**OLATUNJI TEMITOPE OLATUNDE**

**15/ENG02/043**

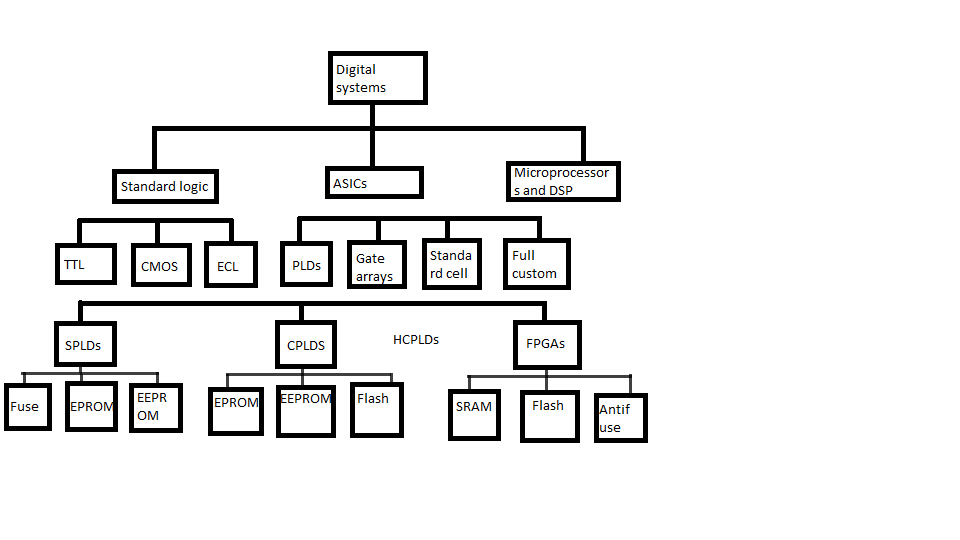
**COE 506 ASSIGNMENT**

**VHDL**

Digital Systems Family Tree

A digital system family tree showing most of the hardware choices that are currently available can be used in sorting out the many categories of digital devices.

The major digital system categories consist of standard logic, application-specific integrated circuits (ASICS) and microprocessor/digital signal processing (DSP) devices.

*A digital family tree*

**The first category of standard logic devices** refers to the basic functional digital components (gates, flip-flops, decoders, multiplexers, registers, counters, etc.) that are available as SSI and MSI chips. These devices have been used for many years to design complex digital systems.

**The microprocessor/digital signal processing (DSP) category** is a much different approach to digital system design. These devices actually contain the various types of functional blocks that have been discussed throughout this text. With microcomputer/DSP systems, devices can be controlled electronically, and data can be manipulated by executing a program of instructions that has been written for the application. Using a hardware solution for your digital system design is always faster than a software solution.

**The third major digital system category** is called application-specific integrated circuits (ASICs). This broad category represents the modern hardware design solution for digital systems. Four subcategories of ASIC devices are available to create digital systems: programmable logic devices, gate arrays, standard-cell, and full-custom.

**Programmable logic devices (PLDs)**, also known as field-programmable logic devices (FPLDs), can be custom-configured to create any desired digital circuit, from simple logic gates to complex digital systems. This ASIC choice for the designer is very different from the other three subcategories. This option can be very expensive and usually requires that your company purchase a large volume of parts to be cost effective.

**Gate arrays** are ULSI circuits that offer hundreds of thousands of gates. The desired logic functions are created by the interconnections of these prefabricated gates. Individually, these devices are less expensive than PLDs of comparable gate count, but the custom programming process by the chip manufacturer is very expensive and requires a great deal of lead time.

**Standard-cell ASICs** use predefined logic function building blocks called cells to create the desired digital system. The IC layout of each cell has been designed previously, and a library of available cells is stored in a computer database. The needed cells are laid out for the desired applications, and the interconnections between the cells are determined. Design costs for standard-cell ASICs are even higher than for MPGAs because all IC fabrication masks that define the components and interconnections must be custom designed. Standard cells do have an advantage over gate arrays. The cell-based functions have been designed to be much smaller than equivalent functions in gate arrays, which allows for generally high-speed operation and cheaper manufacturing costs.

**Full-custom ASICs** are considered the ultimate ASIC choice. All components and the interconnections between them are custom-designed by the IC designer. This requires a significant amount of time and expense, but it can result in ICs that can operate at the highest possible speed and require the smallest die (individual IC chop) area.

**PLDs** can be described as being one of three different types: simple programmable logic devices (SPLDs), complex programmable logic devices (CPLDs), or field programmable gate arrays (FPGAs).

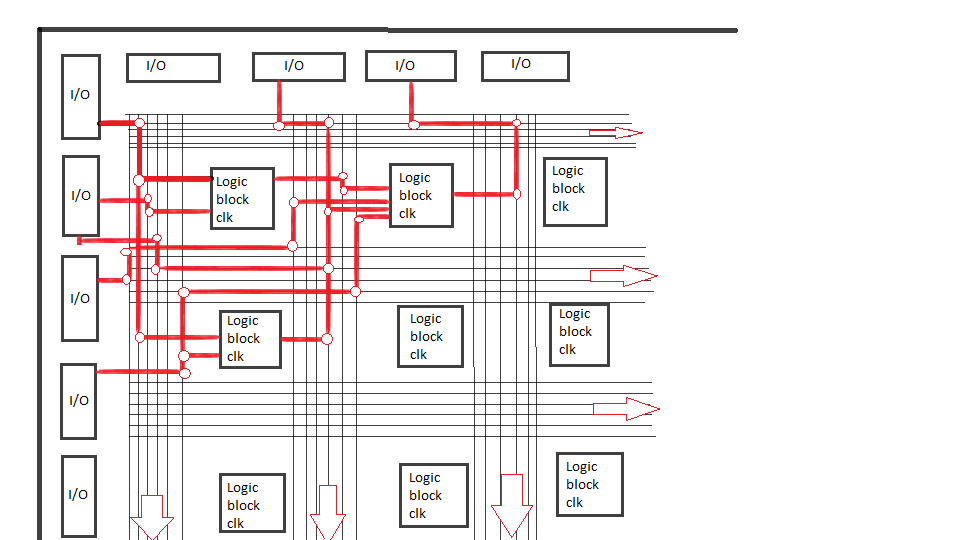
The distinction between CPLDs and FPGAs is often a little fuzzy, with the manufacturers constantly designing new, improved architectures and frequently muddying the waters for marketing purposes. Together, they are often referred to as high-capacity programmable logic devices (HCPLDs). The programming technologies for PLD devices are actually based on the various types of semiconductor memory.

The amount of logic resources available is the major distinguishing feature between SPLDs and HCPLDs. Internal programmable signal interconnect resources are much more limited with SPLDs. SPLDs are generally much less complicated and much cheaper than HCPLDs. Many small digital applications need only the resources of an SPLD. On the other hand, HCPLDs are capable of providing the circuit resources for complete complex digital systems, are larger, more sophisticated HCPLD devices are designed every year.

**The SPLD** classification includes the earliest PLD devices. The amount of logic resources contained in the early PLDs may be relatively small by today's standards, but they represented a significant technological step in their ability to create easily a custom IC that can replace several standard logic devices The first PLD type to gain the interest of circuit designers was programmed by literally burying open selected fuses in the programming matrix. The fuses that were left intact in these one-time programmable (OTP) devices provided the electrical connections for the AND/OR circuits to produce the desired functions. This logic device was based on the fuse links in PROM memory technology and was most commonly referred to as a programmable logic array (PLA). PLDs didn't really gain widespread acceptance with digital designers until the late 1970s, when a device called a programmable array logic (PAL) was introduced. With the development of the ultraviolet erasable PROM came to the EPROM-based PLDs in the mid 1980s, followed soon by PLDs using electrically erasable (EEPROM) technology.

**CPLDS** are devices that typically combine an array of PAL-type devices on the same chip. The logic blocks themselves are programmable AND/fixed-OR logic circuits with fewer product terms available than most PAL devices. Each logic block called a macrocell can typically handle many input variables. The flip-flop used to implement the register in the macrocell can often be configured for D, JK, T, or SR operation. Input and output pins for some CPLD architectures are associated with a specific macrocell, and typically additional macrocells are buried (that is, not connected to a pin). The programming technology used in CPLD devices are all non-volatile and include EPROM, EEPROM, and flash, with EEPROM being the most common. ALl three are erasable and reprogrammable.

**FPGAs** have a few fundamental characteristics that are shared. They normally consist of many relatively small and independent programmable logic modules that can be interconnected to create larger functions. Each module can usually handle only up to four or five variables. Most FPGA logic modules utilize a look-up table (LUT) approach to create the desired logic functions. A look-up table basically functions like a truth table in which the output can be programmed to create the desired combinational function by storing the appropriate 0 or 1 for each input combination. The logic modules are not associated with any I/O pin. Instead, each I/O pin is connected to a programmable input/output block that, in turn is connected to the logic modules with selected routing lines. The I/O blocks can be configured to provide input, output, or bidirectional capability, and built-in registers can be used to latch incoming or outgoing data. All of the logic blocks and input/output blocks can be programmed to implement almost any logic circuit. The interconnections are accomplished via lines that run through the rows and columns in the channels between the logic blocks. The programming technologies used in FPGA devices include SRAM, flash, and antifuse, with SRAM being the most common. SRAM-based devices are volatile and therefore require the FPGA to be reconfigured (programmed) when it is powered up.

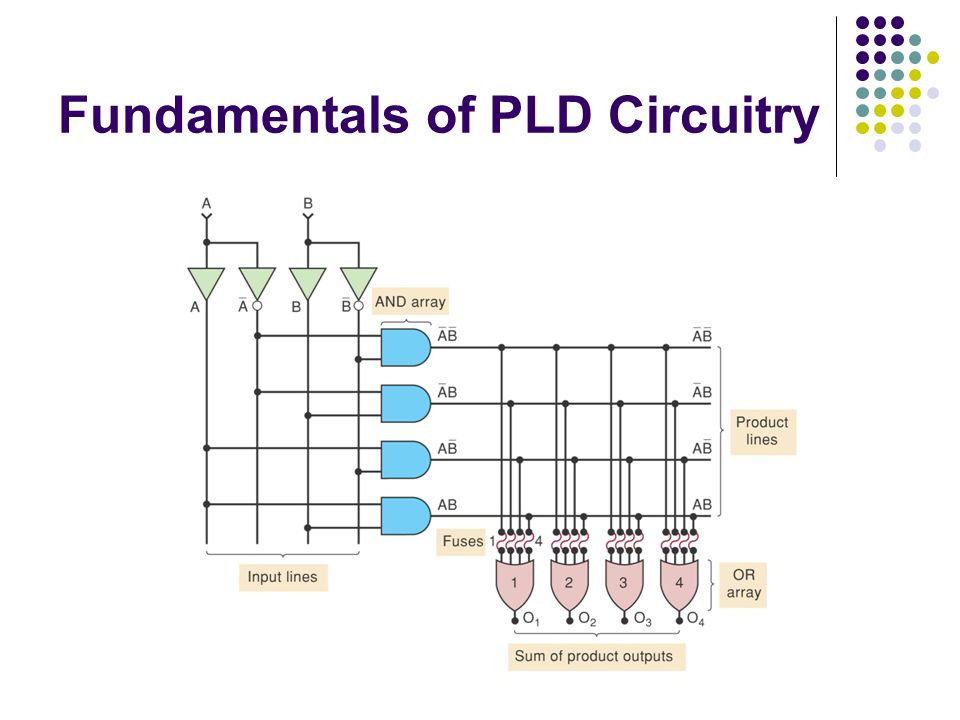


*Fpga architecture*

The differences in architecture between CPLDs and FPGAs, among different HCPLD manufacturers, and among different families of devices from a single manufacturer can affect the efficiency of design implementation for a particular application.

**A simple PLD device.**

Each of the four OR gates can produce an output that is a function of two input variables, A and B. Each output function is programmed with the fuses located between and AND gates and each of the OR gates.



*A programmable logic device*

Each of the inputs A and B feed both a non-invering buffer and an inverting buffer to produce the true and inverted forms of each variable.

Each of the product lines is connected to one of the four inputs of each OR gate thtough a fusible link. The proof is below

O1 = A'B' + A'B + AB' + AB

= A'(B'+B) + A (B' +B)

= A' + A = 1

Each of the four outputs O1, O2, O3, and O4 can be programmed to be any function of A and B by selectively blowing the appropriate fuses. For example, if fuses 1 and 4 are blown, we can see the output O1 below

O1 = 0 + A'B + AB' + 0 = A'B + AB'

**PLD Symbology**

The figure below shows the same PLD circuit seen above using simplified symbols.

The inputs to each of the OR gates are also designated by a single line representing all four inputs. An X represents an intact fuse connecting a product line to one input of the OR gate. The absence of an X (or a dot) at any intersection represents a blown fuse.

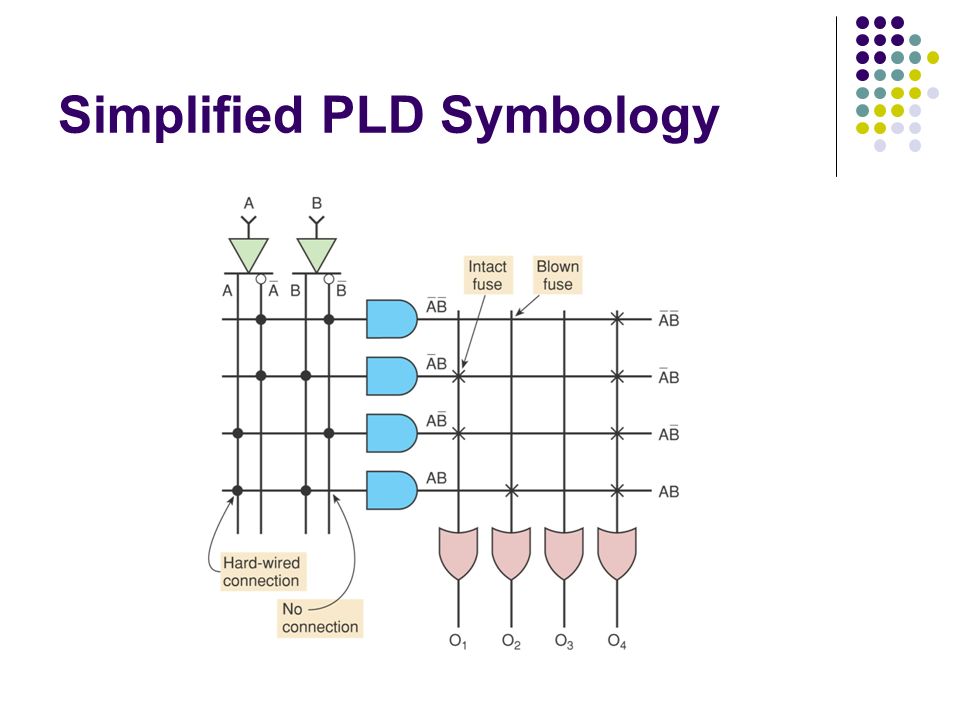
In the example, the outputs are programmed as

O1 = A'B + AB

O2 = AB

O3 = 0

O4 = 1



*simplified PLD symbology*

**PLD Architectures**

**PROMS**

The architecture of the programmable circuits in the previous section involves programming the connections to the OR gate. The AND gates are used to decode all the possible combinations of the input variables. For any given input combination, the corresponding row is activated. If the OR input is connected to that row, a HIGH appears at the OR output. If the input is not connected, a LOW appears at the OR input.

**Programmable Array Logic (PAL)**

The PAL has an AND and OR structure similar to a PROM but in the PAL, inputs to the AND gates are programmable, whereas the inputs to the OR gates are hard-wired. This means that every AND gate can be programmed to generate any desired product of the four input variables and their complements. Each OR gate is hard-wired to only four AND outputs. If a function requires having more than four product terms, it cannot be implemented wit this PAL; one having more OR inputs would have to be used. If fewer than four product terms are required, the unneeded ones can be made 0.

**Field Programmable Logic Array (FPLA)**

The FPLA was developed in the mid 1970s as the first non-memory programmable logic device. FPLAs are used mostly in state-machine design where a large number of product terms are needed in each SOP expression.

**The GAL 16V8 (Generic Array Logic)**

The GAL chip uses an EEPROM array to control the programmable connections to the AND matrix, allowing them to be erased and reprogrammed at least 100 times. GAL 16V8 contains optional flip-flops for register and counter applications, tristate buffers for the outputs, and control multiplexers used to select the various modes of operation. It can be used as a generic, pin-compatible replacement for most PAL devices.

Theis device has 8 dedicated input pins, two special function inputs, and 8 pins that can be used as inputs or outputs. The major components of the GAL devices are the input term matrix; the AND gates, which generate the products of input terms; and the output logic macrocells (OLMCs).

The flexibility of the GAL 16V8 lies in its programmable output logic macrocell. Eight different products are applied as inputs to each of the eight output logic macrocells. Within each OLMC, the products are ORed together to generate the sum of products (SOP).