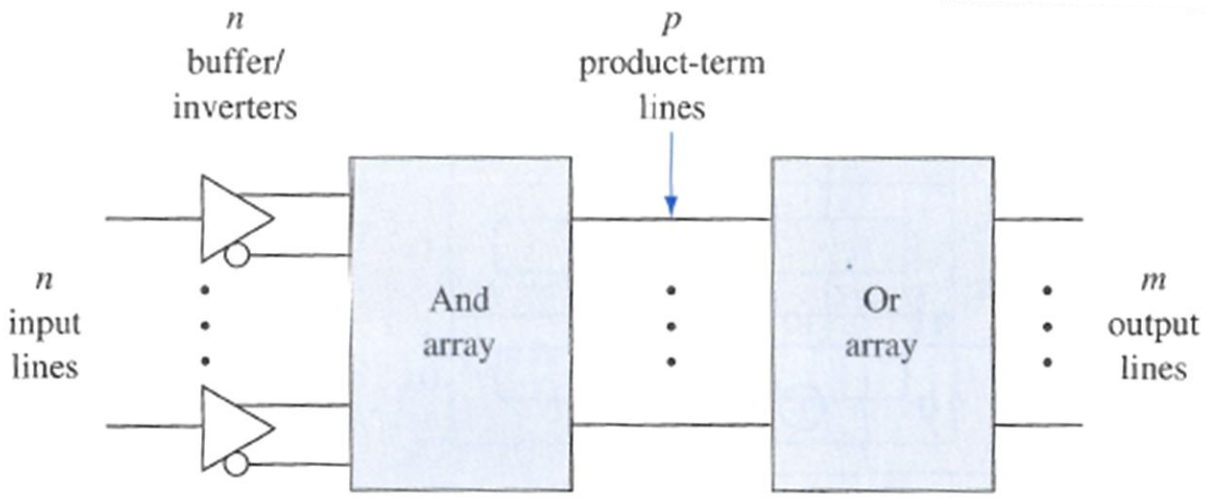


Figure 5.48 General structure of PLDs.

General Structure of PLD

- One or both of the gate arrays are programmable.
- The logic designer can specify the connections within an array.
- PLDs serve as general circuits for the realization of a set of Boolean functions.

Device	AND-array	OR-array
PROM	Fixed	Programmable
PLA	Programmable	Programmable
PAL	Programmable	Fixed

Programming a PLD

- In a programmable array, the connections to each gate can be modified.
- Simple approach is to have each of the gate inputs connected to a fuse.

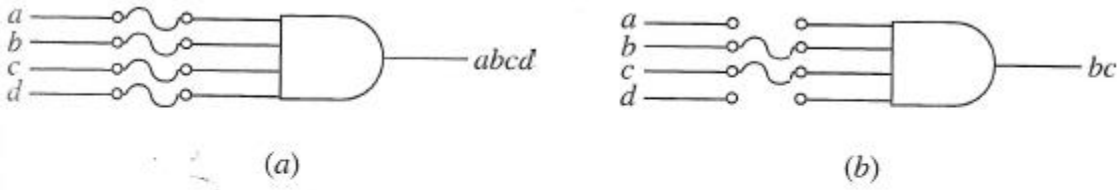


Figure 5.50 Programming by blowing fuses. (*a*) Before programming. (*b*) After programming.

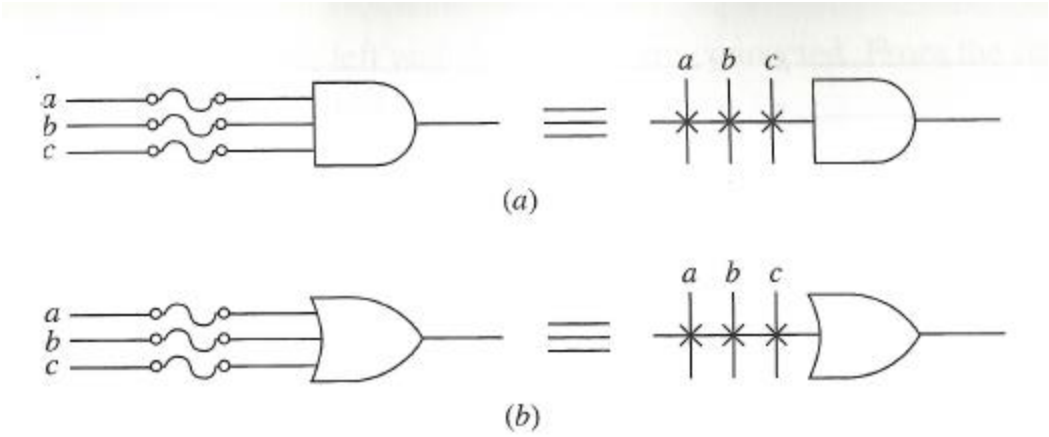
- Gate realizes the product term $abcd$.
- To generate the product term bc we remove the a, d connections by blowing the corresponding fuses.
- Thus, programming is a hardware procedure. Specialized equipment called programmers is needed to carry out the programming of a PLD.

Programming a PLD

- Erasable PLD—connections can be reset to their original conditions and then reprogrammed.
  - Can be achieved by exposing the PLD to ultraviolet light or using electrical signals
- PLDs programmed by a user are called field programmable.
- User can also specify the desired connections and supply the information to the manufacturer. Manufacturer prepares an overlay that is used to complete the connections as the last step in the fabrication process.
- Such PLDs are called mask programmable.

PLD Notation

- Simplified notation. Each gate has only a single input line.
- Inputs are indicated by lines at right angles to the single gate lines.
- A cross at the intersection denotes a fusible link is intact.



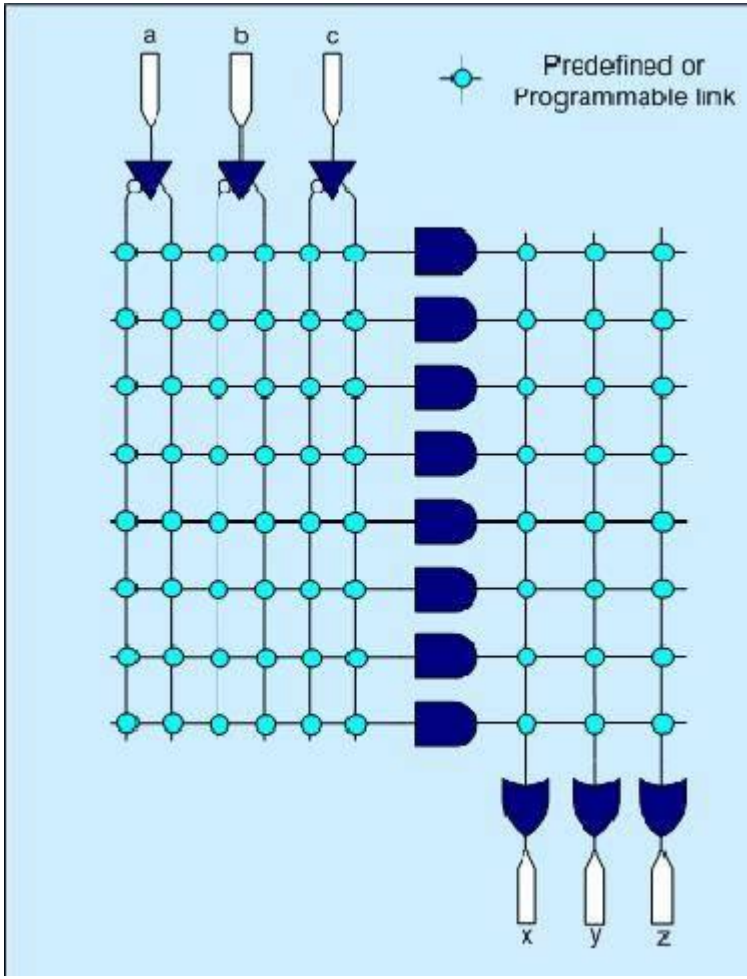
Types of Programmable Logic:

- Programmable logic devices are available in many different types. The current range of devices span from small devices capable of implementing only a handful of logic equations to huge FPGAs that can hold an entire processor core and peripherals. In addition to this incredible difference in size, there are many variations in the architecture.

- Programmable logic devices can be divided into three distinct architectural groups.
- Simple Programmable Logic Devices – SPLDs
- Complex Programmable Logic Devices – CPLDs
- Field Programmable Gate Arrays – FPGAs

1. Simple Programmable Devices:

- SPLDs are the **simplest, smallest and least-expensive type** of programmable logic device. These devices typically have logic gates laid out in arrays where the ***interconnection between these arrays is configurable by the user.***
- The term SPLD covers several types of device:
- Programmable Logic Array (PLA) – This device has ***both programmable AND and OR planes.***
- Field Programmable Logic Array (FPLA) – Same as PLA but can be ***erased and reprogrammed.***
- Programmable Array Logic (PAL) – This device has a programmable AND plane and a fixed OR plane.
- GAL – This device has the same logical properties as the PAL but can be ***erased and reprogrammed.*** The GAL is very useful in the prototyping stage of a design, when any bugs in the logic can be corrected by reprogramming.
- The ***connection link across two wires*** can either be predefined or programmable depending on the type of SPLD.

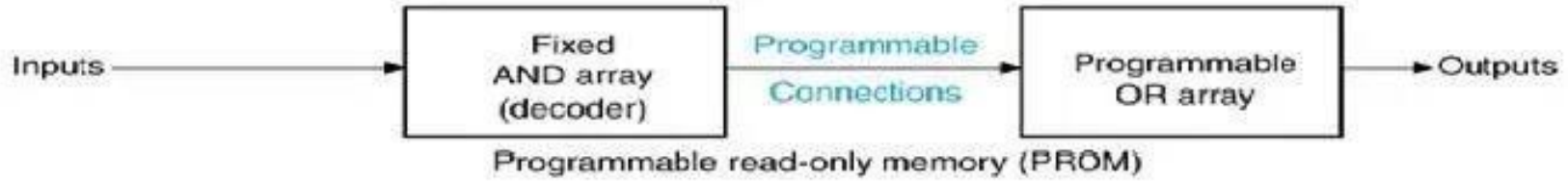


Programmable Logic Array:

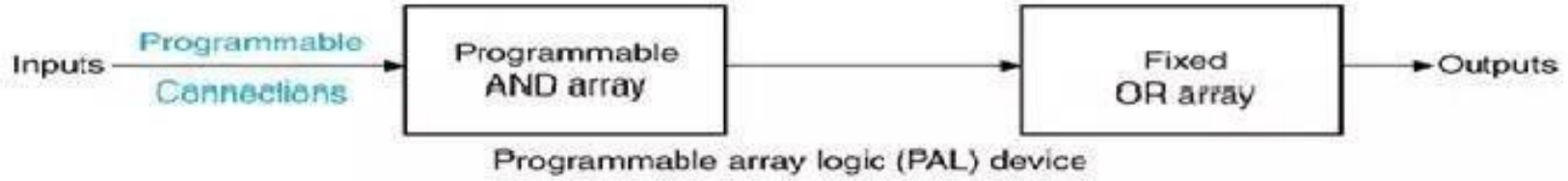
- These are the most configurable of the SPLDs. They consist of two levels of logic gates, an array of AND gates and an array of OR gates, both arrays of which are user programmable.
- The structure of a PLA allows any of its inputs (and the complement of its inputs) to be AND'ed together in the AND plane which will correspond to the product term of its inputs.
- Also, each output of the OR plane can be configured to give the logical sum of any outputs of the AND plane. This structure allows the implementation of logic functions in the sum-of-products form.
- PLAs are particularly useful for large designs that require many common product terms that can be used by several outputs.
- The downside of the PLA device is the price of manufacture and speed. This device has two levels of programmable links and signals take a relatively long time to pass through programmable links as opposed to pre-defined ones.

Programmable Array Logic:

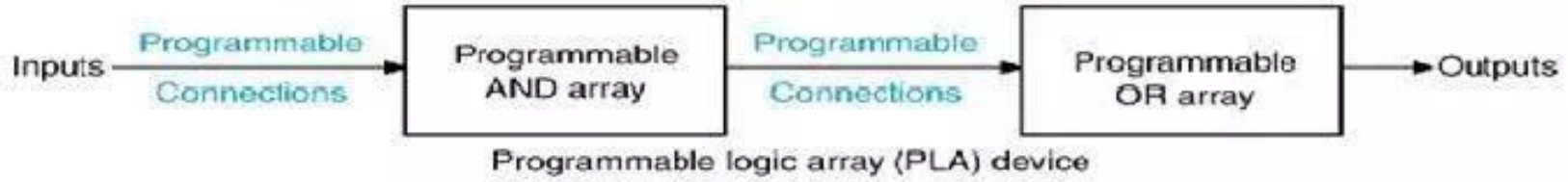
- The ***speed problems associated with the PLA*** were addressed with the development of the PAL. The advantage of a PAL is that they are ***faster due to having only a single programmable array.*** However, this limits the number of product terms that can be OR'ed together. Because of this, several variants of the device are produced with different numbers of inputs and outputs, and different sizes of OR gate arrays. Also, many PALs support ***registered or latched outputs*** therefore if necessary ***the output can be stored in the flip-flop until the next clock edge and sequential designs can be realized.*** The structure for such a device is shown in Fig below.



Programmable read-only memory (PROM)



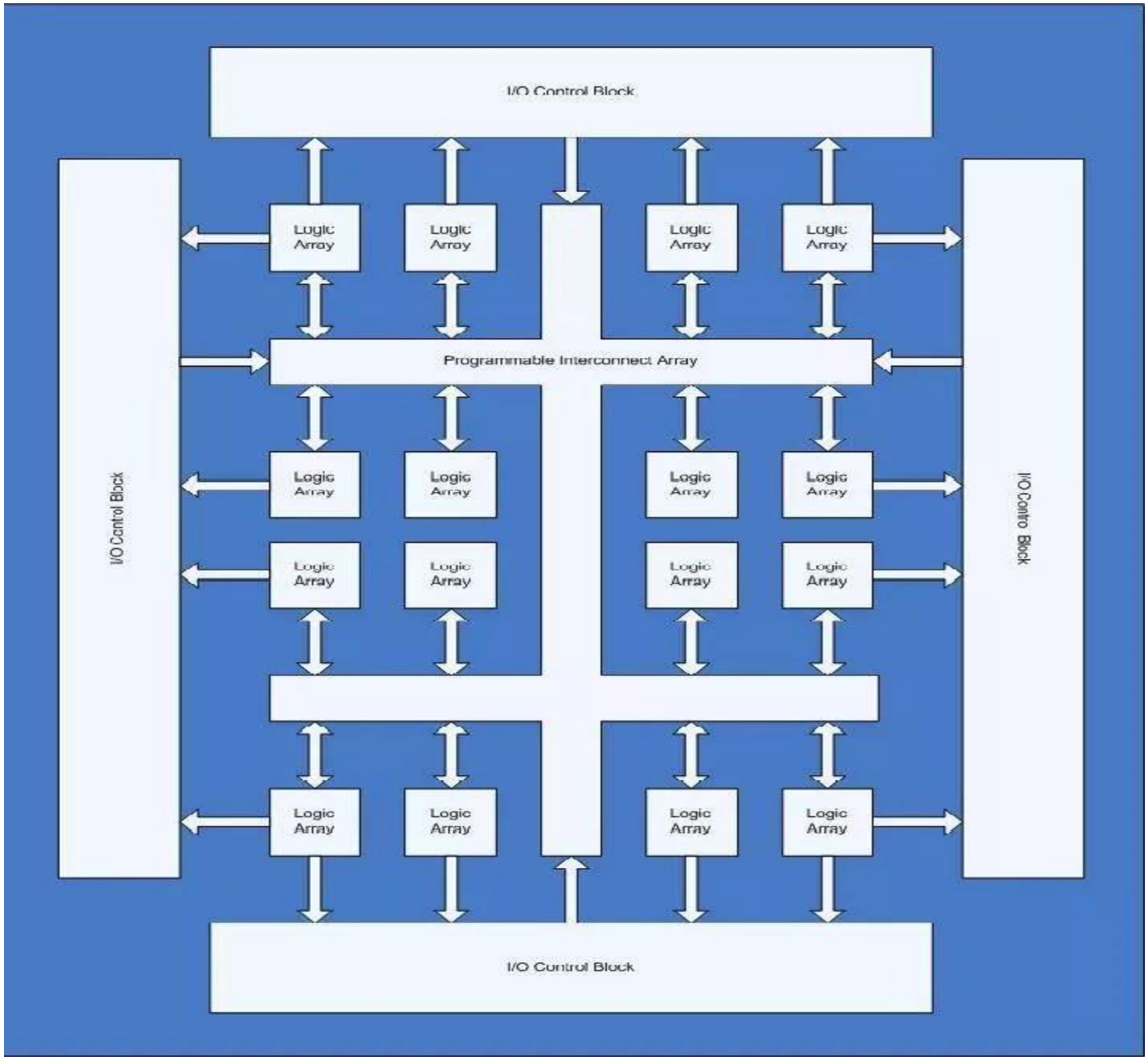
Programmable array logic (PAL) device



Programmable logic array (PLA) device

2. Complex Programmable Logic Device

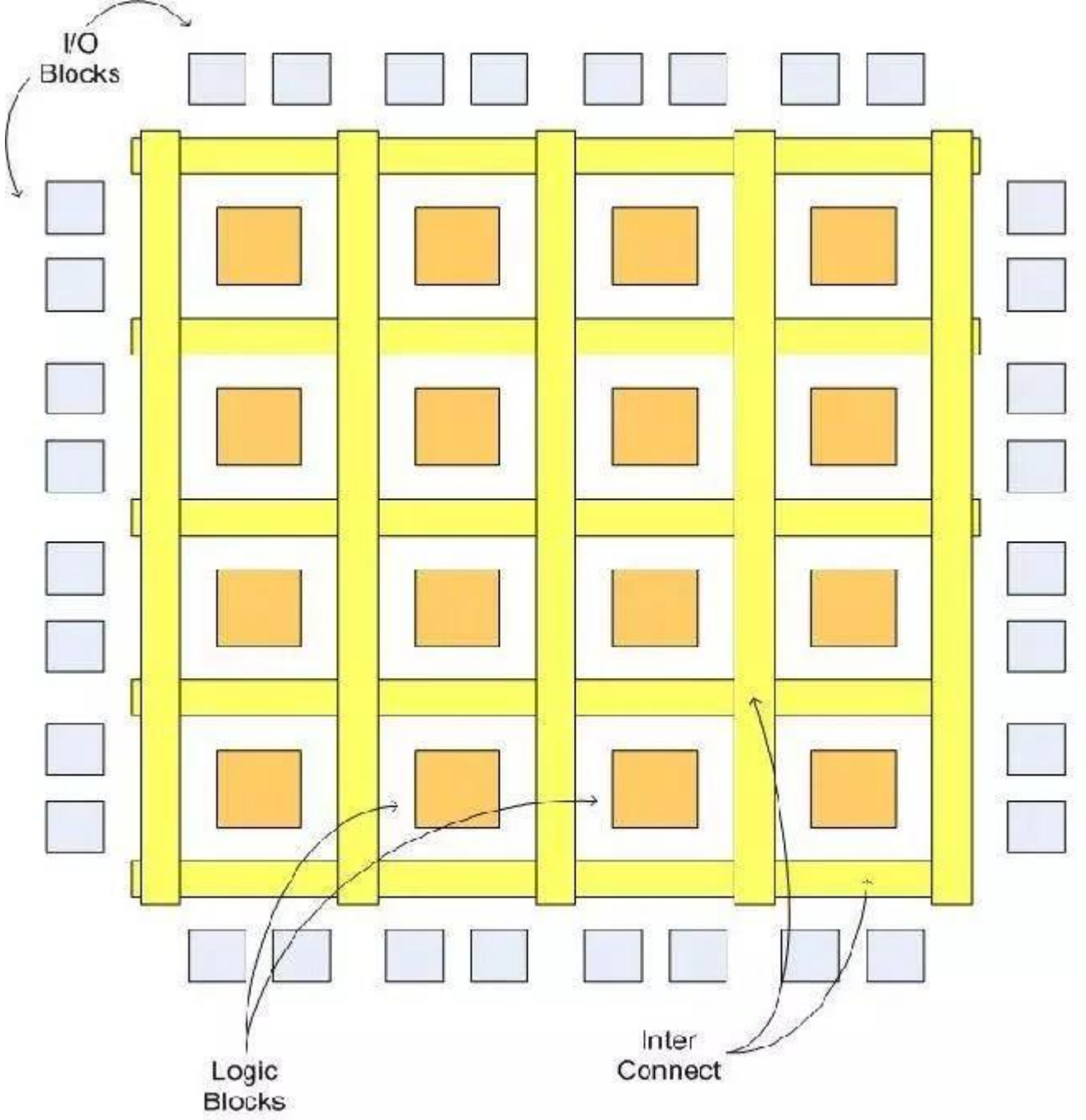
- This group covers the middle ground in terms of complexity and density between SPLDs and FPGAs.
- CPLDs can handle significantly larger designs than SPLDs, but provide less logic than field programmable gate arrays (FPGAs).
- CPLDs contain several logic blocks, each of which includes eight to 500 macrocells.
- For most practical purposes, CPLDs can be thought of as multiple SPLDs (plus some programmable interconnect) in a single chip.
- The typical structure of a CPLD is shown in Fig. Each of the 16 logic array blocks shown is the equivalent of one SPLD. However, in an actual CPLD, there may be more (or less) than 16 logic array blocks.
- Also, each of these logic array blocks are themselves comprised of macrocells and interconnect wiring, just like an ordinary SPLD.


I/O Control Block
Logic Array
Programmable Interconnect Array
I/O Control Block
I/O Control Block
I/O Control Block

- The larger size of a CPLD allows you to implement either more logic equations or a more complicated design. Most complex programmable logic devices contain macro cells with a sum-of-product combinatorial logic function and an optional flip-flop.
- Complex programmable logic devices feature predictable **timing characteristics that make them ideal for critical, high-performance control applications.**
- Typically, CPLDs have a **shorter and more predictable delay** than FPGAs and other programmable logic devices. Because they are ***inexpensive and require relatively small amounts of power***, CPLDs are often used in **cost-sensitive, battery-operated portable applications.**
- CPLDs are also used in simple applications such as **address decoding.**
- Because CPLDs can hold larger designs than SPLDs, their potential uses are more varied. They are still sometimes used for simple applications like ***address decoding, but more often contain high-performance control-logic or complex finite state machines***. At the high-end (in terms of numbers of gates), there is also a lot of overlap in potential applications with FPGAs. Traditionally, CPLDs have been chosen over FPGAs whenever **high-performance logic is required**. Because of its less flexible internal architecture, the delay through a CPLD (measured in nanoseconds) is more predictable and usually shorter.

3. Field Programmable Logic Arrays:

- Field Programmable Gate Arrays are a two-dimensional array of logic blocks and flip-flops with electrically programmable interconnections between logic blocks.
- The interconnections consist of electrically programmable switches which are why FPGA differs from Custom ICs, as Custom IC is programmed using integrated circuit fabrication technology to form metal interconnections between logic blocks.
- FPGAs can be used to implement just about any hardware design. One common use is to prototype a system that will eventually find its way into an ASIC. However, especially if the product has to be available as soon as possible then there is no reason why the FPGA can't be in the final product. Whether it does or not depends on the balance between development time and costs, and cost of the final device (number of required parts).
- Figure 6 illustrates a typical FPGA architecture. There are three key parts of its structure: logic blocks, interconnect, and I/O blocks. The I/O blocks form a ring around the outer edge of the part.

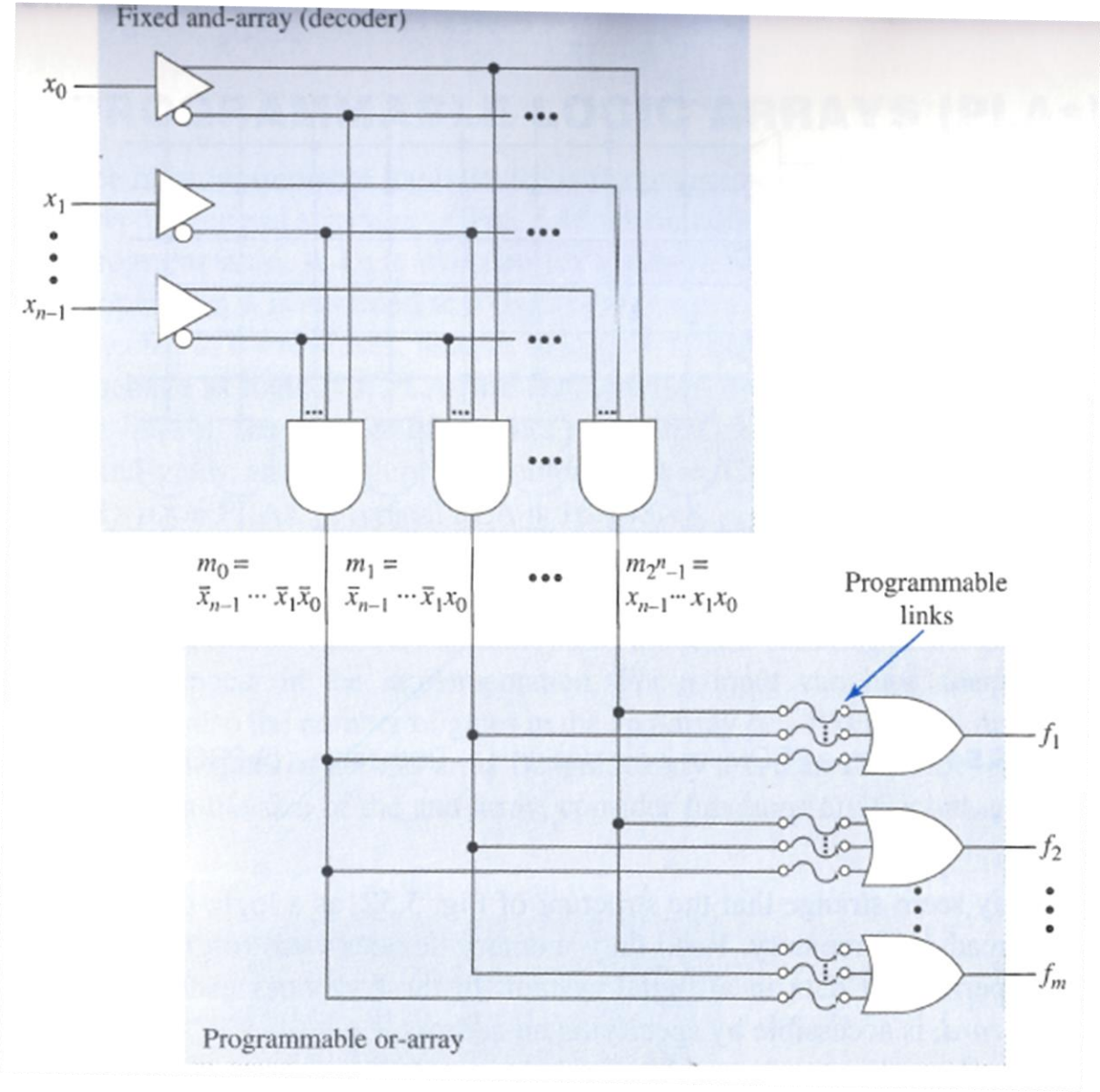

I/O
Blocks
Logic
Blocks
Inter
Connect

- Each of these provides individually selectable input, output, or bi-directional access to one of the general-purpose I/O pins on the exterior of the FPGA package. Inside the ring of I/O blocks lies a rectangular array of logic blocks. And connecting logic blocks to logic blocks and I/O blocks to logic blocks is the programmable interconnect wiring.
- The logic blocks within a FPGA can be as **small and simple as the macro cells in a PLD** (a so-called fine-grained architecture) or larger and more complex (coarse-grained). However, they are never as large as an entire PLD, as the logic blocks of a CPLD are. The logic blocks in a FPGA are generally nothing more than a couple of logic gates or a look-up table and a flip-flop.
- Because of all the extra flip-flops, the architecture of a FPGA is much **more flexible than that of a CPLD.** This makes FPGAs better in register-heavy and pipelined applications. They are also often used in place of a processor-plus-software solution, particularly where the processing of input data streams must be performed at a very fast pace. In addition, FPGAs are usually denser (more gates in a given area) and cost less than their CPLD cousins, so they are the de facto choice for larger logic designs.

Programmable Read-Only Memory (PROM)

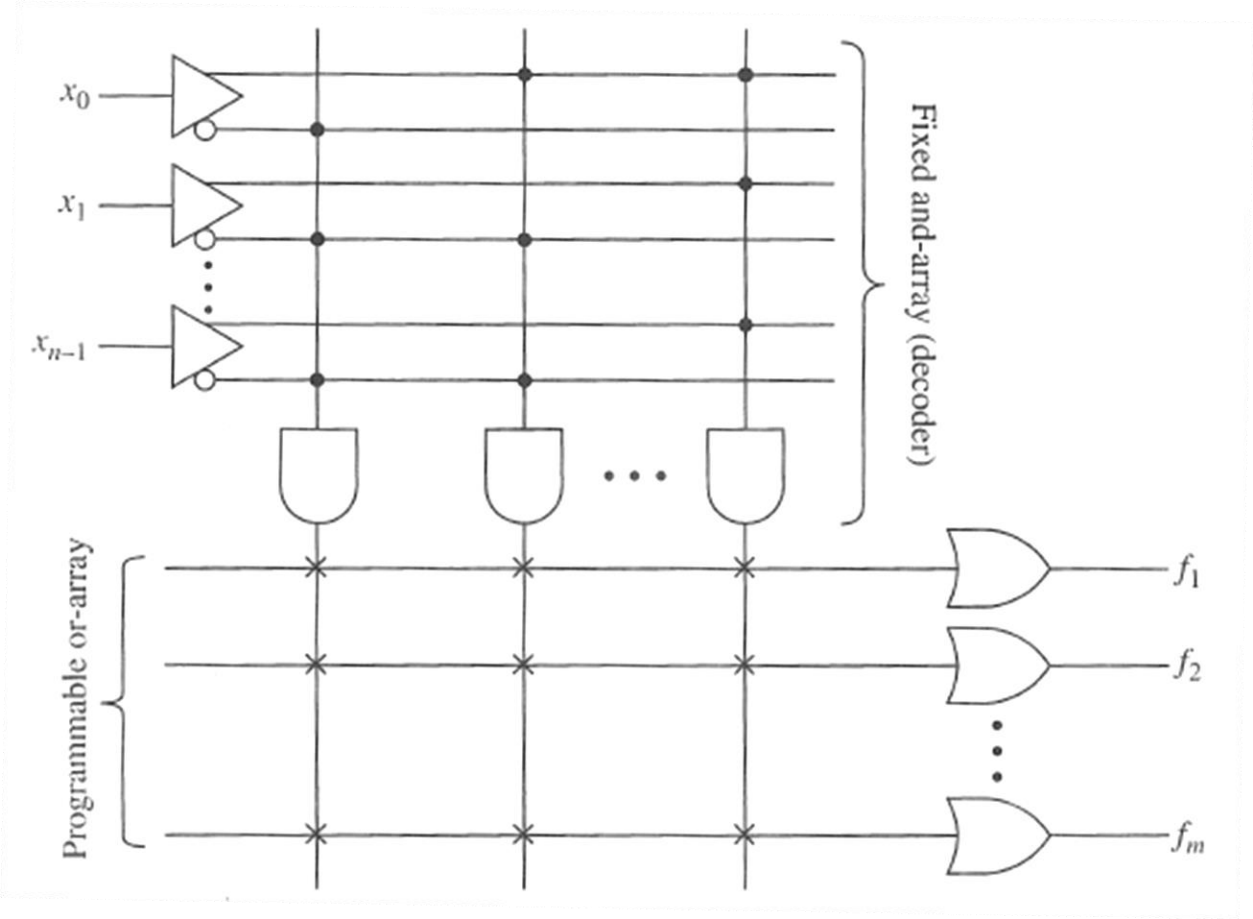
- AND-array with buffer/inverter is an n-to-2^n-line decoder.
- OR-array is a collection of programmable or-gates.
- Decoder is a min-term generator.
- n-variable minterms appear on the 2^n lines at the decoder output. These are also known as word lines.
- n input lines called address lines, m output lines called bit lines.
- $2^n \times m$ PROM.
- Realization of Boolean expressions same as realization using decoder discussed previously.

PROM Structure



Logic Diagram

PROM Structure



PLD Notation

Why is it called PROM?

- 3-bit input combination to the x_0, x_1, x_2 lines is regarded as an address of one of the word lines.
- As a consequence of selecting a given word line, a pattern of 0's and 1's, a word, as determined by the fusible connections to the selected word line appears at the bit lines of the device.
- This 0-1 pattern is considered the word stored at the address associated with the selected word line.
- E.g. the word stored at address $x_2x_1x_0 = 100$ is $f_1f_2 = 01$.
- "Read only": The fact that the connections associated with the fusible links normally cannot be altered once they are formed.

Programmable Logic Array

- PLAs are characterized by three numbers:
  - Number of input lines n
  - Number of product terms that can be generated p (the number of AND gates)
  - Number of output lines m
- $n \times p \times m$ PLAs
- Typical PLA is $16 \times 48 \times 8$.

Programmable Logic Array

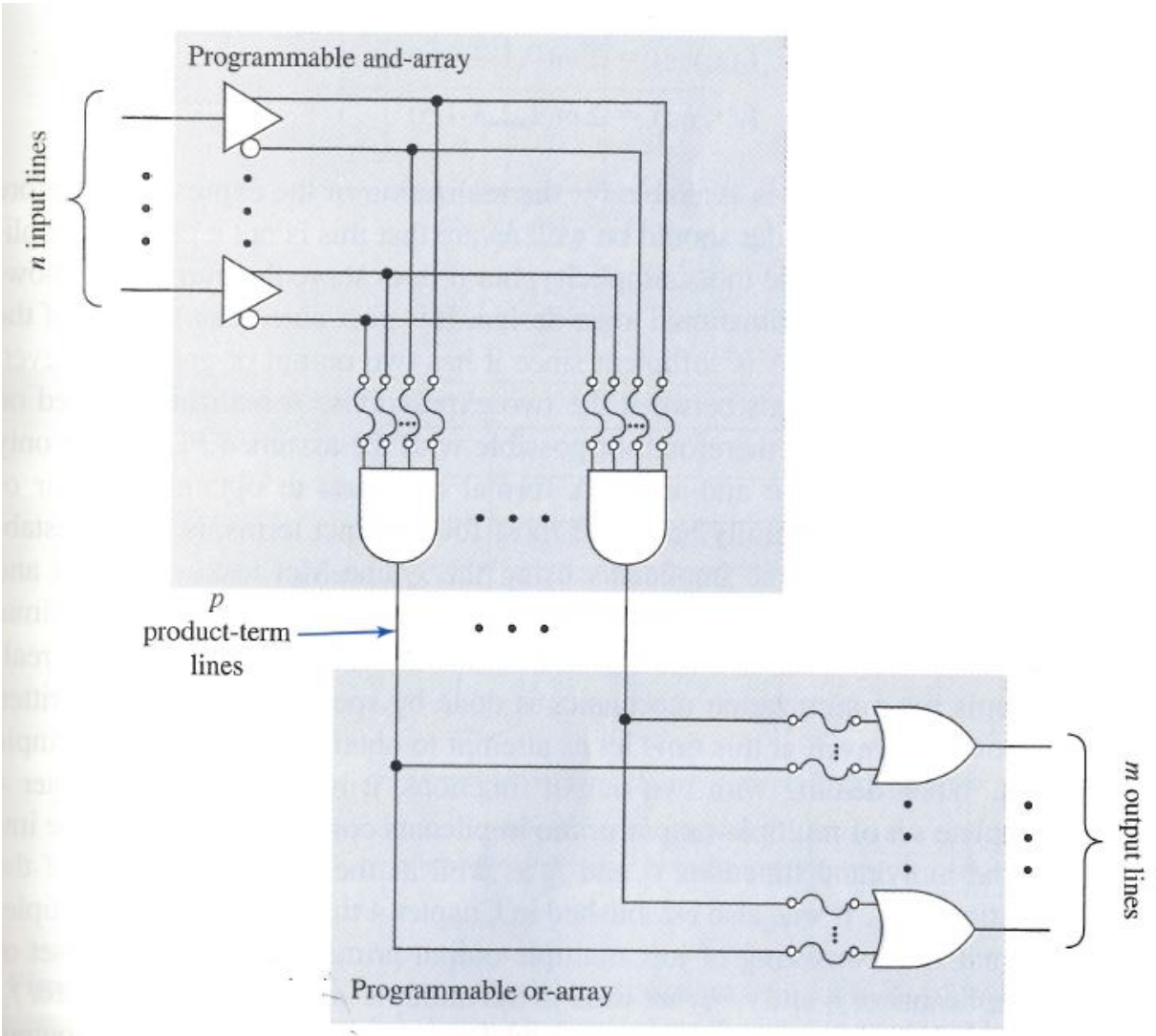


Figure 5.55 Logic diagram of an $n \times p \times m$ PLA.

Programmable Logic Array

- In many logic design situations not all the minterms are needed for a realization.
- For n input variables, 2^n minterms.
- This is the number of gates in the AND-array of a PROM.
- PLA's with 16 input lines
  - $2^{16} = 65{,}536$ minterms
  - In a $16 \times 48 \times 8$ PLA only 48 product terms.
- $2n$ inputs appear at each AND gate.
- For our examples, assume $3 \times 4 \times 2$ PLA.

PROM vs PLA

- PROM: realization of a set of Boolean functions is based on minterm canonical expressions.
  - No minimization necessary.
- PLA: the AND gates are capable of generating product terms that are not necessarily minterms.
  - Realization using PLA is based on sum-of-product expression that may not be canonical.
  - Logic designer is bounded by the number of product terms that are realizable by the AND-array.
  - Simplifications is necessary.

Programmable Array Logic (PAL) Devices

- OR-array is fixed by the manufacturer of the device.
  - PAL device is easier to program and less expensive than the PLA.
  - Less flexible.

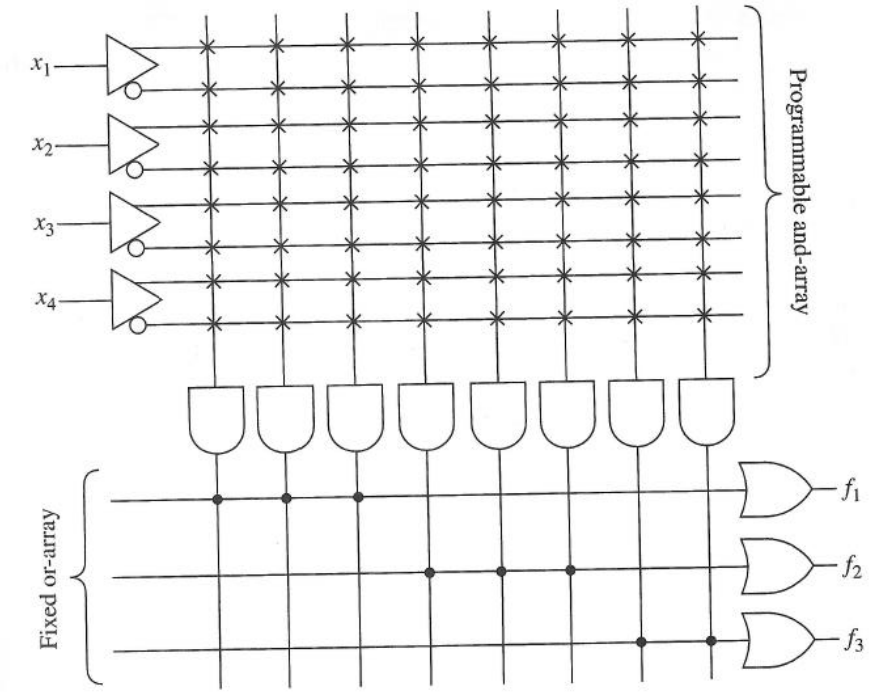


Figure 5.62 A simple four-input, three-output PAL device.

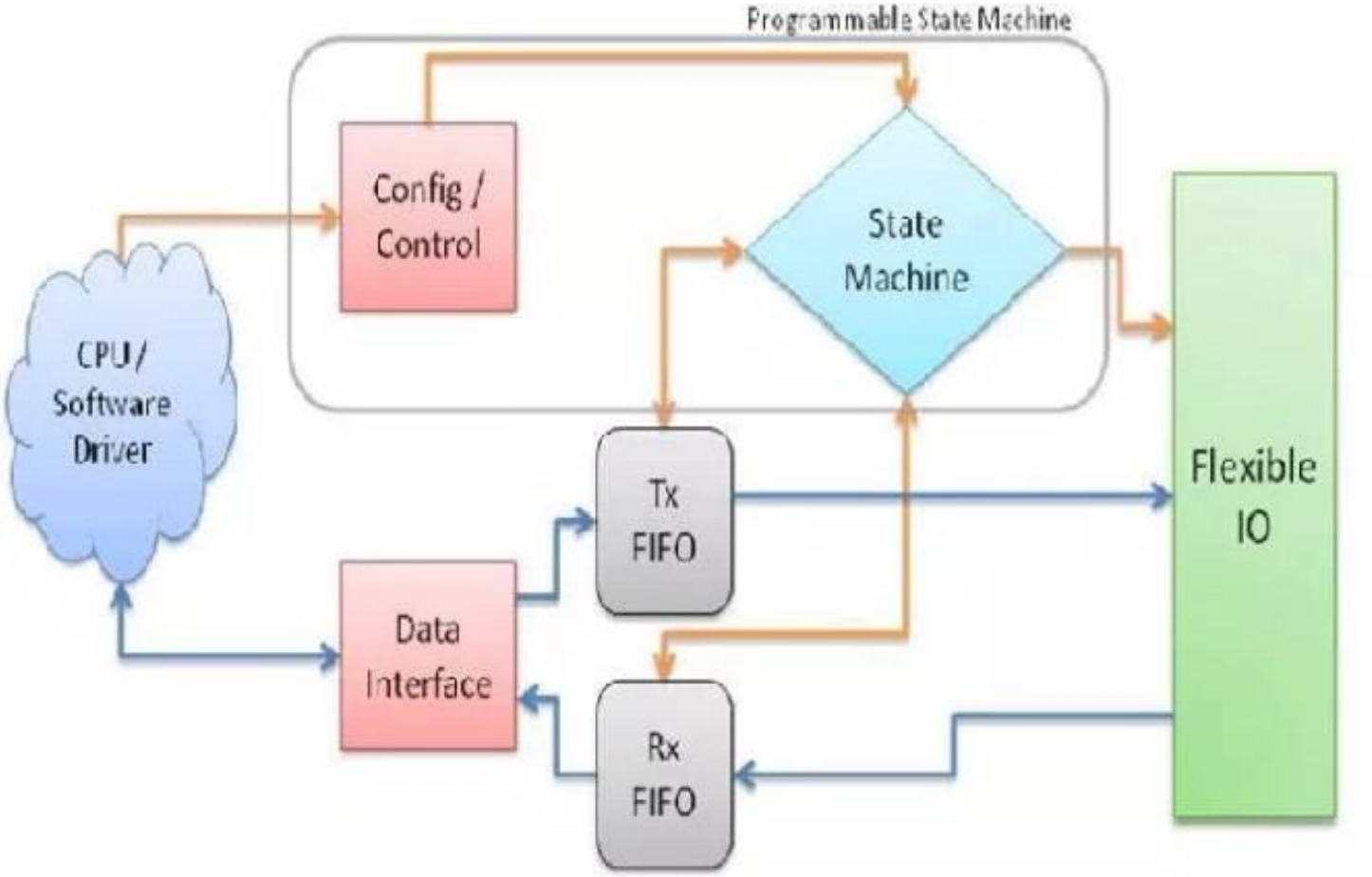
  - For our examples:
    - 4-input, 3-output PAL device
    - Three Boolean expressions can be realized in which two expressions can have at most 3 product terms and one expression can have at most 2 product terms.

Advantages of Programmable Logic Devices:

- Programmable logic devices offer a number of important advantages over fixed logic devices, including:
- **Design Flexibility:** PLDs offer customers much more flexibility during the design cycle because design iterations are simply a matter of changing the programming file, and the results of design changes can be seen immediately in working parts.
- **Improved Reliability:** Lower power plus fewer interconnections and packages translate into greatly improved system reliability.
- **Lower Power:** CMOS and fewer packages combine to reduce power consumption.
- **Reduced complexity:** Since PLDs consume lower power requirements less board space simpler testing procedures.
- **PLDs are field-programmable** i. e. can be programmed outside of the manufacturing environment
- **PLDs are erasable and reprogrammable** i.e allows updating a device or correction of errors and allows to reuse the device for a different design – the ultimate in reusability!

Applications of Programmable Logic Devices

- **Glue Logic:** Glue logic is the Simple logic circuits used to connect together more complex circuits which are not perfectly compatible. For example, an ASIC (application specific Ics) chip may contain large functions, such as a microprocessor, memory block or communications block, which are tied together via small amounts of glue logic. At the printed circuit board (PCB) level, glue logic may be implemented with simple “jelly bean” chips (“glue chips”) that contain a few gates all the way to programmable logic devices.
- This glue logic was straight forward for the IP04 as the Blackfin CPU has a rich set of interfaces with good DMA support. The Atheros SoC is not quite as feature rich. It does have an SPI bus for talking to the SPI flash. This only has one chip select line. Another example of glue logic is the address decoder which with older processors like the 6502 or Z80 had to be added externally to divide up the addressing space of the processor into RAM, ROM, and I/O. Newer versions of the same processors (such as the WDC565816 or Zilog eZ80) instead have internal address decoders so glueless interfacing to the most common external devices becomes possible


Programmable State Machine
Config / Control
State Machine
CPU / Software Driver
Tx FIFO
Flexible IO
Data Interface
Rx FIFO

State Machines:

- A state machine is a sequential logic function that controls other logic functions by sequencing control signals in response to input signals. The most common implementation of a state machine is a digital logic function in which the outputs are a function of both the present inputs and the previous state. The previous state is usually stored in a state register.
- For example, a state machine may wait in a particular state until a certain signal is asserted and then move to a new state. The machine travels from state to state in response to the input signals. Implementing control functions as state machines is a common application for PLDs (programmable logic devices). For an example, there are three main components to implementing PLIO [Programmable Low-Latency Input Output]: the software driver, the programmable state machine, and the flexible IO. In order for PLIO to address the highest number of bus protocols, part of the bus interface is controlled in software while the low-latency portions of the protocol are programmed into the state machine. The software driver handles the high latency parts of the bus protocol and the supplying and receiving of data and addresses to the interface, while the programmable state machine manages the control signals and interface timing. The flexible IO of PLIO allows for pin compatibility across the control, address and data signals.

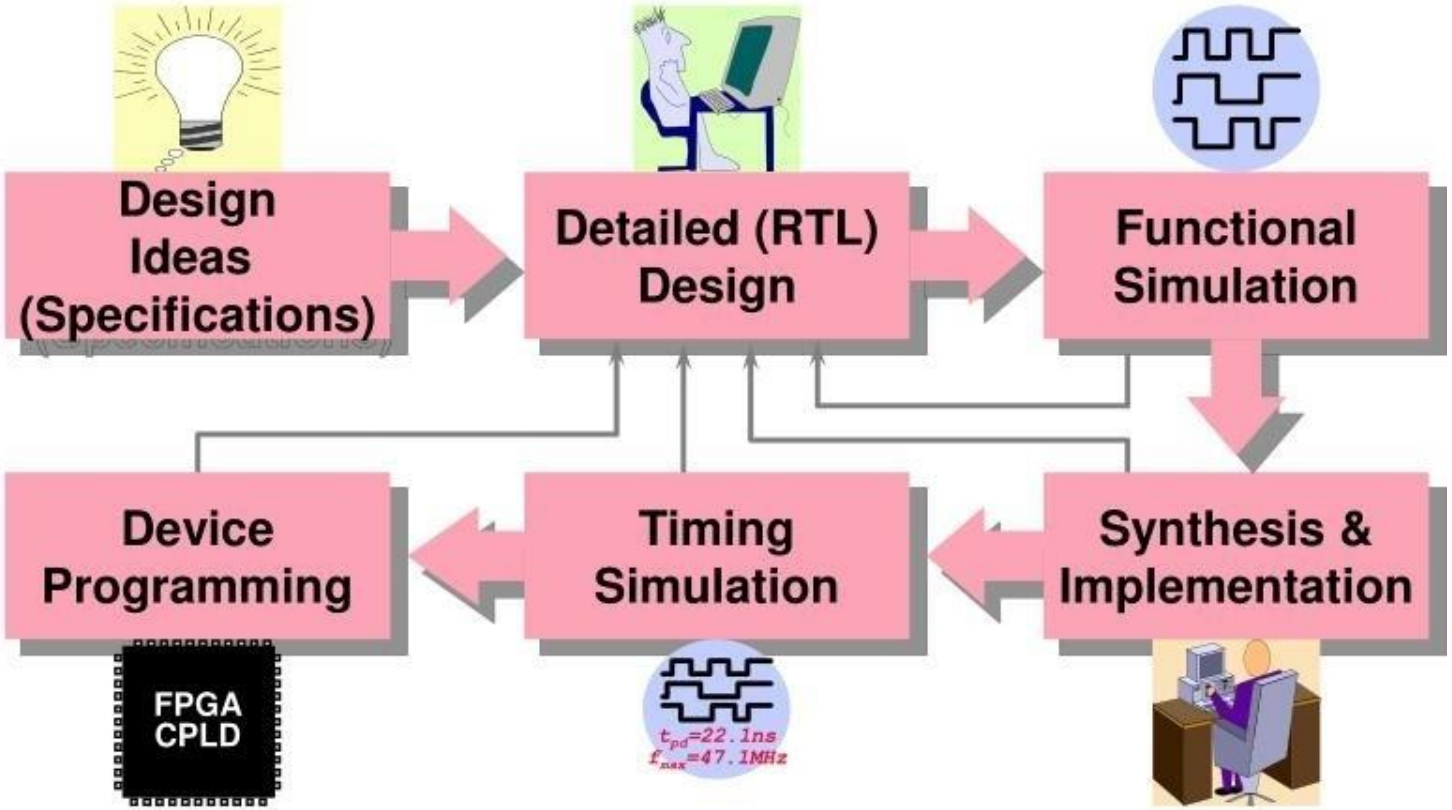
Counters:

- **Counters:** An instrument which, in its simplest form, provides an output that corresponds to the number of pulses applied to its input. Most digital counters operate in the binary number system since binary is easily implemented with electronic circuitry. Binary allows any integer (whole number) to be represented as a series of binary digits, or bits, where each bit is either a 0 or 1 (off or on, low or high, and so forth).
- For an example, a digital watch contains numerous counters in its large-scale integration (LSI) chip, usually implemented with complementary metal oxide semiconductor (CMOS) technology. For another example, digital computers may contain counters in the form of programmable interval timers that count an integral number of clock pulses of the known period, and then generate an output at the end of the count to signal that the time period has expired.

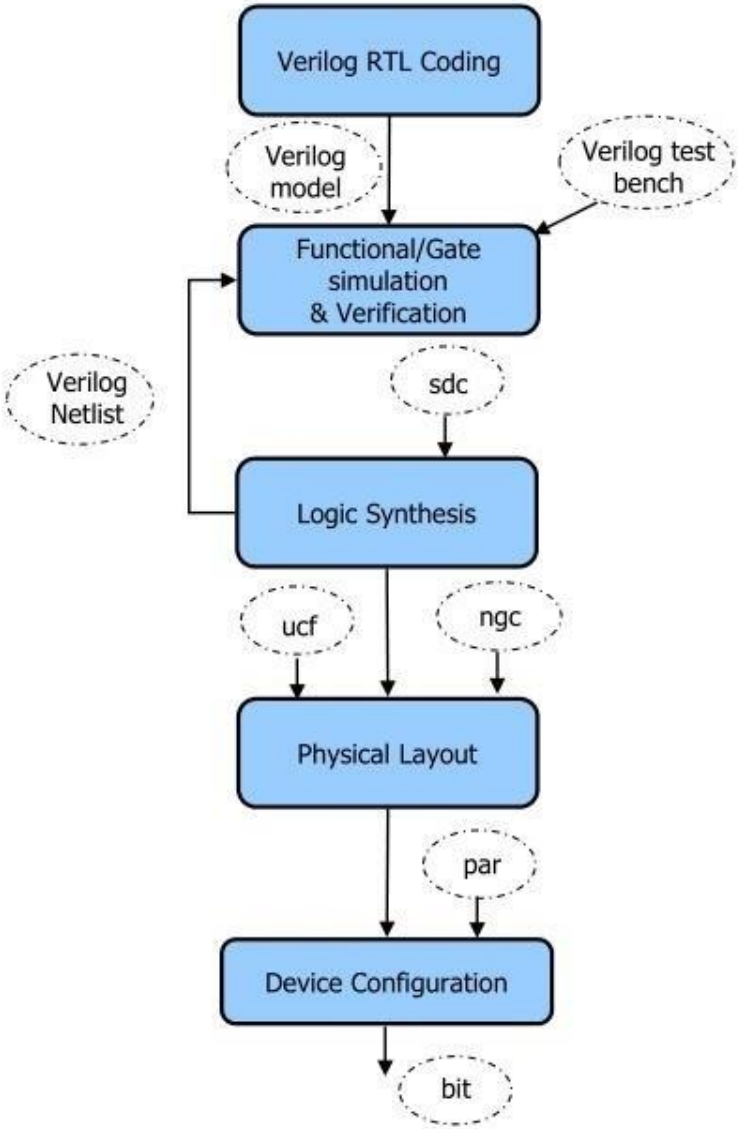
Other Uses:

- Synchronization
- Decoders
- Bus Interfaces
- Parallel-to-Serial
- Serial-to-Parallel

FPGA Design Flow



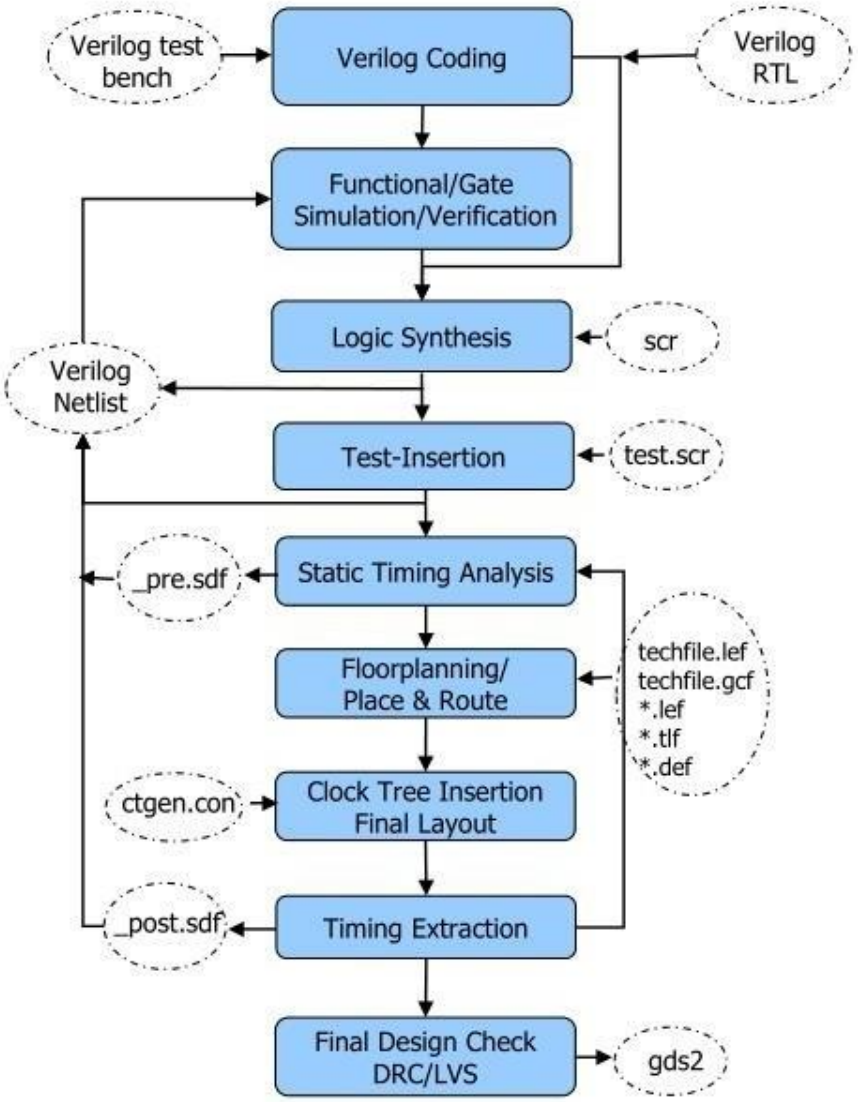
FPGA Design Flow



Design Stage	Tools
Verilog Design	Text Editor Emacs, Nedit, Vi
Verification	Modelsim SE Leda
Synthesis	Xilinx ISE - XST Synplify Pro
Pyhsical Design & Implementation	Xilinx ISE Xilinx Impact

Digital Design Flow

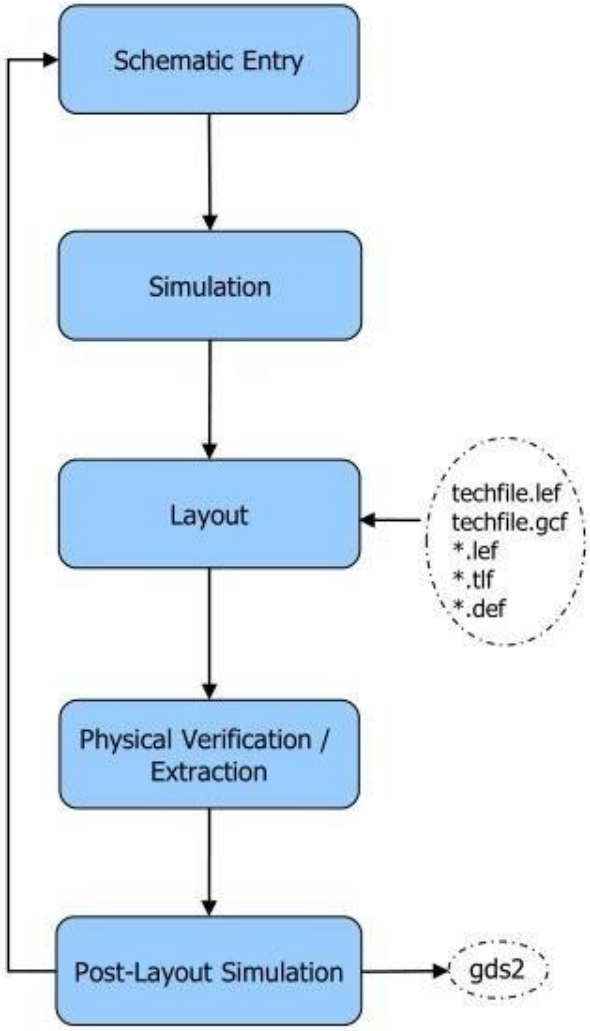
csrc
circuits and systems research centre



Design Stage	Tools
Verilog Design	Text Editor Emacs, Nedit, Vi
Verification	Mentor - Modelsim SE Synopsys - Leda
Synthesis	Synposys - Design Compiler
Test Insertion	Synopsys - TetraMax Mentor - Fastscan
Static Timing Anal.	Synopsys - Primetime
Place & Route	Cadence - Sensemble/ SOC Encounter Synopsys - Apolllo
Clock Tree Insertion	Cadence - CTgen
Timing Extraction	Synopsys - StarRXT Cadence - Pearl
DRC/ANT Checking	Cadence - Assura, Dracula Mentor – Callibre
LVS	Cadence - Assura, Dracula Mentor – Callibre

Analogue Design Flow

csrc
circuits and systems research centre



Design Stage	Tools
Schematic Entry	Composer
Simulation	Spectre
Layout	Virtuosso
Pyhsical Verification/ Extraction	Assura Calibre
Post-Layout Simulation	Spectre

Mixed Signal Design Flow

csrc
circuits and systems research centre