Basic Design with PLDs

Advanced Micro Devices

INTRODUCTION

The Programmable Array Logic device, commonly known as the PAL device, was invented at Monolithic Memories in 1978. The concept for this revolutionary type of device sprang forth as a simple solution to the short comings of discrete TTL logic.

The successfully proven PROM technology which allowed the end user to "write on silicon" provided the technological basis which made this kind of device not only possible, but very popular as well.

The availability of design software made it much easier to design with programmable logic. As designers were freed from the drudgery of low-level implementation issues, new complex designs were easier to implement, and could be completed more quickly.

This chapter outlines some basic information essential to those who are unfamiliar with Programmable Logic devices (PLDs). The information may also be useful to those who are current users of programmable logic. The specific issues which need to be addressed are:

- What is a PLD?
- What other implementations are possible?
- What advantages do PLDs have over other implementations?

WHAT IS A PLD?

In general, a programmable logic device is a circuit which can be configured by the user to perform a logic function. Most "standard" PLDs consist of an AND array followed by an OR array, either (or both) of which is programmable. Inputs are fed into the AND array, which performs the desired AND functions and generates product terms. The product terms are then fed into the OR array. In the OR array, the outputs of the various product terms are combined to produce the desired outputs.

PAL Devices

The PAL device has a programmable AND array followed by a fixed OR array (Figure 1). The fact that the AND array is programmable makes it possible for the devices to have many inputs. The fact that the OR array is fixed makes the devices small (which means less expensive) and fast.

WHAT OTHER IMPLEMENTATIONS ARE POSSIBLE?

There are essentially four alternatives to programmable logic:

- Discrete Logic
- Gate Arrays
- Standard Cell Circuits
- Full Custom Circuits

Discrete Logic

Discrete logic, or conventional TTL logic, has the advantage of familiarity; hence its popularity. It is also quite inexpensive when only unit cost is considered. The drawback is that the implementation of even a simple portion of a system may require many units of discrete logic. There are "hidden" costs associated with each unit that goes into a system, which can render the overall system more expensive.

Designing with discrete chips can also be very tedious. Each design decision directly affects the layout of the board. Changes are difficult to make. The design is also more difficult to document, making it harder to debug and maintain later. These items all contribute to a long design cycle when discrete chips are used extensively.

Gate Arrays

Gate arrays have been increasing in popularity. The attractiveness of this solution lies in the device's flexibility. By packing the functions into the device, a great majority of the available silicon is actually used. Since such a device is customized for an application, it would seem to be the optimum device for that application.

However, one also pays substantial development costs, especially in the case of a design which needs changes after silicon has already been processed. Even though the unit costs are generally quite low for gate arrays, the volumes required to make their use worthwhile excludes them as a solution for many designers. This fact, added to the long design cycle and high risk involved, make this solution practical for only a limited number of designers.

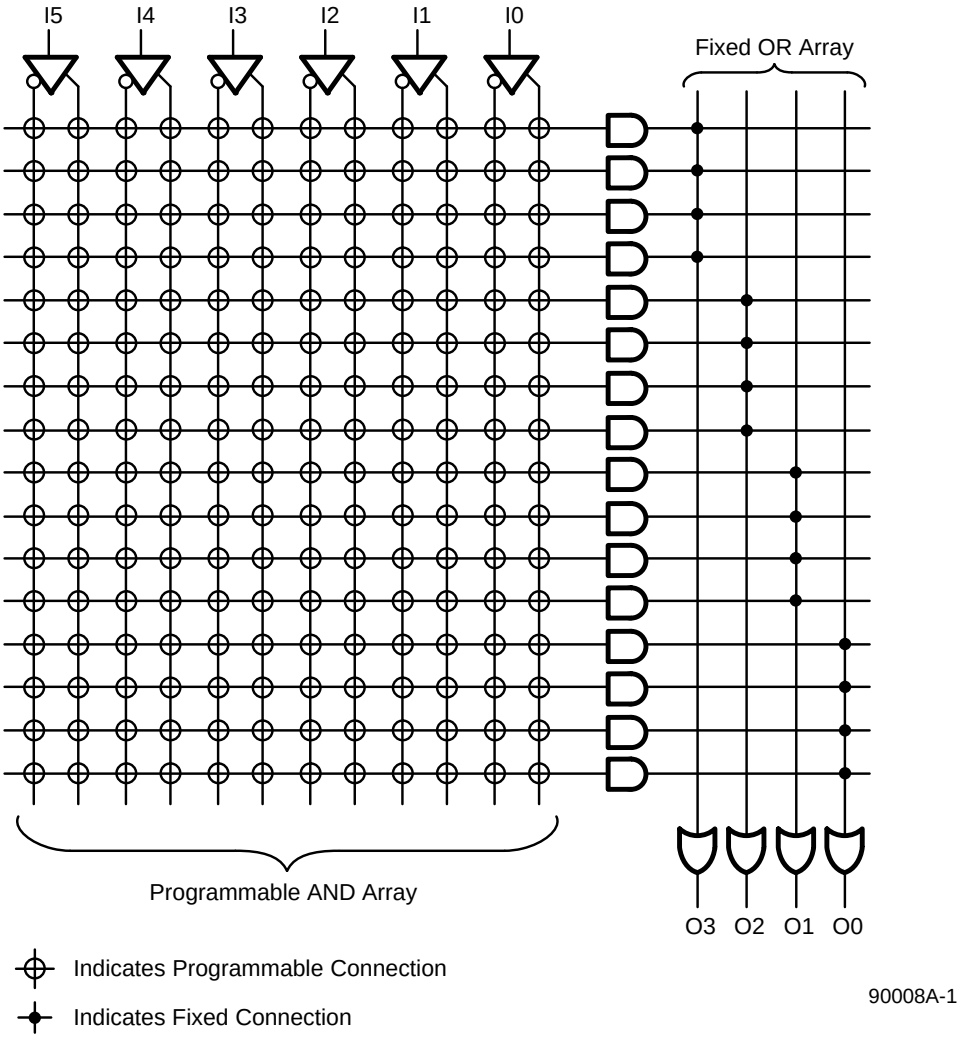

Figure 1. PAL Device Array Structure

Standard Cell Circuits

Standard cell circuits are quite similar to gate arrays, their main advantage being that they consist of a collection of different parts of circuits which have already been debugged. These circuits are then assembled and collected to perform the desired functions. This can ideally lead to reduced turn around from conception to implementation, and a much more efficient circuit.

The drawback is that even though the individual components of the circuit have been laid out, a complete layout must still be performed to arrange the cells. Instead of just customizing the metal interconnections, as is done in a gate array, the circuit must be developed from the bottom up. Development costs can be even higher than for gate arrays, and despite the standard cell concept, turn around time often tends to be longer than planned. Again, the volume must be sufficiently high to warrant the development costs.

Full Custom Circuits

Full custom designs require that a specific chip be designed from scratch to perform the needed functions. The intent is to provide a solution which gives the designer exactly what is needed for the application in question; no more and no less. Ideally, not a square micron of silicon is wasted. This normally results in the smallest piece of silicon possible to fit the needs of the design, which in turn reduces the system cost. Understandably, though, development costs and risks for such a design are extremely high, and volumes must be commensurately high in order for such a solution to be of value.

WHAT ADVANTAGES DO PLDs HAVE OVER OTHER IMPLEMENTATIONS?

As user-programmable semicustom circuits, PLDs provide a valuable compromise which combines many of the benefits of discrete logic with many of the benefits of other semicustom circuits. The overall advantages can be found in several areas:

- Ease of design
- Performance
- Reliability
- Cost savings

Ease of Design

The support tools available for use in designing with PLDs greatly simplify the design process by making the lower-level implementation details transparent. In a matter of one or two hours, a first time PLD user can learn to design with a PAL device, program it, and implement the design in a system.

The design support tools consist of design software and a programmer. The design software is used in generating the design; the programmer is used to configure the device. The software provides the link between the higher-level design and the low-level programming details.

All of the available design software packages perform essentially the same tasks. The design is specified with relatively high-level constructs; the software takes the design and converts it into a form which the programmer uses to configure the PLD. Most software packages provide logic simulation, which allows one to debug the design before actually programming a device. The high-level design file also serves as documentation of the design. This documentation can be even easier to understand than traditional schematics.

Many PLD users do not find it necessary to purchase a programmer; it is often quite cost effective and convenient to have either the manufacturer or an outside distributor do the programming for them. For design and prototyping, though, it is very helpful to have a programmer; this allows one to implement designs immediately.

The convenience of programmable logic lies in the ability to customize a standard, off-the-shelf product. PLDs can be found in stock to suit a wide range of speed and power requirements. The variety of architectures available also allows a choice of the proper functionality for the application at hand. Thus, a design can be implemented using a standard device, with the end result essentially being a custom device. If a design change is needed, it is a simple matter to edit the original design and then program a new device, or, in the case of reprogrammable CMOS devices, erase and reprogram the old device.

Board layout is vastly simplified with the use of programmable logic. PLDs offer great flexibility in the location of inputs and outputs on the device. Since larger functions are implemented inside the PLD, board layout can begin once the inputs and outputs are known. The details of what will actually be inside the PLD can be worked out independently of the layout. In any cases, any needed design changes can be taken care of entirely within the PLD, and will not affect the PC board.

Performance

Speed is one of the main reasons that designers use PAL devices. The PAL devices can provide equal or better performance than the fastest discrete logic available. Today's fastest PAL devices are being developed on the newest technologies to gain every extra nanosecond of performance.

Performance cannot come strictly at the expense of power consumption. Since PLDs can be used to replace several discrete circuits, the power consumption of a PLD may well be less than that of the combined discrete devices. As more PLDs are developed in CMOS technology, the option for even lower power becomes available, including zero standby power devices for systems which can tolerate only minute standby power consumption.

Reliability

Reliability is an area of increasing concern. As systems get larger and more complex, the increase in the amount of circuitry tends to reduce the reliability of the system; there are "more things to go wrong." Thus, a solution which inherently reduces the number of chips in the system will contribute to higher reliability. A programmable logic approach can provide a more reliable solution due to the smaller number of devices required.

With the reduction in units and board space, PC boards can be laid out less densely, which greatly improves the reliability of the board itself. This also reduces crosstalk and other potential sources of noise, making the operation of the system cleaner and more reliable.

Cost

For any design approach to be practical, it must be cost effective. Cost is almost always a factor in considering a new design or a design change. But, the calculation of total system cost can be misleading if not all aspects are considered. Many of the costs can be elusive or difficult to measure. For example, it is difficult to quantify the cost of market share lost due to late product introduction.

The greatest savings over discrete design are derived from the fact that a single PLD can replace several discrete chips. Board space requirements can drop 25% or more when PLDs are used. The relationship between the various alternatives is summarized in Figure 2.

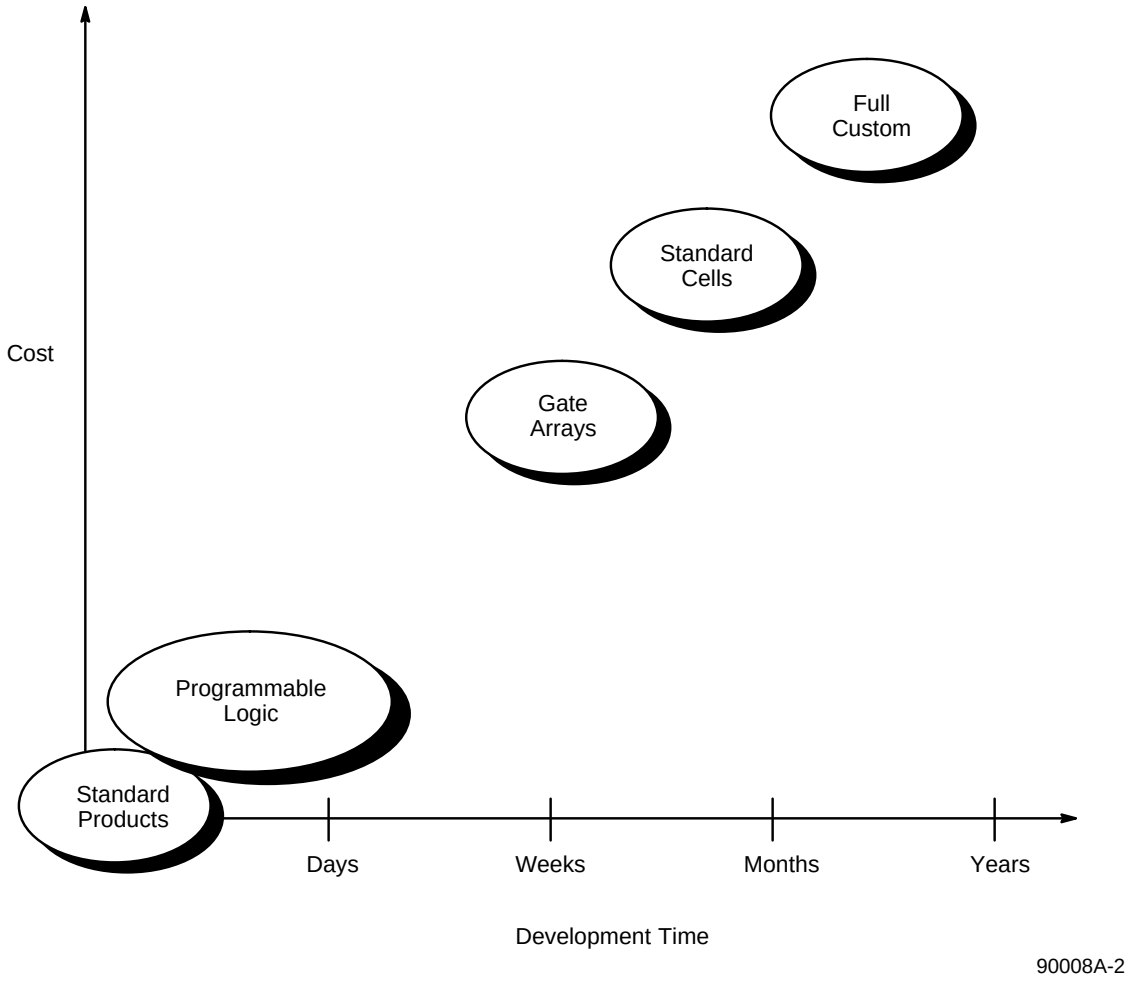

Figure 2. Development Cost vs. Time for Alternative Logic Implementations

Another economic benefit of the use of PLDs is that when one PAL device is used in several different designs, as is often the case, the user has not committed that device to any one of the particular designs until the device has been programmed. This means that inventory can be stocked for several different designs in the form of one device. As requirements change, the parts can be programmed to fit the need. And in the case of reprogrammable CMOS devices, one is not committed even after programming.

One final subtle cost issue is derived from the ease with which a competitor can copy a design. PLDs have a unique feature called a security bit, whose purpose is to protect a design from being copied. By using secured PLDs extensively in a system, one can safely avoid having one's system easily deciphered. The added design security provided by this feature can buy extra market time, forcing competitors to do their own original design work rather that copying the designs of others.

Summary

Programmable logic provides the means of creating semi-custom designs with readily available standard components. There is a wide variety of PLDs; PAL devices are most widely used, and perform well for basic logic and some sequencing functions.

By assuming some of the attributes of gate arrays, programmable logic provides the cost savings of any other semicustom device, without the extra engineering costs, risks, and design delays. Reliability is also enhanced as quality increases and board complexity decreases.

The design tasks are greatly simplified due to the design tools which are now available. Design software and device programmers allow top-down high-level designs with a minimum of time spent on actual implementation issues. Simulation allows some design debug before a device is programmed.

For all of these reasons, programmable logic has become, and will continue to be, the design methodology of choice among digital systems designers.