Programmable Logic and Application Specific Integrated Circuits

by

Dave Landis, Ph.D., P.E.
Professor of Electrical Engineering
The Pennsylvania State University
Center for Electronic Design, Communications, and Computing
11 EE West Building
University Park, PA 16802
dll2@psu.edu

Table of Contents:

I. INTRODUCTION TO ASICS AND PROGRAMMABLE LOGIC 3

A. OVERVIEW OF ASIC DESIGN OPTIONS 4
1. Full Custom Design 5
2. Standard Cell Design 5
3. Gate Array Design 6
4. Field Programmable Logic 6
B. ASIC DESIGN STYLE SELECTION 7

II. ASIC FABRICATION TECHNOLOGIES 8

A. OVERVIEW OF PROCESSING TECHNOLOGIES 8
B. COMPARISON OF TECHNOLOGIES 10

III. FIELD PROGRAMMABLE LOGIC 12

A. PROGRAMMABLE LOGIC DEVICES 13
B. FIELD PROGRAMMABLE GATE ARRAYS 17
1. FPGA Programming Technologies 17
2. FPGA Architectures 21
a) Symmetrical Array FPGA Example: Xilinx 23
b) Row Based FPGA Example: Actel 25
c) Hierarchical PLD Example: Altera 28
d) Sea of Gates FPGA Example: Algotronix 30
3. FPGA Design Flow 32
4. Summary of FPGA Selection Criteria 35
a) Manufacturing and Reliability Issues 35
b) Design and Test Issues 36

IV. MASK PROGRAMMABLE GATE ARRAYS 37

V. STANDARD CELL AND CUSTOM ASICS 41

VI. ASIC PACKAGING 43

VII. SYSTEM LEVEL ASIC DESIGN ISSUES 44

A. BEHAVIORAL MODELING AND SYNTHESIS 44
B. DESIGN FOR TESTABILITY 46
1. Structured Design for Testability: Scan Design 48
2. IEEE 1149.1 Boundary Scan Architecture and Test Bus 50
C. MIXED-SIGNAL ASICS 52

VIII. REFERENCES 53

Programmable Logic and Application Specific Integrated Circuits

Chapter II: Handbook of Components for Electronics, Vol. I

List of Figures

FIGURE 1. VLSI MARKET HIERARCHY....4
FIGURE 2. TOTAL ASIC PRODUCTION COST VS. VOLUME....8
FIGURE 3. TAXONOMY OF IC FABRICATION TECHNOLOGIES....9
FIGURE 4. SPEED / POWER PERFORMANCE PROJECTIONS GAAS AND SILICON IC TECHNOLOGIES....11
FIGURE 5. CMOS AND GAAS POWER CONSUMPTION VS. FREQUENCY....12
FIGURE 6. TYPICAL PROGRAMMABLE-AND FIXED-OR STRUCTURE OF 22V10 PAL....14
FIGURE 7. EPROM FLOATING GATE CROSS SECTION....14
FIGURE 8. EEPROM CELL LAYOUT VIEW....15
FIGURE 9. PLICE™ ANTI-FUSE PROGRAMMABLE INTERCONNECT....19
FIGURE 10. THE FOUR FPGA ARCHITECTURAL CLASSES....21
FIGURE 11. XILINX XC4003 LOGIC CELL ARRAY (LCA)....23
FIGURE 12. XILINX XC4000 FAMILY CONFIGURABLE LOGIC BLOCK (CLB)....25
FIGURE 13. XILINX XC5000 ARRAY ARCHITECTURE....25
FIGURE 14. XC5000 VERSABLOCK AND CLB STRUCTURE....25
FIGURE 15. ACTEL ACT1 LOGIC MODULE (LM)....27
FIGURE 16. ALTERA MAX 7000 INTERNAL ARCHITECTURE....28
FIGURE 17. ALTERA FASTTRACK INTERCONNECT ARCHITECTURE....29
FIGURE 18. LAB CONNECTION TO ROW AND COLUMN INTERCONNECT....30
FIGURE 19. ALGOTRONIX ARRAY ARCHITECTURE....31
FIGURE 20. ALGOTRONIX LOGIC CELL FUNCTION UNIT DESIGN....32
FIGURE 21. TYPICAL CAD SYSTEM DESIGN FLOW FOR FPGAS....33
FIGURE 22. SEA OF GATES (SOG) MPGA ARCHITECTURE....40
FIGURE 23. LSSD POLARITY HOLD SHIFT REGISTER LATCH (SRL)....49
FIGURE 24. IEEE 1149.1 TEST INTERFACE....51

List of Tables

TABLE 1. ASIC MARKET FORECAST (PREDICTED WORLDWIDE SALES IN MILLIONS OF DOLLARS)[3]....7
TABLE 2. SUMMARY OF ASIC DESIGN STYLE ATTRIBUTES....7
TABLE 3. PAL AND PLD COMMERCIAL PRODUCT EXAMPLES....16
TABLE 4. SUMMARY OF FPGA PROGRAMMING CHARACTERISTICS....20
TABLE 5. COMMERCIAL FIELD PROGRAMMABLE GATE ARRAY PRODUCTS....22
TABLE 6. XILINX FPGA FAMILY CHARACTERISTICS....23
TABLE 7. ACTEL FPGA FAMILY CHARACTERISTICS....26
TABLE 8. ALTERA EPLD (FPGA) FAMILY CHARACTERISTICS....29
TABLE 9. COMMERCIAL CMOS MASK PROGRAMMABLE GATE ARRAY PRODUCTS....37
TABLE 10. BIPOLAR, BICMOS, AND GAAS GATE ARRAY PRODUCTS....39
TABLE 11. COMMERCIAL CMOS STANDARD CELL PRODUCTS....41
TABLE 12. BIPOLAR, BICMOS, AND GAAS CELL-BASED PRODUCTS....42
TABLE 13. IEEE 1149.1 INTERFACE PIN/SIGNAL DEFINITIONS....50

I. Introduction to ASICs and Programmable Logic

A goal of this chapter is to provide the technical community with a basic understanding of the technologies and options available in the Integrated Circuits (ICs) that they design and use. Advancements in Very Large Scale Integration (VLSI) technology have brought chips with millions of transistors into our laboratories, offices, and homes. In order to be competitive, companies must develop new products and enhance existing ones by incorporating the latest commercial off-the-shelf VLSI chips; and more-and-more by designing chips which are uniquely tailored for their own applications. These so-called "Application Specific Integrated Circuits" (ASICs) are changing the way electronic systems are designed, manufactured, and marketed. Thus, it is important in this competitive environment to understand the nature, options, design styles, and costs of ASIC technology and of the related programmable logic IC families.

During the 1980's, Full Custom and Application Specific Integrated Circuit designs were approached from two quite different directions. ASIC design focused on meeting time-to-market and customer specific requirements. ASIC design methodologies used chips containing arrays of prefabricated gates (gate arrays), or chips based on libraries of standard function cells (standard cell designs). These designs sacrificed density but allowed the customer to perform major portions of the design in-house; albeit with significant help from the semiconductor manufacturer. Full-custom or hand-crafted IC design addressed maximum density and performance for high-volume standard products. These full custom chip designs were the sole purview of the semiconductor manufacturers who targeted only the largest markets; frequently providing customers with multiple sources for most products.

Since the 1980s, product offerings of the typical semiconductor manufacturer have undergone dramatic changes. Memories, Microprocessors, and other traditional high volume standard products that are still available are now referred to as "commodity products". In addition, current standard product portfolios include a significant proportion of so-called Application Specific Standard Products (ASSPs). These ASSPs differ from commodity products through the incorporation of value-added features for specific market segments (e.g. complete SCSI controllers targeted for workstations). Furthermore, ASSPs from different vendors may have similar functionality but they are seldom pin-for-pin compatible; resulting in intensive "design-in competition" among vendors offering comparable solutions to the same problem. Figure 1 illustrates this hierarchy in the present Integrated Circuit market. The introduction of the ASSP designation is a natural result of semiconductor manufacturers use of the ASIC design methodology for standard products. However, this points to a problem of accuracy with the current use of the term ASIC. ASIC chips are not generally just application specific, they are customer specific. It can be argued that customer programmable logic devices are the only true ASICs. To compound this problem, the term is used differently by different people in different contexts. ASIC is defined variously as gate array, any custom, semi-custom, or programmable technology, and as a design methodology.

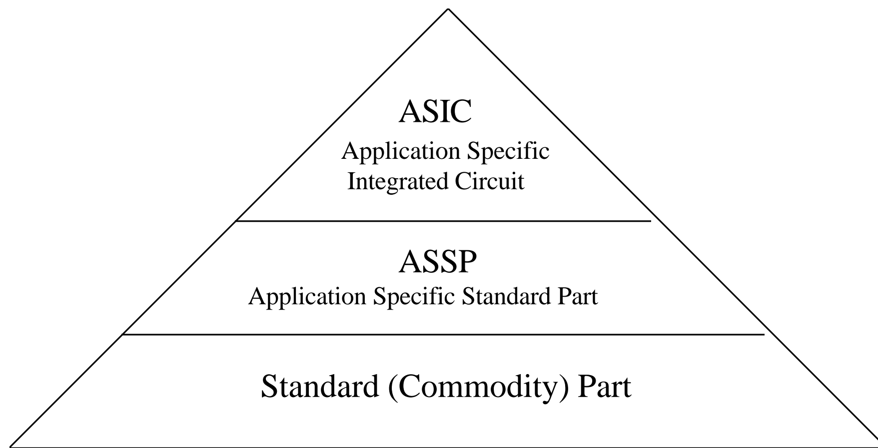

Figure 1. VLSI Market Hierarchy

An even more gray area is created by the previously discussed ASSPs. Although they are built using traditional ASIC technology, they are sold like standard parts. Nonetheless, the term ASIC is still quite useful. In addition to describing gate array, standard cell, and PLD chip designs, ASIC describes a methodology (or group of methodologies) for designing electronic systems, and it describes a technology (or group of technologies) used to build electronic systems. Most of today's flexible and cost effective electronic systems are designed using ASIC methodologies and built using ASIC technology.

It is clear that a wide range of Integrated Circuit (IC) design options are available for electronic system construction. Many factors must be considered in order to make informed choices about the chips that we purchase, specify, or design. The remainder of this introduction provides a broad comparison of the ASIC and programmable logic technologies, and identifies both technical and economic selection criteria. Section II provides an overview of the semiconductor processing technologies in use today, and provides a brief comparison of technology attributes. In Section III an in-depth evaluation of field-programmable logic devices is presented together with examples of currently available commercial chips. Sections IV and V present information about the more traditional gate array and standard cell ASIC design methodologies. Finally, Sections VI and VII discuss system implementation and application issues including ASIC packaging, behavioral modeling, design for testability, and mixed-signal design..

A. Overview of ASIC Design Options

This section introduces the range of options and styles available for integrated circuit design. Although the bulk of this chapter will focus on the programmable logic design style, this section places programmable logic in context alongside the alternate design techniques. The following sections are loosely organized in order of decreasing design investment (non-recurring engineering costs) and corresponding maximum chip complexity.

1. Full Custom Design

In the classic full custom design style, each primitive logic function or transistor is manually designed and optimized. This results in the most compact chip design with the highest possible speed and lowest power dissipation. However, the initial investment or Non-Recurring Engineering (NRE) cost is highest compared to all other design styles. The designer must manipulate the individual geometric shapes which represent the features of each transistor on the chip; hence the often applied term for full custom design: "polygon pushing". A relatively simple 3000 gate design might require the handling of 300,000 rectangles per chip.[1] Although this design style was used exclusively in early ICs, engineers rarely use it for today's ASICs due to the high engineering costs and low designer productivity. Productivity for full custom logic designs is typically only 6 to 17 transistors per day.[2] The exception is in high volume commodity products such as memories which must be hand-crafted to meet density and performance requirements. In addition, at least portions of high-end products such as microprocessors are full custom designed for performance reasons. Worldwide sales of full custom ASIC designs are predicted to grow only slightly from the current level of $2.7 Billion to $2.9 Billion in 1998 (a declining market share from 23% to 16%).[3]

2. Standard Cell Design

In the standard cell design methodology, pre-defined logic and function blocks are made available to the designer in a cell library. Typical libraries begin with gate level primitives such as AND, OR, NAND, NOR, XOR, Inverters, flip-flops, registers, and the like. Libraries generally include more complex functions such as adders, multiplexers, decoders, ALUs, shifters, and memory (RAM, ROM, FIFOs, etc.). In some cases, the standard cell library may include complex functions such as multipliers, dividers, microcontrollers, microprocessors, and microprocessor support functions (parallel port, serial port, DMA controller, event timers, real-time clock, etc.).

Standard cell designs are created using schematic capture tools or via synthesis from a Hardware Description Language (HDL). Section VII of this chapter will discuss behavioral modeling and synthesis options in more detail. Automated tools are then used to place the cells on a chip image and wire them together. Standard cell layouts are easily identified by rows of equal height cells separated by wiring channels. Large macro-cells such as multipliers or microcontrollers may span multiple cell rows and block some of the wiring channels. Standard cell designs operate a lower clock rates and are generally less area efficient than a full custom design due to the fixed cell size constraints and requirements for dedicated wiring channels. However, very high layout density is achieved within the cells themselves, resulting in densities which can approach that of full custom designs with substantially shorter design times. In the field of CMOS ASICs, standard cells are the fastest growing market segment with worldwide sales of $2.9 billion on 1993. This 26% market share is expected to grow to $6.2 billion or 34% of the total ASIC market by 1998.[3] The growing market success of standard cell design is largely due to the increasing availability of "mega-cell" and "core" functions in their libraries. These simplify ASIC design by providing entire sub-system or chip level functional blocks (e.g. a RISC processor

core, a DMA or memory controller, or a complete I/O subsystem) from which the designer can compose a complex ASIC.

3. Gate Array Design

Full custom and standard cell design methodologies require custom chip fabrication using a complete set of unique masks which define the semiconductor processing of the design. Thus, both the NRE cost for the mask set and the design turn-around time through the foundry are quite high. As an alternative, a chip design can be created using a custom interconnection pattern on an array of uncommitted logic gates (i.e. a gate array). Wafers of chips containing the uncommitted logic gate arrays can be pre-fabricated up to the point of the final metalization steps which create the logic personalization. Compared to standard cell or full custom designs, the design turnaround time and cost are reduced because only the top level interconnect and contact mask steps (2-5 masks) need to be applied. Several other advantages of these "mask programmed" gate arrays relate to the economies of scale for this design style. Wafer costs are low because many different customer designs can be created from the same base wafer design. Similarly, foundry packaging and test costs are low due to standard pin-outs and common test fixtures which can be used for multiple designs. Section IV will describe gate array design in more detail.

Gate array sales are also growing substantially, but at a slower rate than standard cells. They are expected to maintain a relatively constant 34% share of the market through 1998, corresponding to a predicted dollar sales growth from $3.5 billion to $5.1 billion.[3]

4. Field Programmable Logic

A field programmable logic device is a chip whose final logic structure is directly configured by the end user. By eliminating the need to cycle through an integrated circuit production facility, both time to market and financial risk can be substantially reduced. The two major classes of field programmable logic, Programmable Logic Devices (PLDs) and Field Programmable Gate Arrays (FPGAs), have emerged as cost effective ASIC solutions because they provide low-cost prototypes with nearly instant "manufacturing". This class of device consists of an array of uncommitted logic elements whose interconnect structure and/or logic structure can be personalized on-site according to the user's specification. Section III of this chapter will discuss the design alternatives for this class of ASIC in more detail. Field programmable logic devices are expected to grow from the 1993 world wide sales level of 11% to at least 17% by 1998.[3] Table 1 summarizes the current ASIC worldwide market and indicates the forcasted trends. Although field programmable logic represents only a small percentage of total ASIC market sales, statistics indicate that *approximately one half of all chip design projects today are begun using FPGAs*.[4]

Table 1. ASIC Market Forecast (predicted worldwide sales in millions of dollars)[3]

ASIC Design Style	1994	1996	1998
Full Custom	2,725	2,800	2,900
Standard Cell	3,375	4,600	6,200
CMOS Gate Array	3,765	4,350	5,100
Bipolar Gate Array	750	615	460
CMOS PLD	785	1,150	1,550
Bipolar PLD	185	120	75
Field Programmable Gate Array	500	900	1,350

B. ASIC Design Style Selection

Each of the ASIC design styles fills a niche in the IC system marketplace. Full custom designs are appropriate for high volume markets where the large NRE costs can be amortized over a high production volume. Furthermore, full custom designs typically consume less power and operate at higher speeds than the other design styles. At the opposite end of the spectrum, FPGAs offer much lower prototype costs and have drastically shorter production times. However, the three main disadvantages of FPGAs are their low speed of operation, low logic density, and high cost per chip. When implemented with equivalent semiconductor processes, FPGAs are approximately 3 times slower, 10 times less dense, and 5 times more expensive than their mask programmed gate array equivalents. Table 2 estimates the differences between the various design style alternatives for a particular ASIC, using relative numbers which are normalized to the full custom design style.

Table 2. Summary of ASIC Design Style Attributes

Attribute	FPGA	MPGA	Std. Cell	Full Custom
Development Cost	.15	.2	.3	1
Development Time	.05	.15	.25	1
Risk (1st pass success)	.3	.4	.5	1
Final Die Size	4-6	1.5-2	1.1-1.5	1
Chip Cost	10-20	3-4	1.5-2.0	1

When considering design style alternatives from a purely economic perspective, production volume is the dominant variable. For low production volumes the NRE costs are the dominant factor, while for high production volumes the cost per chip is the dominant factor. Furthermore, the NRE cost per chip increases as we move from FPGA to MPGA, to Standard

Cell, and finally to Full Custom design. Thus, for a particular IC design, each design style has an optimal range of production volumes for which it gives the lowest overall production cost. Figure 2 shows the total production cost vs. volume for a simple ASIC design produced in each style; the optimal style from a cost perspective is always the lowest line segment on the figure.

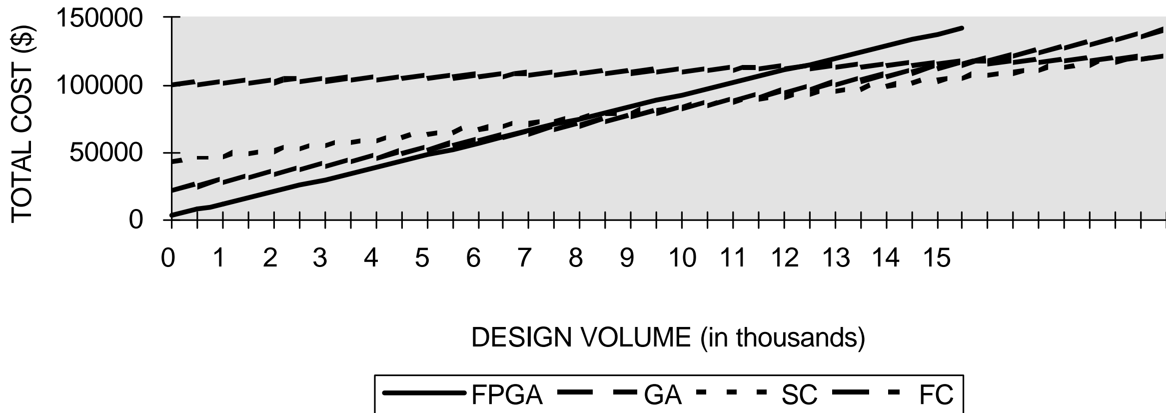

Figure 2. Total ASIC Production Cost vs. Volume

II. ASIC Fabrication Technologies

A. Overview of Processing Technologies

This section identifies and discusses the underlying semiconductor technologies used to fabricate programmable logic and ASIC devices. This treatment is not intended as an in-depth tutorial on semiconductor devices and fabrication, but rather to identify the minimum information necessary to enable designers to make intelligent technology choices. In order to appreciate the capabilities and limitations of a particular technology, the ASIC user or designer must know the characteristics of the pertinent fabrication technology.

CMOS is currently the dominant ASIC fabrication technology due to its many advantages including cost, performance, density, and manufacturing / designer experience. Referring back to the Gate Array and FPGA entries in Table 1, the prediction is that CMOS will continue to gain market share over bipolar technologies. Although designers typically lump ASIC designs into two groups, CMOS and "other" technologies, this section attempts to take broader view of the technology alternatives. Bipolar, BiCMOS, and GaAs ASICs each have unique advantages for many high performance applications.

Figure 3 presents a taxonomy of available semiconductor process technologies for ASICs. At the topmost level, the tree splits into silicon and gallium arsenide (GaAs) technologies. GaAs

has been slowly expanding from its historical markets in the military and aerospace fields; and may be ready to expand into the mainstream digital IC market.[5] It has inherent performance advantages over silicon, mainly due to its higher carrier mobility. Electrons travel four to five times faster through GaAs than bulk silicon, which means that GaAs logic will operate at much higher clock frequencies. Furthermore, lower electric fields are necessary to achieve the maximum mobility when compared to CMOS. Thus, as system operating voltages are reduced, this performance advantage becomes even greater. Although significant progress has been made to solve the historical materials and processing problems of GaAs, it still has several fundamental disadvantages when compared to silicon technology. For example, unlike silicon there is no native oxide to act as an insulator to produce simple MOS style logic elements. Also, holes in GaAs move more slowly than in silicon which makes complementary (CMOS) style circuit operation inefficient.

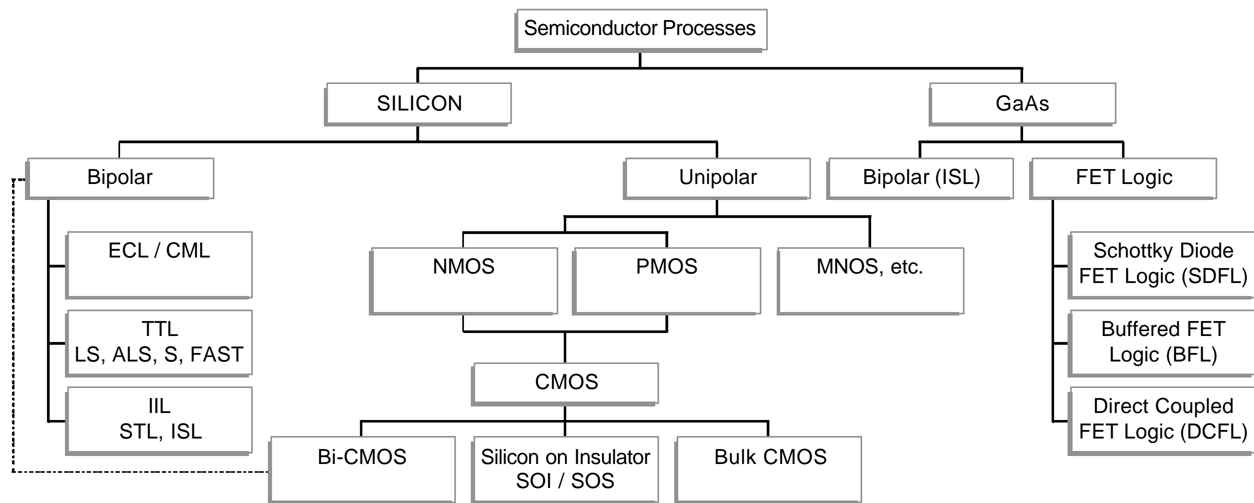

Figure 3. Taxonomy of IC Fabrication Technologies

As indicated on the right hand side of Figure 3, GaAs has many circuit topologies and device types. The most dominant commercially available GaAs technologies are Direct Coupled FET Logic (DCFL) and Source-Coupled FET Logic (SCFL). DCFL is similar in design to NMOS and because of its low transistor count circuits provides higher gate-count chips. It's speed is comparable to bipolar Emitter Coupled Logic (ECL) with a 60% reduction in power dissipation. SCFL has significantly higher speed than DCFL, with a correspondingly higher power dissipation. Other common GaAs technologies include Buffered FET Logic (BFL) and Bipolar Integrated Shottkey Logic (BSL).

Looking at the left side of Figure 3, it can be seen that silicon technologies split into two main categories; bipolar and unipolar. This distinction is made because in bipolar devices both majority and minority carriers participate in transistor operation. Bipolar processes were

dominant during the 1960's and 70's, and offer the potential for very high speed operation using Bipolar Junction Transistors (BJTs). However, the power dissipation in bipolar circuits is quite high, and the device density is not as great as in MOS designs. The still popular Transistor Transistor Logic (TTL) logic family, as well as Emitter Coupled Logic (ECL) and Integrated Injection Logic (IIL) families fall in this category. An important point about TTL logic is that its input/output voltage levels still constitute a de facto standard which is followed by other logic families.

The most important class of unipolar devices for ASICs are the Metal-Oxide Semiconductor (MOS) devices used in the PMOS, NMOS, and CMOS processes. While other unipolar technologies exist, such as the Metal-Nitride Oxide Semiconductor (MNOS) process used in nonvolatile memories, they do not represent a significant part of the ASIC market. Although universally used today, the acronym MOS is an outdated term. Metal refers to the gate layer, Oxide refers to the silicon dioxide insulator, and Semiconductor to the channel being controlled by the gate. MOS processes today make almost exclusive use of polysilicon rather than metal for the gate material.

MOS Field Effect Transistors (MOSFETS) are available in two basic flavors; P-Channel MOSFETS and N-channel MOSFETS. The term PMOS refers to an MOS process which exclusively uses P-channel MOSFET transistors; and similarly NMOS refers to an MOS process which exclusively uses N-channel transistors. Although used extensively in early MOS designs, PMOS technology is not used today because of the poor electrical characteristics of P-channel transistors. This is because the mobility of holes (the majority carrier in PMOS) is considerably poorer than the mobility of electrons (the majority carrier in NMOS). NMOS design offers excellent density and reasonable performance; but it is seldom used today because of the difficulties in designing ratioed NMOS logic and because it dissipates static power. However, some designs such as Dynamic Random Access Memories (DRAMs) utilize NMOS style circuits for the bulk of the chip array while providing CMOS support logic and I/O circuits. The term CMOS (Complementary MOS) refers to an MOS process that simultaneously provides both P- and N-channel transistors. Although CMOS logic circuits have increased fabrication costs and increased area when compared to PMOS or NMOS, their ease of design, potential for very low static power dissipation, and high speed operation make CMOS the technology of choice in the 90's.

Finally, BiCMOS is a relatively recent technology introduction which incorporates both Bipolar and CMOS devices on the same chip. Typically, most of the logic in a BiCMOS ASIC is CMOS, while the bipolar devices are used for on-chip and off-chip drivers. The advantage of the bipolar drivers is that they are capable of driving much higher loads without sacrificing speed. Compared to CMOS, BiCMOS is significantly faster, but chip cost can be two or three times higher.[6]

B. Comparison of Technologies

It is a tautology in the electronics industry that each new generation of systems must be of smaller size, lower weight, lower cost, higher speed, and higher reliability than the previous one. Integrated circuits, particularly ASICs, constitute much of the added value of an electronic system and thus the pressure to improve IC technology is great. Historically, silicon replaced germanium, NMOS replaced PMOS, CMOS supplanted NMOS and then to a large part bipolar and emitter coupled logic. GaAs is currently a contender for the high performance market, but has not yet achieved a dominant position in this market over traditional high speed ECL components. Note that a disadvantage of both GaAs and ECL technologies is their use of non-standard power supply voltages and the consequent difficulty of interfacing with "standard TTL" or CMOS signal levels.

Processing technology selection is a tradeoff between three main factors; operating speed, circuit area, and power dissipation. While CMOS has the major advantages of high density and low power dissipation compared to other processing technologies in use today, it is not the speed leader. As indicated in the previous section, GaAs technology shows great promise for high speed circuits, at the expense of higher power dissipation. In general, speed and power can be traded-off within certain limits for each process technology; creating an operating region for that technology. Figure 4 shows the major speed (propagation delay) and power dissipation tradeoffs for GaAs as well as the commonly used Silicon technologies.[7]

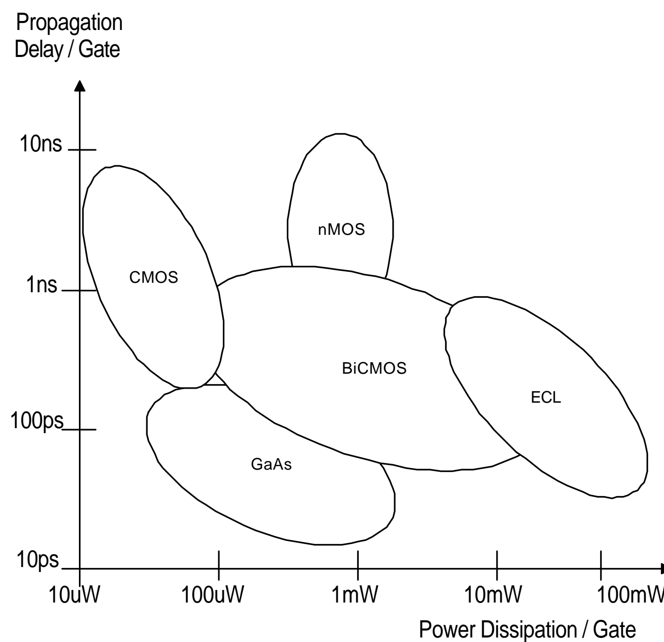

Figure 4. Speed / Power Performance Projections GaAs and Silicon IC Technologies

Although a detailed comparison of processing technology alternatives is beyond the scope of the chapter, several key points merit discussion. First is the fact that as operating frequency (f) increases, power dissipation in CMOS chips increases linearly according to the relationship $C*Vdd^2*f$ (where C is the driven capacitance and Vdd is the chip operating power supply voltage). Power dissipation in GaAs also increases with frequency, but at a much slower rate. As indicated

in Figure 5, present day CMOS has the advantage below 100-150mhz. However, as system clock rates exceed 150-200mhz GaAs currently has a clear advantage in terms of its speed-power product, especially for low-voltage applications.[5] Note that this crossover point moves as process technology scales; and the CMOS crossover point can be expected to move to higher clock frequencies for deep sub-micron CMOS processing.

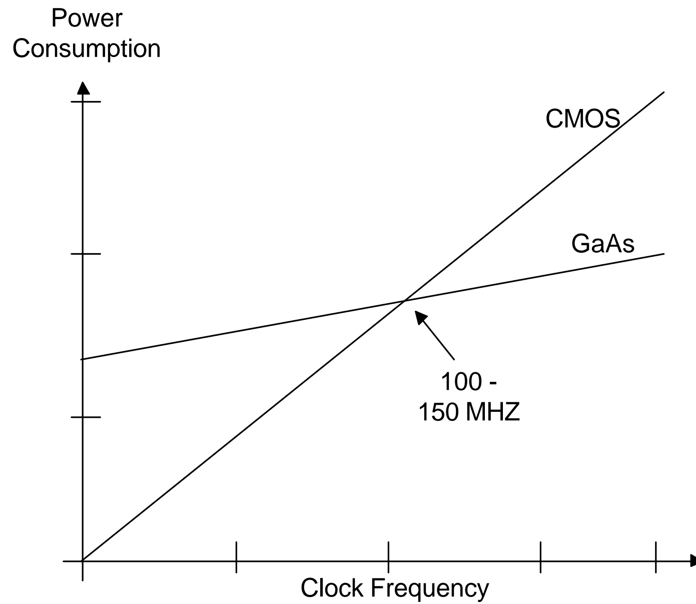

Figure 5. CMOS and GaAs Power Consumption vs. Frequency

III. Field Programmable Logic

Programmable logic devices have over two decades of history of use in electronic systems. Their origins coincided with the development of Programmable Read-Only Memorie (PROM) and Erasable PROM (EPROM) technologies around the same time as the microprocessor. However, until recently their development and use has been overshadowed by the headlong advances of microprocessor technology. Their main historical role was in replacing the multitude of Small Scale Integration (SSI) and Medium Scale Integration (MSI) "glue logic" parts which were needed in microprocessor based systems. It is only within the past seven years that programmable logic devices have become dense enough and fast enough to play more major roles in digital systems

A. Programmable Logic Devices

Programmable Read-Only Memories were the first programmable devices to achieve widespread use in digital systems. These devices are one-time programmable memories organized as an array of read-only cells. Simple programmable logic functions can be created using PROMs as a look-up table which stores the truth table of the logic function. The function inputs are connected to the address lines and the function truth table is programmed into the memory array for each function output (PROM output data bit).

Mask programmed ROMs can achieve high speed and density because they are programmed by the manufacturer in the final semiconductor processing steps. These devices are analogous to Mask Programmed Gate Arrays. Field Programmable PROMs which are programmed by the end user are inherently slower and less dense because of the overhead associated with the internal programmable switch or fuse. Field Programmable devices offer the advantages of lower cost for low volume production and design turn-around times measured in minutes rather than weeks. The Erasable PROM (EPROM) and Electrically Erasable PROM (EEPROM) offer the additional advantage that they can be erased and reprogrammed many times. This can be attractive early in the design cycle, or when in-circuit re-programmability using EEPROMs is desirable.

Although PROMs can be used to construct simple combinatorial logic functions, their structure is much more efficient in creating computer memory. Programmable Logic Devices (PLDs) were developed to fill the need for chips which are optimized for logic circuit construction. The oldest of the PLD architectures is the Programmable Array Logic (PAL) design. These devices generally implement two-level sum-of-products Boolean expressions using a programmable AND-plane followed by a fixed OR-plane. Generic Logic Array (GAL) devices are marketed which also feature a programmable AND-plane. Figure 6 shows the internal array architecture of the 22V10, a typical PAL device. Each AND-OR term has a variable number of input terms which are selected from the input signal set during programming. The OR-gate output feeds an I/O cell which allows the output to be registered as well as providing feedback of the stored result into the AND-OR plane. A more flexible version of the PAL is the Programmable Logic Array (PLA) chip which has both programmable AND- and OR-planes.

PAL and PLD devices come with a wide range of numbers of inputs, outputs, and product terms. For example, the industry standard 22V10 PAL comes in a 24-pin package and has 12 inputs and 10 outputs with an average of 12 product terms per output. Typical CMOS 22V10 chips have operating speeds of 40mhz with typical input to combinational output delays of 15ns.[8]

Commercial PALs use EPROM, EEPROM, or fuse technology for end-user programming. Fusible link PALs employ a metal such as titanium tungsten or platinum silicide to form links which are one-time programmable. These links are blown by applying higher than normal operating voltages to create currents which exceed the capacity of the fuse material. Fusible link PALs or more common in Bipolar than CMOS technologies because high current capacity bipolar devices are more easily constructed.

TI Programmable Logic Databook pg. 2-160

Figure 6. Typical Programmable-AND Fixed-OR Structure of 22V10 PAL

EPROM and EEPROM PAL programming technologies are identical to those used in the corresponding memory technology. In an EPROM cell, a two-gate transistor structure is used rather than a simple MOS transistor. As shown in the circuit cross section of Figure 7, a floating gate is positioned between the regular MOS transistor control gate and the channel. In its normal (unprogrammed) state, the floating gate is uncharged and the channel can be turned off and on according to the voltage applied to the control gate. To program the cell, a high voltage (e.g. 14 volts) is applied to the control gate while the drain of the transistor is held at approximately 12 volts. This causes a large electric field and high current to flow between the source and drain. The high field in the drain depletion region accelerates electrons to high velocity and a small fraction of them traverse the thin oxide and become trapped on the floating gate. Because the floating gate is surrounded by an insulating layer, it becomes "permanently" negatively charged and the transistor is permanently turned off for all normal operating voltages. "Permanent" generally means about 10 years at 125 degrees C; at higher temperatures this time is reduced. All programmed cells can be erased by exposing the chip to Ultra-Violet (UV) light. The electrons stored on the floating gates are excited by the UV light and discharged to the substrate, resulting in a blank unprogrammed chip. EPROM chips are typically packaged in glass lidded packages to allow the UV light to illuminate the silicon. Vendors sometimes package EPROM chips in opaque plastic packages and market these parts as one-time-programmable units.

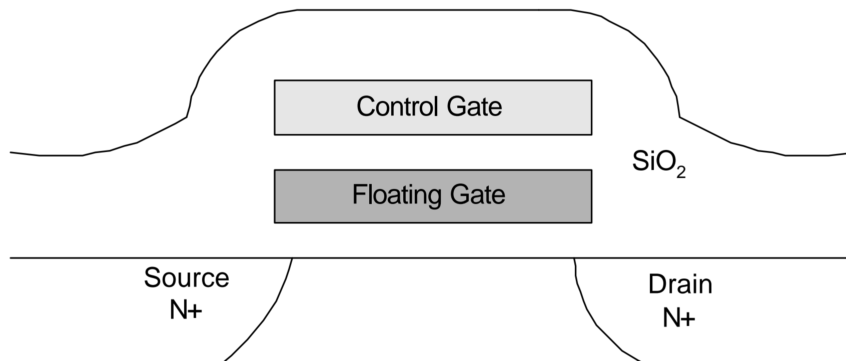

Figure 7. EPROM Floating Gate Cross Section

EEPROMs use a technology which is somewhat similar to EPROMs, except that these devices can be electrically erased and reprogrammed in-circuit. Like the EPROM cell, the

programmed transistor uses a floating gate structure with a control gate on top. Figure 8 shows a layout view of a typical EEPROM cell, and illustrates that a different physical structure is required for writing and erasing. In the "tunnel region" identified on the figure, the dielectric between the floating gate and the substrate is reduced to 100 Angstroms or less. When the programming voltage is applied across this thin film, electrons flow to the floating gate by the Fowler-Nordheim tunneling mechanism.[9] This tunneling mechanism is completely reversible, and the cell is electrically erased by simply reversing the voltage applied for writing. A disadvantage of EEPROM technology is that a cell consumes about twice the area of an EPROM cell; reducing the on-chip gate density.

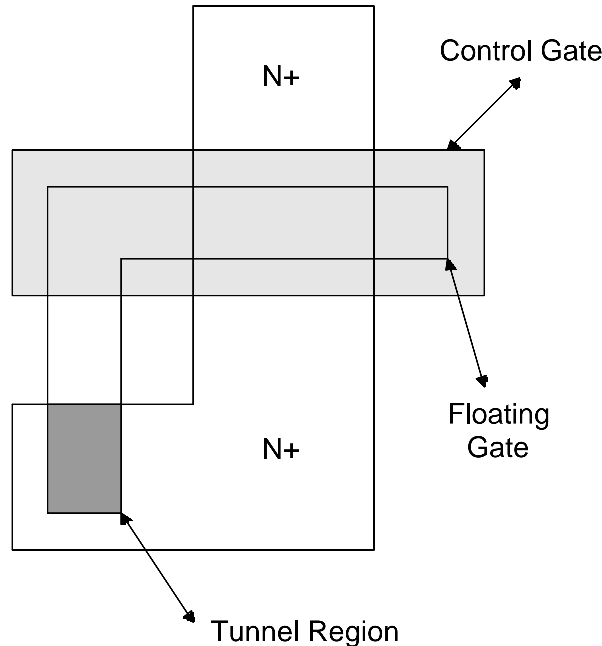

Figure 8. EEPROM Cell Layout View

PAL and PLD devices are available from a number of different commercial vendors in both CMOS and Bipolar technologies. Table 3 shows some of the commercial offerings and lists the product family name, basic cell type, programming technology, and semiconductor fabrication technology.[10]

Unlike programmable logic devices, Gate Array chips contain an array of completely uncommitted logic elements. Mask programmable gate arrays which are programmed at the silicon foundry will be discussed in Section IV. The following section describes Field Programmable Gate Arrays (FPGAs) which combine the programmability of a PAL or PLD with the flexible logic and interconnect structure of a mask programmed gate array.

Table 3. PAL and PLD Commercial Product Examples

Company	Product Family	Basic Cell Type	Programming Technology	Fabrication Technology
Advanced Micro Devices	CMOS PALS	Macrocell	EEPROM	CMOS
Advanced Micro Devices	Bipolar PALS	Macrocell	One-Time Programmable	CMOS
Altera	FLEXlogic	sum-of-products macrocell or 128 X 10 RAM	SRAM and Flash EPROM	CMOS
Altera	Classic	sum-of-products macrocell	UV-PROM	CMOS 0.65μm MAX 9000
Atmel	V750	22V10 sum-of-products style	UV-EPROM	0.65μm CMOS
Atmel	22V10 / 16V8 20V8	sum-of-products	Flash EEPROM	0.65μm CMOS
Cypress Semiconductor	PAL22V10 family	EPROM PLD	Flash EEPROM, EPROM	CMOS BiCMOS
Cypress Semiconductor	PALLE16V8	PLD	Flash EEPROM	CMOS
ICT	PEEL	SOP macrocell	EEPROM	CMOS
Lattice Semiconductor	ispGAL and GAL families	Programmable AND - Fixed OR w/ regs.	EEPROM isp: in-circuit prog.	CMOS
Philips Semiconductor	GAL family	macrocell	one-time prog., and reprogrammable	Bipolar, BiCMOS EPROM, EEPROM
Philips Semiconductor	PAL family	combinational & registered outputs	one-time programmable	Bipolar
Philips Semiconductor	Prog. Logic Arrays (PLAs)	bidirectional I/O or combinational outputs	one-time programmable	Bipolar
Philips Semiconductor	Prog. Macro Logic and Sequencers	JK, D registered and combinational I/O	one-time prog. and UV-EPROM	Bipolar and EPROM CMOS
Texas Instruments	PLDs and Field Prog. Sequencers	PLD	TiW fuses	Bipolar
Xilinx	XC7200A EPLD	21-input, 9-output sum-of-products macrocell w/ ALU	EPROM	Submicron CMOS
Xilinx	XC7300 EPLD	9 macro-cell blocks	EPROM	Submicron CMOS

B. Field Programmable Gate Arrays

Field Programmable Gate Arrays are a relatively new class of integrated circuit; first introduced by the Xilinx company in 1985.[11] Since that time, the FPGA market has expanded dramatically with many different competing designs developed by companies including, Actel, Advanced Micro Devices, Algotronix , Altera, Atmel, AT&T, Crosspoint Solutions, Cypress, Intel, Lattice, Motorola, QuickLogic, and Texas Instruments. A generic FPGA consists of an array of logic elements together with an interconnect network which can be configured by the user at the point of application. User programming specifies both the logic function of each block and the connections between the blocks. Programming can take two forms; one-time programmable chips (analogous to fusible link PROMs), and reprogrammable chips (analogous to EEPROMs or Static RAMs).

In one sense, FPGAs represent an evolutionary improvement in gate array technology which offers potential reductions in prototype system costs and product time-to-market. However, recent applications of FPGA technology suggest their impact on electronic systems may be much more profound. Consider the fact that reprogrammable FPGAs are capable of dynamically changing their logic and interconnect structure to adapt to changing system requirements. This offers a new computing paradigm which blurs the traditional lines between hardware and software. For example, traditional digital hardware simulation is performed by a computer program which simulates operation of the target hardware on an event-by-event basis. Reprogrammable FPGA technology provides the opportunity to take the target design, download it into an array of FPGA chips, and actually emulate the target hardware at only a slightly reduced clock rate. This method offers orders of magnitude reduction in simulation time. As another example, one computer is often used to emulate another using software that translates the native code of the target processor to that of the host. Using FPGA technology, the internal host processor organization can actually be re-mapped to correspond to that of the target; again providing orders of magnitude speedup. The potential for future products whose actual hardware configuration can be specified (and changed) by the user in the end-product environment is virtually limitless.

In order to understand the opportunities and limitations of FPGA technology, a detailed understanding of the programming technology alternatives and the internal logic and interconnect architectural styles is needed. The following two sub-sections provide this information.

1. FPGA Programming Technologies

Before describing the features of alternative FPGA architectures, it is necessary to gain an understanding of the technologies which allow these devices to be "field-programmable". Programming technologies used in today's commercial FPGA products include EPROM and EEPROM transistors, anti-fuses, and Static Random Access Memory (SRAM) cells. The selection of a programming technology is important because it affects chip density, signal and logic propagation delays, and chip power consumption. Like PALs, FPGAs can be programmed by changing the characteristics of a switching element using EPROM and EEPROM programming

technologies. These techniques are identical to those described for PALs and PLDs in the previous section, and will not be discussed further here.

An alternative method to personalize FPGA devices is to physically program the routing connections. Commercial products which use programmable routing approaches have been introduced by Actel, QuickLogic, and others. In Actel FPGAs, a Programmable Low-Impedance Circuit Element (PLICE™) anti-fuse element is used.[12] The anti-fuse resistance which is normally very high (>100MΩ) is permanently changed to a low resistance (200-500Ω) by the application of appropriate programming voltages. The programmed anti-fuse is used to make a direct electrical connection between two metal lines as shown in the layout view of Figure 9. The PLICE anti-fuse is manufactured by adding three specialized masks to a standard CMOS process. The physical structure, identified at the center of Figure 9, consists of an Oxide-Nitride-Oxide dielectric layer sandwiched between a top polysilicon layer and a bottom N+ diffusion layer. Programming is accomplished by applying a relatively high voltage (18V) across the device and driving a high current through the link dielectric. This causes the dielectric to melt and results in a conductive link between the top and bottom terminals.

QuickLogic also adds a unique three layer structure to the standard CMOS process to create their anti-fuse element; which they call a ViaLink™.[13] The ViaLink uses an amorphous silicon layer which is sandwiched between the first and second metal layers. An unprogrammed ViaLink has greater than 1GΩ resistance and, like the PLICE anti-fuse, is programmed by applying a higher than normal voltage. The resulting high current through the amorphous layer causes it to permanently change to a conductive state with a typical resistance of only 80Ω. The area occupied by these anti-fuse elements is very small when compared to the other programming alternatives. While this contributes to improved on-chip gate density, it is somewhat offset by the large area required for the high-voltage transistors needed to support programming. An additional disadvantage of the anti-fuse technologies is that they require modifications to the standard CMOS process.

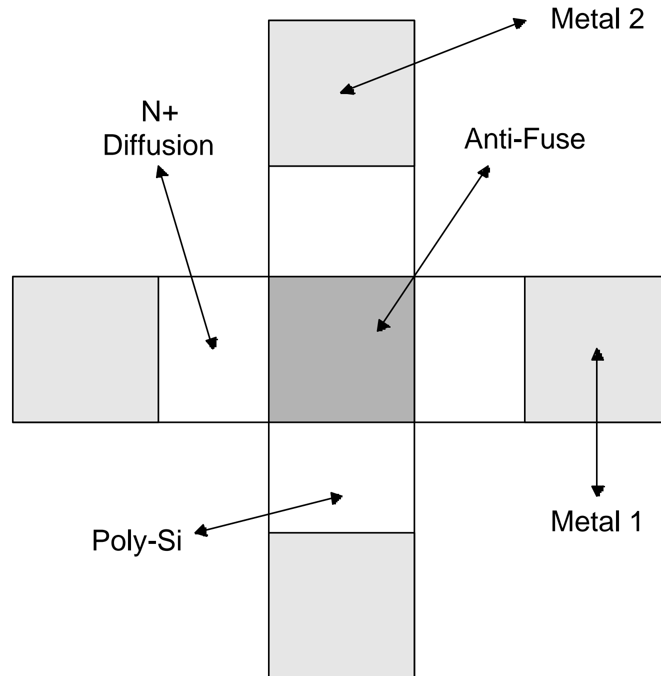

Figure 9. PLICEÔ Anti-fuse Programmable Interconnect

The Static Random Access Memory (SRAM) FPGA programming technology which was first introduced by Xilinx is also used in designs by Algotronix, Plessey, AT&T, and others. Programmable connections in these FPGAs are made using multiplexers, transmission gates, or pass transistors that are controlled by information stored in SRAM cells. Since the static RAM is volatile, these FPGAs must be programmed to set the circuit configuration each time that power is applied to the chip. This can be accomplished automatically through a serial connection to an attached ROM (PROM) or controller; or in parallel by an attached processor or controller which address the FPGA in parallel as a normal static RAM. The chip area required by the SRAM logic and interconnect programming circuitry is the largest of the FPGA technologies described here. A typical SRAM cell uses 5 or 6 transistors, and additional devices are needed for the transmission gates or multiplexers. However, these devices are produced using standard CMOS SRAM fabrication techniques, and thus can immediately benefit from advances in SRAM CMOS processes.

The non-volatile nature of the SRAM programming can be viewed alternatively as either a disadvantage or as a major advantage. Programming imposes a system level overhead in terms of ROM storage and power-on initialization time. The fact that additional programming power-supply voltages are not required is an advantage for SRAM-based devices. However, the greatest system level benefit of this technology is that SRAM FPGAs can be re-programmed in-circuit to make system modifications and upgrades; or to perform multiple functions with the same base hardware.

Table 4 provides a summary of the characteristics of the four FPGA programming styles just discussed. Exact comparisons by class are difficult since different vendors products are fabricated using different CMOS processes. Thus, the cell area and the resistance and capacitance values for the on-state of the programmable elements are only meant for relative comparison.

Table 4. Summary of FPGA Programming Characteristics

Programming Technology	Re-Programmable	Volatile Storage	Series Resistance	Capacitance in ff.	Cell Area
Static RAM	in-circuit	yes	1KΩ	15	5X
Anti-Fuse	no	no	50-500Ω	1.2-5.0	1X
EPROM	outside circuit	no	2KΩ	10	1X
EEPROM	in-circuit	no	2KΩ	10	2X

2. FPGA Architectures

FPGAs are commercially available in many different architectures and organizations. Although each company's offerings have unique characteristics, FPGA architectures can be generically classified into one of four categories: Symmetrical Array, Row Based, Hierarchical PLD, and Sea of Gates.[4] Figure 10 illustrates this classification based on the general internal organization of the design. In the interest of space, only a single vendor for each design style will be described in detail in the following four subsections. However, this does not indicate any preference or endorsement for these particular product offerings. Table 5 lists examples of the extensive range of choices in commercial FPGA products.[10]

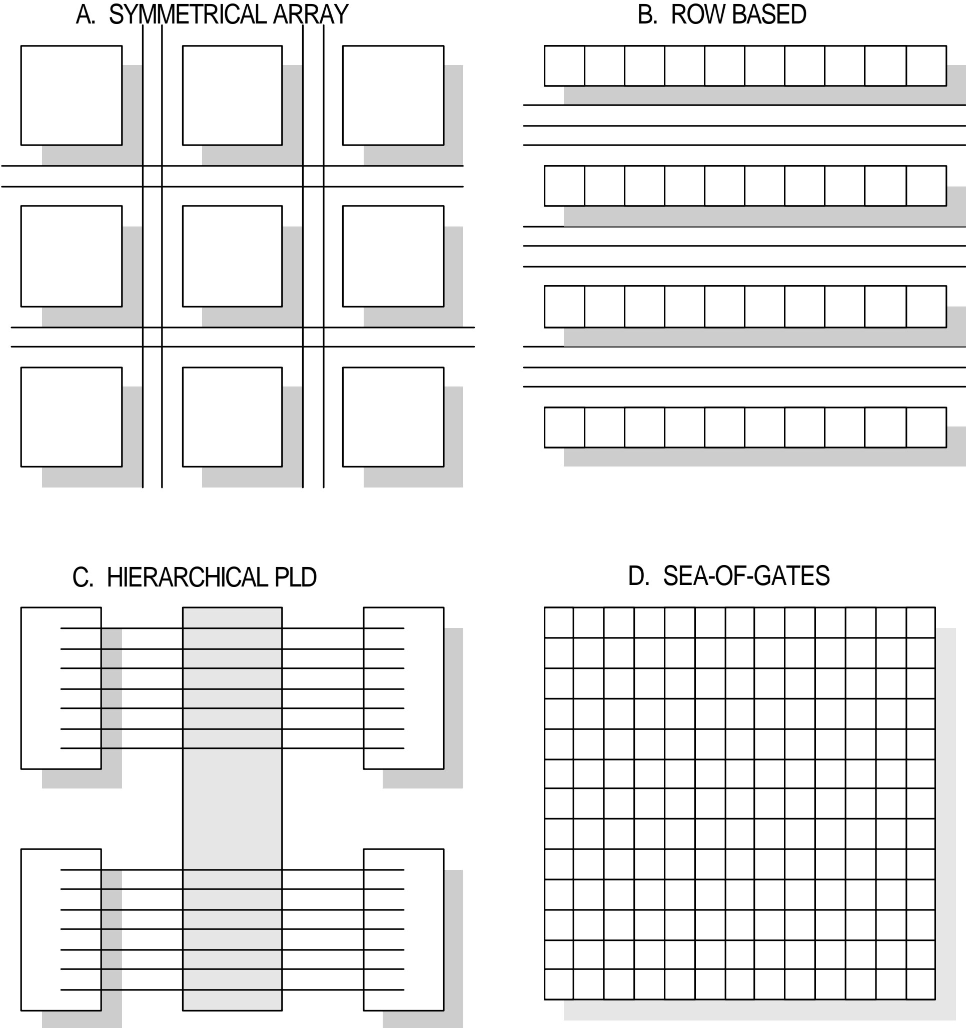

Figure 10. The Four FPGA Architectural Classes

Table 5. Commercial Field Programmable Gate Array Products

Company	Product Family	Basic Cell Type	Programming Technology	Fabrication Technology
Actel	ACT-1, ACT-2	4-or 5-input comb. / sequential functions	One-Time Prog. Antifuse	1.0μm CMOS
Actel	ACT-3	5-input comb. / sequential functions	One-Time Prog. Antifuse	0.8μm CMOS
Advanced Micro Devices	MACH 1xx/2xx, MACH 3xx/4xx	EEPROM	Electrically erasable and prog. (3xx/4xx in-circuit prog.)	0.65-0.8μm CMOS
Altera	MAX 5000	sum-of-products macrocell	EPROM	CMOS
Altera	MAX 7000, MAX 9000	sum-of-products macrocell	EEPROM	CMOS 0.65μm MAX 9000
Atmel	V5000	20-prod. & 3-sum terms, 2 DFF	UV-EPROM	0.8μm CMOS
Atmel	V25000	13-prod. & 3-sum terms, 2 DFF	UV-EPROM	0.65μm CMOS
AT&T	ATT3000	Configurable Logic Block	Reprogrammable SRAM	0.6μm CMOS
AT&T	ORCA	Programmable Function Unit	Reprogrammable SRAM	0.5-0.6μm CMOS
Crosspoint Solutions	CP20K	Transistor-Pairs	One-Time Prog. Antifuse	0.8μm CMOS
Cypress Semiconducto r	MAX340, CY7C335	EPROM PLD	UV-EPROM	0.8μm CMOS
Cypress Semiconducto r	FLASH370	PLD	Flash EEPROM	0.8μm CMOS
Cypress Semiconducto r	pASIC380	metal-to-metal antifuse	One-Time Prog. Antifuse	CMOS
Lattice Semiconducto r	ispLSI and pLSI 1000, 3000, & 3000 families	Generic Logic Block PLA with prog. registers	EEPROM isp line is in-circuit programmable	CMOS
Motorola Semiconducto r	MPA1000	2-input NAND / DFF W-OR, XOR	Reprogrammable SRAM	0.8μm CMOS
QuickLogic	pASIC1	14-in, 5-out Mux-based cell with FF	One-Time Prog. Antifuse	0.8μm CMOS
Texas Instruments	TPC12 TPC10	Comb. & Sequential Logic Modules	One-Time Prog. Antifuse	1.2-1.0μm CMOS
Texas Instruments	PLDs and Field Prog. Sequencers	PLD	TiW fuses	Bipolar
Xilinx	XC2000, XC3000 XC4000, XC5000	Config. Logic Blocks w/ look-up table, ff(s).	Reprogrammable SRAM	Submicron CMOS

a) Symmetrical Array FPGA Example: Xilinx

Figure 11 shows a placed and wired Xilinx XC4003 chip design, and illustrates how the general architecture of a Xilinx FPGA resembles the symmetrical array FPGA architecture presented earlier in Figure 10.A. The Configurable Logic Blocks (CLBs) are organized in a two-dimensional array separated by horizontal and vertical wiring channels. Each CLB contains flip-flop(s), multiplexers, and a combinatorial function block which operates as an SRAM based table look-up. Connections between CLBs are personalized by turning on pass transistors which selectively connect the CLBs to the interconnection resources, or interconnect lines between the horizontal and vertical wiring channels. SRAM cells which are distributed around the chip hold the state of the interconnect switches. Surrounding the CLB array and interconnect channels are the programmable I/O blocks which connect to the package pins.

Figure 11. Xilinx XC4003 Logic Cell Array (LCA)

There are four families of Xilinx SRAM based FPGAs (XC2000, XC3000, XC4000, and XC5000) corresponding to their first through fourth generation designs. Table 1 indicates the characteristics of each family in terms of numbers of I/O, numbers of flip-flops, numbers of CLBs, and estimated number of usable 2-input gates per chip. Note that the XC5000 family was just introduced in 4th quarter 1994 with three initial product offerings.[14]

Table 6. Xilinx FPGA Family Characteristics.

Family Series	Number of I/O Blocks	Number of CLBs	Number of FFs	Estimated Typical Usable Gates
XC2000	58-74	64-100	122-174	800-1,800
XC3000	64-176	64-484	256-1,320	1,300-9,000
XC4000	64-256	64-1,024	256-2,569	1,600-25,000
XC5000	148-244	196-484	784-1,936	6,000-15,000

The design of the Xilinx CLB and routing architecture is slightly different for each product family. The first generation XC2000 family contains a CLB with a single D flip-flop and a look-

up table capable of generating any Boolean function of four variables, or two functions of three variables. The routing architecture uses three resource types: direct connection, general purpose interconnect, and longlines. Direct connection lines are used to interconnect a CLB with adjacent CLBs or I/O blocks either above, below, or to the right. General purpose interconnects are used for connections which span more than one CLB. There are four horizontal and five vertical general purpose interconnect lines between the array rows and columns, respectively. Each segment runs only the length of a CLB, then enters a switch matrix which provides programmable connections to adjoining row or column general purpose interconnects. Finally, each horizontal wiring channel has one longline and each vertical wiring channel has two long lines which span the entire array. These longlines bypass the switch matrices and are intended for global signals (e.g. clocks), or other signals which must have minimum skew at multiple fan-out points.

In the second generation XC3000 family the logic block (CLB) is expanded and additional routing resources are provided. The CLB can implement any Boolean function of five variables or two functions of four variables. Two D-type flip-flops are now provided to latch both cell outputs if required. The routing architecture is similar to the XC2000 family except that each resource type has been enhanced: direct connections are now permitted to all nearest neighbors, an extra wiring segment is added to the horizontal general purpose interconnect, and an additional longline is added to both the horizontal and vertical channels.

Compared to its predecessors, the XC4000 family adds another level of evolutionary improvements to the basic Xilinx architecture. Greater logic capacity per CLB is achieved using a two-level look-up table as illustrated in Figure 12. The 13 input and four output CLB can generate any of the following combinatorial logic functions: two independent functions of up to four variables, any single function of five variables, any function of four variables together with some functions of five variables, or some functions of up to 9 variables. Compared to earlier families, the routing resources of the XC4000 family have been more than doubled. The number of globally distributed signals has increased from two to eight, and there are twice as many horizontal and vertical long lines. The number of wiring segments has also more than doubled, and CLB connectivity is improved by allowing most CLB pins to connect to a high percentage of the wiring segments. However, the switch matrix connectivity was reduced to 50% of that of the XC3000 family. Justification for these changes in routing resources is supported by the research of Rose and Brown.[15] They concluded from place and route experiments with multiple designs that FPGA connection blocks need high flexibility to achieve a high percentage of routing completion, and that relatively low flexibility is needed in the switch blocks.

Figure 13, page 2-21, 1994 Xilinx Data Book

Figure 12. Xilinx XC4000 Family Configurable Logic Block (CLB)

The XC5000 architecture, while still a symmetrical array with SRAM based programmable logic and interconnections, represents a more significant departure from the earlier family members. Although pin-for-pin and programming/control interface compatibility is maintained with the XC4000 family, the internal chip organization has been dramatically changed. As indicated in Figure 13, the logic blocks and their local routing connections have been combined into a larger entity called a VersaBlock™. The VersaBlocks provide logic and connectivity for efficient assembly of local logic functions. These local functions are then globally interconnected through a General Routing Matrix (GRM). This architecture provides five levels of interconnect hierarchy, which can be used to efficiently exploit the locality of logic in typical digital designs. Heavily interconnected logic macro-functions placed in adjacent CLBs can be locally connected within the VersaBlock. This allows the GRM resources to be devoted to connections between macro-function blocks.

Figure 1, page 13, XCELL 4th quarter 1994 newsletter

Figure 13. Xilinx XC5000 Array Architecture

As shown in Figure 14, the VersaBlock contains a CLB constructed from four separate logic cells (LC0-LC3), together with a local interconnect matrix. Note that each of the four logic cells within the XC5000 CLB is similar in structure to the original XC2000 family CLB; with a single D flip-flop and a four variable Boolean function generator. However, the grouping of four of these independent logic cells in a tightly coupled VersaBlock unit allows efficient high speed carry chains or high fan-in logic functions to be easily created.

Combine Figures 2 & 3, pg 16, XCELL 4th Quarter 1994 Newsletter

Figure 14. XC5000 VersaBlock and CLB Structure

b) Row Based FPGA Example: Actel

In the Actel ACT™ family FPGAs a logic module matrix is arranged as rows of cells separated by horizontal wiring channels, as previously illustrated in Figure 10.B. This organization is similar to that found in the traditional style of Mask Programmed Gate Arrays (MPGAs). Vertical interconnect segments of varying lengths are available. Vertical segments in input tracks are permanently connected to logic module inputs, and vertical segments in output tracks are permanently connected to logic module outputs. Long vertical segments are available which are uncommitted and can be assigned during routing. The horizontal wiring channel resources are also segmented into varying lengths. The minimum horizontal segment length is the width of a single logic module, and the maximum horizontal segment length spans the full channel. Any segment that spans more than one-third of the row length is considered a "long horizontal segment". Connections between interconnect segments are permanently formed using the PLICE™ antifuse which was described earlier in this chapter. Dedicated routing tracks are used for global clock distribution and for power and ground tie-off connections. Actel has three generations of FPGAs, denoted ACT1, ACT2, and ACT3[16], whose characteristics are listed in Table 7.

Table 7. ACTEL FPGA Family Characteristics

Family Series	Number of User I/O	Number of Logic Modules	Number of FFs	Typical Gate-Array Equivalent Gates
ACT-1	57-69	295-547	147-273	1,200-2,000
ACT-2	83-140	451-1,232	382-998	2,500-8,000
ACT-3	80-228	200-1,377	264-1,153	1,500-10,000

In contrast to the Xilinx FPGA which uses a relatively complex CLB cell, the Actel approach uses small and simple logic modules. This does not imply that the Actel design has inherent disadvantages compared to the Xilinx approach. Research by Singh et. al. has shown that both of these approaches have merit.[17] Their results indicate that the best choice for a programmable block depends on the speed performance and the area requirements of the routing architecture. The low-impedance and small area Actel antifuse structure is better suited for use with a simple logic module; whereas the larger area and higher resistance Xilinx SRAM controlled transistor switch is more appropriate for a complex logic cell.

The ACT1 family Logic Module (LM) is an 8-input, one output function which can be used to construct the four primitive logic function (AND, OR, NAND, NOR) with two through four inputs. Figure 15 shows how the basic ACT1 Logic Module circuit uses multiplexers to create programmable logic functions. The LMs can also be used to construct latches, flip-flops, XORs, AND-ORs and other logic structures. Actel does not include dedicated hardwired latches or flip-flops in the ACT1 array since they can be constructed from LMs wherever needed in the design. The ACT1 family uses 22 metal signal wiring tracks in each horizontal channel and 13 vertical tracks that lie on top of each column of LMs.

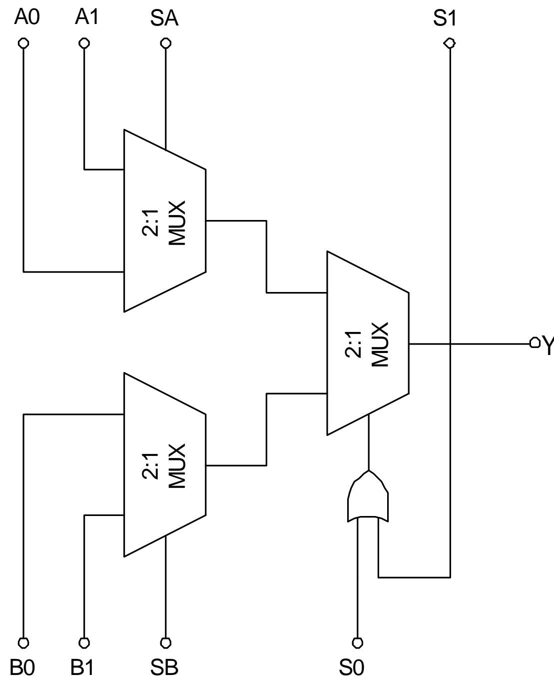

Figure 15. Actel ACT1 Logic Module (LM)

The ACT2 family is Actel's second generation of FPGAs. It uses two different types of logic modules; a Combinational (C) Module and a Sequential (S) Module. The C module with eight inputs and one output is similar in functionality to the LM used in the ACT1 family. The S-Module is designed to efficiently construct high-speed D flip-flops or latches within a single cell. An S-module can create an up to 7-input Boolean function followed by a D-type flip-flop or a latch. The S-Module can also be configured with a transparent latch so that like the C-Module it can also implement a purely combinatorial 8-input function. C-Module and S-Modules are paired and then grouped together in alternating pairs to construct the rows of the ACT2 array. The ACT2 routing structure is also similar to that of ACT-1, with the same four types of routing resources: vertical input and output segments, clock tracks, and horizontal wiring tracks. However, there are 14 additional tracks in each horizontal wiring channel and 2 additional tracks in each vertical column.

ACT3 is Actel's third generation FPGA family which uses the same basic array architecture with enhanced versions of the ACT2 family logic modules. The new C-Module is functionally equivalent to that of ACT2, while the S-Module has been expanded to include a full C-Module driving a flip-flop. The ACT3 architecture contains four clock networks; two dedicated high-performance clock networks, and two general purpose networks. The ACT3 architecture continues to use the routing resource structure of the ACT2 design with horizontal wiring channels and vertical wiring tracks which overlay the logic modules.

c) Hierarchical PLD Example: Altera

The Altera Multiple Array MatriX (MAX) architecture is quite different from those discussed previously in this chapter. This architecture represents a hierarchical arrangement of Erasable Programmable Logic Devices (EPLDs) using a two-dimensional array structure. The design provides multiple level logic, uses a programmable routing structure, and is user re-programmable based on EPROM or EEPROM technology. Thus, although these devices are marketed by Altera as EPLDs, they clearly meet the criteria to be called FPGAs in the context of this chapter.

The MAX 5000 series and second generation MAX 7000 series architectures consist of an array of large programmable blocks, called Logic Array Blocks (LABs).[18] [19] Each LAB in the MAX 7000 family is constructed from 16 macrocells. Each macrocell in turn has a programmable-AND / fixed-OR array and a configurable register. Thus, each macrocell represents a small PLD with five programmable product terms, and it can be configured for either sequential or combinatorial operation. Complex logic functions can be constructed using multiple macrocells. In addition, the Altera LAB architecture provides both shareable and parallel expander product terms ("expanders") that can be used to create additional product terms directly to any macrocell in the same LAB. Finally, at the top level of the design hierarchy, signals are routed between LABs by a Programmable Interconnect Array (PIA). This global routing resource connects any signal source to any destination on the chip. Figure 16 shows a portion of the MAX 7000 architecture; and illustrates the Programmable Interconnect Array (PIA), Logic Array Block (LAB) with its constituent macrocells, and I/O Control (IOC) block.

figure 1 from MAX 7000 data book........

Figure 16. Altera MAX 7000 Internal Architecture

Table 8 lists a summary of the characteristics of the first and second generation MAX 5000 and MAX 7000 families. The FLEX 8000 series Altera design listed in the table departs from the earlier families in its use of SRAM programming technology.[20] This series continues Altera's fine-grain hierarchical architecture by using 4-input look-up table Logic Elements (LEs) as the basic functional building block. LE's are grouped into sets of eight to create LABs as in the earlier family designs; and these blocks are arranged into rows and columns. Connections between LEs are provided by horizontal and vertical FastTrack interconnect channels which span the chip. Both the Logic Elements and the FastTrack interconnects are SRAM programmed in an analogous fashion to the Xilinx technology discussed earlier. The MAX 9000 family identified in the table is a third generation Multiple Array MatriX architecture which has features of both the MAX 7000 and FLEX 7000 families.[21] It uses EEPROM non-volatile programming, and logic is

hierarchically constructed from macrocells which are grouped into LABs as in the MAX 7000 family. However, the routing architecture of the MAX 9000 family uses the FastTrack technology introduced in the FLEX 8000 series parts. A special feature of the MAX 9000 family is that the parts are 5.0 volt in-circuit programmable through the JTAG (Joint Test Action Group) Boundary Scan Standard interface.

Table 8. Altera EPLD (FPGA) Family Characteristics

Family Series	Technology	Number of User I/O	Macrocells	Flip-Flops	Usable Gates
MAX 5000	EPROM	16-64	32-192	--	600-3,700
MAX 7000	EPROM	36-164	32-256	--	600-5,000
MAX 9000	EPROM	56-216	320-560	484-772	6,000-12,000
FLEX 8000	SRAM	78-208	208-1296 (Logic Elements)	282-1,500	2,500-16,000

The FastTrack interconnect technology used in the MAX 9000 and FLEX 8000 parts is different from those previously described in this section. Figure 17 shows four adjacent LABs with row and column interconnect; and illustrates how logic device and interconnect resources are arranged in both these FPGA families. The LABs are arranged into a two-dimensional array separated by horizontal and vertical FastTrack wiring channels which span the entire array. An advantage of this device wide routing is that it provides predictable wiring delays when compared to segmented FPGA wiring schemes which use a variable number of programmable interconnection points in the routing path.

figure 6 page 12 from MAX 9000 data book

Figure 17. Altera FastTrack Interconnect Architecture

Each column of LABs has dedicated lines that route signals out of the LABs and into the FastTrack column. The column interconnect can then drive I/O pins or feed into the row interconnect to drive other LABs. The number of wiring channel routing resources vary by family and part type. In the MAX 9000 family there are 96 channels per row and 48 channels per column, while the FLEX 8000 family parts have either 168 or 216 channels per row and 16 channels per

column. Figure 18 illustrates the MAX 9000 FastTrack Row and Column interconnect structure. Each row of LABs has a dedicated row interconnect for routing macrocell inputs and outputs. The row interconnect can then drive I/O pins or feed other LABs on the chip. Each macrocell in the LAB can drive up to three separate column interconnect channels. A row interconnect channel can be fed by the output of a macrocell through a 4-to-1 multiplexer that it shares with three column channels. As indicated in the figure, if the 4-to-1 mux is used for a macrocell-to-row connection, then the three column signals can access another row channel via an additional 3-to-1 multiplexer.

fig. 7 in page 13 in MAX 9000 data sheet.

Figure 18. LAB Connection to Row and Column Interconnect

d) Sea of Gates FPGA Example: Algotronix

Although the Algotronics company was purchased in 1994 and no longer markets their CAL1024 (Configurable Array Logic) product, a few details of the design are presented here to illustrate this generic FPGA style. As illustrated in Figure 19 which shows one sixteenth of the CAL1024 chip, the design has a two-dimensional mesh array structure which resembles the gate array "sea of gates" architecture previously identified in Figure 10.D.[22] Like the Xilinx architecture, Algotronics used Static RAM programming technology to specify the function performed by each logic cell and to control the switching of connections between cells. The CAL1024 design contains 1024 identical logic cells arranged in a 32 X 32 matrix. The design is considered to be a mesh-connected architecture since each cell is directly connected to its nearest north, south, east, and west neighbors. In addition to these direct connects, two global interconnect signals are routed to each cell to distribute clock and other "low skew requirement" control signals. Figure 19 shows the basic array architecture, indicating both nearest neighbor and global connections to the logic cells. In addition to these logical connections, row select lines and bit select lines which are not shown on the figure are connected to program each cell's SRAM bits.

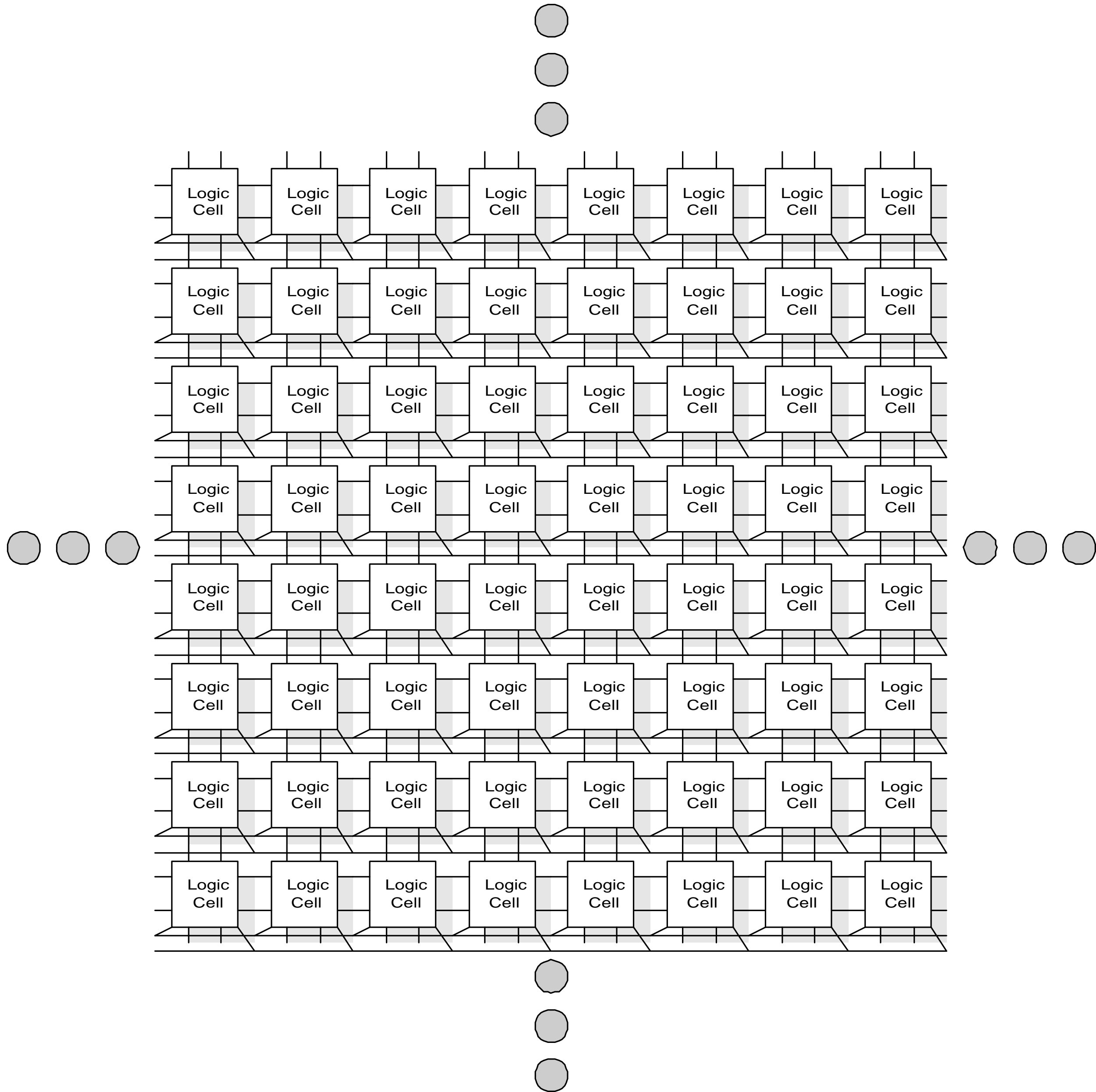

Figure 19. Algotronix Array Architecture

The basic building block of the Algotronix design is a configurable cell containing multiplexers and a function unit. As indicated in Figure 20, the function unit is preceded by multiplexers which select the source for the X1 and X2 inputs.[4] The function unit is capable of generating any logic function of the two inputs, or of operating as a D-type latch. Not shown in the figure are four additional multiplexers which select the function output or one of the external inputs for routing to each of the four outputs (north, south, east, and west).

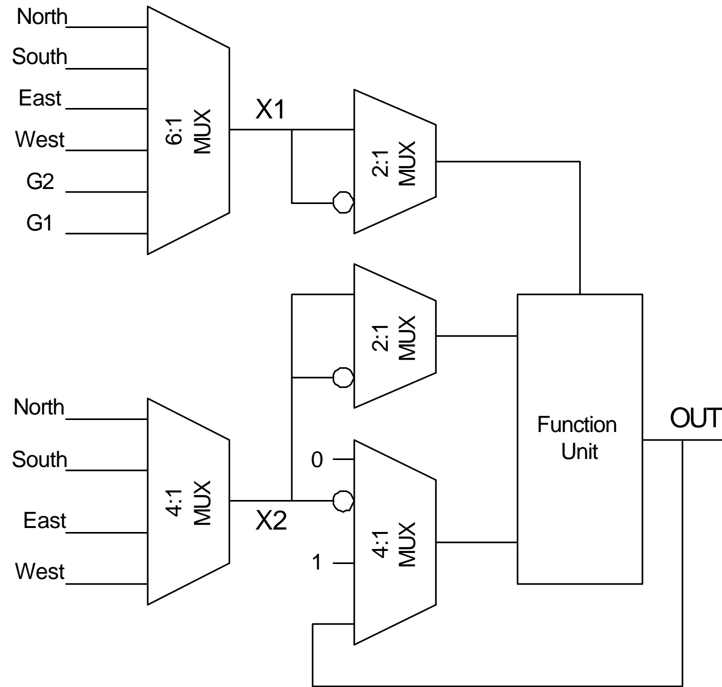

Figure 20. Algotronix Logic Cell Function Unit Design

A unique feature in the Algotronix I/O pad design is its capability to provide simultaneous input and output on the same pin when communicating with another Algotronix chip. This is done through a 3-level (ternary) logic signaling scheme in which I/O pads sense whenever two outputs are driving each other via a contention scheme. Even during contention, the pad can deduce the correct input value and pass it along to the internal circuitry. This makes it easier to partition a single design across multiple FPGAs because the increased connectivity reduces pin limitations on communications bandwidth.

3. FPGA Design Flow

Although early PLD and FPGA designs were generated largely by hand, access to today's complex programmable logic devices requires the use of an integrated Computer-Aided Design (CAD) system. Figure 21 illustrates the typical sequence of operations needed to go from concept to programmed chip. Both commercial CAD tool vendors and FPGA companies offer appropriate tools. For example, traditional Electronic Design Automation (EDA) vendors such as Cadence, Mentor Graphics, Synopsys, and ViewLogic all offer tools to support FPGA design. These tools are typically used for the front-end design entry and simulation operations and provide the necessary interfaces to vendor-specific back-end tools for chip placement and routing.

Examples of vendor specific tools are the Xilinx XACT system and the Altera MAX+PLUS II software. It is worth noting that Altera's MAX+PLUS II software supports the entire design flow illustrated in Figure 21 on either PC or workstation platforms. A detailed discussion of available FPGA CAD tools is outside the scope of this chapter. Rather, the following discussion is meant to be indicative of the general operations and steps required in FPGA design. Where appropriate, examples are taken from the Xilinx and Altera CAD design flows to illustrate the generic operations.

Figure 21. Typical CAD System Design Flow for FPGAs

The starting point in any logic or digital system design is a set of architectural or behavioral specifications. Traditionally, a designer uses schematic capture tools for graphical entry of a logic design which has been manually generated to meet the architectural or behavioral specifications. The upper left hand arrow in Figure 21 identifies some of the commercial CAD tools available for FPGA schematic capture. One of the more significant recent innovations in the EDA industry is the development of tools which allow the designer to move from the gate level to the behavioral level for design entry. A behavioral design specification is created using a Hardware Description Language (HDL), and then a synthesis tool automatically compiles the gate level schematic or netlist from the behavioral description. The upper right hand arrow in Figure 21 indicates some of the HDLs currently being used for FPGA behavioral modeling. Section VII will present a more detailed discussion of behavioral modeling and logic synthesis.

Options for behavioral description of designs include the VHSIC Hardware Description Language (VHDL), the Verilog hardware description language, timing diagrams, logic state diagrams, and PLD description languages such as ABEL. As an example of how pervasive the behavioral design style has become, the PC-based Altera MAX+PLUS II software provides multiple options for behavioral design entry. In addition to traditional schematic capture it will accept VHDL, text design description in the Altera Hardware Description Language (including truth tables and Boolean expressions), and Timing Diagrams which describe the desired input and output waveforms. Whichever behavioral design entry method is chosen, the design system provides logic synthesis which automatically creates gate-level schematics.

No matter what method is used for initial design entry, the next step in FPGA design is to translate the entire design into a standard form which can be processed by a logic optimization tool. The goal of logic optimization is to perform minimization of the Boolean expressions and eliminate redundancy, thus minimizing the area of the final circuit. The tool may also be constrained to maximize speed at the expense of area by limiting the number of logic levels between clocked registers. This optimization process is usually merged with the logic synthesis step when behavioral design entry is employed. Simulation is performed both before and after the logic optimization steps to verify that the design meets the original system requirements for functionality and timing. The next step is to convert the generic gate level design into one which uses the FPGA circuit building blocks of the target technology.

To provide a concrete example, the Xilinx XACT design system flow will be used to illustrate the steps needed to go from logic design to programmed FPGA. In the Xilinx design flow, the native format of the logic design (Cadence, ViewLogic, OrCAD, etc.) must first be translated into the Xilinx Netlist Format (XNF) which is understood by the Xilinx tools. Next, the XNF circuit description must be mapped into Xilinx Configurable Logic Blocks (CLBs). This is the technology mapping step referred to in Figure 21. Xilinx calls this step "partitioning", and the XACT tools also attempt to optimize the circuit during this step. For example, circuitry associated with unused logic block inputs or outputs is eliminated from the design. In addition, the partitioning program attempts to minimize either the total number of CLBs used or the number of logic stages in the critical delay path.

The next step is to place and route the design on the selected chip image. The XACT systems allows manual and/or automatic placement and routing. In the automatic placement

operation, each CLB generated during the "partitioning" step is assigned to a physical location on the chip. Xilinx uses a Simulated Annealing algorithm which starts with a random placement, and then goes through a series of improvement passes. This program can be run multiple times with different starting random seeds in an attempt to generate a more optimal placement. Following placement, interconnections between the CLBs must be routed using the available interconnect segments and switch matrix elements. XACT uses an automatic Maze Routing Algorithm to perform this operation. With the physical placement and routing completed, exact timing values can now be used to determine chip performance. The XACT tools provide a critical path timing analyzer which provides delay information on the longest through shortest paths through the chip. In addition, the physical layout timing information can also be back-annotated to the schematics to get more accurate functional simulation results. The final step in the Xilinx design flow is the creation of the BIT file which contains the binary programming data needed to configure the SRAM bits of the target chip. This file is then downloaded to configure the chip for final functional and timing tests of the programmed chip.

4. Summary of FPGA Selection Criteria

The most common applications of FPGAs are for the prototyping of designs which will eventually be implemented in a high-volume ASIC technology (e.g. gate array or standard cell), or for final use in low production volume products. As previously mentioned, approximately one half of all of today's chip design projects are begun using FPGAs. The programmable nature of these devices create several system level design and manufacturing issues which must be considered when selecting a particular FPGA vendor and programming technology.

a) Manufacturing and Reliability Issues

Although SRAM based FPGAs present no special manufacturing difficulties, there are several concerns related to one-time programmable chips. The first issue with programming is that it is not 100% reliable. For example, fuse-based devices cannot be completely tested by the manufacturer so a higher than normal post-programming component fallout rate may be expected. As with any programmable technology, adequate post-programming test is essential and must be included in the product manufacturing schedule. In the same vein, programming adds an additional manufacturing step which increases production time, expense, and tooling; and proper programmer calibration is essential for low component fallout. Finally, one-time programmable FPGAs are frequently socketed to permit system design revisions and field upgrades by changing the chip. Sockets can add significant cost and size to a product, and may reduce the reliability of the final product. Note that these issues are of prime importance only in very high volume applications which are not the mainstay of FPGAs.

Reliability concerns have been raised in the past as arguments against the use of FPGAs. For example, soft errors in SRAM technology devices are known to be caused by alpha-particle radiation. Although not as severe as in Dynamic RAMs (DRAMs), the soft-error rate in SRAMs

is expected to go up as improved processing technology allows smaller geometry transistors which are more susceptible to transient upset. However, these problems are less likely to occur in SRAM based FPGAs than SRAM memory. FPGA storage cells typically have much higher load capacitance making them more immune to transient upset, and they typically have lower impedance power connections. Also, packaging can affect soft-errors since chips in plastic packages have nearly ten times lower upset rates than ceramic packaged parts. EPROM and fuse-based non-volatile FPGA technologies are sensitive to the calibration of the programmer. High field return rates are frequently based on failure to meet the manufacturers programming specifications or failure to perform adequate post-programming testing. EPROM based technologies are also sensitive to UV light; they can be erased in only a week of direct sunlight exposure or three years of fluorescent light exposure. Thus, proper handling and the use of opaque labels on UV-erasable parts is essential. One-time programmable versions of EPROM based FPGAs packaged in non-windowed packages present no particular reliability concerns in this regard.

b) Design and Test Issues

When an FPGA is used in a system, the overhead related to chip programmability must be factored into the design. With SRAM programmable FPGAs, some non-volatile storage will be required for power-on initialization. In microprocessor-based systems where the FPGA is not required for the processor to boot, programming information can be stored in the processors ROM memory. In this situation the processor can program the FPGA chip as part of its power-on initialization sequence and the on-board programming overhead becomes negligible.

Low pin-count serial PROMs are available which are specifically designed to store configuration bits; and most SRAM FPGAs are capable of initializing themselves from PROM without requiring external control circuitry. With the Xilinx family, multiple FPGAs can be chained together and initialized from a single PROM. However, the extra cost and board space for the PROM must be factored into the design. Finally, the use of in-circuit programmable EEPROM technology or combined EPROM and SRAM technology devices present unique system design opportunities. A system may power-on into a default configuration which is automatically loaded from EPROM into SRAM, then the SRAM programming can be modified during system operation to change the FPGA functionality.

An additional programming overhead must be considered in the form of the added pins per chip. Special programming and mode control pins are needed, which makes the chip more expensive and consumes board area. In some devices, user I/O signals share pins with the programming signals which significantly reduces this programming overhead.

Finally, the choice of FPGA programming technology affects testability in several ways. The use of one-time programmable parts adds additional system test requirements. It is generally not sufficient to rely on device programmer tests. Partially programmed fuses, anti-fuses, or floating gates may read back correctly on the programmer but cause the chip to fail at operating speed or temperature. On the positive side, SRAM based FPGAs add several unique testing opportunities. Special test configurations can be downloaded in the chips to test chip-to-chip

connections, to test adjacent non-FPGA chips, or to test the FPGAs themselves. This test method can also be easily integrated with an IEEE 1149.1 (JTAG) boundary scan test methodology. In fact, as previously mentioned some FPGAs include the JTAG interface and boundary scan resources on-chip. Even if the FPGA does not include dedicated JTAG resources, it can temporarily be configured with a test program to provide these resources for board test prior to loading the application programming for functional test. Section VII provides additional details about the use of the IEEE 1149.1 boundary scan test standard.

IV. Mask Programmable Gate Arrays

The most prevalent style of Mask Programmed Gate Array (MPGA) in current use is the Sea-Of-Gates (SOG) or Sea-Of-Transistors Architecture.[23] The core of an SOG MPGA is composed of a continuous array of transistors in fixed positions, surrounded by I/O circuits and bonding pads. Wafers containing the core design are pre-fabricated up to the final metalization steps. These master or base wafers are stocked by the vendor until a customer design is received. Then, one or more custom masks are created to define the user's circuit with design specific metalization and contacts. The cost of gate arrays is lower than standard cell or full custom designs because:[8]

1.) a single "base wafer" design can be used for many different customer designs,
2.) only 2-5 masks are required (compared to >14 for full custom or standard cell),
3.) automated Placement and Routing tools keep design costs low,
4.) use of standard bonding patterns and packages keeps packaging costs low,
5.) design turn-around time is short because only top metalization steps are run, and
6.) use of common test fixtures and programs keeps testing costs low.

MPGAs are available with varying core array sizes, package pin counts, and fabrication technologies. Table 9 shows the range of commercially available CMOS gate array product offerings.[24] Package pin counts range from 64 to 752, and manufacturers claim maximum usable gate counts up to 1,750,000. Table 10 indicate the range of vendors and options available for MPGAs in Bipolar/GaAs technologies.[25] Note that the entries in Table 10 correspond to approximately an order of magnitude fewer gates per chip; however, the propagation delay per gate is up to an order of magnitude faster than in CMOS gate arrays.

Table 9. Commercial CMOS Mask Programmable Gate Array Products

Company Name	Family	Process Technology (Drawn feature size)	Operating Voltage Range	Max. Usable Gate Count	Max. # of I/Os

American Microsystems, Inc.	GCx, GDx 8Gx	0.8 - 1.25µm	2.5 - 5.5V	65,000 - 250,000	330 - 732
Aspec Technology Inc.	Gate Array HDA	0.5 - 0.8µm 2-4 level metal	2.7 - 5.5V or 2.7-3.6V	280,000 - 1,000,000	460
Aspec Technology Inc.	Embedded Array HDEA	0.5 - 0.8µm 2-4 level metal	2.7 - 5.5V or 2.7-3.6V	550,000 - 1,750,000	460
Atmel Corporation	ATL (60, 80, V)	0.6 - 1.0µm 2-3 level metal	1V, 2V, 3.3V, and 5V options	18,000- 600,000	192 - 440
AT&T Microelectronics	ATT656	1.0µm 2-3 level metal	4.0 - 6.0V	4,000 - 141,600	84 - 448
Chip Express	QYH400 QYH500	1.2µm 0.8µm		16,000 - 71,000	244 - 424
Fujitsu Microelectronics	CE31, 51 CG24, 31, 51	0.5 - 0.8µm	5V, or 3.3V core w/ 3.3V or 5.0V I/O	14,160 - 490,000	256 - 496
GEC Plessey	CLT, CLA, 80K	0.8µm (0.7µm Leff)	3V or 5V	157K-513K raw gates	64 - 456
Harris Semiconductor	AUA20K TAGC40K	1.2µm CMOS/SOS 1.2µm Bulk EPI	5V 5V	16,000 10,000	256 180
Hitachi America	HG62G,72G	0.5 - 0.8µm	3V and 5V	31.7K-500K	136-672
IBM Microelectronics	CMOS 4L CMOS 4LP CMOS 5L	0.8µm (3LM) 0.8µm (3LM) 0.5µm (3-5LM)	4.75 - 5.25V 2.8 - 3.6V 3.0 - 3.8V	230,000 230,000 1,200,000	304-376 304-616 376-752
LSI Logic	LCA 300K, 500K	0.5 - .60µm	5/3V, 3.3/3/2.5V	100,000 - 1,100,000	896 - 992
Mitsubishi Electronics	M6005x/6x7x /8x - GP, EP	0.5 - 1.0µm	3V and 5V	35,000 - 700,000	234 - 512
Motorola ASIC Division	HDC, H4C, H4-5Cplus	0.5 - 1.0µm	3V,5V 1.8V, 3.3V	41,500 - 636,000	88 - 668
National Semiconductor	SCX6Bxx SCX62xx	1.5µm 2.0µm	3V - 5.5V 5V	14,250 8200	200 155
NCR Microelectronics	VGX500 VGX700	0.75µm 0.95µm	3.0 - 5.5V 3.0 - 5.5V	110,000 50,000	304 304
NEC Electronics	CMOS-6 CMOS-8	0.5 - 1.2µm	3V and 5V	155,000 - 439,000	64 - 908
OKI Semiconductor	MSM series	0.5 - 0.8µm 2-3 level metal	3V and 5V	63,000 - 541,000	420 - 624
Orbit Semiconductor	Encore!	1.2µm	3V, 5V	40,000	240
Samsung Semiconductor	KG50000 KG60000	1.0µm 0.8µm	4.5 - 5.5 V 4.5 - 5.5 V	40,000 200,000	300 400
SGS-Thompson Microelectronics	ISB24K, 28K, 35K, 350K	0.5 - 0.7µm	3.3V and 5V	216,000 - 1,094,000	48 - 648
Siemens	SCxE6 SCxF8	0.8µm 0.5 / 0.6µm		200,000 - 560,000	408 - 470
SMOS	SLA 9000, 30000, 20000	0.6 - 1.0µm	down to 0.9V	32,000 - 200K raw gates	100 - 592
Texas Instruments	TGC / TEC families	0.5 - 0.65µm	2.7 - 3.6V and 4.5 - 5.5V	171,000 - 717,000	108 - 700

Toshiba America Electronic Comp.	TC183/180/163/160/14	0.4 - 1.0µm	3.0/3.3V and/or 5.0V	20,000 - 707,000	208 - 652
UTMC	UTDR UTER	1.5µm 1.2µm	4.5 - 5.5V 4.5 - 5.5V	11,000 100,000	212 342
VLSI Technology	VGC720/650/450/453/350	0.5 - 1.0µm	4.5-5.5V, 2.7-3.6V, 2.7-5.5V	132,800 - 800,000	80 - 672

Table 10. Bipolar, BiCMOS, and GaAs Gate Array Products

Company Name	Family	Process Technology	Unloaded Worst Case Gate Delay	Max. Usable Gate Count	I/O Types
Applied Micro Circuits Corp.	Q20000 Q24000	Bipolar BiCMOS	0.1ns 0.4ns	18,700 13,440	ECL, TTL ECL/TTL/CMOS
GEC Plessey	DX Mixed Signal Arrays	Bipolar	0.3ns	9,400	ALL
Motorola ASIC Division	MCA3 ETL MCA3 ECL MCA5ETL	Bipolar Bipolar Bipolar	0.2ns 0.175ns 0.085ns	800-7,000 2,400-12,000 20,000+mem.	ECL, TTL, PECL ECL ECL, TTL
National Semiconductor	NGA, NGR, NGM, NGL, array	Bipolar, ECL, CMOS, BiCMOS	0.19ns ECL 0.48ns BiCMOS 0.36ns CMOS	78,000 - 190,000 (or RAM)	ECL, CMOS, TTL
NEC Electronics	102ECL, BiCMOS 5,8	Bipolar BiCMOS	0.126ns ECL .29/.72ns BiCMOS	24,000 - 165,000	ECL, TTL, PECL, GTL
Rockwell International	Cyclone Lightning	GaAS HMESFET GaAs HBT	0.048ns 0.026ns	35K - 50K 300	ECL, TTL, PECL ECL, CML
Sony	E3G	Bipolar	0.300ns	4,000	ECL, TTL
Synergy Semiconductor	System Elements	Bipolar	0.08ns	58,000	ECL, TTL, CMOS, analog
Texas Instruments	TGB2000E	BiCMOS	0.18ns	112,000	ECL, TTL, PECL, GTL, BTL, cmos
Vitesse Semiconductor	FURY FX VIPER	GaAs GaAs GaAs	0.067ns 0.057ns 0.072ns	30,500 175,000 21,000	ECL, TTL ECL, TTL ECL, TTL

A typical CMOS Sea-Of-Gates structure consisting of adjacent P- and N-channel transistors is shown in Figure 22. Continuous horizontal rows of transistors are created across the chip, and these rows are then repeated vertically to create the array core. While most CMOS SOG designs use single rows of P- and N-channel transistors, double rows are sometimes used to aid in the construction of memories and dynamic logic.[26]

Routing tracks in a SOG design run directly on top of unused transistors in the array. In the traditional MPGA designs which preceded the SOG architecture, fixed-height wiring channels

were used as is done in standard cell designs. This fixed size of the routing channels constrains the maximum number of available routing tracks; and represents silicon area which cannot be used for active devices. Since each design will require different wiring resources, the SOG structure provides a more flexible architecture because of its inherent tradeoff between wiring resources and array transistors. For these reasons, the SOG architecture is the dominant MPGA design style in use today.

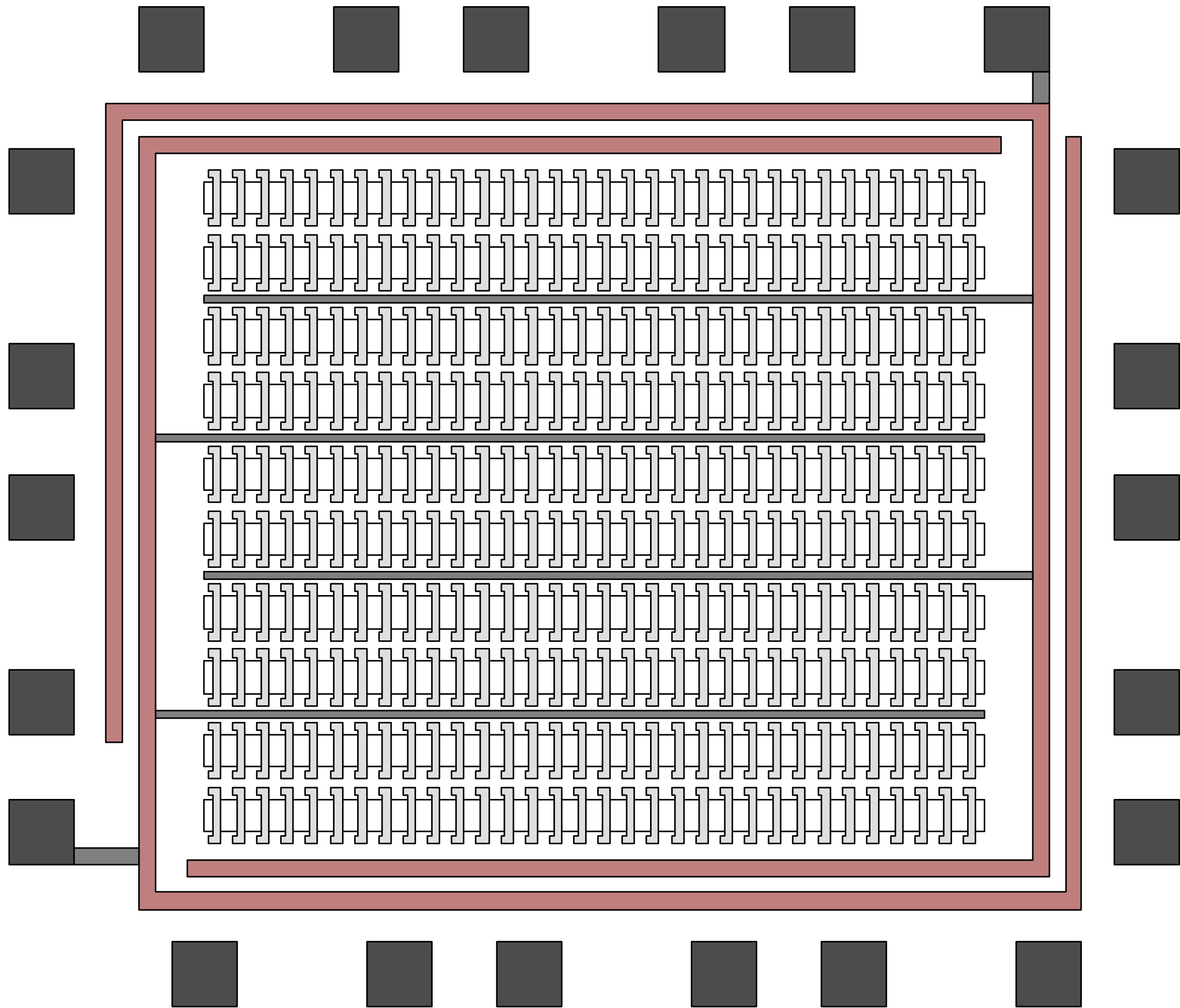

Figure 22. Sea of Gates (SOG) MPGA Architecture

Although the basic layout primitives of the SOG MPGA array are individual transistors, the designer never needs to deal with the physical implementation details. Just as in FPGA design, libraries of gates and macro-functions are provided which are automatically converted to mappings of primitive transistors by the CAD system. The general design flow previously discussed in the FPGA section applies equally well to MPGAs. The only significant difference is the added time and cost to generate and apply the custom masks which define the user's design.

V. Standard Cell and Custom ASICs

In a standard cell (cell-based) design, a library of “full custom components” (cells) is provided which contains each logic gate and primitive macro-function available to the designer. Thus, where an MPGA provides standardized transistor structures, a standard cell design provides standardized function blocks. As discussed in Section I, popularity of the standard cell approach continues to grow, fueled by the increasing availability of advanced cell libraries which include core analog, microprocessor, and special purpose blocks. These advances, together with increasingly sophisticated design tools, are attracting high end users who traditionally relied on full custom design. Table 11 and Table 12 indicate the range of vendors and options available for Standard Cell Design in CMOS and in Bipolar/GaAs technologies respectively.[27] [28]

Table 11. Commercial CMOS Standard Cell Products

Company Name	Family	Process Technology (Drawn feature size)	Operating Voltage	Maximum Gate Count	Max. die size - mm^2
ABB Hafo Inc.	SIG1, 2, 3 SOS4	1.5-3.0µm CMOS 2.0µm CMOS/SOS	1V to 7, 20, 25V 3V - 20V	10K - 60K 25K	196 196
American Microsystems, Inc.	SDx AMI8Sx	1.0µm CMOS 0.8µm CMOS	2.5 - 5.5V 2.5 - 5.5V		
Aspec Technology Inc.	Std. Cell HDC	0.5 - 0.8µm CMOS 2-4 level metal	2.7 - 5.5V and 2.7 - 3.6V	330,000 - 1,200,000	
AT&T Microelectronics	HS, HL, and LP series	0.5 - 0.9µm CMOS 2-3 level metal	3V, 5V	200,000 - 675,600	232
ES2, Inc.	ECL, ECP, ECD	.6 - 1.2µm	3.3V, 5V	230,000 - 320,000	224
GEC Plessey	GSC90K	0.6µm CMOS	3V, 5V	800,000	225
Harris Semiconductor	AUA Cell	1.2µm CMOS/SOS	5V	30,000	161
Hitachi America	HG51 series	0.6-0.8µm CMOS	3V, 5V, or 3/5V	133-232K	
IBM Microelectronics	CMOS 5S CMOS 4L CMOS 4LP CMOS 5L	0.5µm (4-5LM) 0.8µm (3LM) 0.8µm (3LM) 0.5µm (4-5LM)	3 - 3.6V 4.75 - 5.25V 2.8 - 3.6V 3.0 - 3.8V	1,600,000 260,000 260,000 1,400,000	322 161 161 322
LSI Logic	LCB 300K, 500K	0.5 - 0.6µm CMOS	5/3V, 3.3/3/2.5V	600,000 - 1,500,000	
Mitsubishi Electronics	M65200, 65300,GP, EP	0.5 - 1.0µm CMOS	3V and 5V	70,000 - 1,000,000	
National	SCLA80, 10,	0.8 -2.0µm CMOS	3V and 5V	35,000 -	

Semiconductor	20		5V	125,000	
NCR Microelectronics	VS500 VS700	0.75µm CMOS 0.95µm CMOS	3.0-5.5V 4.5- 5.5V	200,000 100,000	
NEC Electronics	CB-C7 CB-C8	0.8µm CMOS 0.5µm CMOS	3.3V or 5V 3.3V	237,000 680,000	225 400
OKI Semiconductor	MSM series	0.5 - 0.8µm CMOS 2-3 level metal	3V and 5V	220,000 - 774,000	
Philips Semiconductors	Hi-IQ	0.8µm BiNMOS		260,000	
Samsung Semiconductor	STD50 STD60	1.0µm CMOS 0.8µm CMOS	4.5 - 5.5 V 4.5 - 5.5 V		
SGS-Thompson Microelectronics	CB22K CB35K	0.7µm CMOS 0.5µm CMOS	3.3V and 5V 3.3V	170,000	284
S-MOS Systems	SSC5000	0.8µm CMOS	down to 0.9V	112,240	225
Toshiba America Electronic Comp.	TC183/180/ 256	0.5 - 0.8µm CMOS	3.0/3.3V and/or 5.0V	200,000 - 383,000	
UTMC	UTDR UTER	1.5µm 1.2µm	4.5 - 5.5V 4.5 - 5.5V	75,000 150,000	232 232
VLSI Technology	VSC family	0.5 - 0.8µm	4.5-5.5V, 2.7-3.6V	300,000 - 800,000	

Table 12. Bipolar, BiCMOS, and GaAs Cell-Based Products

Company Name	Family	Process Technology	Unloaded Worst Case Gate Delay	No. of Macros in Library	I/O Types
GEC Plessey Semiconductors	PCA Compiled ASIC	Bipolar	0.2ns	500	ALL
Synergy Semiconductor	System Elements	Bipolar	0.08ns	160	ECL, TTL, CMOS, analog
TriQuint Semiconductor	QLSI QLSI2	GaAs	0.09ns	>45	ECL, TTL

The full custom design style is the method that was historically used in virtually all commercial high-volume commodity ICs. Full custom continues to maintain its niche for leading-edge technology applications where no other ASIC product meets the system requirements. In this design style, virtually every function is optimized at the transistor layout level. Often, non-conventional circuit designs and clocking methods are used to decrease size and increase speed. Today, companies rarely make exclusive use of full custom design except in memory arrays and commodity parts such as FPGAs.[8] Even in high volume parts such as microprocessors,

combinations of full custom and standard cell design methodologies are used to increase designer efficiency. Elements containing repetitive structures such as register files, caches, and arithmetic pipelines may be full custom; while control logic and interface circuits are created using a standard cell methodology.

VI. ASIC Packaging

An IC package must provide electrical connections to the chip for both signal and power transfer. The package must also physically support the relatively small and fragile IC die, and must protect it from moisture, dust, gases, and other potential contaminants. Finally, the package must provide heat transfer from the die to the ambient environment or to the second level package in order to prevent performance and reliability degradation. The "price" of providing these functions is the imposition of system level constraints which can be summarized as follows:[29]

1.) degraded electrical performance (speed and power)
2.) increased size and weight
3.) reduced testability
4.) reduced reliability
5.) increased cost

IC and system packaging can affect system performance as much or more than the selection of IC design style or process technology. Today's IC packaging costs can also exceed the price of the silicon, a trend that is expected to continue. IC packaging costs rose from 17% of the 1987 merchant semiconductor market ($4.9 Billion) to 22% of the 1993 market ($15.2 Billion).[3] Because of this influence, packaging has received quite a bit of research and development attention. This section only highlights a few of the ASIC packaging issues. The interested reader is referred to one of the many text and reference handbooks devoted to this subject.[30] [31] In addition, hybrid and multi-chip module packaging techniques are discussed in a separate chapter of this handbook.

The traditional low-cost package of choice has been the Dual-In-line Package (DIP). This package has a row of pins on each side which are mounted into holes drilled through a Printed Circuit Board (PCB). Commercial DIP packages grow quite large as pin-count goes up, and are generally limited to a maximum of 64 pins. The Pin Grid Array (PGA) package was developed to increase pin density by placing leads under the entire bottom surface of the package. This technology easily provides well in excess of 300 pins per package, and like DIP packages requires through-hole PCB mounting with 100 mil lead spacing.

Surface mount IC packages have now overtaken the traditional through-hole package market, even for cost sensitive products. In surface mount technology, a chip carrier which may have leads on all four sides is soldered directly onto pads on the surface of the PCB. Lead pitch on a surface mount component is typically 20 to 50 mils, compared to the 100 mil pitch of DIPs and PGAs. Higher system packaging density is possible since the IC packages are smaller, through-holes are not needed, and components can be placed on both sides of the board without

interference. This higher component density reduces parasitic capacitance and inductance and results in higher system operating speed. However, testing of high density surface mount boards is much more difficult than through-hole PCBs. For example, traditional bed-of-nails style board level testers cannot be used to drive and observe signals from the back side of the board since all IC pins are not available on through-holes. In fact, the increasing use of surface mount PCBs was a driving factor in the development and overwhelming acceptance of the IEEE 1149.1 Boundary Scan test standard which will be discussed in Section VII. Surface mounted IC packages come in a variety of styles including the Small Outline IC (SOIC), Plastic Leaded Chip Carrier (PLCC), Leadless Ceramic Chip Carrier (LCCC), and ceramic, metal, or plastic Flat Packs..

A fundamental constraint imposed by IC packaging is the limited number of available pins. During the time when on-chip IC gate count has increased by nearly six orders of magnitude, the number of available package pins has only increased by about two orders of magnitude. The most popular ASIC packages today are the pin-grid arrays (PGAs), Quad Flat Packs (QFPs) and Thin Quad Flat Packs (TQFPs). The current trend in packaging is toward very tight lead pitches, staggered lead pitches, advanced array packages such as Ball-Grid-Array (BGA) and flip-chip (C4), and non-standard surface mount packages such as Tape-Automated Bonding (TAB).[32]

The most promising new ASIC packaging technology is the Ball-Grid-Array. A BGA package provides high I/O density through its array of solder bumps on the underside of the package without requiring ultra-fine pitch connections to the PCB. For example, a 1 inch square QFP package with a 50 mil lead pitch can provide 80 I/O connections. For the same package dimensions, a BGA package can provide 400 I/Os. Since their introduction just a few years ago, BGAs have received quite a bit of attention from both semiconductor manufacturers and end users. Motorola is developing a BGA package for its microcontrollers, Hitachi plans to offer a micro-BGA package for its 0.5µ MPGAs with up to 672 I/Os, and Sandia National Labs is developing a mini-BGA package only slightly larger than the chip die which can accommodate more than 200 I/Os.[33]

VII. System Level ASIC Design Issues

Many system level ASIC design issues have been introduced earlier in this chapter in the context of design style and fabrication technology selection; and in the detailed discussions of ASIC technology alternatives. This section provides additional discussion on three particularly important subjects: Behavioral Modeling and Synthesis, and Structured Design for Testability, and Analog/Mixed Signal ASICs.

A. Behavioral Modeling and Synthesis

The most significant recent change in the way FPGAs (and digital systems in general) are designed is the transition from schematic capture to behavioral modeling and synthesis. The

functionality of the system (IC) is specified with a behavioral model, simulated to verify correct operation, and then a synthesis tool creates an optimized logic circuit. The concept of behavioral synthesis has been around for more than a decade, but efficient and general purpose EDA behavioral synthesis tools have only become widely available in the past few years. During this same ten year period, the essential IC designer skills have changed from an ability to efficiently handicraft a transistor-level full custom IC layout to an ability to use schematic capture tools to create optimal gate-level circuits. According to Devadas, et.al., "The importance of each of these skills is now becoming secondary to the skill of writing an efficient *hardware description language* (HDL) model of an *integrated circuit* (IC)".[34] This section only provides a brief overview of ASIC synthesis, and the interested reader is referred to one of the textbooks devoted entirely to this subject.[34 35]

The traditional approach to synthesis uses a Register Transfer Level (RTL) design description for data path synthesis. The designer specifies an architecture in terms of registers, data transfer operations between registers, and Boolean operations or expressions modifying the data during these transfers. The bit widths of the data paths are fixed in the RTL description, and logic operations must be manually bound to specific clock intervals. Also, since RTL focuses on operations between clocks, the designer must manually specify the control logic needed for the synthesized data path. Thus, an RTL synthesis tool is severely restricted in its ability to optimize across register or clock interval boundaries.

In behavioral synthesis a design is specified at a more abstract level than in RTL. The emphasis is on the behavior of the system as defined by the operations which must be performed; not by a specific Boolean or hardware implementation. The most common behavioral modeling style is one involving algorithmic descriptions of operation sequences and control decisions. The dataflow style is also supported by many tools for designs where sequential inputs must stream through a series of data transformation stages.

There are many potential benefits to the behavioral synthesis style of design. Behavioral models promote the top-down design style which is generally accepted as the most effective approach to creating a target design which meets system specifications. Behavioral code is much easier and faster to write, simulate, and debug than RTL code or gate level schematics. It is also much easier to incorporate structured design for testability; many tools are introducing automatic synthesis of test circuitry as an option to the designer. Finally, because synthesis takes a global perspective by looking at the entire design, it has the potential to explore tradeoffs across functional partitions which would not be obvious to the designer.

While behavioral synthesis offers enhanced design quality, reusability, and designer productivity it is not appropriate for all situations. The following issues should be considered when evaluating design synthesis options:[36]

1.) Behavioral synthesis tools make assumptions about the target architecture in order to constrain the design space to one suitable for automatic mapping and optimization. The designer must determine if the candidate target architecture space provided by a synthesis tool is suited to the design before selecting a given tool. For example, some tools are

better suited for designs dominated by datapath constructs, and may be poorly suited for synthesizing a complex control logic problem.

2.) Behavioral models generally contain little timing information; this information must be added in the form of constraints on the synthesis process. Designs whose I/O signal timing requires irregular patterns will be difficult to automate because the tool cannot know when they must be scheduled. A design with extensive critical timing may require so much effort to manually guide the tool that synthesis cannot be justified.

3.) Most behavioral synthesis tools cannot automatically generate asynchronous designs since they assume fully synchronous circuit operation. Furthermore, most synthesis approaches assume a single-phase edge-triggered clocking scheme. If a design requires multiple phase clocks or multiple clock signals, then some of the less automated synthesis tools may perform better.

The two major languages for behavioral synthesis are the Verilog HDL and the VHSIC HDL (VHDL). Although Verilog has the advantage of a longer time on the market, the standardization of VHDL by the military and by the IEEE standards group (IEEE 1076) has made it a strong contender. Both languages are flourishing today and Verilog may soon be approved as an IEEE standard. Thus, it is not clear that either language will become a dominant standard in the near future. Commercial behavioral synthesis tools are available which use both VHDL and Verilog, as well as a host of proprietary HDLs. An indication that behavioral synthesis is entering the mainstream of ASIC design is given by the June 1994 survey of HDL tool vendors which identifies 75 companies who offer HDL tools.[37]

B. Design for Testability

A high confidence manufacturing test is needed to verify that fabricated ASICs or programmed logic devices are structurally correct. End users typically apply functional test vectors derived from simulation to verify simple low volume PLD designs. However, foundries generally employ a high coverage Single-Stuck-At (SSA) test to meet manufacturing test requirements. An SSA test identifies nodes within the chip which are "stuck-at-one" or "stuck-at-zero"; and may also identify many faults which fall into the other two principal static fault categories of bridging and stuck open.[38] When designing a testable ASIC we must consider any added hardware and I/O overhead necessary to meet manufacturing test requirements (typically >98% SSA fault coverage). Testability enhancement techniques that can improve ASIC manufacturing test coverage fall into three categories:

- Ad Hoc
- Structured Design for Testability
- Built-In Self-Test

Ad hoc methods generally consist of techniques such as partitioning and test point insertion. The specific techniques will vary from design to design, and a common methodology for applying these testability enhancement techniques does not exist. However, structured approaches such as Level Sensitive Scan Design (LSSD), Scan Path, Random Access Scan, and Scan/Set Logic are aimed at solving the general testing problem by employing a specific design methodology.[39] In some cases, an ASIC foundry will provide the appropriate cell libraries and design tools so that a particular design for testability discipline is generated by default. The foundry can then take responsibility for automatic generation of manufacturing test vectors. In other cases, it is the responsibility of the designer to impose a structured design for test discipline.

Structured approaches to testability have historically been viewed as a necessary evil. Their increased silicon overhead and design time was viewed as a burden by design engineers, who were able to get by with ad-hoc testability approaches on designs of relatively low complexity. Today's high complexity systems with substantial functionality embedded in ASICs can no longer be cost effectively designed and implemented without considering testability early and at all stages in the system design for the following reasons:

1.) ASIC test development time is increasing to as much as 40% of the design cycle. Manual test vector generation for a 100K-gate design can take from three to six months.[40] Clearly, the gains in productivity and time to market achieved by synthesis can easily be lost to test development.

2.) ASIC debug time can also contribute to a reduction in time-to-market. When foundries are capable of providing three-week turnaround times on Gate Arrays, traditional chip debug times of hundreds of hours can disastrously affect system integration time.

3.) System level debug time is increasing with the introduction of ASICs. As more logic is integrated onto ASICs, an increasing proportion of the system functionality and timing becomes inaccessible to traditional debug tools.

4.) Increased field diagnosis time for complex ASIC-based systems results in a deterioration of field-service time-to-restore.

5.) Lost opportunity cost for system manufacturers; a designer who is occupied with manual test development and debug on the current product cannot work on the next generation product.

The use of a structured test discipline either eliminates or ameliorates most test problems. Boundary scan and the IEEE 1149.1 standard reduce system level test, debug, and diagnosis by giving serial access to IC primary inputs and outputs. Internal scan or imbedded test solves the test development problem in hours instead of weeks, and can reduce the ASIC debug time by as much as 50%. Both approaches, as well as the use of Built-In Self-Test improve field diagnostics.

Cost for structured test is often less than anticipated. Internal scan or embedded test can add 15% to the silicon area. However, more than 50% of today's designs are pad-limited and the

area impact is not acute because of the die size required to accommodate the I/O pin requirements. In a typical ASIC design, the cost of the silicon is typically only 20 to 30% of the manufacturing cost of the delivered chip. The test costs and the cost of the high pin-count package easily exceed the cost to produce the silicon. Thus, an area impact of 15% may actually only increase the manufacturing cost of the delivered ASIC by 8% or less.[40] More significantly, the use of a structured test discipline can result in a reduction in time-to-market of three to six months. This can be worth millions of dollars in increased sales, easily negating the increased cost of silicon; without even considering the money saved on reduced system test development and debug time.

1. Structured Design for Testability: Scan Design

The structured design for testability approach is based on the concept that any sequential network can be represented by: 1.) a combinational logic network with primary inputs and primary outputs, and 2.) memory elements whose inputs are fed by the combinational logic and whose outputs feed back into the combinational logic.[39] The key to the success of this technique is to find an efficient means to control and observe the latches (memory elements). The most common method is to incorporate a shift register (scan) mode into all on-chip memory elements that allows test input data to be shifted into the memory elements and test results to be shifted out. This scan technique has many names: NEC calls it Scan Path; Sperry Computer Systems called it Scan/Set Logic; Honeywell Inc. calls it Synchronous Scan Design (SSD), and IBM calls it Level Sensitive Scan Design (LSSD).[39] These scan techniques are frequently used in conjunction with a standardized test bus interface such as the IEEE 1149.1 Boundary Scan standard, and vary somewhat with regard to the specific gate level implementation, clocking scheme, and corresponding area overhead.[41 42] In order to provide a concrete example of a structured design for testability technique, the LSSD approach will be briefly described.

A logic subsystem is said to be "Level Sensitive" when the steady-state response to any allowed input state change is independent of the circuit and wiring delays within the subsystem. Also, if an input state change involves the changing of more than one input signal, then the response must be independent of the order in which the signals change. Steady state response is the final value of all logic gate outputs after all change activity has terminated.[43] "Scan Design" refers to the requirement that all sub-system memory be contained in latches. These latches interconnect to form a shift register, and data can be scanned in to or out of the shift register to control and/or observe the subsystem state.

The basic shift register building block is a polarity hold Shift Register Latch (SRL), shown in Figure 23. The SRL uses three clocks: C (the system clock); A (a shift clock); and B (another shift clock). System data is presented to the D (data) input, and test scan data is presented to the I (scan) input. During normal system operation the C clock is used to clock data into the L1 stage, the B clock transfers data from L1 into L2, and the A clock is inactive. The A clock is used in conjunction with the B clock during the test mode. Scan-in data (I) from the L2 output of another SRL (or from a primary input pin) is latched into L1 by the A clock. By cycling the A and B clocks, data is serially shifted through the scan path from the scan-in (I) terminal to the scan out terminal.

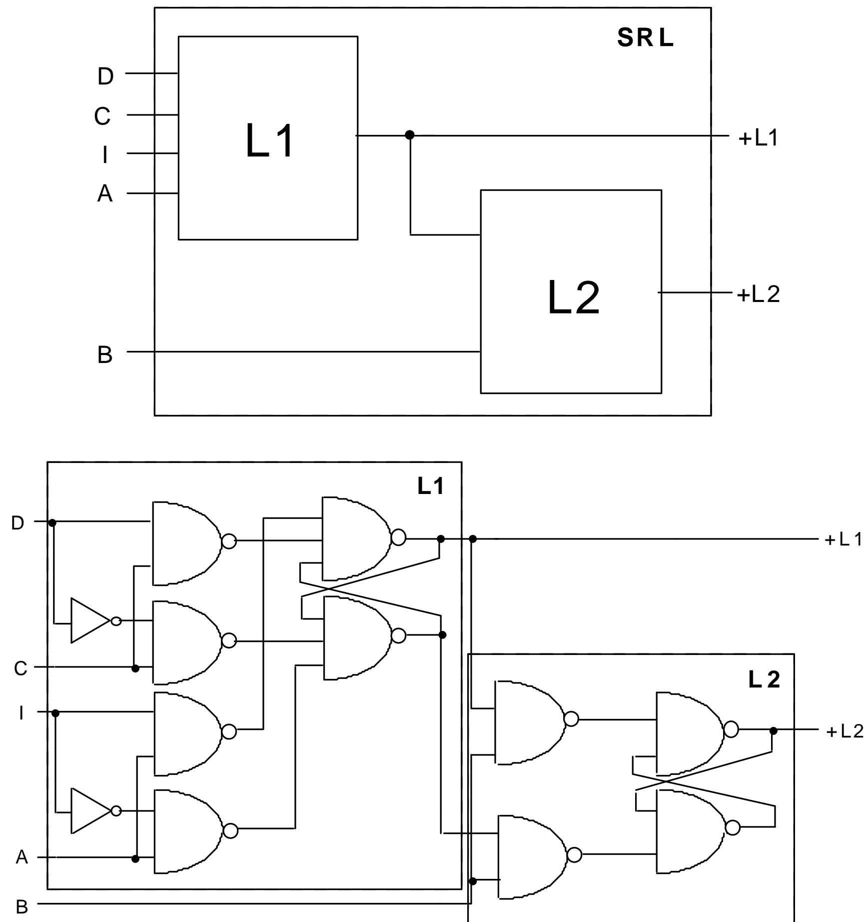

Figure 23. LSSD Polarity Hold Shift Register Latch (SRL)

Boundary-scan is another structured design for testability technique that is used for both chip and system level testing. The boundary-scan technique adds a shift register element (boundary scan cell) at each I/O pin so that signals at component boundaries can be controlled and observed using scan test techniques. The boundary scan cells are bypassed during normal chip operation and data is passed directly between the I/O pins and on-chip logic. However, in the test mode of operation, the boundary cells pass test input and output data along their shift-register path. Data loaded into the boundary scan cells can be used instead of the functional data flowing to or from the pins so that either the internal logic or the external chip-to-chip connections can be tested. The next section discusses the generic boundary scan technique in more detail by describing the IEEE Boundary scan standard.

2. IEEE 1149.1 Boundary Scan Architecture and Test Bus

From 1986 to 1988 the Europe and North America based Joint Test Action Group (JTAG) developed and published a series of proposals for a standardized boundary-scan test architecture. In 1988 the IEEE and JTAG agreed to cooperate in developing a standard to be known as IEEE P1149.1 The final draft of this standard, ANSI/IEEE 1149.1, was accepted as a standard in February 1990.[44]

The 1149.1 standard defines a boundary scan architecture and test interface. Boundary scan requires that scanable registers be placed at all inputs and outputs of an IC. During normal system operation, the boundary scan cells are transparent and do not interfere with system operation. However, during the test mode stimulus data can be serially shifted into the boundary scan registers and results can be captured and serially shifted back out. The 1149.1 test interface consists of a controller and a timing protocol which coordinates all the test circuitry through four required pins: Test Data Input *(TDI),* Test Data Output *(TDO),* Test Mode Select *(TMS),* and Test Clock *(TCK).* Functional IEEE 1149.1 signal definitions are given in Table 13. These four signal pins, along with an optional test reset pin (*TRST**), allow ICs manufactured by different vendors to be accessed through a standard bus for both chip and board level testing.

Table 13. IEEE 1149.1 Interface Pin/Signal Definitions

TCK	Test Clock provides clocking of the test interface and boundary scan circuitry, and provides chip level system clock signals necessary for test / self-test.
TMS	Test Mode Select is used to control the modes of operation of the test interface circuitry as defined by the test access port state machine.
TDO	Test Data Output returns test data from the boundary or scan registers within the chip, or from the test interface.
TDI	Test Data Input is used to load functional test patterns into the boundary or scan paths, or to load test instructions into the test interface.
TRST*	Test Reset is an optional active-low input pin to the chip under test which provides an external hardware reset to the test access port controller.

The test interface can initialize chips for functional testing, provide a mechanism for isolating assembly faults (opens/shorts), and provide for reusable test vectors at the chip, board, and system level. The behavior of the 1149.1 test interface is governed by a Test Access Port (TAP) which includes a 16 state finite state machine controlled by the *TMS* and *TCK* inputs. Other required features of the TAP, illustrated in Figure 24, include an instruction register for

interpreting commands, a boundary scan register for implementing device I/O tests, and a bypass register used to pass test data through the TAP to other Ics.

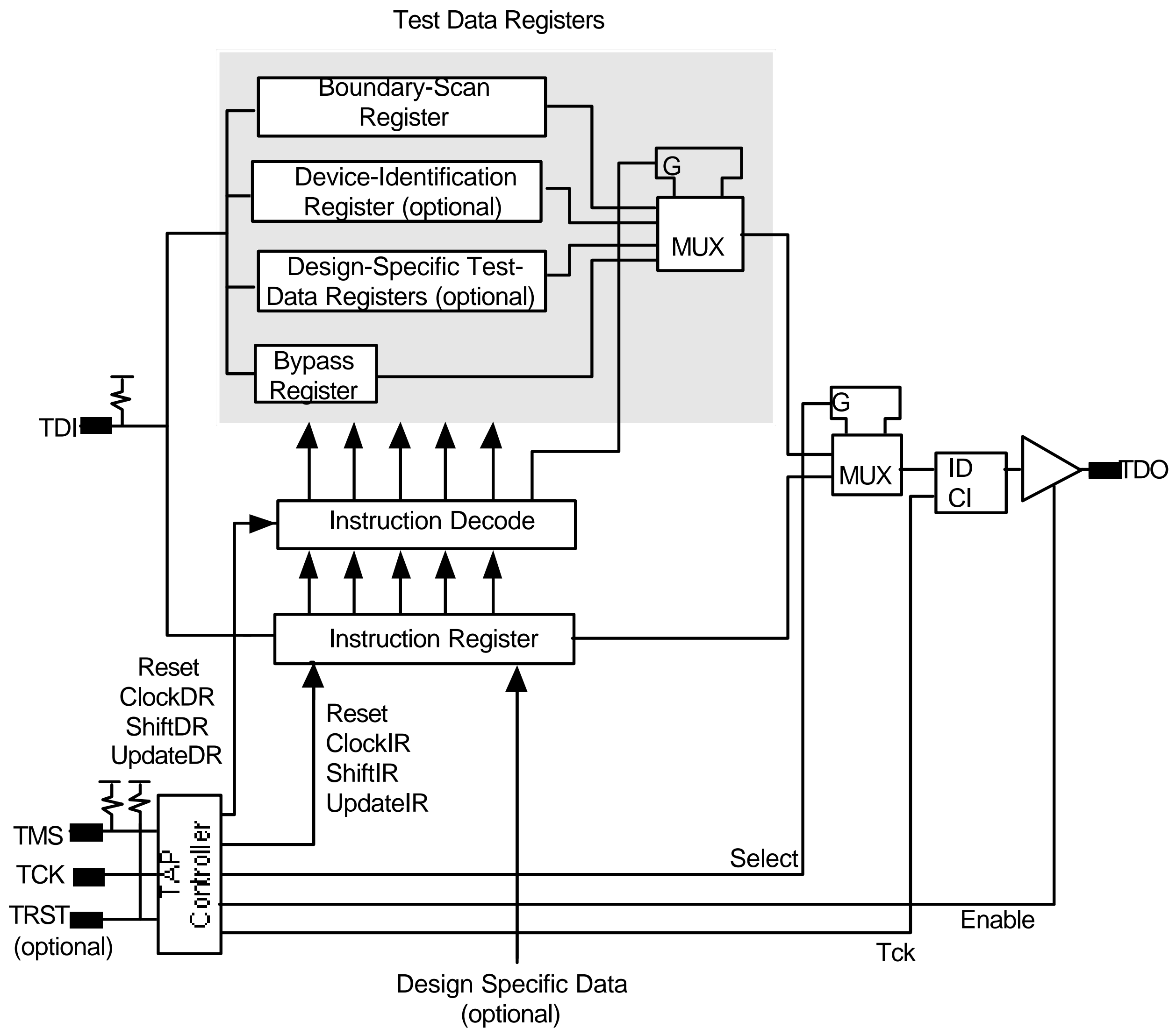

Figure 24. IEEE 1149.1 Test Interface

As indicated in Figure 24, there are three primary TAP registers. The Instruction Register is constructed using a shift-register and a parallel output register. An instruction that is shifted in through the TDI pin selects the test to be performed and/or the test data register to be accessed. The Boundary-Scan register is a single shift-register-based scan path connecting all cell inputs and outputs. Finally, the Bypass register is a single shift-register stage between TDI and TDO which provides a short circuit route for the test data during a scanning cycle. Device-specific test data register connections shown in Figure 24 provide scan path access to the internal registers of the circuit, allowing all internal parallel registers to be operated as shift-registers. The number of

scan paths will vary according to the structure of the design and the total number of on-chip registers. Decoding logic identifies a selected test data register according to the code in the instruction register. The unselected registers maintain their previous values. The contents of the instruction register and the state of the test interface controller determine the mode of operation of the chip and test circuitry. The various operating modes of the TAP are generically defined as follows:

Functional (Test-Logic-Reset): Following power-up reset, the instruction register is initialized to the functional mode and remains in this state until a specific instruction is scanned into the instruction register. Holding TMS high for more than five clock cycles, or asserting the TRST* signal, will also force the controller into the functional (reset) state.

Internal test (INTEST or RUNBIST): In this mode the boundary scan registers are used to isolate the chip I/O from other chips in the system so that the on-chip circuitry can be tested. This test may proceed by sequentially shifting in scan path test vector data (INTEST), or automatically through Built-In Self-Test circuitry (RUNBIST).

External test (EXTEST): This mode permits the testing of inter-chip interconnects using the boundary-scan registers. This is accomplished by scanning interconnect test vector data into the output boundary of one chip and then observing the inputs of all attached chips using their boundary scan input features.

Sample test (SAMPLE/PRELOAD): In this mode the boundary-scan registers sample the input and/or output of the chip without interfering with functional operation; providing an in-circuit "logic analyzer" type feature which is very useful for design debugging.

C. Mixed-Signal ASICs

Through the continuous advances in digital VLSI technology, increasingly powerful microprocessors and digital signal processors have displaced many of the traditional applications of analog circuits. However, the real world is still in the analog domain, and digital signal processors must operate on continuous time analog signals which are translated to and from the digital domain by analog-to-digital (A/D) and digital-to-analog (D/A) converters. Thus, there is frequently a need to provide analog A/D and D/A functions on the same chip as the digital signal processor. Furthermore, the maximum signal bandwidth that digital processing can accommodate is limited by the processor clock speed and the Nyquist sampling rate. Thus, many designs will require analog pre-processing to reduce the bandwidth to a level suitable for digital processing.

Mixed signal and analog ASICs which provide continuous time analog signal processing capabilities represent a small but significant portion of the ASIC market. As in the digital market, these analog/mixed signal designs are driven by time-to-market, testability, performance, and density concerns. Growing application areas for mixed signal technology include audio circuits in communications and PC products; telecommunications and personal communications systems,

industrial applications such as motor controllers; and mass storage devices such as hard drives, optical drives, and tape units. For example, although a hard drive controller interface may use 100% digital technology, the servo motor controller is 50% analog and the read channel processor is 40% analog. The stringent size, weight, and power restrictions on the latest generation of 1.8" hard drives for portable applications almost mandates use of mixed signal ASICs.[45]

Analog/Mixed signal ASICs are available as Linear Arrays and as Standard Cell products from a number of vendors. A 1994 survey lists 17 vendors of Analog/Mixed Signal Linear Array products and 28 vendors of Analog/Mixed Signal Standard Cell products.[45] Special care is needed when designing with any analog technology. Noise can be a significant problem, and analog and digital functions need to be carefully isolated on the chip to prevent digital noise from coupling into sensitive analog circuits. Mixed-signal designs typically need much more of a "hand-crafted" approach than all digital designs, resulting in longer design times. Analog CAD tools comparable to digital tools which allow a designer to work at a high level of abstraction (and hide the low-level design details) are just not available yet. Simulation at the transistor level is necessary to predict analog circuit performance, and it is difficult to incorporate all the parasitics, crosstalk, and noise problems of a mixed signal ASIC in the device level model. Thus, mixed-signal ASICs are generally much more expensive than their digital counterparts, and design volume is quite critical in determining whether the high engineering NRE will be recovered. Thus, it is nontrivial to make the economic decision of whether to design a mixed-signal ASIC or to keep analog and digital circuits separate and have a system with more components and larger board area.

Test development and test time represent a more significant cost risk in an analog/mixed signal ASIC development project than in an all digital design. Mature structured design for test methodologies for analog/mixed signal designs do not exist, and a custom test approach must be developed for each chip. The major test tradeoff areas which must be considered include: test development cost, implementation schedule, production test cost, and production test coverage. If the user or the vendor attempted to test every design parameter over all voltage and temperature extremes, the test cost could be many times greater than the cost of the chip itself.[46]

VIII. REFERENCES

[1] S. Murog, VLSI System Design: When and How to Design Very-Large-Scale Integrated Circuits, John Wiley & Sons, New York, NY 1982.

[2] C. F. Fey & D. E. Paraskevopoulos, "Studies in LSI Technology Economics IV: Models for gate array design productivity", IEEE JSSC, vol. SC-24, no. 4, August 1989, pp. 1085-1091.

[3] B. Arnold, "Standard Cells Pace CMOS ASIC Growth", ASIC & EDA - Technologies for System Design, March 1994, pp. 36-51.

[4] S. Brown, R. Francis, J. Rose, & Z. Vranesic, Field Programmable Gate Arrays, Norwell MA, Klewer Academic Publishers, 1992.

[5] K. Rousseau, “Gallium Arsenide Revisited”, ASIC & EDA - Technologies for System Design, May 1994, pp. 28-36.

[6] B. Arnold, “Bipolar, BiCMOS, and GaAs Vie with CMOS for Digital ASIC Sockets”, ASIC & EDA - Technologies for System Design, August 1994, pp. 46-52.

[7] D. Pucknell and K. Eshraghian, Basic VLSI Design, Prentice Hall, third edition, 1994.

[8] N. Weste & K. Eshraghian, Principles of CMOS VLSI Design: A Systems Perspective, Second Edition, Reading MA, Addison-Wesley Publishing Company, 1993.

[9] D. Hodges and H. Jackson, Analysis and Design of Digital Integrated Circuits, McGraw-Hill, New York, NY, Second Edition, 1988.

[10] Programmable Logic Devices, Vendor Guide 1995 Supplement, ASIC & EDA - Technologies for System Design, December / January 1995, pp. 85-88.

[11] The Programmable Logic Data Book, Xilinx Inc., San Jose, CA, 1994.

[12] ACT Family Field Programmable Gate Array Databook, Actel Corporation, San Jose, CA, 1992.

[13] pASIC 1 Family Data Book, QuickLogic Corporation, Santa Clara, CA, 1992.

[14] XCELL, The Quarterly Journal for Xilinx Programmable Logic Users, Xilinx, Inc, San Jose, CA, Fourth Quarter 1994

[15] J. Rose and S. Brown, “Flexibility of Interconnection Structures in Field Programmable Gate Arrays”, IEEE Journal of Solid State Circuits, Vol.26, No.3, March 1991, pp. 277-282.

[16] Actel ACT3 Field Programmable Gate Arrays, Actel Corporation, Sunnyvale CA, 1993.

[17] S. Singh, J. Rose, D. Lewis, K. Chung, and P. Chow, “Optimization of Field-Programmable Gate Array Logic Block Architecture for Speed”, Proc. 1991 Custom Integrated Circuits Conference, May 1991, pp. 6.1.1-6.1.6.

[18] MAX 7000 Data Book, Altera Corporation, San Jose, CA, 1993.

[19] MAX 7000 Programmable Logic Device Family; Data Sheet, Altera Corporation, San Jose, CA, August, 1994, ver. 2.

[20] FLEX 8000 Programmable Logic Device Family; Data Sheet, Altera Corporation, San Jose, CA, August 1994, ver. 4.

[21] MAX 9000 Programmable Logic Device Family; Data Sheet, Altera Corporation, San Jose, CA, October 1994, ver. 1.

[22] The Configurable Logic Data Book, Algotronix Ltd, Edinburgh, Scotland, 1990.

[23] E. Hollis, Design of VLSI Gate Array ICs, Englewood Cliffs, NJ, Prentice-Hall, Inc, 1987

[24] L. Waller, "A Tale of Two ASICs - CMOS Gate Array Products", Integrated System Design, February 1995, pp. 36-51.

[25] Bipolar, BiCMOS, and GaAs Digital ASICs - *Array Based*, Vendor Guide 1995 Supplement, ASIC & EDA - Technologies for System Design, December / January 1995, pp. 80.

[26] H. Veendrick, J. Kernhof, and B. Hoefflinger, "The CMOS Gate Forest, an Efficient and Flexible High-Performance ASIC Design Environment", IEEE JSSC, vol.25, no.5, October, 1990, pp. 1153-1157.

[27] L. Waller, "A Tale of Two ASICs - CMOS Cell-Based Products", Integrated System Design, February 1995, pp. 36-51

[28] Bipolar, BiCMOS, and GaAs Digital ASICs - *Cell Based*, Vendor Guide 1995 Supplement, ASIC & EDA - Technologies for System Design, December / January 1995, pp. 80.

[29] B. Johnson, "Overview of Chip Level Packaging", Electronic Materials Handbook: Volume 1 - Packaging, ASM International, Materials Park, OH, 1989, pp. 398-407.

[30] H. Bakoglu, Circuits, Interconnections, and Packaging for VLSI, Addison-Wesley Publishing Co., Reading, MA, 1990.

[31] Electronic Materials Handbook: Volume 1 - Packaging, ASM International, Materials Park, OH, 1989.

[32] R. Heitmann, "Ultra-Fine Pitch Technology: Assembly Challenges and Considerations", Electronic Packaging and Production, December 1993, pp. 34-37.

[33] A. Bindra, "BGAs Making Micro Move", Electronic Engineering Times, 1994 Special Report, October 34, 1994, pp. 32, 56, 82.

[34] S. Devadas, A. Ghosh, and K. Keutzer, Logic Synthesis, McGraw-Hill, Inc., New York, NY, 1994.

[35] G. De Micheli, Synthesis and Optimization of Digital Circuits, McGraw-Hill, Inc., New York, NY, 1994.

[36] S. Schulz, "Behavioral Synthesis: Concept to Silicon", ASIC & EDA - Technologies for System Design, May 1994, pp. 13-26.

[37] L. Vereen, "HDL Momentum Grows", ASIC & EDA - Technologies for System Design, June 1994, pp. 50-70.

[38] N. JHa, "Detecting Multiple Faults in CMOS Circuits", *Proceedings of the 1988 International Test Conference*, pp. 514-519.

[39] T. W. Williams, "VLSI Testing", *Computer*, Volume 17, October 1984, pp. 126-136.

[40] T. Gheewala, "Structured Test is Free", ASIC & EDA - Technologies for System Design, July 1994, pp. 71.

[41] L. Schramm and D. Landis, "Evaluation of VHSIC Chip Level Test Architectures", Proc. 1992 Florida Microelectronics Conf., May 1992, pp. 24-27.

[42] L. Schramm, Evaluation of VHSIC Chip Level Test Architectures, Master's Thesis, University of South Florida, April 1992.

[43] E. B. Eichelberger and T.W. Williams, "A Logic Design Structure for LSI Testability", *Proceedings of the 14th Design Automation Conference*, New Orleans, June 1977, pp. 462-468.

[44] ANSI/IEEE Standard Test Access Port and Boundary-Scan Architecture. IEEE Std. 1149.1-1990.

[45] B. Arnold, "Finding Success with Mixed-Signal ASICs", ASIC & EDA - Technologies for System Design, January 1994, pp. 36-48.

[46] R. Pate & J. Rogers, "Putting together a test strategy for analog and mixed-signal ASICs", Computer Design ASIC Design Supplement, February 1994, pp. A29-A34.