LOGIC ARRAYS

Logic integrated circuits, (ICs) or logic "chips," are found in an increasing number of applications that affect our everyday lives. The use of logic integrated circuits has gone from the exclusive world of government projects and huge computer systems to personal computers, fax machines, home appliances, automobiles, and children's toys. The creation and explosive growth of an entirely new class of personal portable devices, such as pagers, cellular phones, and personal-data assistants have been made possible with increasingly smaller, faster, logic ICs that consume less power than ever before. The amount of logic that can be integrated on a single silicon die has grown from the four devices interconnected on the first integrated circuit in 1958 to 12 million individual circuits in 1998. In that same time span, the speed at which these circuits operate has increased from less than 1 MHz to more than 400 MHz. As circuits become smaller, they also become less expensive. Personal computers that surpass the million-dollar mainframes of the 1970s in terms of function, performance, memory and storage can be purchased at the local office supply store for less than $1000 (1).

Amazing advances and innovations made in manufacturing technology, circuit design, and the software programs to design these chips made the explosion of highly complex, inexpensive integrated logic circuits possible. The exponential growth in function and complexity is expected to continue well past the year 2000, allowing integration of entire systems on a chip. The minimum device geometry defining transistor size is expected to decrease to below one tenth of one micron (Table 1) allowing 100 million transistors to be packed in a square centimeter of silicon. With the successful integration of materials, such as copper and silicon germanium, circuits manufactured in the future will run faster and dissipate less power, fueling development of new logic-IC-based applications. The logic-IC market represents approximately 16% of the rapidly expanding semiconductor market. It generated more than $23 billion in revenue in 1996, a figure expected to more than double by the year 2001.

GROWTH IN CIRCUIT DENSITY

Modern-day electronics began with the invention of the transfer resistor, also known as the transistor, in 1947 by William Shockley and his team at Bell Laboratories (3). In 1958 the integrated circuit was born when Jack Kilby at Texas Instruments successfully interconnected, by hand, several transistors, resistors, and capacitors on a single substrate. During the following 40 years, scientists and engineers developed methods to integrate hundreds, thousands, and eventually millions of circuits onto a single piece of silicon, and at the same time, improved the speed and reduced the cost of those circuits. In 1965, Gordon Moore, founder of both Fairchild Semiconductors and the Intel Corporation quantified the amazing increase in the density of the integrated circuit with "Moore's law" which states that the number of circuits on a chip roughly doubled every 12 to 18 months, a trend he expected will continue into the foreseeable future (4). This prediction has proved to be extremely accurate during the last 30 years and has driven the pace of innovation and integration in the semiconductor industry.

The generations of circuit growth are classified as small-scale integration (SSI), medium-scale integration (MSI), large-scale integration (LSI), very large scale integration (VLSI), and ultralarge scale integration (ULSI), corresponding roughly to 10^2, 10^3, 10^4, 10^5, and 10^6 circuits per chip. The era of ULSI became a reality in 1995 when the million-gate logic chips were successfully manufactured (5).

WAFER FABRICATION

Logic ICs consist of logic devices (circuits) interconnected with wires on a single piece of silicon called a "die" or chip. The chips are manufactured on thin disks of purified silicon called wafers. In the 1970s silicon wafers 75 mm in diameter (approximately 3 in) were the standard (6). Average wafer size increased to 125 mm in the 1980s and again to 200 mm in the 1990s, with 300 mm wafer capability predicted to be online before 2000. Hundreds of chips are manufactured simultaneously on a single 200 mm wafer. The exact number per

Table 1. Semiconductor Industry Association's Technology Roadmap[a]

	1997	1999	2001	2003	2006	2009	2012
Feature size (μm)	0.25	0.18	0.15	0.13	0.10	0.07	0.05
Logic transistors/cm^2	8M	14M	16M	24M	40M	64M	100M
Logic chip size (mm)	22 × 22	25 × 32	25 × 34	25 × 36	25 × 40	25 × 44	25 × 52
Wafer diameter (mm)	200	300	300	300	300	450	450
Maximum wiring levels	6	6–7	7	7	7–8	8–9	9
Minimum mask count	22	22–24	23	24	24–26	26–28	28
Chip frequency (MHz)							
Across chip ASIC (HP)[b]	400	600	700	800	1100	1400	1800
Across chip ASCI (CP)[b]	300	500	600	700	900	1200	1500
Max power watt/die							
With heat sink (HP)	70	90	110	130	160	170	175
Battery/hand-held	1.2	1.4	1.7	2.0	2.4	2.8	3.2
Minimum power (V_{dd})	1.85–2.5	1.5–1.8	1.2–1.5	1.2–1.5	0.9–1.2	0.6–0.9	0.5–0.6
Number of chip I/Os							
Chip-to-package (HP)	1450	2000	2400	3000	4000	5400	7300
Chip-to-package (CP)	600	975	1195	1460	1970	2655	3585
Number of package pins/balls							
ASIC (HP)	1100	1500	1800	2200	3000	4100	5500
Microprocessor/(CP)	600	810	900	1100	1500	2000	2700

[a] Ref. 2.
[b] HP = high performance; CP = cost performance.

wafer varies with chip size. Logic circuits are created by using glass masks, UV light, and successive layers of insulators and conductors to print detailed circuit patterns on a wafer. Polysilicon is used to create transistor gates, and "dopants," such as phosphorous or boron, are embedded into the wafer by diffusion or implantation to form negative- and positive-conducting regions. Then the circuits are interconnected by wires formed by depositing patterns of successive layers of metal (usually aluminum) and an insulator with additional masks. Wires are connected to the circuits with "contacts" made of metal, such as tungsten. A completed wafer is sealed with a final layer of insulator before it is diced into individual chips. The number of masks required to create a logic chip varies among semiconductor manufacturers, but it can easily exceed 20, depending on the wiring levels used (Table 1).

DEVICE SCALING VERSUS PACKING DENSITY

The incredible growth rate in the number of circuits that can be manufactured on a chip is the result of reducing the minimum transistor feature size, larger chips, and improving the packing efficiency of the circuits on those chips (7). The continued reduction in transistor size is known as device "scaling." The transistor size is typically measured by the length of the "channel" or "gate" through which the electrons flow when the transistor is active. Transistors with gate lengths less than 0.5 μm have been in volume production since the early 1990s (8). Leading-edge logic suppliers have announced devices with gate lengths of 0.15 μm that will enter volume production before the year 2000.

The theory of device scaling (9) was used in the 1970s and 1980s to predict the speed, power, and packing density of circuits that could be achieved as device sizes were reduced (scaled). Ideal scaling predicts that the delay through a device decreases at the same rate as device size. As circuit size decreases, delay decreases, resulting in faster circuits. If the voltage level of the circuit is reduced at the same rate, the resulting power dissipation per device drops at an even greater rate. Reducing the size of the transistor by a factor of two results in a circuit that runs twice as fast and dissipates 25% of the power if the input voltage to the circuit is also reduced by a factor of 2. Continued reduction in circuit size through repeated device scaling would logically result in faster, more densely packed chips that dissipate less power than previous generations.

The advantages of ideal scaling, however, cannot be fully realized because of certain second-order effects that require modifying the simple scaling approach (7). Compatibility with existing power supplies and ICs from previous generations prevents input voltages from being scaled with the transistor size. Providing multiple voltage supplies is costly and impractical in many applications. Continued scaling of the device size, while keeping the voltage level constant, causes electromigration (the transport of atoms due to current flow) and other reliability-related problems forcing reduction in the supply voltage every four to five years.

Continued scaling of the device also leads to wiring-related or "interconnect" delay problems at the chip level. Although circuits become smaller from one generation to the next, average chip size has increased as designers integrate more functions on a given die. Larger chips result in longer cross-chip wires (global wires), which causes an in increase in delay through those wires. Critical paths that determine the overall performance of a circuit are usually dominated by global wires, so faster circuits do not necessarily result in faster chips (7,10).

Through the eras of SSI, MSI, and LSI, the effect of interconnect delay (caused by wire resistance and capacitance) on an integrated circuit's maximum performance was dominated by intrinsic gate delay (the speed at which a signal propagates through a device). When transistor gate length decreased below 1 μm in the VLSI era, the interconnect-related delay began to equal or exceed the gate delay and ICs could no longer be accurately predicted by simply applying the device scaling theory (10). At deep-submicron gate lengths (less than 0.5 μm), the interconnect delay can account for more than 80% of the overall chip delay. The intrinsic gate delay for 0.35 μm designs is around 100 ps, whereas the potential estimated delay for a 2 mm interconnect wire can be as high as 600 ps (11).

Calculating accurate chip delays at the deep-submicron level required changes to chip-design methods, design tools, and the basic algorithms used to calculate device and interconnect delay. The basic equations used to calculate delay through a circuit have evolved from 5 to 7 term equations in the early 1990s to equations with more than 20 terms in 1998 that provide nonlinear scaling of temperature and voltage, input transition degradation caused by RC effects, and more complex capacitive loading.

Interconnect-related delay has increased because of increasing average wire length (L) and also because of decreasing average wire width (W). As wire widths are scaled down with the devices, they become more resistive, slowing down the electron flow through the wire. Resistance (R) is proportional to the width of the wire and, therefore, increases as wires become thinner (10,11):

$$R = L/W \qquad (1)$$

In addition to becoming more resistive as they become thinner, aluminum wires used for on-chip interconnects also become more susceptible to electromigration (12). Electromigration generates electrical opens and shorts between on-chip wires, causing circuits to fail. This has become a major interconnection failure mechanism in VLSI and ULSI circuits (7), impacting chip reliability. The problem is exacerbated by increasing on-chip clock frequencies.

Aluminum (specifically an aluminum/copper alloy) has been the interconnect material of choice for semiconductor circuits for more than 30 years because of its relatively low resistivity (compared to polysilicon), good adhesion to silicon and silicon dioxide, bondability, patternability, and ease of deposition. Aluminum is also easily purified (it does not contaminate the IC with undesirable impurities) and is a readily available, low-cost material (7). For high-performance logic designs with features below 0.25 μm, however, the delay caused by increased wire resistance and electromigration failure associated with aluminum interconnects is a barrier to meeting the density, speed, and power targets of the marketplace (13).

The successful use of copper metallization for on-chip interconnects was announced by IBM in late 1997 and will be used in volume chip production in 1998 (Fig. 1). When com-

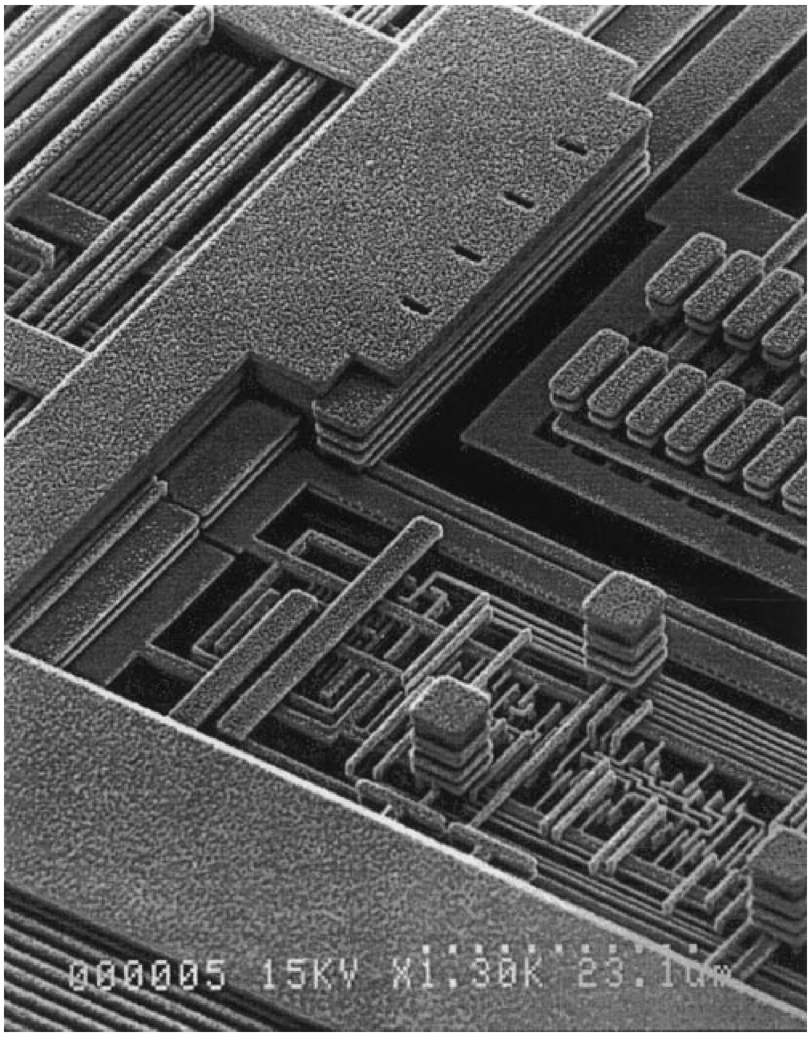


Figure 1. The insulator between the chip wiring levels has been etched away to show the six individual layers of wiring. Both the metal lines and the vias that connect the wiring levels are made of copper. The transistors connected by these wires are 0.12 μm long. Photograph courtesy of the IBM Corporation, © 1997.

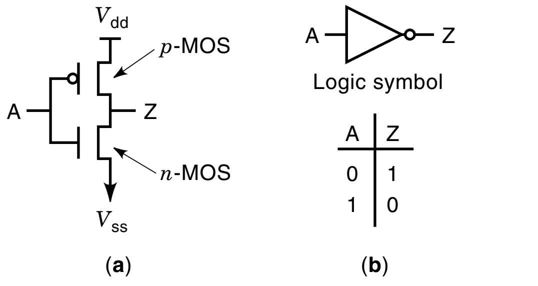
pared to aluminum, copper is 40% less resistive, is a better conductor, and is far less susceptible to electromigration. There are major drawbacks to using copper, such as lack of a viable etch technology to pattern the wires on the silicon wafer and copper's tendency to diffuse into silicon and the other layers that form the logic device, which changes basic electrical properties and destroys the transistors (13,14). A protective barrier that isolates the copper wires from the active devices and can be used in volume production took decades to develop (14). Metal is deposited and planarized, rather than etched, by a damascene process and chemical-mechanical processing (CMP), originally developed for aluminum and extended to dual-damascene for use with copper.

DIGITAL PROCESS TECHNOLOGY

Approximately 94% of digital logic circuits are designed in the complementary metal-oxide semiconductor (CMOS) process. Bipolar, once the dominant IC technology, now represents only 6% of the logic IC market and is expected to decline to less than 2% by the year 2001 (16). *n*MOS devices, which dominated the late 1970s and 1980s are no longer in production. CMOS replaced these technologies because it combines high packing density, high speed, and high yield with low power dissipation (7). Although most bipolar devices were faster than CMOS and *n*MOS circuits were denser, CMOS prevailed because it dissipates far less power per circuit. As transistor counts approached one million, power dissipation became the fundamental barrier to integrating more circuits (15). Densely packed ICs running at high speeds generated more heat than IC packages could effectively dissipate, creating high-temperature-induced reliability problems. As a result CMOS became the technology of choice for VLSI and beyond (7,15).

The active device used in all MOS circuits is the metal-oxide semiconductor field-effect transistor (MOSFET). The name CMOS stands for complementary MOS and refers to the use of both types of MOSFET transistors, *n*-channel and *p*-channel (17) [Fig. 2(a)]. The term metal-oxide semiconductor stems from using metal to form the gate in early transistors. Modern devices have polysilicon gates, making the term CMOS a misnomer (7).

The layout view of a simple CMOS inverter function illustrates the major process levels of the device. The schematic view of the inverter [Fig. 2(b)] and the corresponding truth table [Fig. 2(c)] demonstrate how the circuit operates. When the value at inverter input (A) is a positive voltage bias (a logic 1), the *n*MOS portion of the circuit turns on and the *p*MOS turns off. The output of the inverter (Z) becomes a logic zero because it is connected to ground through the *n*MOS. When the value at the input changes to ground (a logic 0), the *n*MOS transistor turns off and the *p*MOS turns on. The output is pulled up to the power-supply (V_{dd}) value through the *p*MOS transistor and becomes a logic 1 (18). Because the *n*MOS and the *p*MOS are not on at the same time (overlapping states are minimal), a dc-conducting path is never formed between power and ground, and the steady-state current from V_{dd} to V_{ss} is almost zero. Minimum power dissipation translates into circuits that run cooler (release less heat), which enables the use of inexpensive plastic packages and reduces the need for space-consuming external heat sinks and cooling fans. Minimum power consumption permits the circuits to run longer from battery-powered sources. Both of these features are critical for success in the growing market of portable personal devices, such as cell phones and laptop computers.



A	Z
0	1
1	0

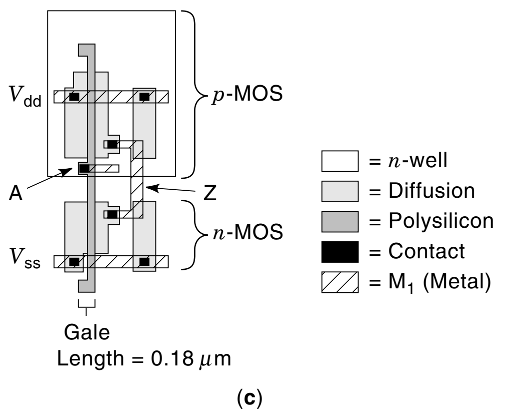


Figure 2. (a) The transistor schematic view for a simple CMOS inverter. "A" is the input to the inverter. "Z" is the output. The logic symbol used to represent the inverter function and the corresponding truth table. (c) A simplified layout view of the inverter showing the polygon shapes drawn on silicon wafer to create the transistors.

LOGIC CLASSIFICATIONS

Although most digital logic circuits use the CMOS process and therefore contain the same basic *p*- and *n*-MOS transistors, different methods have been developed to create and interconnect these transistors to serve different chip-level density, speed requirements, market cost, and turnaround-time (TAT) objectives. Logic ICs designed to implement a unique function for a single application are classified as "application-specific integrated circuits," or ASICs (Fig. 3). ASICs can be further subdivided according to the method used to design and integrate the base logic circuits on a chip. Choosing the correct design approach for any given logic application must

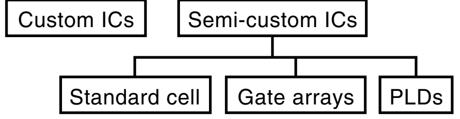


Figure 3. Logic family classifications.

take into account performance, density, cost, and turnaround time and also the design experience required to implement the product.

Full-Custom Design

Full-custom IC design is the oldest and most complex of the ASIC design methods. In full-custom, every device on-chip (transistor, diode, resistor, etc.) and their interconnections are individually designed to occupy minimum area and yield optimum performance. Full-custom IC designers must be intimately familiar with the logic function they are designing and the technology in which it is implemented. They must also understand and adhere to the detailed circuit design rules governing the placement of each geometric shape on each of the technology mask levels used to manufacture the IC.

Each logic circuit implemented, such as an inverter or NAND, may have unique height and width depending on the size of the transistors within the circuit. The absence of regularity in circuit size makes it difficult, if not impossible, to use automated methods to place and route the circuits. So chips using custom-designed circuits often require time-intensive manual layout.

Full-custom ICs are generally the most expensive ASIC type to design because of the expertise level and time required to customize each circuit. However, a custom chip usually occupies the least silicon area and therefore can cost less per manufactured piece when compared with other design methods (Table 2). The time and effort used for custom design can be easily justified for performance-driven logic chips, such as microprocessors, that are produced in volumes of millions per year (Fig. 4). Alternative design methods are required for the majority of logic applications.

In 1996, custom ICs accounted for approximately 10% of the total ASIC market (23). They are expected to be replaced by standard-cell circuits, and the percentage will decline to less than 1% of ASICs produced in 2001.

Semi-Custom Design

Standard-cell, gate-array and programmable-logic devices share the characteristic of using predesigned circuits placed in a structured array on a die. The interior of the array consists of a series of circuit rows that contain the logic circuits and input/output (I/O) circuits for off-chip drivers and receivers, usually arranged in a ring around the periphery of the chip. The number of I/O locations and circuit rows, the circuits per row, and the spacing between the rows are predefined by the ASIC manufacturer. The three semi-custom design approaches differ in the method used to design and interconnect the base logic circuits which translates into performance, density, and turnaround-time trade-offs. In standard-cell circuits, all process layers are customized. With gate-array only the metal layers vary from one circuit type to another. Programmable-logic devices containing identical logic circuit patterns that are customized or "programmed" after they are manufactured. Because all three types of logic ICs can be customized for a particular logic application but do so through the use of predesigned circuits, they are referred to as "semicustom" logic (24).

Standard Cell. Standard cell is currently the fastest growing segment of the ASIC market, projected to grow from 44.5% of the ASIC market in 1996 to 65% of ASICs in 2001, a compound annual growth rate of almost 28% (23). Standard cell was developed to reduce the engineering time and expertise required for IC design (19). It provides high-density, high-performance solutions for applications that require a faster time to market than is possible for full-custom design but at the expense of less efficient area utilization.

Unlike custom ICs, transistor designs for specific logic functions are predefined and placed in a "circuit library" by the ASIC manufacturer (or ASIC vendor). The vendor attempts to include the functions commonly used to build a logic IC, including simple logic (inverters, NANDs, NORs); more complex functions (AOIs, flip-flops, latches, muxes); functions that may be unique to a product family (clock drivers, splitters, choppers); and a family of input/output (I/O) drivers and receivers. A range of static random access memories (SRAMs), read-only memories (ROMs), and/or register arrays (RAs) are also usually provided using a memory compiler program.

An average ASIC library contains approximately 500 elements, excluding the variable memory elements, and is designed by the ASIC vendor to the design-rule requirements of the targeted process. An instance or occurrence of each logic circuit type (e.g., NAND2) is usually manufactured on a test chip and "characterized" by the ASIC vendor before it is used by a logic-chip designer. During characterization, the vendor verifies the circuit functionality, performance (speed), and reliability.

Logic IC designers use elements from this circuit library to implement their individual chip's function. Because the logic circuits have been predesigned by the ASIC vendor, the logic-chip designer is not required to be an expert in transistor layout or the ASIC vendor's target process. Because the circuits have been preverified or characterized on a test chip, the risk of finding performance or functional errors in an individual logic circuit has been mitigated. Standard cells are de-

Table 2. Design Approach Comparisons[a]

	Relative Density	Relative NRE Cost	Relative Cost per Chip	Relative Design Time	Relative Time to Prototype	Unique Processing
Full-custom	1.7	5.0	0.3	4.0	2.2	100%
Standard-cell	1.4	2.0	0.7	2.0	2.0	100%
Gate-array	1.0	1.0	1.0	1.0	1.0	30%
PLD	n/a	0	2–4	0.1	0	0%

[a] Refs. 19–22.

[b] n/a = not available. PLDs are less dense than gate-arrays, but because of the varying architectures which implement different numbers of equivalent gate-array 2-way NANDs per PLD cell, an accurate comparison cannot be made.

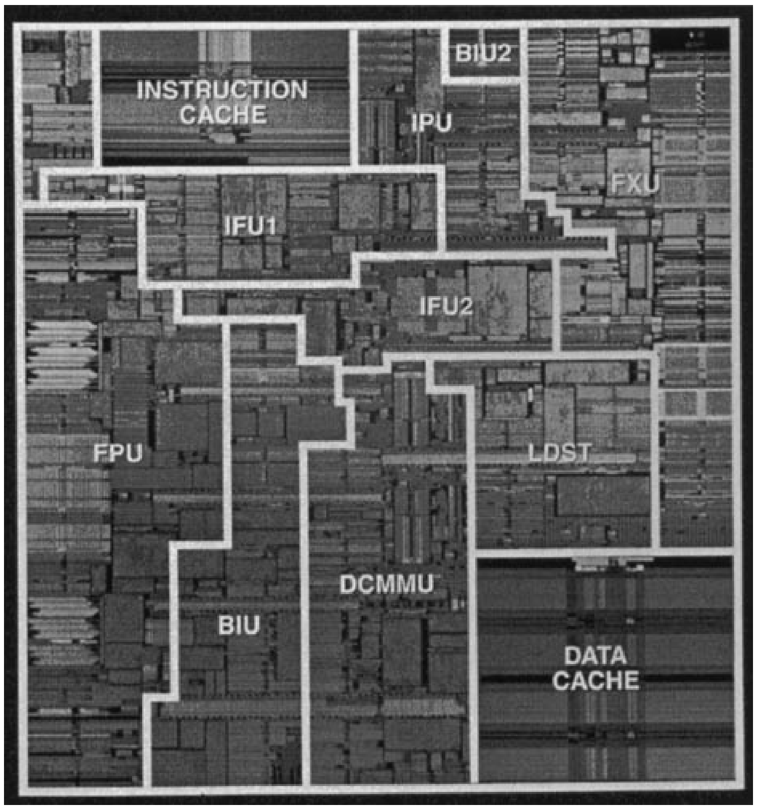


Figure 4. The Power3 microprocessor is an example of custom logic and contains over 15 million transistors. It was fabricated in a hybrid 0.25 to 0.35 μm process and is used in IBM RS6000 workstations. The high density is achieved using highly customized circuits and layout techniques. Photograph courtesy of IBM Corporation, © 1998.

signed to a common height, so that automated layout programs can be used. The combination of all these features allows standard-cell ICs to be designed in far less time, and without requiring the circuit design skills needed for the full-custom approach.

All mask layers in standard-cell circuits are customized for each unique circuit type (e.g., inverter). Therefore, chips containing standard-cell circuits must begin silicon processing with a blank wafer and use custom masks at all levels. Once designed, the transistors and corresponding mask/polygon shapes of a standard-cell inverter are reused each time the inverter is used. These transistor sizes and mask shapes, however, may have little commonality with those used in the standard-cell AND circuit (Fig. 5). Standard-cell circuit designers can change the size of the diffusion area to create the optimal size pMOS and nMOS transistors required by the circuit. Circuits designed to drive a large number of other circuits on-chip require larger transistors than those designed to drive a single element. Polysilicon gates can be jogged to accommodate optimal placement of contacts and wires. Individual gates within a circuit can be electrically isolated from each other by creating separate diffusion polygons. This level of customization results in extremely dense circuit layouts using only the number of transistors required to implement the circuit function. Efficient circuit layouts enable denser chip designs.

Unlike custom circuits, standard cells are designed within certain restrictions, or boundary conditions, to provide efficient automated placement and wiring at the chip level. Consistent or "standard"-cell height, consistent location of the power and ground busses, and common diffusion boundary conditions on the right and left of the cell (Fig. 5) permit automated layout programs that pack the cells efficiently in circuit rows. Logic circuits, RAMs, register arrays, and logic "cores" (large, predesigned blocks of circuits) are placed in the internal area of the logic array. I/O circuits are placed around the periphery of the array in a ring-like structure [Fig. 6(a)] where they can be wired to the IC package connections (20).

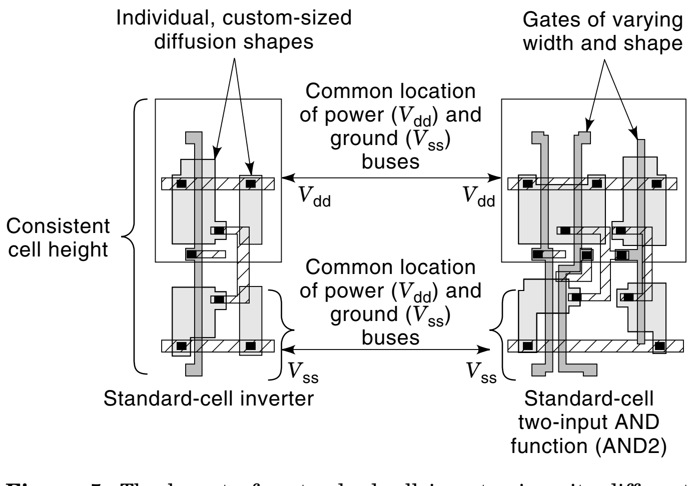

Individual, custom-sized diffusion shapes
Gates of varying width and shape
Common location of power (V_{dd}) and ground (V_{ss}) buses
V_{dd}
V_{dd}
Consistent cell height
Common location of power (V_{dd}) and ground (V_{ss}) buses
V_{ss}
V_{ss}
Standard-cell inverter
Standard-cell two-input AND function (AND2)

Figure 5. The layout of a standard-cell inverter is quite different from the layout of a standard-cell two-input AND function. The shape of each diffusion area is unique. The polysilicon gate in the inverter is essentially straight as opposed to the middle gate in the AND which jogs to the left to avoid overlaying a contact shape. The height of the inverter and AND circuit are the same, a feature that allows an automated placement program to pack into circuit rows efficiently.

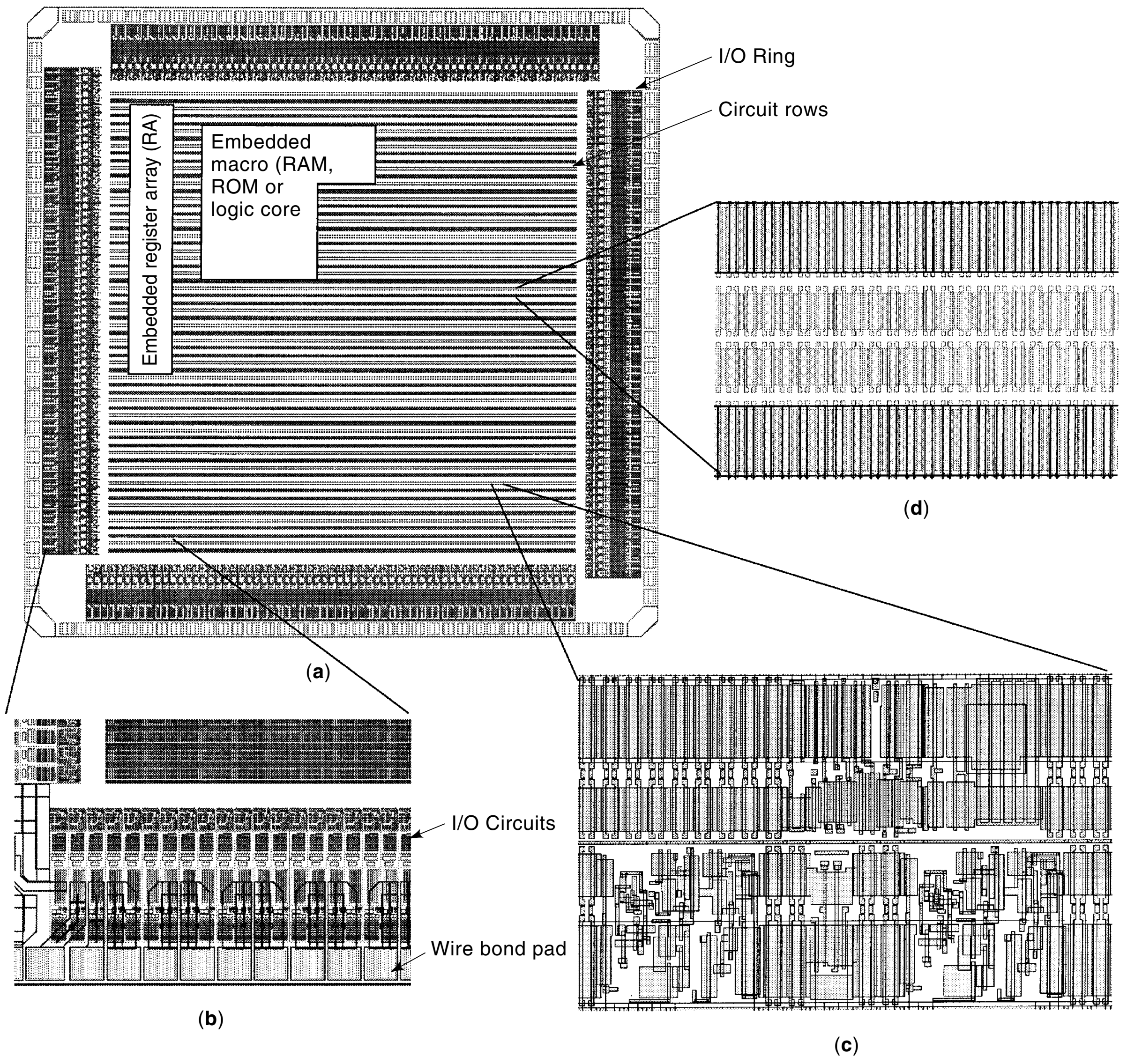

I/O Ring
Circuit rows
Embedded register array (RA)
Embedded macro (RAM, ROM or logic core
(a)
(d)
I/O Circuits
Wire bond pad
(b)
(c)

Figure 6. (a) A sample logic IC footprint. Circuit rows occupy the interior. I/O circuits form a ring structure around the periphery of the array. The white rectilinear shapes represent register arrays or logic cores that can be incorporated into the circuit row area. V_{dd} and V_{ss} metal lines are not included in Figs. 6(a), (b), (c), and (d) to make the circuit structures more viewable. (b) An enlarged view of several I/O circuits and their corresponing wire bond pads. (c) An enlarged view of several circuit rows containing both standard-cell and gate-array circuits. If the two circuit types are designed using common boundary conditions, they can coexist on the same chip. This technique, known as "gate-array backfill," is supported by some ASIC vendors to enable metal-only changes to a standard-cell design (41). (d) An enlarged view of circuit rows that contains only uncommitted gate-array circuits. In the array architecture depicted, the circuit layouts are mirrowed or "flipped" from one row to the next to allow the *p*-MOS and *n*-MOS transistors to share mask shapes at the boundary.

Routing programs contact the input and output nodes on each circuit and connect them to other circuits with metal wires. Routing within a circuit (intraconnect) is predefined by the circuit layout and is usually accomplished using only the first metal layer (M_1). Megacells, such as RAMs, register arrays, or logic cores, may use two or more levels of metal within the circuit, but these are exceptions.

Making connections between individual circuits (interconnect) requires additional layers of metal. Metal wires on odd-numbered levels (e.g., M_1, M_3, and M_5) generally run in the same direction (e.g., horizontally or east to west), while wires on the even-numbered levels (e.g., M_2, M_4, and M_6) run in perpendicular direction (e.g., vertical or north to south direction). The wire direction is alternated on each level to help minimize cross talk between the wiring planes (25). Wires on different levels are connected to each other by a hole, or "via," made between the wiring planes and filled with metal to complete the connection (Fig. 7).

A very large portion of a VLSI chip's area is used for interconnecting the circuits (25). Allowing wires to be placed on

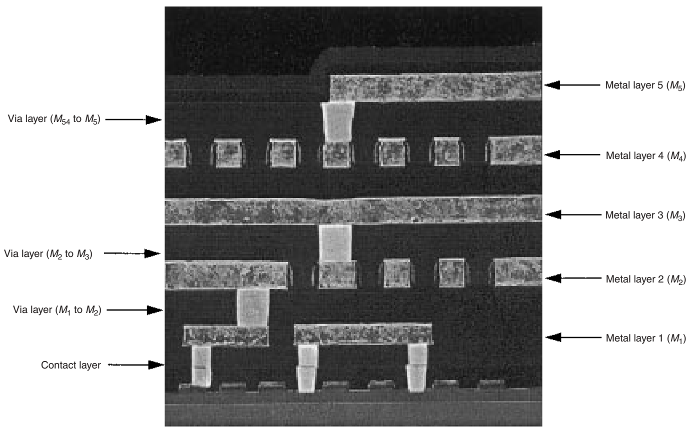


Figure 7. This photograph, taken by a scanning electron microscope (SEM) shows a cross section of the multiple layers of interconnect on a logic-array chip. The metal lines on layers M_1, M_3, and M_5 run in the horizontal direction (East to West), and metal lines on layers M_2 and M_4 run in the vertical (North to South) direction. Photograph courtesy of the IBM Corporation, Copyright 1995.

multiple levels that can via down to the circuits and to each other means that less chip area has to be reserved for wires between the circuits and the circuits can be more densely packed. The number of available metal levels for wiring varies between two and five, depending on the ASIC vendor. Standard-cell products with six levels of interconnection will be in production in the second half of 1998 (13).

Gate-Array Design. Gate arrays differ from standard-cell designs because the individual transistors within a circuit are not customized. Only the contact, wiring, and via mask levels are unique to a particular chip design (21). A predefined pattern of identical transistors is used in gate-array designs, irrespective of individual chip function. Therefore, many ASIC vendors prefabricate wafers containing the transistor-definition layers (e.g., diffusion, implants, polysilicon, and *n*-well) and "stockpile" them for later use.

A gate-array design consists of rows of transistors, usually arranged in two pairs of *n*- and *p*-channels, the minimum number required to form a NAND gate. Before contact and metal shapes are added, these transistors have not been committed to any particular logic function. Contact shapes and the first level of metal are used for the intracell connections that turn the uncommitted transistors into circuits (Fig. 8). Subsequent metal and via layers are used to interconnect the newly formed circuits to each other.

Gate-array designs are similar to standard-cell designs in that they also use predesigned and preverified circuit libraries in the targeted semiconductor process (21). The organization of the die area, or die "footprint", is also similar. Logic circuits are placed in the interior of the chip, and I/O circuits are placed at the perimeter. RAMs and other megacells can coexist with gate-array circuits. However, they are usually designed by custom or standard-cell methods and require additional custom mask levels.

Because the cost of generating masks is part of a design's nonrecurring engineering (NRE) cost, gate arrays incur less NRE than the equivalent standard-cell designs. A three-metal-level gate-array design requires generating only six wiring-related custom mask whereas the equivalent standard-cell design can require more than 20 masks, depending on the process parameters (20). The manufacturing turnaround time (TAT) can also be faster for gate-array designs than for standard-cell if the ASIC vendor has prefabricated wafers in stock.

There are significant drawbacks to gate-array design when compared with other logic design methods. Gate-array circuit layout is limited to one wide diffusion shape having many transistors located in it (Fig. 8). Transistor size cannot be optimized for individual circuit implementations. Because the diffusion cannot be personalized for each circuit, it cannot be broken into two shapes to create two isolated diffusion nodes as in standard-cell. Selected gates must be tied to ground to create the isolation, thereby wasting cell area. Other transistors within a gate-array circuit boundary may not be required to implement a particular cell function, resulting in more wasted silicon area. In simple circuits, such as NANDs and NORs, the difference may be negligible, but for more complex circuits, such as a two-stage latch, the gate-array circuit may be significantly larger than the corresponding standard-cell

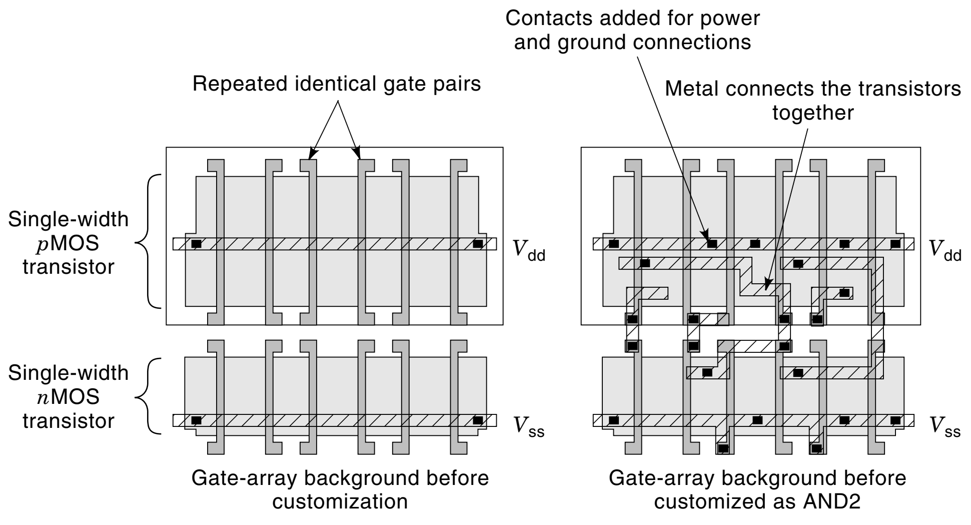

Contacts added for power and ground connections
Repeated identical gate pairs
Metal connects the transistors together
Single-width *p*MOS transistor
Single-width *n*MOS transistor
V_{dd}
V_{ss}
V_{dd}
V_{ss}
Gate-array background before customization
Gate-array background before customized as AND2

Figure 8. A gate-array circuit is shown before and after customization. The transistors in (a) have not been interconnected to implement any logic function. In (b) metal and contact shapes make the required *p*-MOS to *n*-MOS and gate-to-gate connections to implement a two-input AND function (AND2).

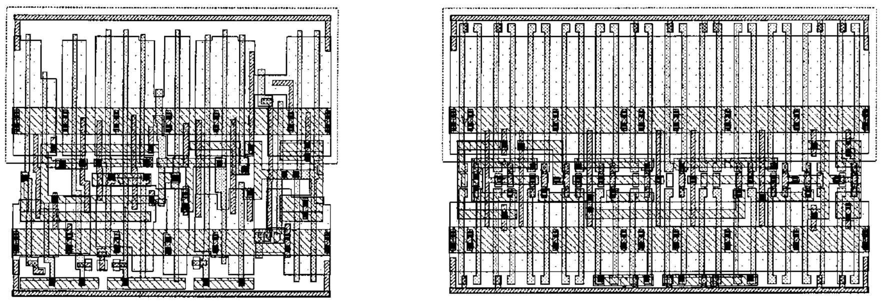


Figure 9. The difference in standard-cell and gate-array is that the circuit area density becomes obvious in larger circuits, such as a two-stage latch. The standard-cell implementation occupies a total of nine array cell locations. The gate-array implementation of the same function requires twelve cells, a 30% area increase.

circuit (Fig. 9). Larger gate-array circuits translate into larger chips, which may translate into higher costs per chip.

Gate-array circuits are usually slightly slower (26,27) and use more M_1 wiring for intracell connections than equivalent standard cells, which impacts wireability at the chip level.

Gate-array design, which dominated standard-cell in the 1980s (20), has been declining in popularity in the 1990s in favor of standard-cell ICs and programmable-logic devices (PLDs). Standard-cell ICs are taking over the high-density, high-performance market whereras PLDs are moving in on the low-end, gate-array market (less than 50,000 gates). Gate-array designs are expected to represent less than 20% of the ASIC market in 1998, down from almost 40% in 1995. During the same period, both standard-cell and PLD design starts are expected to experience a growth rate of approximately 25% (23).

Programmable-Logic Devices. Programmable-logic device (PLD) is a general term that refers to an integrated circuit that can be customized by the end user to implement different designs (28). Users purchase PLDs "off the shelf" from the manufacturer and program the circuits by using software programs and sometimes specialized hardware to implement a particular function. PLDs are usually divided into three categories: simple programmable-logic devices (SPLDs), complex programmable-logic devices (CPLDs), and field-programmable gate arrays (FPGAs).

The SPLDs, in existence since the early 1970s, have a maximum capacity of several hundred gates and are typically used to implement simple control logic and "glue" logic. They consist of planes of interconnected AND and OR circuits "programmed" by fuses or EPROM switches. A CPLD has multiple SPLD-like blocks on a single chip and can accommodate thousands of gates. CPLDs are used to integrate several SPLDs onto a single chip and can be used to implement counters, decoders, and more complex glue logic. The CPLDs are programmed using EPROM or electrical-EPROM (EEPROM) technology. Those using EEPROM technology have the significant advantage of supporting "in-system" programming, that is, the device can be programmed or reprogrammed without removing it from the circuit board. Devices using EPROM technology must be removed from the board and erased using UV light before they can be reprogrammed.

FPGAs became available in the mid 1980s and support the highest gate density of the programmable devices. They can integrate more than 100,000 devices on a single chip and are configured using programmable switches built from either SRAM cells or antifuses. An SRAM-based FPGA can be reprogrammed an unlimited number of times without removing the part from its circuit board. The programming of an antifuse-based FPGA is irreversible and hence supports only one-time programming. Designs implemented as FPGAs are mapped into small units of logic which correspond to the number of devices in a single FPGA logic cell. The content of the logic cell can vary between FPGA products. The most common cell types are look-up table (LUT) and multiplexer-based (28).

Programmable-logic devices are the most expensive of the logic-array products (Table 2) on a per circuit basis and have the lowest performance but offer the fastest time to market. These characteristics make PLDs the device of choice for low-volume, low-gate count, and short-life cycle applications and for prototyping designs that eventually migrate to gate-array or standard-cell. The on-site programmability provided by certain CPLD and FPGA architectures enables designers to get prototype systems up and running and then incrementally refine the design and add new features (29). PLDs also offer a cost advantage over other logic-array types because they do not require paying NRE to the logic manufacturer on a per design basis. Application customization is accomplished electrically by the end user, not by applying unique mask levels during chip fabrication.

PLDs are one of the fastest growing areas in the ASIC market and are expected to grow at a compound annual growth rate (CAGR) of nearly 25% between 1995 and 2001 (23). The average gate count in PLDs has been steadily increasing, allowing PLD use in applications traditionally implemented in gate-array. Many vendors have increased FPGA and CPLD performance enough to serve the needs of many CMOS gate-array applications (29). In 1995, 60% of all CMOS gate arrays were used in applications at 40 MHz or less. Several vendors offer programmable devices running at speeds of 50 MHz to 70 MHz, and some exceed 100 MHz. As a result, PLDs are capturing the low-end, gate-array market (less than 50,000 gates) and starting in 2001 are expected to replace gate arrays at densities under 100,000 gates (23).

TESTING LOGIC INTEGRATED CIRCUITS

After logic-ICs are manufactured, they must be tested to determine if any of the chips contain manufacturing defects or "faults." Initial testing is performed at the wafer level to avoid placing defective ICs in packages. Wafers are placed in a test fixture called a probe station that makes temporary electrical contact with the I/O pads on each die. The tester applies electrical signals to the input pads and compares the signals on the output pads to expected values. The series of input signals and expected output signals applied to the die are called test patterns or test vectors. Chips that pass wafer-level testing are diced, packaged, and tested again before being shipped to the end customer.

The patterns applied to test logic ICs can be derived in different ways. The goal is to create a set of vectors that tests the internal logic circuits or "nodes" in a design and isolates the failing circuit if a defect is present. Traditional methods use a subset of the patterns written to test the function of the logic as it was being designed and simulated. As chip sizes began to exceed 75,000 to 100,000 circuits, writing functional test cases that considered all possible permutations became virtually impossible. The test coverage (the ratio of faults that can be detected to all possible faults) that could be achieved practically by the functional test method was approximately 90%, leaving many circuits untested (30).

Effective testing of VLSI-scale logic requires alternative methods based on design-for-test (DFT) technique (31). Scan-based DFT techniques, used to test large logic ICs by companies, such as IBM for more than 20 years, are gaining in popularity as average chip size is expected to increase from 200 thousand gates in 1997 to over 1.3 million gates by 2002 (42). In scan-based design the internal storage elements (e.g., flip-flops) in an IC are connected in series to form one or more shift-register structures called scan chains. The beginning of each scan chain is connected to a unique chip primary input. The end of each chain is connected to a chip primary output. When testing the IC, data values (test patterns) can be shifted or "scanned" into the individual flip-flops using special test-clock control signals. Then the clock signal(s) that control the logical operation of the IC are activated for a single cycle to exercise the combinational logic which propagates new values into the flip-flops. The test clocks are activated to scan out these new values and compare them to expected values. Register arrays and logic cores can be included in an IC scan chain and tested by this method if designed using full-scan DFT techniques. Embedded RAMs and ROMs are typically tested using built-in self test (BIST) circuitry. The BIST logic itself is tested with full-scan. Using scan, a million-gate IC with hundreds of pins can be tested using a small subset of the total chip pins, which allows the chip manufacturer to use less expensive, lower pin-count testers (33) (Fig. 10).

Test patterns for chips designed by full-scan DFT techniques can be automatically generated with automatic-test-pattern-generation (ATPG) software. The resulting test coverage (the ratio of faults that can be detected to all possible faults) achieved can be very high (99.5%+), resulting in high-quality components (34).

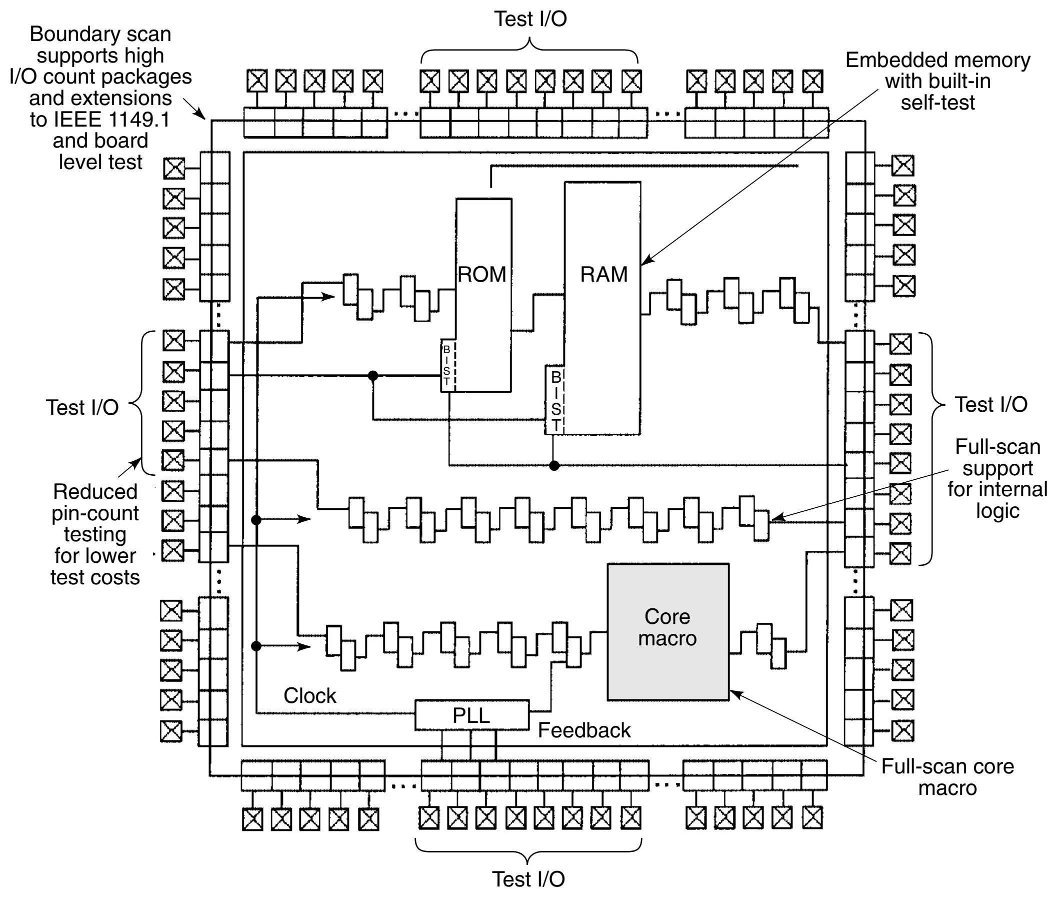

Test I/O
Boundary scan supports high I/O count packages and extensions to IEEE 1149.1 and board level test
Embedded memory with built-in self-test
ROM
RAM
BIST
BIST
Test I/O
Test I/O
Reduced pin-count testing for lower test costs
Full-scan support for internal logic
Core macro
Clock
PLL
Feedback
Full-scan core macro
Test I/O

Figure 10. Figure 10 depicts scannable flip-flops, RAM and ROM BIST logic, and a scannable logic core connected into scan chains. Each scan chain originates at a chip primary input and terminates at a primary output. Boundary scan techniques can also be supported including the IEEE 1149.1 standard for board-level testing.

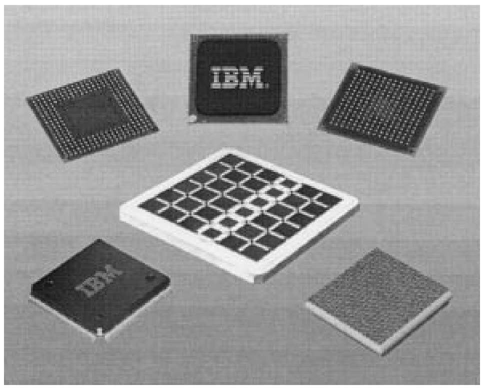
LOGIC PACKAGING AND PIN COUNTS

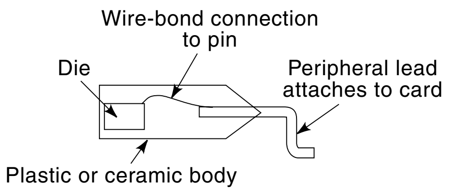
Packaging plays an increasingly important role in determining overall speed, cost, and reliability of logic ICs and the systems in which they are used. An IC package must connect the chip to external signals and power, remove the heat generated by the circuit, and provide physical protection from the surrounding environment. Increases in circuit speed and density drive the need for higher performance packages with more I/O. On-chip circuitry has become so fast that more than 50% of the total system delay now is attributed to the packages. This figure is expected to increase to more than 80% by the year 2000 (7). Increased use in consumer applications demands cheaper, smaller, lighter, and more reliable packages.

In the 1970s and early 1980s, dual-in-line pin (DIP) and pin-grid-array (PGA) packaging technologies dominated. Both package types had metal pins extending from the package body that were mounted and then soldered in holes drilled in a circuit board. The DIP package body was plastic and had a maximum of 64 pins extending from two opposing sides. PGAs were generally made of ceramic, had an array of pins along the entire bottom surface of the package, and could support 300 or more pins.

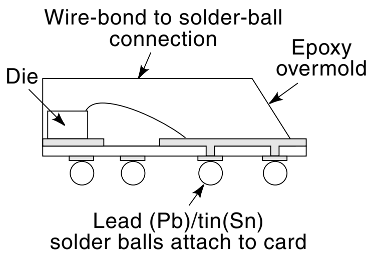
In the 1980s, surface-mount packages became popular because chip packages could be soldered directly to the board's surface without requiring drilled through-holes. Packages could be mounted on both sides of a board, greatly increasing board-level density. Then quad-flatpack (QFP) designs (plastic and ceramic) with densely packed leads on all four sides evolved and are the dominant logic packaging technology of the 1990s.

A bare logic die is usually connected to the package leads (internal package connections) with wire-bonding or "flip-chip" techniques. The wire-bonding process attaches the I/Os on a chip to the package lead frame with individual wires, usually aluminum. As the I/O circuits are more densely packed together, it becomes more difficult to bond them to the package with individual wires without causing shorts (7). Bonding wires are characterized by large parasitic inductance


(**a**)



(**b**)


Wire-bond to solder-ball connection
Epoxy overmold
Die
Lead (Pb)/tin(Sn) solder balls attach to card
(**c**)

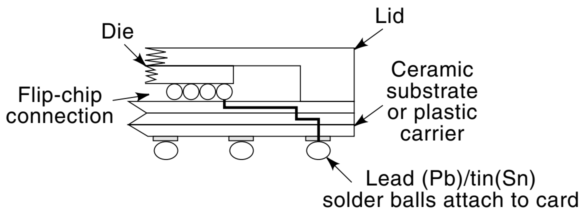

Die
Lid
Ceramic substrate or plastic carrier
Flip-chip connection
Lead (Pb)/tin(Sn) solder balls attach to card
(**d**)

Figure 11. (a) A plastic QFP is shown in the bottom left-hand corner. Three package views of a plastic PGA (PBGA) package appear at the top of the picture. The leftmost view shows multiple rows of solder-ball connections on the bottom surface of the package. The empty space in the center of the package is where the die is attached on the opposite side. The view in the center is the top view of the package. A wire-bonded die is underneath the black epoxy bearing the ASIC vendor logo. The rightmost view shows the bottom view of a PBGA that has solder-ball connections covering the entire bottom surface of the package. The solder-ball pattern used is determined by the number of connections required by the chip and the spacing of the balls required for card attachment. The package in the lower right-hand corner is the bottom view of a CBGA, containing several hundred, densely packed, solder-ball connections. Cutaway views of the die-attach methods used in QFP and the BGA packages are depicted in (b), (c), and (d). The package view in the center of (a) is a multilayer ceramic MCM. This MCM contains 36 chips. Connections between these chips are in the wiring layers internal to the white ceramic substrate. Photograph courtesy of the IBM Corporation, Copyright 1998.

Table 3. Percent Package Usage by Type[a]

	1995	1996	1997	1998	1999	2000
DIP (%)	1.0	1.1	0.7	0.6	0.5	0.3
PGA (%)	12.0	8.0	6.0	4.0	3.0	1.0
QFP (%)	74.0	76.3	65.3	45.9	44.5	39.0
BGA (%)	9.0	11.0	24.0	34.0	44.0	50.7
Other[b] (%)	4.0	3.6	4.0	5.5	8.0	9.0
Percent Package Usage—by Pin-Count Range						
Under 44-pin	0	0	0	0	0	0
44–132-pin	9.5	7.7	6.7	5.4	3.0	1.0
133–195-pin	26.7	25.7	21.3	19.0	15.0	11.0
196–244-pin	29.0	29.0	26.0	24.0	20.0	17.0
245–304-pin	19.1	19.9	22.0	22.0	24.0	26.5
305–388-pin	14.0	14.1	15.1	15.9	17.9	18.9
389–456-pin	0.4	1.0	4.5	5.1	7.0	9.8
457–503-pin	0.1	0.8	1.0	3.1	4.2	6.0
Over 503-pin	0.1	0.3	0.4	0.5	0.7	0.8
Bare die	1.1	1.5	3.0	5.0	8.0	9.0

[a] Ref. 35, courtesy Dataquest Inc./Gartner Group.
[b] Includes chip carrier and bare die.

and can vary in length within a single package, making accurate calculation of package electrical parameters difficult.

Flip-chip technology, invented more than 30 years ago by IBM, solves many of the problems associated with wire-bonding techniques and is becoming increasingly popular in the merchant market. It is a direct-chip-attach technology that attaches the bare die to the package substrate by using solder bumps placed on the surface of the IC. After bumping, the chip is flipped over and the bumps are aligned with the contact pads on the package substrate. The entire assembly is placed in a furnace to reflow the solder balls and establish a bond between the chip and package. Flip-chip technology allows distributing contact pads over the chip's entire surface instead of confining them to the perimeter, as required by wire bonding, which enables many more (over 1000 versus several hundred) I/O connections per die. The connection length with flip-chip packaging is constant for every I/O, and the associated parasitic inductance is minimal.

Many factors must be considered when selecting the appropriate package for a logic array, including cost, speed, pin count, reliability, power dissipation, thermal-expansion characteristics, and package height and weight. Application-specific criteria, such as product life, number of machine on/off cycles predicted, air flow across the package, and simultaneous switching characteristics can also affect package selection.

Quad-flatpacks (QFPs) [Fig. 11(a), (b)] using wire-bond chip attach are the package of choice for low-cost and low-pin count (less than 200 pins) logic ICs and account for 65% of the logic packaging market in 1997. This percentage is expected to decline steadily as chip I/O and average clock speeds continue to increase.

QFPs can support approximately 300 I/Os (36). Although this satisfied the pin-count need of 76% of logic chips in 1997, applications requiring over 300 pins are rapidly growing (Table 3). Higher QFP I/O counts are limited by the fine pitch of the leads which exit the package (pitch is the distance measured from the edge of a pin or wire to the same edge of the adjacent pin or wire). The 0.4 mm pitch on current QFPs makes package-to-board connections extremely difficult and creates a fragile package highly susceptible to lead deformation prior to assembly (36). QFPs are not well suited to high-performance applications because of the long signal paths, high inductance associated with the lead frame and the lack of impedance control.

Ball-grid array (BGA) packages are the fastest growing logic package type, with market share expected to grow from 24% in 1997 to nearly 75% of the market in the year 2000. Ball-grid array packages connect to board assemblies with solder balls (90% lead/10%tin) distributed across the bottom surface of the package [Fig. 11(a),(c),(d)]. Because the I/O connections are not limited to the perimeter of the package, BGAs can support much higher I/O counts in a smaller package size than GFPs. The solder balls are also more rugged than the narrow QFP leads and handle well during the manufacturing process (36). However, BGAs are currently more expensive than QFP alternatives and are more difficult to inspect after board assembly because the leads are not visible. Visibility of the solder joints on the bottom of the BGA package is possible only by X ray.

BGAs with over 600 I/Os are currently in production, and 1000 pin versions using flip-chip attach methods have been announced. Internal connections from the die to the package solder balls can be made using either wire-bond or flip-chip connections [Fig. 11(c), 10(d)]. With BGA packages, the signal lines between chip and package I/Os are shorter because the pin connections do not have to be pushed to the periphery of the package. This feature, in addition to lower inductance on the leads, makes BGAs a preferred option for high-performance applications.

Multichip modules (MCMs) integrate two or more individual chips into a single packaged module. Chips are attached to a module substrate using the flip-chip method. Chip-to-chip connections within the module are accomplished with multiple wiring layers in the ceramic substrate. The number of available wiring planes ranges from as few as three to nearly one hundred (for multilayer ceramic substrates), allowing many chips [Fig. 11(a)] with high I/O counts to be interconnected.

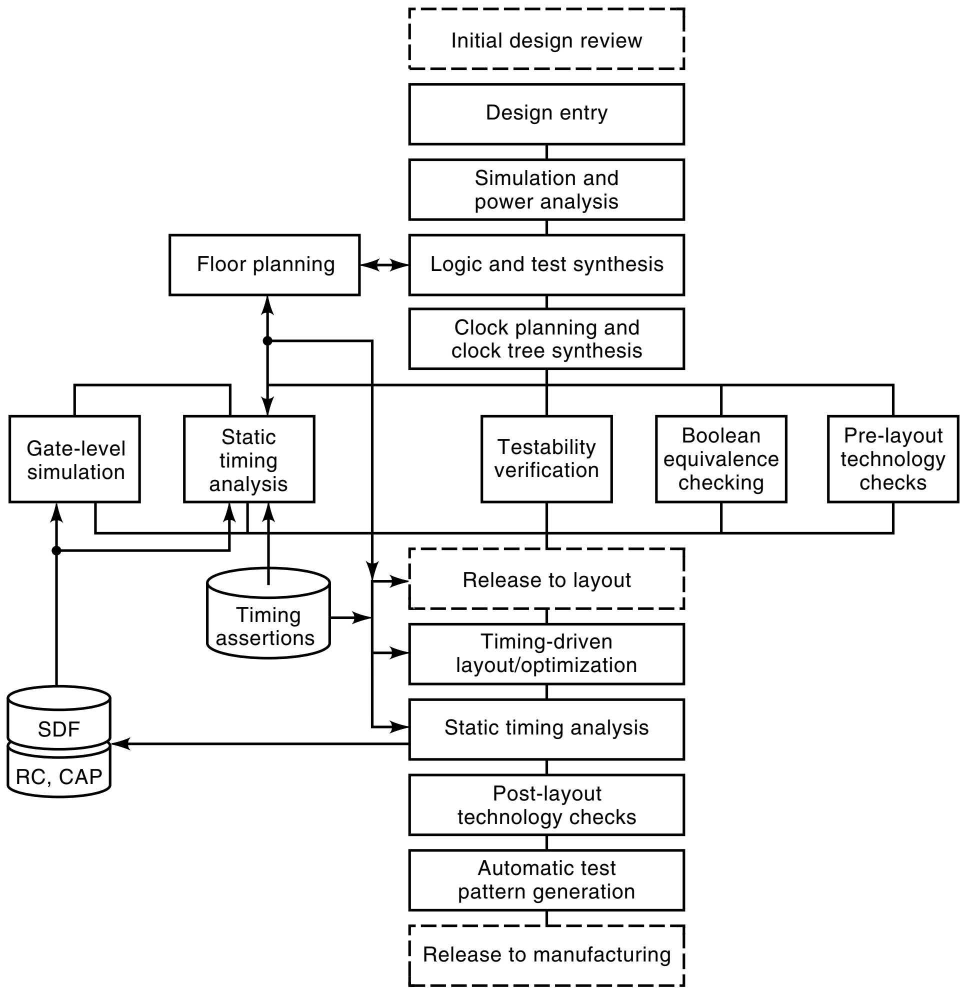


Figure 12. A representative ASIC design flow. Boxes outlined with bold lines are sign-off points between the chip designer and the ASIC vendor. The process steps between the Initial Design Review and the Release to Layout are generally performed by the ASIC logic designer. The process steps between Release to Layout and Release to Manufacturing are generally performed by the ASIC vendor.

HDL view

```
Process (CLK)
 begin
   if (CLK + ‘¡’) and (not CLK’stable) then
   s_counter_output ,+ s_counter_input and not s_reset;
   end if;
end process;
```

Netlist view

```
begin
 U68: INVERT A port map (Z => s_load, A => n265);
 U87: NOR3_4 port map (Z => n275, A => COUNT(3),
 B => COUNTb(4), C => COUNTb(0);
 U88: NOR3_4 port map (Z => n275, A => COUNT(3),
 B => COUNT(4), C => COUNT (0));
 s_ref_ctr_out_reg : D–F–LPH0001–4 port map (L2 =>
 s_ref_ctr_out,D => s-load, E => CLK);
end SYN_refctr_rtl;
```

Physical view

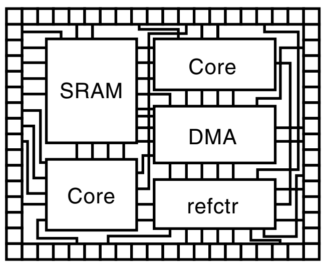


Figure 13. A portion of logic at different levels or abstraction or “views.” The HDL view is independent of any particular ASIC vendor’s circuit library. The netlist view shows the design in its technology-dependent postsynthesis form. The implementation is more detailed than in the HDL view and has been mapped to specific circuits in an ASIC vendor’s library. The physical view depicts the logic in terms of the area it occupies on the silicon die.

Because of the extremely short chip-to-chip connections and the ability to dissipate a large amount of heat, MCMs are the highest performing and most reliable of the logic packages. MCM technology has been in production since the 1970s but is used in relatively few applications. The cost of the multilayer ceramic technology makes MCMs the most expensive package option (7), so that they have traditionally been used only in the highest performing applications, such as mainframe processor units.

LOGIC DESIGN PROCESS

Logic chips are designed by using a combination of hardware and software tools. Workstations are the most common hardware platform for designing custom, standard-cell and gate-array chips. PLDs are programmed predominantly by using Windows-based personal computers. The logic designers use computer-aided design (CAD) software to transform their high-level logic descriptions into the individual circuits manufactured on the silicon wafers. The logic design methodology (Fig. 12) described here assumes the use of standard-cell and gate-array circuit libraries and does not address the design of the individual circuits.

Design Views

During the course of the design process, design data exists in several different formats or views. As the design progresses, it becomes less abstract, more specific to, and optimized for a particular technology. Each step in the design methodology serves a different purpose and requires unique tools. These views evolve through three major phases:

In the initial phase the design is described in a technology-independent Hardware Description Language (HDL), a format very similar to a programming language, to describe the design's functionality (Fig. 13). In the second phase, the design is translated into a technology-dependent netlist that consists of a series of instances of circuits from the ASIC vendor's library, interconnected to implement the functionality described in the previous view. In the last phase, the design is translated into a physical view, in which the logic circuits described in the previous view are physically placed on a piece of silicon, or die, and are interconnected by various layers of wiring.

An ASIC design must go through four key phases to create working silicon: design entry and analysis; technology optimization and floor planning; design verification; and layout.

Design Entry

The designer's first task is to describe the design's intended function. This functionality is typically specified in a document, such as a functional specification, and written in a natural language (English) to facilitate its development and to make it accessible for review by all project team members. Once the specification is finalized, the designer translates the specification into a form that can be understood by software tools to direct the creation of silicon. The two principal design description methods are HDLs, generally used for designs of 20,000 to 30,000 gates or more, and schematic capture, an older method, suitable only for sub-30,000 gate designs and generally less often used today.

The two dominant HDLs are Verilog and VHDL. Both are entered using a text editor on a Windows or UNIX-based workstation. Verilog and VHDL are much like programming languages, like C or Pascal, but they have been specifically designed for describing hardware behavior. Verilog and VHDL are functionally equivalent. The choice of one over the other is driven primarily by the experience base of the design group, the tool set available to the designers to process the HDL, and, possibly, by organizational dictates, such as those of the US government, which requires that all designs be written in VHDL. Verilog dominates the US merchant ASIC market, whereas VHDL prevails in Europe, the US government, and some large US companies, such as IBM.

HDLs allow designers to describe the function of their designs at a high level, often independent of the eventual implementation in silicon, much as a programmer describes a function in the C language without knowing the specific compiler that will create the executable object code.

Figure 14 contains a portion of a direct memory access (DMA) controller written in two different HDLs, VHDL and Verilog. Notice that, though there are syntactical differences between the two languages (for example, VHDL's "entity DMA1 . . ." versus Verilog's "module DMA1 . . ."), the types of language statements and level of description are essentially equivalent. Both HDLs have execution control statements based on the state of a signal called CLK, and both propagate certain design values based on the status of CLK. The language statements are independent of any particular ASIC vendor's library and are at a level of abstraction above any particular logic circuit implementation. For example, such statements might be at a behavioral level or register transfer language (RTL) level. Whatever the level, an HDL can be implemented in several different ways, using different combinations of circuits from any one of several different ASIC vendors' libraries.

Design Analysis

After entering a design in an HDL, the designer begins the process of analyzing what was entered to determine if it correctly implements the intended function. The traditional method is through simulation, which evaluates design behavior. Simulation is a mature, well-understood process, and there are many simulators available that accept HLDs written in VHDL, Verilog, or increasingly, both languages. The HDL that describes the design is read into the simulator tool. Then the simulation process is driven by a set of simulation stimuli, called "input vectors" to which the design reacts. The resulting output values from the design or "output vectors" are captured and compared to expected values. If the output values compare, the simulation is said to "pass." If the actual output values differ, then the simulation is said to "fail," and the design needs to be corrected.

A more recent addition to the design analysis phase is power analysis. For a growing number of customers, the power consumption and dissipation of their designs are becoming critical factors. Early feedback on design power requirements allows designers to make timely design trade-offs to achieve power targets.

The traditional method for calculating power has been primarily pen and paper calculations using technological information provided by the ASIC vendor and switching informa-

DMA Controller

VHDL

```
entity DMA1 is

  port(CLK            : IN STD_LOGIC;
        RESET         : IN STD_LOGIC;
                    . . .

         FIFO_RESTART: BUFFER STD_LOGIC;

                    . . .
--*process to create latches

architecture DATAFLOW of DMA1 is

   process

     begin
         wait until (CLK'EVENT and CLK='1');
         OUT_END1_L2 <= OUT_END1_SIG;
         OUT_END1_L1L2 <= OUT_END1_L2;
         OUT_END2_L2 <= OUT_END2_SIG;
         OUT_END2_L1L2 <= OUT_END2_L2;
            . . .
      end process;
```

Verilog

```
module DMA1(CLK, REST, FIFO_RESTART,...)

     input CLK;
     input RESET;
                  . . .

     output FIFO_RESTART;

//* process to create latches


  always

   begin : block_578
     @ (posedge CLK);
     OUT_END1_L2 <= OUT_END1_SIG;
     OUT_END1_L1L2 <= OUT_END1_L2;
     OUT_END2_L2 <= OUT_END2_SIG;
     OUT_END2_L1L2
      <= OUT_END2_L2;
       . . .
   endmodule;
```

Figure 14. Logic from a DMA controller can be described in multiple HDLs. This figure shows equivalent VHDL and Verilog descriptions.

tion supplied by the customer. This method is inadequate in terms of scope and accuracy for today's power-conscious designs. Estimates of the amount of power a design consumes and dissipates before it has been mapped to a specific technology can vary by more than 50% from the actual silicon even when using CAD tools. Therefore, power estimation should be repeated after technology implementation to obtain more accurate predictions.

Technology Optimization

The technology optimization process takes a technology-independent description of a design and maps it to a library of logic circuits provided by an ASIC vendor, thereby making the design technology-dependent. This phase seeks a correct mapping and also the most efficient one in terms of the customer requirements. The optimization process is divided into subprocesses: logic synthesis, test insertion, clock planning and insertion, and floor planning.

Logic Synthesis. Logic synthesis transforms a design's HDL representation into technology-specific logic circuits. An ASIC vendor provides the logic circuits in a form called a "synthesis library." As the synthesis tool breaks down high-level HDL statements into more primitive functions, it searches this library to find a match between the functions required and those provided in the library. When a match is found, the synthesis tool copies the function into the design (instantiates the circuit) and gives it a unique name (cell-instance name). This process continues until all statements are broken down and mapped (synthesized) to logic circuits. Potentially hundreds, even thousands, of different logic circuits combinations can implement the same logical function. The combination chosen by a synthesis tool is determined by the design constraints provided by the designer. The constraints define the design's performance, power, and area targets. A design driven primarily by performance criteria may use larger, faster circuits than one driven to minimize area or power consumption. Synthesis tools have matured during the past 5 to 8 years and are used in virtually all ASIC design starts today.

The inputs to the logic synthesis process are the HDL design description (VHDL or Verilog), the design constraints, and the synthesis library provided by the ASIC vendor. The output of the synthesis process is a netlist, which is a list of circuit instances interconnected to implement the logical function of the design. The netlist can be written in several different formats or languages. The dominant netlist languages are VHDL, Verilog, and Electronic Design Interchange Format (EDIF). The interconnected circuits may also be graphically represented as schematics.

The HDL design description (in VHDL) shown in Fig. 15(a) is a technology-independent description of a counter function called refctr. Take note of the statements in the dotted box that assign the value of a signal s_load to a signal s_ref_ctr_out based on the status of CLK.

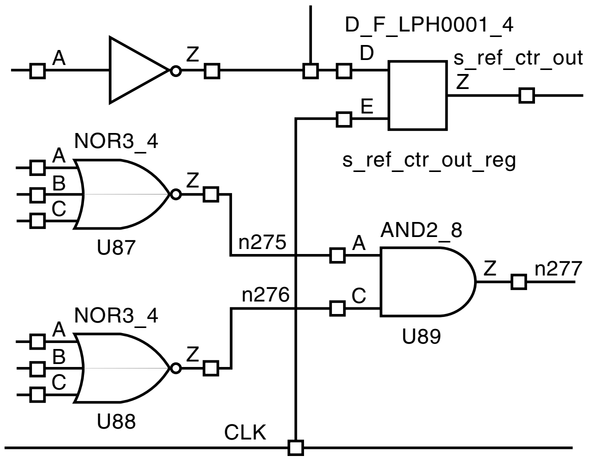
Figure 15(b) depicts a post synthesis schematic view of a portion of refctr. Notice that the design was mapped to specific logic-circuit functions, such as INVERT_A, NOR3_4, and D_F_LPH0001_4. These names correspond to circuit names found in an IBM ASIC CMOS 5S Databook. Each circuit has a unique name, such as U87 for one instance of NOR3_4 and U88 for another instance of NOR3_4. The instance names U87 and U88 were generated by the synthesis tool as it mapped the HDL function into logic circuits, such as NOR3_4.

Signals generated by the synthesis tool as it mapped the HDL to logic circuits appear with names, such as n275 and n276. Signals explicitly named in the HDL, such as sload and CLK, are retained. Notice that sload and CLK feed into a circuit that generates the signal s_ref_ctr_out, as described in the technology-independent source in Fig. 15(a).

Figures 15(c) and (d) contain the refctr postsynthesis netlist output in VHDL and Verilog, respectively. The circuits

```
entity
    port (COUNT    in  std_ulogic_vector(5 downto 0);
    CLK            in  std_ulogic,
    RESET          out std_ulogic);

architecture refctr-rtl of refctr is
  signal s_ref_ctr_out       : std_ulogic;
  signal s_load              : std_ulogic;
  s_next_ctr_val             : std_ulogic-vector(5 downto 0);
  s_counter_input            : std_ulogic-vector(5 downto 0);
  s_counter_output           : std_ulogic-vector(5 downto 0);
  s_reset                    : std_ulogic-vector(5 downto 0);

begin
    s_reset(0) <= RESET;

process(CLK)
 begin

  if (CLK = '1') and (not CLK'stable) then
    s_counter_output <= s_counter_input and not s_reset;
    s_ref_ctr_out <= s_load;
  end if;

 end process;

end refctr_rtl;
```

(a)



(b)

```
entity refctr is
  ...
architecture SYN_refctr_rtl of refctr is
  ...
component INVERT_A
  port(Z:out std_logic; A:in std_logic),
end component

component NOR3_4
  port(Z:out std_logic, A,B,C:in std_logic);
end component;

component AND2_8
  port(Z:out std_logic;A,B:in std_logic);
end component,

component D_F_LPH0001_4
  port(L2:out std_logic,D,E:in std_logic);
end component,
  ...
begin
  U69:INVERT-A port amp(Z=>s-load,A=>n265);
  U87:NOR3_4 port map(Z=>n275,A=>COUNT(3),
      B=>COUNT(4),C=>count(0));
  U88:NOR3_4 port map(Z=>n276,A=>COUNT(5),
      B=>COUNT(2),C=>count(1));
  s_ref_ctr_out_reg:D_F_LP0001_4 port map(L2+>
      s_ref_ctr_out_reg,D+>s_load,E=>CLK);
end SYN_refctr_rtl;
```

(c)

```
module refctr (COUNT, CLK, RESET,REF);
    ...
  INVERT_A U68(.Z(s_load), A(n265)),
  NOR_4 U87(.Z(n275),.A(COUNT[3]),
    .B(COUNT[4}), C(COUNT[0]) );

  NOR3_4 U88(.Z(n276),.A(COUNT[0}),
    .B(COUNT[2]),.C(COUNT[1]) );

  AND2_8 U89(.Z(n277),.A(n275),
    .B(n276));

D_F_LPH0001_4 s_ref_ctr_out_reg(.L2(s_ref_ctr_out),
    .D(s_load).E(CLK));

    ...
end module;
```

(d)

Figure 15. (a) A technology-independent description of a function called refctr. The portion of the VHDL within the dotted box is shown in three equivalent technology-dependent views. (b) A logic schematic. (c) Technology-dependent gate-level VHDL. (d) Technology dependent gate-level Verilog.

described, along with net names and instance names are exactly the same. The difference is in the descriptive syntax. EDIF syntax is more verbose, and the associated data volume of an EDIF netlist is a drawback. Nonetheless, EDIF is an industry standard and is accepted by almost every electronic design automation (EDA) tool on the market.

Test Insertion. Test insertion consists of inserting structures into the design to enable a complete and efficient manufacturing test. Nonscan latches and flip-flops are replaced with scannable versions and connected into scan chains. Scan test clocks are introduced and connected to the appropriate pins on the latches, memory elements (SRAMs or ROMs), and scannable logic cores (Fig. 10). All scan test clocks are brought out to predesignated chip primary inputs that will be contacted at wafer and module test time. Scan chains are connected to chip primary inputs and outputs to allow scanning test patterns in and out by tester software. Test insertion may also include inserting boundary scan logic (scannable latches that control each chip primary input and output) and self-test logic. The test-insertion process must be done in accordance with the ASIC manufacturer's design-for-test (DFT) requirements to allow automatic generation of test patterns for manufacturing test.

Clock Planning and Insertion. The last phase of the technology optimization process is planning and inserting the clock network. Every ASIC design has at least one clock. Many of the large and more complex ASIC designs have multiple clocks, in some cases, twenty or more. The manner in which the clock network is propagated throughout the design to the clocked circuits (such as latches, flip-flops and other logic circuits that need to be synchronized with the clock signal) can vary from vendor to vendor and involves trade-offs among various design parameters: die area, delay through the clock network to the clocked circuits (latency), the variation in clock arrival time at the various clocked elements (skew), and the power generated by the clock network as it switches.

Because clock networks are usually the most power-hungry nets in a design, design techniques such as clock-gating are particularly important for chips used in portable applications (37). The clocking methodology must comply with the DFT requirements to maintain design testability.

Floor Planning. Floor planning is the process of placing groups of circuits on a die and analyzing the effect of that placement in terms of design performance and routability. The need for floor planning arose as circuits became smaller and the length of the wires that interconnected those circuits began to dominate design performance trade-offs. This is often referred to as one of the "deep-submicron" ($<0.5\ \mu$m) design paradigms where interconnect delay dominates the delay through the individual circuits or gates. Integrating floor planning into the prelayout portion of the methodology allows the designer to consider the physical design implementation during the logic-design process. With floor planning, trade-offs on design partitioning, I/O assignment, and macro location assignments can be made early, thereby avoiding costly design iterations between layout and synthesis.

By physically placing groups of logic on a die, more accurate estimates can be made of the wire lengths within the logic groups (shorter, faster nets) and the wires interconnecting them (longer, slower nets). More accurate estimation of wire lengths that interconnect the logic on-chip translates into more accurate wire-delay predictions, which greatly affect the overall design timing. The wire-length estimates from floor planning can be passed back to the synthesis tool and used to further optimize the selection of logic gates chosen to implement a function. The floor-plan grouping information can also be passed directly to the ASIC vendor's detailed place-and-route tools. This can improve the turnaround time through the design center for the die layout. Floor planning also helps to monitor the actual design size which eliminates discovering later (during the layout phase) that a design has outgrown its target die size.

Design Verification

The design verification performed at this point in the design process ensures through automated checking that the design is functionally correct and meets physical constraints in terms of performance, testability, power, and technology-specific electrical checks.

Functional Verification. As we have seen, designs are functionally verified before synthesis by using simulation. Now, after synthesis, the design is resimulated to ensure that its function has not been corrupted by the technology optimization process. As synthesis tools have matured, the likelihood of introducing functional errors during synthesis has been drastically reduced. Nonetheless, it is still advisable to verify the technology-mapped version of the design. The traditional verification method is to resimulate the gate-level version of the design. The process is straightforward. The gate-level version of a design should produce the exact same functional results as the presynthesis version of the design, given the same set of stimuli (input vectors). Unfortunately, as designs exceed 100,000 gates, the elapsed time required to rerun simulation vectors becomes prohibitive. Designs of up to one million gates can take weeks or more of simulation time to complete functional verification. Boolean equivalency checking is an alternative verification method that requires less time.

Boolean Equivalency Checking. Boolean equivalency checking (BEC), also called "formal verification," achieves the same purpose as gate-level simulation, which is to guarantee that the function of the design was not altered or corrupted by the technology optimization process, but the method is very different. A BEC tool breaks down a design into a set of Boolean or logical expressions. This process is repeated on a second version of the design, and the logical expressions are compared for equivalence. Although the comparison of the two designs is exhaustive, it is not driven by evaluating different design states created by input vectors. No input or output vectors are required. Compared to the hundreds of hours required by simulation, verification of a 500,000 gate design through formal verification can be done in approximately three hours.

BEC can also be used to compare two technology-dependent versions of a design for equivalence, such as comparing the post-test insertion version of the design against the netlist from logic synthesis or the postlayout version of a design against the prelayout version.

Testability Verification. Testability verification ensures that the design, as implemented by a specific set of circuits, can be tested in the manufacturing facility. Most new ASIC designs are migrating to scan-based design techniques that allow automatically analyzing the logic for compliance with test requirements by using DFT software. Compliant designs require no further action on the part of the customer to support manufacturing test. Test patterns are automatically generated by ATPG software. A variety of scan-based methods exist (e.g., level-sensitive scan design, mux-based scan) and are supported in varying degrees by ASIC vendors.

Timing Verification. The purpose of timing verification is to determine if a design, once mapped to a specific library of circuits, meets the specified performance target. Traditional methods based on gate-level simulation are being replaced by static timing analysis because of long run times and design-coverage issues.

Static timing analysis allows examining all paths on a die (under best- and worst-case conditions) in a single timing run. Static timing on a large design (for example, 860,000 gates) can be achieved in two to three hours, compared to the many days or weeks required to get equivalent coverage (if possible) using delay simulation. The move away from timing simulation and toward static timing analysis is the industry trend, and ASIC vendor support for static timing is becoming more common.

Prelayout Technology Checks. A final set of technology- and library-specific design verification checks is usually provided by the ASIC vendor. These checks verify a variety of ASIC vendor requirements. Examples include verifying that all input pins on each circuit in the design are used (connected to another circuit) and verifying that all circuits that communicate to tester equipment are located in the required I/O slots.

Layout

The layout process involves physically implementing the design in silicon. Layout is traditionally performed by the ASIC vendor at ASIC design centers. The design centers may be located at the actual silicon foundry site or at satellite locations.

Layout can be broken into two different steps: place and route and the return of postlayout timing values to the chip designer (back-annotation). Floor planning can be considered part of the layout process, part of the technology optimization process, or both, and can be performed by either the customer or the ASIC vendor. Floor planning straddles the traditional front end (that is, the logic-design process) and the back end (the physical-design process) and helps to yield optimum results from both. Because early, prelayout floor planning is an essential ingredient to successfully designing deep-submicron ASICs, floor planning has already been discussed in the Technology Optimization section.

Traditionally, designs were placed and routed by the ASIC vendor and then retimed to see if the original performance target was achieved. If the performance target was missed, the customer had to change the logic. This often meant resynthesizing blocks of logic and then repeating the design verification and place-and-route steps. Multiple iterations through this process to achieve timing closure could add weeks or even months on top of the original schedule. With floor planning, earlier analysis can be done by the customer and the design reoptimized before entering the layout process.

Place and Route. Floor planning consists primarily of placing and interconnecting groups or clusters of logic, whereas place and route involves placing and interconnecting each circuit on a die. With today's large chips (containing over 1 million logic gates), place and route is comparable to solving a jigsaw puzzle with hundreds of thousands of pieces. From the chip designer's perspective, the end result must fit in the allocated area and must also meet the performance targets, and all this must be achieved on schedule. In addition, the ASIC vendor has more criteria for success in terms of technological constraints (e.g., no electromigration) and testability (i.e., test-related circuits are properly connected).

Most layout tools place circuits most efficiently from an area standpoint. Adjustments to that placement to improve timing are largely a manual process and may require the chip designer to change the actual logic. Advanced place-and-route methods are timing-driven, that is, the placement algorithm in the tools consider the performance constraints of the design as the circuits are placed. To drive the layout process, some ASIC vendors use the timing information developed by the chip designer for timing sign-off with a static timing analyzer (5,38). Then the placement tools work to create a layout that has the most efficient area utilization and meets the timing assertions. If, after placement, paths remain that do not meet the specified timing, a series of automated placement optimization routines are run, varying the drive strength of logic circuits and relocating clock driver cells until timing closure is achieved.

Because the assertions completely describe the timing performance targets of the design, the optimization can be performed without intervention from the chip designer and without requiring resynthesizing the design, which can translate into real time-to-market savings. Timing-driven layout can handle a range of design sizes from less than 50,000 to over 2 million gates, and can accommodate a flat (all circuits are placed on the die simultaneously), partitioned (circuits are placed using the grouping and preplacement information from floor planning), and hierarchical (individually placing routing sections of the die and interconnecting these sections with global wires) placement approach.

After the circuits are placed, they are interconnected, according to the netlist specification, by automated routing programs. The manner in which the circuits are interconnected depends on many factors, including the minimum and maximum wire widths and spacing between wires allowed by the manufacturer; the number of available wiring connection points on each circuit; and the number of layers of wiring available. More levels of wiring translate into the ability to interconnect more circuits on a given die, allowing for denser, more integrated designs. Wiring programs attempt to make all the interconnections described in the netlist. If the automated program cannot find a solution for all nets, manual routing of some nets may be required. Because manual intervention on a large, complex design is extremely time-consuming, it may be more efficient to adjust the placement of the circuits and rerun the routing program than attempt to wire the remaining nets by hand. Final timing of a chip is performed after placement and routing is complete.

Timing Back-Annotation. Timing back-annotation is the process of extracting timing information from one design step to analyze in an earlier step. For example, classic back-annotation uses postlayout delay information in a timing simulation. Because the actual distance between logic circuits on the die is known once a design is placed and routed, the corresponding wire delay can be calculated with great accuracy. Then this delay information is extracted from the layout and is written in a form that the simulator can understand. The industry standard for this type of delay information is the Standard Delay File (SDF).

The SDF can be read into a simulator for postlayout, gate-level timing verification. This process is orders of magnitude slower than gate-level simulation without timing and is impractical for large designs. Repeating static timing analysis after layout, using back-annotated resistance and capacitance

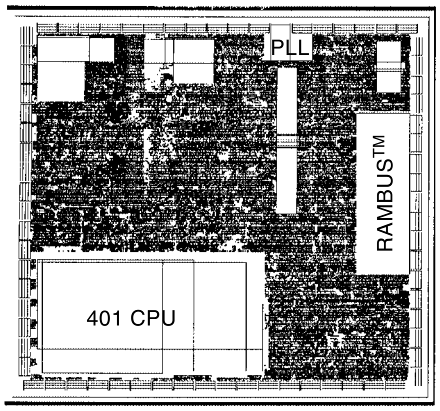


Figure 16. This system-on-a-chip contains several cores: a PowerPC microprocessor, a Rambus high-speed memory interface, an analog phase-locked loop, multiple RAMs and register arrays; all integrated with hundreds of thousands of standard-cell logic gates. Each of the cores was predesigned and preverified by the ASIC vendor and can be reused in multiple logic designs. Only the standard-cell logic gates are unique to the chip design.

information, is becoming a popular alternative for postlayout timing sign-off.

Final Checking and Release to Manufacturing. The last step in the logic design process involves generating the mask shapes used in the fabrication process and the manufacturing test vectors. A design rule check (DRC) is run to ensure that the mask shapes meet the technology specifications for minimum widths and spacing. Lay-out-versus-schematic (LVS) checking is run to ensure that the function implemented by the shapes exactly matches the function described by the schematic representation. Manufacturing test patterns are generated using ATPG software and transmitted to the test floor when the completed wafers arrive.

TRENDS

One of the most important trends in logic-IC design in system-level integration (SLI), also called system-on-a-chip (SOC) design. Now microprocessors, ASIC logic, and analog functions, once discrete components on a board, can be integrated onto a single piece of silicon. SLI is a result of millions of available circuits and also of the increasing availability of predesigned, preverified logic functions called "cores." A core can be either "hard" or "soft," implemented as fixed mask shapes or as synthesizable logic, respectively. Both types have the advantage of being preverified and available "off the shelf" for incorporation into a new design. The availability of predesigned, preverified logic allows chip designers to integrate functions more quickly and get designs to market faster than if all logic were designed from scratch (39). SLI ASICs are entering the market at a rapid pace, with many high-volume applications slated to use the technology, including personal electronics, video games, set-top boxes, portable computing, portable communications, and multimedia (32). Figure 16 shows an SLI ASIC containing an embedded microprocessor (PowerPC 401), a mixed analog-digital high-speed memory interface (Rambus), an analog phase-locked loop (PLL), multiple SRAMs and register arrays (RAs), and application-specific standard-cell logic. The processor, Rambus memory interface, PLL, RAMs, and RAs were implemented as hard cores. Soft-core logic for a serial port unit was also used.

SLI ASICs are expected to grow from approximately 20% of the standard-cell market in 1997 to over 60% of the market by 2002 (42), thereby dominating new ASIC design starts. As designs continue to grow in size and complexity, the ability to verify design function adequately is becoming the gate to shortening development cycles. Approximately two-thirds of design team resources are devoted to verification. The volume of test-bench code is growing at a much faster rate than the design itself. SLI ASIC design is driving new verification methodologies that involve hardware-software cosimulation, cycle-based simulation, and more abstract design languages (system-level design languages or SLDs).

The ability to produce faster logic ICs that consume and dissipate less power will continue to be a significant driver of logic design, circuit design, and silicon process design in the future. Several silicon manufacturers are investigating a new technology that integrates bipolar devices enhanced with germanium with CMOS for use in high-speed, low-power communication applications. The resulting BiCMOS silicon germanium technology shows promise because of its incredible performance (65 GHz) and because it can be integrated into existing CMOS manufacturing lines (40).

The affordability of building new and more advanced manufacturing facilities is also a major challenge for the logic-IC industry and the semiconductor industry at large. The capital requirements for new technologies have risen exponentially along with component densities. The cost of building a fabrication plant has gone from $14 million in 1966 to $1.5 billion in 1995 and is expected to hit $3 billion before the year 2000 (3), a fact that has motivated competing semiconductor manufacturers to build joint facilities to make costs more affordable. It may be economics, not the lack of innovation or the laws of physics, that brings an end to the exponential growth in IC density quantified by Moore's law (3).

BIBLIOGRAPHY

1. L. M. Terman, The role of microelectronics in data processing, *Sci. Amer.,* **237** (3): 162–179, 1977.
2. Semiconductor Industry Association (SIA) Roadmap 1997, *The National Technology Roadmap for Semiconductors,* 1997 Ed., Austin, TX: SEMATECH, 1997.
3. R. R. Schaller, Moore's Law—Past, Present, and Future, *IEEE Spectrum,* **34** (6): 53–59, 1997.
4. G. E. Moore, Cramming more components onto integrated circuits, *Electron. Mag.,* April 19, 114–117, 1965.
5. A. M. Rincon, M. Trick, and T. Guzowski, A proven methodology for designing one-million-gate ASICs, *IEEE Custom Integr. Circuits Conf.,* 1996, pp. 45–52.
6. C. C. Chuang, Semiconductors (Fabrication), *Kirk-Othmer Encyclopedia of Chemical Technology,* vol. 20, 3rd Ed., New York: Wiley, 1982, pp. 634–654.
7. H. B. Bakoglu, *Circuits and Packaging for VLSI,* VLSI System Series, Reading, MA: Addison-Wesley, 1990.
8. J. G. Petrovick et al., A 300 K Circuit ASIC Logic Family, *IEEE Custom Integr. Circuits Conf.,* 1990.
9. R. H. Dennard et al., Design of ion-implanted MOSFET's with very small physical dimensions, *IEEE J. Solid State Circuits,* **SC-9**: 256–267, 1974.

10. K. C. Saraswat and F. Mohammadi, Effect of scaling on interconnection on the time delay of VLSI circuits, *IEEE J. Solid State Circuits,* **SC-17**: 275, 1982.
11. J. Gallant, Deep-Submicron geometries dictate new approaches to ASIC design, *EDN Mag.,* 65–73, June 8, 1995.
12. S. A. Schwartz, Semiconductor theory and applications, *Kirk-Othmer Encyclopedia of Chemical Technology,* vol. 20, 3rd ed., New York: Wiley, 1982, pp. 601–633.
13. R. Dornseif, IBM transfers first copper metallization process into production, *Dataquest Perspective, ASICs Worldwide Technology Analysis,* ASIC-WW-DP-9710, October 13, 1997.
14. Soul of a new chip, *Think Mag.,* **64** (1): 16–27, 1998.
15. J. Y. Chen, CMOS-the emerging technology, *IEEE Circuits Dev. Mag.,* **2** (1): 16–31, 1986.
16. M. A. Olsson, Logic 2001, *Dataquest Perspective,* Semiconductors Worldwide, SEMI-WW-DP-9707, June 16, 1997.
17. G. L. Moss, Logic elements: IC logic family operation and characteristics, in J. C. Whitaker, (ed.), *The Engineering Handbook,* Boca Raton, FL: CRC Press, 1996, pp. 1613–1622.
18. M. R. Zargham, S. Tragoudas, and J. L. Seely, Integrated circuits: Layout, placement and routing, in J. C. Whitaker, (ed.), *The Electronics Handbook,* Boca Raton, FL: CRC Press, 1996, pp. 581–590.
19. J. L. Seely, Integrated circuits: Application-specific integrated circuits, in J. C. Whitaker, (ed.), *The Electronics Handbook,* Boca Raton, FL: CRC Press, 1996, pp. 591–602.
20. C. K. Erdelyi et al., Custom and semi-custom design, in S. Groto (ed.), *Design Methodologies,* New York: Elsevier, 1986, pp. 3–41.
21. D. E. White, *Logic Design for Array-Based Circuits, A Structured Design Methodology,* New York: Academic Press, 1992.
22. N. E. Einspruch and J. L. Hilbert (eds.), *Application Specific Integrated Circuits (ASIC) Technology,* New York: Academic Press, 1991.
23. Worldwide ASIC Forecast, Spring 1997, *Dataquest Market Statistics, ASICs Worldwide,* ASIC-WW-MS-9702, June 23, 1997.
24. S. O. Agbo and E. D. Fabricius, Integrated circuit design, in R. C. Dorf, ed., *The Electrical Engineering Handbook,* Boca Raton, FL: CRC Press, 1993, pp. 654–674.
25. P. S. Ho, Part 3 VLSI interconnect metallization, *Semicond. Int.,* 128–133, August, 1985.
26. *CMOS 5S ASIC Products Databook,* Int. Bus. Mach. Corporation, 1996.
27. *ASIC SA-12 Databook,* 2nd ed., Int. Bus. Mach., January 1998.
28. S. D. Brown, *Field-Programmable Devices, Technology, Applications, Tools,* Los Gatos, CA: Stan Baker Associates, 1995.
29. F. Caruthers, Programmable logic muscles in on gate-array designs, *Comput. Des.,* 91–100, April 1995.
30. B. Tuck, Complex ASICs straining verification resources, *Comput. Des.,* 47–60, January 1997.
31. E. Eichelberger et al., *Structured Logic Testing,* Englewood Cliffs, NJ: Prentice-Hall, 1991.
32. B. Lewis, SLI to dominate ASIC market by 2000, *Dataquest Perspective, ASICs Worldwide,* ASIC-WW-PD-9502, December 19, 1995.
33. A. Rincon and D. Lackey, Whose job is it to design testable ASICs?, *Comput. Des.,* November 1996.
34. P. S. Gillis et al., Test methodologies and design automation for IBM ASICs, *IBM J. Res. Dev.,* **40** (4): 461–474, 1996.
35. Integrated Circuit Packaging, *Dataquest Focus Report, Semiconductors Worldwide,* SEMI-WW-FR-9702, June 9, 1997.
36. M. Kuzawinski, Plastic ball grid array chip carriers, *IBM Micronews,* **2** (4): 5, 4th quarter 1996.
37. J. Lipman, Growing your own clock tree, *EDN Mag.,* **42** (6): 41–48, March 14, 1997.
38. J. J. Engel et al., Design methodology for IBM ASIC products, *IBM J. Res. Dev.,* **40** (4): 387–406, 1996.
39. A. M. Rincon et al., Core Design and System-on-a-Chip Integration, *IEEE Des. Test Comput.,* **14** (4): 26–35, 1997.
40. R. Wilson, IBM leads the charge to SiGe production—surge in foundry activity threatens GaAs's role in RF, *Electron. Eng. Times,* January 1998.
41. R. Gregor et al., A one-million-circuit CMOS ASIC logic family, *Custom Integr. Circuits Conf.,* Piscataway, 1993, pp. 23.1.1–23.1.4.
42. B. Lewis, ASIC suppliers target system-level integration, *Dataquest, ASIC/SLI Worldwide, Market Trends,* April 16, 1998.

ANN MARIE RINCON
International Business Machines Corporation

LOGIC, CELLULAR. See CELLULAR AUTOMATA.

LOGIC CIRCUITS. See SEQUENTIAL CIRCUITS.

LOGIC CIRCUITS, BIPOLAR AND MOS. See BIPOLAR AND MOS LOGIC CIRCUITS.

LOGIC CIRCUITS, COMBINATORIAL. See COMBINATIONAL CIRCUITS.

LOGIC CIRCUITS, GALLIUM ARSENIDE FET. See FIELD EFFECT TRANSISTOR MEMORY CIRCUITS.

LOGIC, CURRENT-MODE. See CURRENT-MODE LOGIC.