**Chapter 1**

**INTRODUCTION**

The present day requirements of electronic circuits or processor are speed and flexibility. Speed of a circuit can be increased by designing the circuit for the task it has to perform. By building custom or dedicated circuits for the operation to be performed, the performance of an electronic circuit can be increased drastically. Flexibility increases the ease and the reusability of a circuit. More the flexible, more easily can a circuit be used for different purposes. For imparting flexibility to a circuit, it should contain more general purpose components. These components can do more number of tasks using the same circuit. But this can cause a reduction in the performance of the circuit. This is because the general purpose components in the circuit doesn’t deliver the performance shown by their dedicated counterparts.

The above problem can be solved by a circuitry that combines the flexibility of general purpose circuits as well as the performance of dedicated circuits. This is made possible by Reconfigurable Processors. Reconfigurable Processors (RP) or Reconfigurable Computing. Reconfigurable processor is a computer architecture combining some of the flexibility of software with the high performance of hardware by processing with very flexible high speed computing fabric. RPs have a reconfigurable fabric that can be programmed to be made any custom or dedicated circuit. This will enable the RP to deliver the same performance delivered by dedicated circuits. Once the required task is done, the fabric can then be re-programmed again to another circuit and thus is highly flexible.

Modern day RPs have found applications in areas where flexibility and speed are primary concern. These include signal processing, image processing, data encryption and decryption, avionic circuits, military applications, computer aided design etc.

**Chapter 2**

**HISTORY**

In 1960, Gerald Estrin, the well-known American computer scientist, proposed the concept of a computer make of a standard processor and an array of reconfigurable hardware. In his model the main processor would control the behaviour of the reconfigurable hardware while the latter would be tailored to perform a specific task, such as image processing or pattern matching as quickly as a dedicated piece of hardware. Once the task was done, the hardware could be adjusted to do some other task. This resulted in a hybrid computer structure combining the flexibility of software with the speed of hardware.

**2.1 ROM as PLD**

The first devices that were used to create arbitrary combinational logics where ROMs. Consider a ROM with m inputs (the address lines) and n outputs (the data lines). When used as a memory, the ROM contains 2m words of n bits each. The advantage of using a ROM is that any conceivable function of all possible combinations of the m inputs can be made to appear at any of the n outputs, making this the most general-purpose combinational logic device available for m input pins and n output pins.

When EPROMs and EEPROMs where available this introduced the concept of programmability and re-programmability. But ROMs were still not dedicated hardware and it had the following de-merits:

* They are usually much slower than dedicated logic circuits
* They consume more power
* They are often more expensive than programmable logic, especially if high speed is required
* The cannot be used stand alone for sequential circuits as most of the ROMs lack input or output registers

**2.2 Early Programmable Logic**

In 1969, Motorola offered the XC157, a mask-programmed gate array with 12 gates and 30 uncommitted input/output pins. In 1970, Texas Instruments (TI) developed a mask-programmable IC based on the IBM read-only associative memory. This device, the TMS2000, was programmed by altering the metal layer during the production of the IC. TI coined the term programmable logic array (PLA) for this device.

In 1971, General Electric Company (GE) was developing a programmable logic device based on the new UV erasable PROM technology. The GE device was the first erasable PLD ever developed. In 1973 National Semiconductor introduced a mask-programmable PLA device - DM7575. This was more popular than the TI part but cost of making the metal mask limited its use. The device is significant because it was the basis for the Field Programmable Logic Array (FPLA) produced by Signetics in 1975, the 82S100.

In 1974 GE entered into an agreement with Monolithic Memories to develop a mask- programmable logic device incorporating the GE innovations. The device was named the 'Programmable Associative Logic Array' or PALA. The MMI 5760 was completed in 1976 and could implement multilevel or sequential circuits of over 100 gates. The device was supported by a GE design environment where Boolean equations would be converted to mask patterns for configuring the device. Unfortunately, due to reasons, the part was never brought to market.

MMI introduced a breakthrough device in 1978, the programmable array logic (PAL). The architecture was simpler than that of Signetics FPLA, because it omitted the programmable OR array. This made the parts faster, smaller and cheaper. The PALASM design software (PAL assembler) converted the engineers' Boolean equations into the fuse pattern required to program the part. The PAL devices were soon second-sourced by National Semiconductor, Texas Instruments and AMD.

An innovation of the PAL was the Generic Array Logic device (GAL), invented by Lattice Semiconductor in 1985. 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. GALs are programmed and reprogrammed using a PAL programmer, or by using the in-circuit programming technique on supporting chips.

PALs and GALs are available only in small sizes, equivalent to a few hundred logic gates. For bigger logic circuits, complex PLDs or CPLDs can be used. These contain the equivalent of several PALs linked by programmable interconnections, all in one integrated circuit. CPLDs can replace thousands, or even hundreds of thousands, of logic gates. Some CPLDs are programmed using a PAL programmer, but this method becomes inconvenient for devices with hundreds of pins. A second method of programming is to solder the device to its printed circuit board, then feed it with a serial data stream from a personal computer. Some manufacturers use JTAG to program CPLD's in-circuit.

**2.3 Field Programmable Gate Array**

While PALs were busy developing into GALs and CPLDs, a separate stream of development was happening. This type of device is based on gate array technology and is called the Field Programmable Gate Array (FPGA). Early examples of FPGAs are the 82S100 array, and 82S105 sequencer, by Signetics, introduced in the late 1970s. The 82S100 was an array of AND terms. The 82S105 also had flip flop functions.

FPGAs use a grid of logic gates, and once stored, the data doesn't change, similar to that of an ordinary gate array. The term "field-programmable" means the device is programmed by the customer, not the manufacturer.

FPGAs are usually programmed after being soldered down to the circuit board, in a manner similar to that of larger CPLDs. In most of the larger FPGAs, the configuration is volatile and must be re-loaded into the device whenever power is applied or different functionality is required. Configuration is typically stored in a configuration PROM or EEPROM. EEPROM versions may be in-system programmable (typically via JTAG).

The difference between FPGAs and CPLDs is that FPGAs are internally based on Look-up tables (LUTs) whereas CPLDs form the logic functions with sea-of-gates (e.g. sum of products). CPLDs are meant for simpler designs while FPGAs are meant for more complex designs. In general, CPLDs are a good choice for wide combinational logic applications, whereas FPGAs are more suitable for large state machines (i.e. microprocessors).

Altera was founded in 1983 and delivered the industry’s first reprogrammable logic device in 1984 – the EP300 - which featured a quartz window in the package that allowed users to shine an ultra-violet lamp on the die to erase the EPROM cells that held the device configuration. Xilinx co-founders Ross Freeman and Bernard Vonderschmitt invented the first commercially viable field-programmable gate array in 1985 – the XC2064. The XC2064 had programmable gates and programmable interconnects between gates, the beginnings of a new technology and market. The XC2064 had 64 configurable logic blocks (CLBs), with two three-input lookup tables (LUTs).

The 1990s were an explosive period of time for FPGAs, both in sophistication and the volume of production. In the early 1990s, FPGAs were primarily used in telecommunications and networking. By the end of the decade, FPGAs found their way into consumer, automotive, and industrial applications.

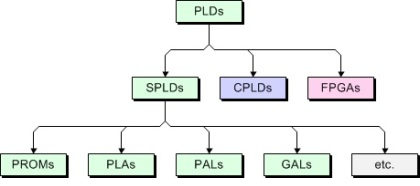
**2.4 Present day developments**

In 2002, Cypress began shipping commercial quantities of their RP called Programmable System-on-Chip (PSoC). A PSoC integrated circuit is composed of a core, configurable analogue and digital blocks, and programmable routing and interconnect.

In 2002, Chameleon Systems stated shipping their Chameleon chips. These RPs could be reconfigured at very high speeds of 20us. Thus they can be used switched between different applications, swiftly and hence without considerable performance reduction. This was a major advantage over traditional FPGAs, as reconfiguring the whole FPGA was a time consuming process, which usually involved resetting the whole FPGA and loading the new circuit configuration into the FPGA.

In the 2000s, both Altera and Xilinx stated shipping FPGAs that supported partial and dynamic reconfiguration on the go. This enabled the FPGA to change its circuitry fast without introducing much delay or hassle and hence made different circuit implementations on the same chip to work very efficiently and fast.

In 2010, Xilinx introduced the first All Programmable System on a Chip branded Zynq-7000 that fused features of an ARM high-end microcontroller with a 28 nm FPGA fabric to make it easier for embedded designers to use. The extensible processing platform enables system architects and embedded software developers to apply a combination of serial and parallel processing system designs. The high level of integration helps to reduce power consumption and dissipation. An alternate approach to using hard-macro processors is to make use of soft processor cores that are implemented within the FPGA logic. Nios II, MicroBlaze and Mico32 are examples of popular softcore processors.



**Chapter 3**