Semicustom and Custom LSI Technology

NOBUO OKUDA, MEMBER, IEEE, MASAMICHI SUGAI, AND NOBUYUKI GOTO, MEMBER, IEEE

Invited Paper

Semicustom and custom LSIs have become very important for system manufacturers because they provide system products with distinctive features that cannot be furnished by using only standard LSIs such as microprocessors. From this point of view, rapid development is essential for semicustom and custom LSIs, but there are other factors to be considered for determining the device technology and design methodology such as performance requirements, allowable development costs, and production quantities.

In this paper, these aspects for semicustom and custom LSI development are discussed. We first review the device technology and then discuss various design methodologies with an emphasis on standard cell designs. New design methodologies, such as silicon compilers and AI approaches, are also included.

I. Introduction

Because of the advancement of LSI technologies, as represented by memories and microprocessors, it is now technologically possible to integrate more than one million transistors on a single chip. As a result, LSI technologies have made various types of electronic equipment, from mainframe computers to home appliances, smaller and less expensive, contributing to the rapid expansion of the electronics industries. This, in turn, is accelerating the growth of the semiconductor industry in two areas: standard products, such as microprocessors and memories, and custom LSIs.

A highly integrated off-the-shelf product, if it can be produced at a low price by mass production, will become a *de facto* standard. On the other hand, systems manufacturers must have unique features built into their systems products so that they can establish a distinct place for themselves in the market.

This apparent conflict is the reason why semicustom LSIs have become so important. In short, systems products enjoy high performance and low cost by incorporating standard devices, while they establish their uniqueness by incorporating semicustom LSIs. In this context, semicustom and/or custom design methods are indispensable for systems manufacturers to establish the unique features of their products. This is why they are greatly concerned with semicustom or custom LSI technologies.

If semicustom and/or custom LSI devices are essential for systems manufacturers to demonstrate the usefullness of their products, manufacturers must be able to design and fabricate these devices more quickly than their competitors. Therefore, the capability for rapid development is another essential characteristic of semicustom and custom technologies.

The types of integrated circuits that are used for semicustom and custom designs include:

1) Field Programmable Logic Devices (FPLDs)
2) Gate Arrays (GAs)
3) Standard Cell LSIs (SCs)
4) Full Custom LSIs (FCs).

Fig. 1 shows the features of these devices as they appear in application system development processes, from systems design through evaluation. Hatched areas denote the processes done by semiconductor manufacturers; other areas denote those done by application systems developers. From Fig. 1, the features of the four devices from the viewpoint of systems designers can be stated as follows:

1) FPLDs are not customized by semiconductor manu-

	FPLD	GA	SC	FC
System Design	o	o	o	o
Logic Design	o	o	o	o
Logic Conversion				
Circuit Design				
Layout Design				
Mask Making				
Fabrication				
Evaluation	o	o	o	o

Fig. 1. Features of custom devices. ○—necessary; △—partially necessary; ×—not required.

Manuscript received September 5, 1985; revised February 21, 1986.

The authors are with Toshiba VLSI Research Center, 1 Komukai-Toshiba-cho, Saiwai-ku, Kawasaki 210, Japan.

0018-9219/86/1200-1636$01.00 © 1986 IEEE

facturers. They are off-the-shelf products that can be purchased and customized for specific applications by systems manufacturers. FPLDs allow systems manufacturers to completely control their development period because there is nothing to be customized by the semiconductor manufacturers.

2) GAs are characterized by a customization method in which logic designs done by systems designers are converted to wiring patterns that interconnect prefabricated standard transistor arrays to obtain specific LSIs. Prefabrication of standard arrays, or master slices, allows reduction of the development period by completing beforehand about two-thirds of the time-consuming fabrication processes and by using the remaining one-third of the processes for customization. It also allows cost reduction by using mass-produced standard master slices.

3) SC methods allow systems designers to design application-specific LSIs, or ASLSI, in the same way they develop breadboards using standard TTL logic. They do this by providing the designers with standard logic functions that are called standard cells. Though SCs require many of the same processes, such as mask making and wafer fabrication, as full custom LSIs, they allow systems designers to skip certain processes, such as TTL to MOS logic conversion (inclusion of transfer gates) and circuit design, which are difficult without the help of semiconductor device designers.

4) Full custom designs allow the designer to optimally customize all the transistors to be used in a system. Therefore, they are the most time-consuming devices to produce, but they permit the highest density LSI designs. Many of the microprocessors have been full custom, or handcrafted, designs.

In Fig. 1, going from left to right, from FPLD to FC, the design method gives more density but requires longer design time.

When selecting a customization method for LSIs to be used in developing systems applications, it is essential to take such factors into account as system size (gate count), allowable development time and cost, and expected die size (chip cost). Fig. 2 shows the chip size difference of LSIs as designed by the gate array, standard cell, and full custom approaches for the same application. This estimate is based on experimental results of redesigning handcrafted LSIs in GAs and FCs, using 2-μm rule CMOS technology. This figure shows that SC methods attain a 25-percent chip reduction at a cost of a 50-percent increase in development time as compared with GA methods.

		Gate Array	Standard Cell	Hand Craft
Chip Size		100%	75%	40%
Development Time	2.2 KG	2.5 month	4.0	13.0
	6.0 KG	3.0 month	4.5	17.5

Fig. 2. Device density versus development time (CMOS 2 μm).

ASICs may be divided into two categories: FPLDs, which are not customized by semiconductor manufacturers, and GAs, SCs, and FCs, which are. In what follows, we will discuss device technologies and design methodologies for these customization technologies, with emphasis on the technologies for shorter turn-around time and large-scale-system realization.

II. Device Technology

A. General Trends

Regardless of the level of customization—semicustom or full custom—the trends are toward higher device density and higher speed. Research efforts are directed to realize these goals in each of the device technologies. Among these, the greatest importance has been recognized for ECL, which can realize ultra-high speed in bipolar technology, and CMOS, which can attain high speed and high device density owing to the low power consumption in MOS technology. TTL, which was once a major technology for realizing logic devices, has proved to be of minor importance for custom LSIs. TTL devices cannot achieve the density of MOS devices, and their speed has almost been matched by MOS devices. PMOS, in the same context, has come to be used only for low-cost calculators and similar applications.

CMOS technologies have achieved remarkable progress as the device technology for middle- to high-speed and large-scale logic circuits. Thus they are considered to be the most promising approach available today. Their major advantage is their very small power consumption, especially for standby power. This is not only most beneficial for battery-operated equipment but essential for VLSIs because heat generation is one of the most serious problems for large-scale integration.

CMOS designs present a disadvantage in that the die size they require is larger than that required by NMOS for the same circuit. New technologies, such as domino circuits and clocked circuits, are reducing the die size difference between CMOS and NMOS designs, however. A hybrid approach, where NMOS designs are used in low-power-consumption circuits and CMOS designs are used in high-power-consumption circuits, such as drivers, is another advantageous technology in that it combines the high-speed operation of NMOS and the low power consumption of CMOS.

In the ultra-high-speed region, ECL and CML technologies are used exclusively, and their superiority is irresistible. Their switching speed is being improved toward the low subnanoseconds, demonstrating the feasibility of less than 100-ps switching time [1]. Their scale of integration, however, is around several hundred to a few thousand gates, and a certain limit is anticipated due to their high power dissipation.

In a much higher speed region, new semiconductor devices are being explored, such as Josephson-junction devices and gallium arsenide devices. Gallium arsenide devices are expected to be applied to real-world systems in the near future and are one of the most promising technologies. The research results show the possibility of super-high-speed operation and a high degree of integration [2]. Switching speeds several times as fast as those for ECLs have been achieved in laboratory experiments, and power consumption is expected to be a fraction of that of ECLs.

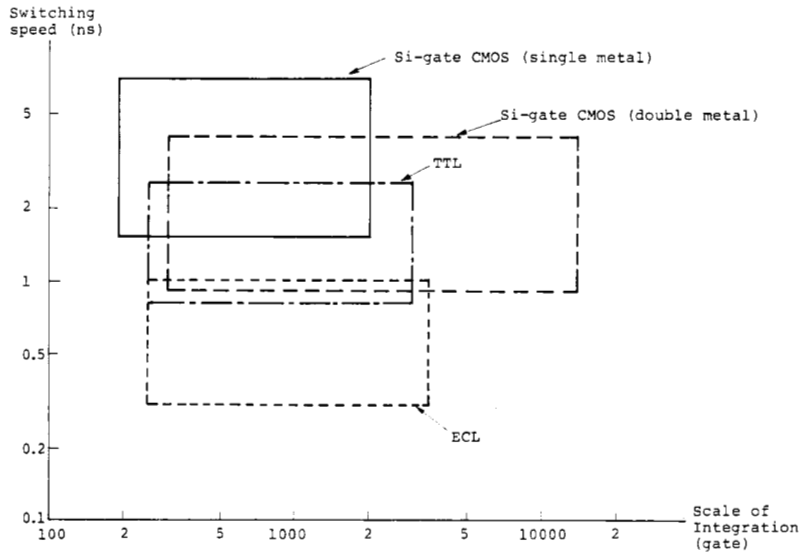

Fig. 3. Speed versus integration scale.

Among other bipolar technologies, Integrated Injection Logic is expected to be extensively applied to custom devices that contain analog circuits because this approach makes it easy to include linear devices and logic devices on the same chip.

Fig. 3 shows the distribution of operation speed and scale of integration for gate arrays achieved by various device technologies [3]. This chart clearly shows the trends discussed above: expansion of the CMOS area covering that of TTL, the superiority of ECL in the ultra-high-speed region, and the overwhelming predominance of CMOS in the large-scale integration region.

B. CMOS Technology Trends

Two CMOS technologies have been used: the silicon-gate MOS and the metal-gate MOS. The silicon-gate technology has become dominant because it allows higher operation speed and a higher degree of integration by using finer pattern geometry. In a gate-array example, a typical propagation delay of 2.0 to 4.0 ns for 3-μm rule devices has been reduced to 1.5 to 2.0 ns by using 2-μm rules. Further, 1.2- to 1.5-μm devices show 1.0- to 1.2-ns delay, and subnanosecond delay is expected by using submicrometer rules, realizing the same operation speed as bipolar devices excluding ECL.

Another technology for achieving higher operating speed is the so-called bi-CMOS technology [4], in which bipolar devices and CMOS devices are combined in basic circuits. This approach adds the features of bipolar devices—high operating speed and high current capability—to CMOS characteristics. Subnanosecond speed has been achieved. Still another emerging approach is silicon-on-sapphire technology, in which a sapphire substrate is used instead of bulk silicon. The high insulation characteristics of the substrate reduce stray capacitance, permitting subnanosecond switching speed. Because it requires special processes such as hetero-epitaxy of silicon on sapphire and because it is difficult to obtain large-diameter sapphire substrates, this approach remains a sophisticated technology for special-purpose LSIs. There are, however, examples of SOS LSIs utilized in general-purpose systems, such as the CMOS/SOS gate arrays used in super minicomputers [5]. In order for the technology to be widely applied, several problems must be solved, such as growing larger sapphire crystals and growing a high-quality silicon single crystal by epitaxy.

C. Multilayer Wiring Technology

Wiring technologies are very important for custom LSIs and, especially, for semicustom LSIs. Metal wiring has become widely used instead of high-resistance materials such as polysilicon and diffusion layers which were used until recently. This has greatly reduced the propagation delay in wiring and at the same time has enabled LSI designers to estimate the delays relatively easily at the design phase. It has also become possible to use multiple layers for wiring, which enable the wiring area on a chip to be reduced. Automatic wiring by CAD is indispensable for semicustom devices such as gate arrays, and multilayer wiring has helped to increase the flexibility for automatic wiring. Two-layer metal wiring [6] has become widely used, and there have been proposals for three-layer wiring [7]. In the future, more layers will be used for wiring, permitting a higher degree of integration by reducing the area for wiring and thus increasing the density of the devices integrated.

III. Design Methodology

A. Customization Technology Trends

Custom LSIs have acquired a very important position in systems product design by offering smaller size, lower cost, higher speed, and higher reliability. It is no exaggeration to say that rapid development of custom LSIs is a matter of life and death for systems manufacturers. Higher integration of devices, however, has led to much longer design time, thus making it difficult to maintain design reliability in reduced development time. On the other hand, custom LSIs to be developed for a particular system tend to be more and more dedicated to the target system and less and less general-purpose. The phrases "system on a chip" and "ap-

plication-specific IC" clearly describe the present situation: many different types of LSIs are being used in small quantities for specific applications. This necessitates reduction of design costs as compared with production costs. To address this issue, research and development work on various customization technologies and automated design technologies is being accelerated, with emphasis on design methodologies.

Fig. 4 shows the general flow of LSI design. Starting at given specifications, the design proceeds from functional

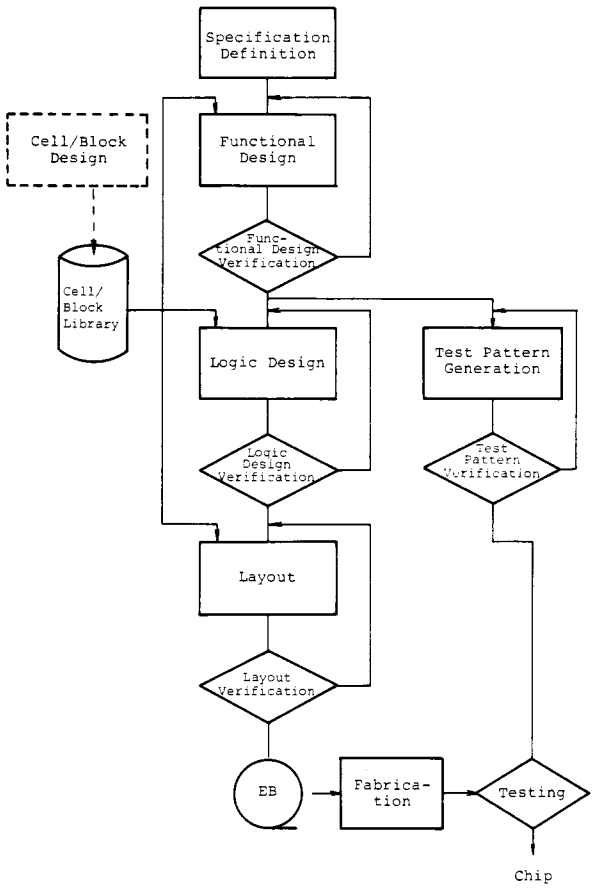

Fig. 4. LSI design flow.

design, through logic design, and then layout, in which masks are prepared and, finally, wafers are fabricated. Efforts have been directed to reducing development time and cost by automating or simplifying the design process. Several approaches have been tried.

1) Using Programmable Devices to Simplify Design: FPLDs have logic devices fabricated in a regular form. Each one of the devices can be programmed to be utilized or omitted, thus realizing the desired logic function. In a predetermined logic form, such as the AND-OR construct, a desirable logic function can be freely constructed requiring almost no time. This is the simplest way of making a custom LSI.

2) Using Wiring Patterns to Customize Arrays: In gate arrays, as mentioned before, transistor arrays are prefabricated and wiring is customized to form logic cells and to connect the cells. It is only necessary to design the wiring patterns and make their masks. Although the basic logic cells are predetermined, there are almost no restrictions for logic constructs as compared with the AND-OR structure of FPLDs. Therefore, the development time is relatively short.

3) Simplification of Layout: The next level of design simplification is to simplify layout design. In this technique, mask pattern designs are completed beforehand for basic logic functions and are maintained in a library. To design a desired LSI, the necessary logic functions are selected from the library and the placement and wiring of the functional blocks are carried out to form an entire layout. These basic logic functions are the SCs, and this method was once called the building-block method. Although this method gives a larger die size than the FC method, the design time is greatly reduced. Advancement of layout techniques continues to attain smaller die sizes. The standard cell approach is considered to be the most useful customization technology.

4) Simplification of Logic Designs: Research is being conducted to further automate the design process by automatically synthesizing the logic circuit from the functional designs. New design methodologies, including this approach, will be discussed in the next section.

B. Field Programmable Logic Devices

The best known type of FPLD is the Field Programmable Logic Array or FPLA [8]. This device has two gate matrices: the AND array and the OR array. Each node point of the arrays has a diode which can be programmed to be intact or open, thus enabling a desired AND-OR logic construct. This field programmability is implemented as fuse blowout or transistor junction breakdown to open the diode circuit. This type of device offers flexibility in designing a desired logic, but the programming is difficult to understand. To remedy this, a device known as Programmable Array Logic, or PAL,[1] has been proposed. This fixes the OR part of the array, which sacrifices the flexibility of logic design, but makes the programming easier.

Most of the current FPLAs are TTL. The trends in FPLAs are similar to those in other devices: achieving higher speed by using ECL [9] and achieving lower power consumption by using CMOS. The latter is also expected to be used for erasability [10], which will enable the programmed devices to be easily adapted to design changes, thus greatly facilitating systems development. "Erase all" type FPLAs have been developed, and research is being conducted on electrical erasability. Proposals for reconfigurable devices [11] have recently been made as a culmination to reconfigurability. One example uses EEPROM switches to electrically program the interconnection of prefabricated functional blocks [12]. Fig. 5 shows an example of such devices.

Device manufacturers can produce FPLAs as a standard product, requiring no extra effort for customization. Systems manufacturers can customize FPLAs in their own facilities, with no interaction with device manufacturers. Thus FPLAs offer the shortest development time of custom LSIs. They are particularly suitable for smaller quantity systems where LSI development costs should be minimal. FPLA design tools are also being improved, enabling the designers to program them using a higher level language description rather than a truth table or Boolean expression level programming.

[1]Trademark of MMI.

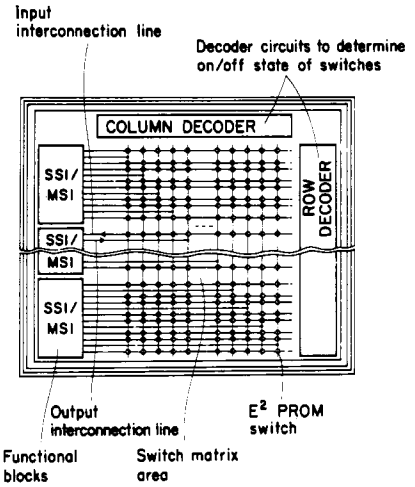

Fig. 5. Electrically reconfigurable IC.

C. Gate Arrays

A gate array is a device in which basic components such as transistors are arranged in an array and preprocessed up to the impurity diffusion process. The semi-finished wafer is called a master. Interconnection of basic cells by metal wiring is added to the master to customize the device. Usually, basic logic elements for constructing logic circuits, such as AND gates and OR gates, and functional elements, such as arithmetic units and registers, are prepared in a library. The user selects the desired elements from the library, places them, and interconnects them by using CAD tools to construct the target system. Gate arrays are suitable for random logic while FPLDs allow regular-logic designs, such as the AND-OR construct.

The trends for GAs are similar to those discussed in the previous section: the proliferation of CMOS and multilayer wiring. The latter is particularly important for higher integration of GAs because the chip area for wiring increases more rapidly as the number of basic cells increases. Multilayer wiring allows the designer to utilize the area occupied by logic devices for wiring, thus reducing the area otherwise dedicated to wiring [7]. As an extreme, Sea-of-Gates devices [13] have been proposed in which, as the name implies, no dedicated space exists for wiring.

Another trend worth mentioning is the combination of different devices. One example is a GA combined with memory [14]. This avoids the inefficiency of constructing memory devices using basic cells and facilitates the design of logic systems with memory.

D. Standard Cell Designs

The standard cell design [15], [16] is a method in which the desired logic circuits are implemented by combining and interconnecting optimally designed circuit blocks, called standard cells. In terms of the completeness of the device fabrication process, FPLDs are completed products, GAs are semicompleted products with the wiring process left to be finished, and SCs start with predesigned patterns of standard cells. In this sense, SCs are considered to be full-custom-design devices because the production process is identical to that for handcrafted designs. Standard cells are usually prepared for standard TTL equivalents, from SSIs up to MSI level devices such as ALUs and registers, which are sometimes called macro cells. It has even become possible to integrate standard LSIs, such as microprocessors and peripheral controllers, in the same manner as standard cells. This technology is known as the core processor [17] method or Super Integration [18] and enables the designer to implement a microprocessor-based system on a single chip. Such elements are often called super macro cells. Fig. 6 shows a photomicrograph of a device designed by such a method.

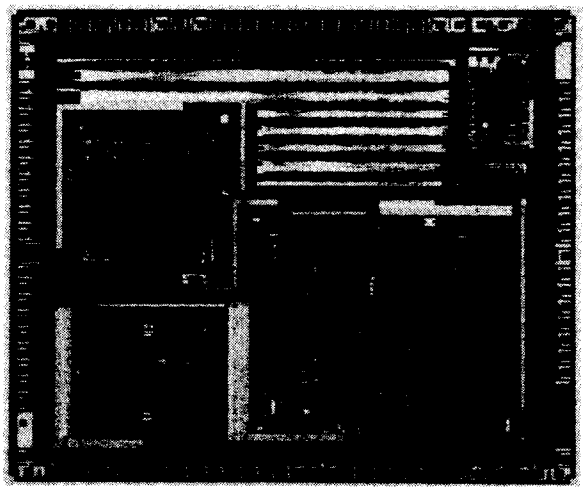

Fig. 6. Super Integration.

E. Selection of Design Methods

When developing a custom LSI, one has to select one design method from the above mentioned various technologies. The points to be considered in selecting a design method are as follows:

1) development costs and time
2) production costs (die size)
3) functions to be implemented
4) production quantities.

If the desired LSI can be developed by several design methods, one has to consider the overall chip costs throughout the life cycle of the device. The overall cost C of an LSI chip is given by

$$C = \frac{C_d}{n} + C_p$$

where C_d is the development cost, n is the total production quantity, and C_p is the production cost. For FPLDs, C_d is considered to be customization program development costs, n is the number of devices used, and C_p is the sum of the device price and programming costs. C_p depends only on the die size, independent of design methods, if the same production process is utilized. The die size, in turn, depends upon the quality of the design. Thus for the same process, a costly design method may be used to reduce die size for devices to be produced in large quantities, while a lower cost design method may be used for devices to be produced in small quantities, where large die sizes, and

higher cost per device, are acceptable. Standard products such as microprocessors have exclusively used handcrafted designs where large C_d is justified by extremely large n to minimize C. The design time, however, may become another factor to be considered because handcrafted designs will require unreasonably long development time as the scale of integration continues to expand. Standard cell and super integration methods are considered to be the best design approaches to develop a full custom device with a die size as small as possible and development time as short as possible. As the integration scale grows, the importance of overall effectiveness has become widely recognized, from specification definition through device evaluation. As a result, the necessity of a comprehensive design system has been generally acknowledged. Realization of automated design systems that achieve the performance of handcrafted designs and the development time of gate array or standard cell designs still faces several problems. Research is being continued in each of the design methods for new technology to realize such an automated system. In the next section we will focus on the standard cell design method and discuss it in more detail.

F. Custom LSI Designs Using the Standard Cell Approach

A general design process for custom LSIs has been shown in Fig. 4. Design engineers are given the specifications for the LSI including the function, algorithm, electrical characteristics, cost, package, and operating environment. Based on these specifications, the designer outlines the register transfer level designs by selecting registers and arithmetic units to be used from the library. The resultant designs are described by flow charts, hardware description languages, and so on. Firmware designs are done in parallel with functional designs in a system where firmware is used.

Functional designs are verified by using functional simulators. If an error is detected, corrections are made to functional and/or firmware design and the series of steps is repeated until satisfactory results are obtained.

When the functional designs are completed, the design is separated into two parallel processes: one to break down the functional designs into logic designs, and the other to generate test sequences to test the LSI chip. This step generates test patterns that can detect as many stuck faults as possible, utilizing the knowledge of the function of the chip through functional designs and logic designs in process. Generated test patterns are evaluated by investigating their fault coverage through fault simulation when logic designs are completed. Fault simulations, however, require extensive computer time, so that an activation check may be used instead. This check tests only whether or not a node changes its logical state.

Test generation and chip testing are time-consuming tasks. Therefore, designs for testability, such as the scan path method, have been developed to make testing easier. Using this technique, automatic test generation and reduction of test time are achieved relatively easily.

Logic designs are continued by selecting logic cells from the library. In this design state, care must be taken to account for the electrical characteristics of the cells and delays caused by wiring. After verifying the designs by logic simulations with timing delays, the design proceeds to the next layout stage.

Layout designs are performed by the hierarchical layout method, in which the designer positions large blocks containing ROMs and RAMs and the blocks containing logic cells and then determines the wiring pattern to connect the blocks. Next, layout designs are continued for these blocks to determine their inner structure as shown in Fig. 7. Layout

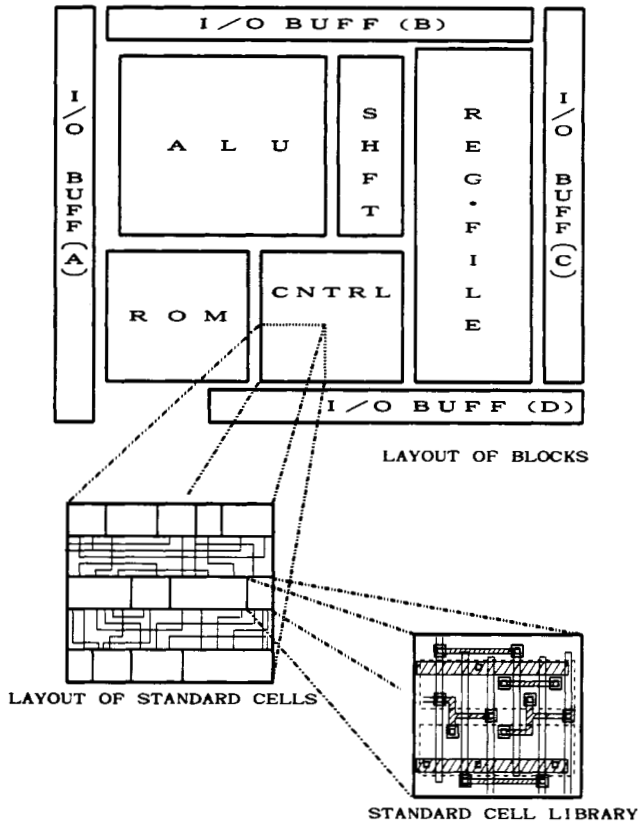

Fig. 7. Hierarchical layout.

designs are verified by checking timing delays, which are evaluated through calculating the wiring lengths. When satisfactory results are obtained, masks are made using layout designs, and the chip is fabricated.

G. CAD Tools

The designs discussed above are conducted by using the appropriate CAD tools. In this section, we will discuss various CAD tools applied in standard cell designs.

1) Design Capture: In order to use CAD tools, a designer must translate his ideas into machine-readable form. Text editors were once used to enter circuit information in the form of a verbal description [19]. This method was inadequate because it did not allow circuits to be viewed in two-dimensional form. Recently, new tools have been developed on various engineering workstations which permit direct input of circuit diagrams [20]. These tools have become widely used. Software for direct diagram input is also being developed for personal computers.

2) Functional and Logic Simulators: Logic simulators are indispensable tools for logic design verification and have been extensively used in LSI design. Recent expansion of integration scales has enabled an entire system to be fabricated on a single chip, which has necessitated the use of

functional simulators that are used in architecture level designs. Functional simulators allow the designers to verify their designs at an abstract level, such as the algorithm level or register transfer level. To facilitate the transfer from functional level designs to logic level designs, mixed-level simulators [21] have been developed. Mixed-level simulators allow both functional and logic-level descriptions within one logic system for simulation. This greatly helps the designers by allowing them to combine functional blocks with modules whose logic designs are completed, thus realizing hierarchical design. At the same time, mixed-level simulators reduce the time required for executing simulation, thus increasing design efficiency. For faster execution of simulation, special-purpose hardware simulators [22] have also become available.

3) Test Generators and Fault Simulators: While automatic test generation is possible for combinatorial circuits by using the D-algorithm, there has been no effective algorithm for automatically generating test patterns for sequential circuits. This has led to the use of heuristic methods combined with fault simulation. Special test generation systems are prepared so that testing procedures such as the scan path method can be used on the design [23]. Fault simulators are used to evaluate the fault coverage of the test patterns generated. They are also used as a part of automatic test pattern generation systems. New algorithms, such as the parallel method, concurrent method, and deductive method, have been developed to improve the performance of fault simulators, which require considerable computer resources. Special-purpose hardware [24] has also been developed for fast execution of fault simulation.

4) Layout: Layout systems require hierarchical application of placement and wiring for arbitrary size rectangles and for the internal blocks containing logic cells, as shown in Fig. 7. These systems are often called hierarchical layout systems [25]. When single-layer metal wiring was used, the areas for devices and the areas for wiring were clearly separated, and the latter were rectangles called channels. Recent two- or three-layer wiring allows a part of the wiring to pass over the device area. Where multilayer wiring is possible, therefore, the wiring areas are no longer rectangles and the channel routers have become insufficient for efficient utilization of wiring areas. In order to improve wiring efficiency for multilayer wiring, new routers are being developed such as line-search routers [26]. There has also been a revival of maze routers. Multilayer wiring necessitates the improvement of placement: rectangles of various sizes, and blocks whose sizes are not determined until internal placement and wiring have been completed, must be placed to minimize the chip area, and plans must be made for the size of the blocks containing logic cells, instead of merely arranging rectangles of the same height. Fig. 8 shows a layout pattern generated by the Hierarchical Layout System[2] that can handle super macrocells.

5) Integrated Design Systems: The CAD tools must be integrated into an overall CAD system in order to be powerful and efficient. The following characteristics are important in an integrated design system:

a) common use of user input data at various design phases

[2]to be presented at CICC '86.

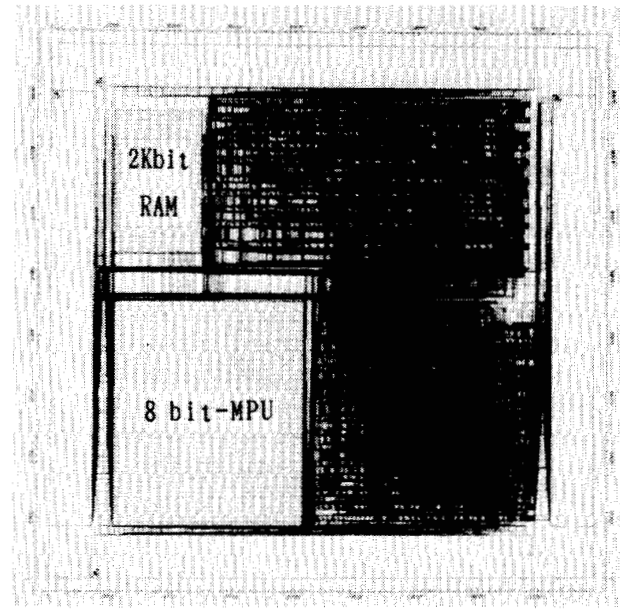

Fig. 8. Example plot of the hierarchical layout system.

b) unification of user interfaces such as commands
c) configuration control to prevent the use of erroneous design data caused by design changes
d) simplicity of the system to exclude erroneous operation.

Research is being done for database systems for CAD as a common base to construct such an integrated system.

IV. New Design Technologies

A. High-Level Design Languages and Silicon Compilers

1) History and Expectations: The high-level design language, or HDL, is a mature tool that has advanced along with the history of general-purpose computer developments [19]. In an earlier phase, the major objectives of HDL were to clarify specification definitions, to serve as documentation, and to input data to design automation systems. General-purpose computers had grown sufficiently large before the advent of LSIs that it was common for a number of designers to design a single computer. HDLs were needed for communication among the designers and between the designers and the computer.

Automatic logic synthesis and logic minimization [27], [28], on the other hand, have no long history of practical applications, although they have been dreamed of by logic designers since the birth of switching theory. Research on automating logic design has been conducted to develop algorithms for simplification of logic equations and synthesis of minimal-cost multistage networks.

The history of custom LSIs in a true sense may be considered to start with the birth and proliferation of gate arrays in the early 1980s. Before that, the major assignments of logic designers were either to design full custom LSIs such as microprocessors or to design minicomputers and control equipment using standard SSIs, MSIs, or LSIs, but never to develop an LSI in so short a period as that required for gate arrays. The size of early gate arrays was around a few hundred gates. Human designers had no difficulty

completing designs up to gate level. Development efforts were focused on the rationalization of lower level design processes, such as miniaturization of die size in layout and rationalization of testing.

As the design houses, which appeared with the progress of gate arrays, made their presence recognized in the world, design and fabrication became separated. This separation has made such phrases as "design house" and "silicon foundry" widely known. Systems developers have recognized the advantages of rationalizing communication between designers and manufacturers. They have encouraged proposals for standard interfaces at various design stages, such as the logic description level and the mask description level. Discussions are currently under way for EDIF, VHDL, and others [29], [30]. It is desirable to include an HDL as one of the standard interfaces.

Standard interfaces are also a means to guarantee second sourcing, which is required by semiconductor users. In this sense, standard interfaces will accelerate the increase of the number of second sources for users, design houses, and manufacturers, because they will benefit all these.

Along with the proliferation of gate arrays, their integration scale is becoming larger by the advancement of semiconductor technologies. In addition, many semicustom LSIs have appeared which are fabricated by design methods approaching full custom designs, such as the standard cell method. A number of designers have been assigned to design a single chip when its circuit size has been estimated to be too large to be designed by a single designer. This has given rise to the association of HDLs and LSI designs.

The concept of special modules, such as basic cells in gate arrays, standard cells in standard cell systems, and ALUs, MPYs, ROMs, RAMs, and PLAs in full custom LSIs, has led to the application of automatic logic synthesis in LSI design. Automatic logic synthesis is used for two purposes: to synthesize logic circuits up to standard cell level constructs and to generate regular modules, such as ALUs and ROMs. In addition, actual evaluation of the results obtained by logic synthesis has become possible, which has made logic synthesis research important again. Silicon compilers [31] have the goal of generating mask patterns directly from HDL descriptions. An elementary version of silicon compilers is the module generator as mentioned above. They aim to provide efficient LSI designs even if the systems designer has insufficient knowledge of semiconductor technology. Although silicon compilers are often discussed with excessive expectations, they do promise to be valuable design tools.

2) Problems for HDLs and Silicon Compilers: When a standard cell description is obtained by logic synthesis of a functionally described circuit using an HDL, there is no guarantee that the LSI fabricated through automatic layout for the SC description will achieve the high density the designer intended. As shown in Fig. 9, assuming 1.2-μm design rules, standard cell LSIs achieve on the average a device density of 1k transistors/mm^2, while circuits of highly regular structure, such as ALUs and MPYs, achieve 3k transistors/mm^2 density. This density is three times as high, while at an extreme, ROMs and RAMs present more than ten times as high device density. This is the reason why conventional logic synthesis systems, in general, have not been adopted into LSI design systems. This also accounts for the fact that many proponents of the current silicon compilers

ROM	- 30,000 Tr/mm^2
RAM	- 10,000 "
REGULAR LOGIC WITH HAND-CRAFT DESIGN	- 3,000 "
RANDOM LOGIC WITH SC DESIGN	- 1,000 "

Fig. 9. Density of macroblock.

recommend that designers adopt high-density modules by releasing module generation to the designers.

Module generation, in itself, does not complete LSI designs. One of the problems left unautomated in hierarchical designs is floor planning, that is, placement and wiring of modules. The shape of high-density modules, generated by module generators, is limited to rectangles, but the size of these rectangles does not have the regularity of standard cells that would be convenient for layout programs. Thus the hard problem of placing and wiring a number of arbitrary size rectangles remains to be solved.

There have been no LSI design systems that have fully automated floor planning. Creators of design systems recommend positive interaction of designers, which shows the difficulty of devising a general solution for this problem [32].

While the problem of placement and wiring for standard cells has successfully been solved with a relatively simple algorithm to place modules based on the cell interconnection specified by interconnection description data, module placement in cells approaching full custom design is difficult without taking into account such things as signal flows, control flows, signal paths, and data buses (a number of signal paths having the same characteristics). In a sense, various techniques used in conventional printed circuit board system designs, such as division of a system into modules, definition of boards, layout within a board, and so forth, correspond to hierarchical floor planning for LSIs. From this point of view, development of expert systems and full experimentation with them, gathering past design knowhow to apply it widely, are expected to lead to the realization of effective and efficient design systems.

B. AI Approaches

In the previous discussions, we have not mentioned the computer resources required for CAD with respect to circuit complexity. Most of the VLSI design problems expand greatly as the integration scale expands, and so the algorithms to solve them become exceedingly complex [33]. This is obvious if such problems are to be solved as placement, wiring, and test generation by testing all the possibilities. In addition, no algorithms that always give the solution to a problem if one exists, or decide that a problem is unsolvable if no solution exists, have yet been found for most of the problems. Therefore, designs are usually conducted by using appropriate procedures that incorporate certain forms of heuristics [34]. Advancing a step further, research has become active to realize automatic design systems that have the same performance as human experts by directly using human experts' knowledge. This approach would apply expert systems [35], known by successful examples such as MYCIN [36] and DENDRAL [37] in the AI field, to LSI design systems [38]. This approach aims to:

a) eliminate the necessity of trying out all the possible combinations to find the desired solution;
b) realize performance comparable to that of human experts;
c) separate knowledge from inference mechanisms to allow performance improvement by improving the knowledge;
d) permit an explanation of the process that has led to the solution;
e) achieve a certain level of efficiency.

In principle, this approach may be applied to any of the LSI design problems. Some of them will be briefly described.

1) Automatic Generation of Functional Descriptions: This derives a functional level description from an algorithm level description. This is a process usually done by human architects. Technology-independent hardware descriptions that consist of registers, arithmetic units, memories, ports, buses, and microcodes are synthesized by using expert knowledge [39].

2) Automatic Logic Synthesis: This approach has been used for many years. Expert systems are used in two ways: one to develop a functional description into gate level by applying the rules about the kinds of gates available in the target technology, fan-ins, and fan-outs [40], and the other to optimize the gate-level descriptions by using conventional logic simplification rules [41].

3) Automatic Layout: This aims to eliminate the necessity of human interaction required by current layout systems. Transistor structure in the target technology, design rules for placement and wiring, heuristics for optimizing wiring length and die size, and others are used as the rules for determining the layout [42].

4) Automatic Design Verification: Gate-level designs and layout may be verified from such viewpoints as conformity to design rules, wiring errors, and the existence of race conditions by applying an expert system approach [43].

5) Automatic Test Generation: Automatic test generation algorithms such as the D-algorithm have not necessarily achieved satisfactory results. Generated test sequences are not directly related to characteristic operation of target systems, and it is not easy to understand the meaning of the sequences, thus prohibiting human interaction to improve their quality. The expert systems approach [44], where test sequences are generated by understanding the characteristic operation of the circuit in the same way humans do, is being extensively investigated.

We have explained the application of the expert system approach in various phases of LSI designs. This is a relatively new research area and few results have been obtained up to now that allow extensive evaluation of their significance. In solving various LSI design problems, the progress in computer power may justify the use of an algorithmic approach if one exists, however complex it may be. If no algorithm is known to exist, one has to resort to the heuristic approach. In this case, expert systems, where knowledge and its manipulation are clearly separated, may facilitate the improvement of their performance with the advancement of the knowledge and/or its manipulation method. Before AI systems are applied to real-world systems, however, thorough evaluation will be necessary based on extensive research efforts.

V. Conclusions

The market for custom LSIs, or application-specific LSIs, is estimated to exceed $20 billion by 1990. No one will doubt that ASICs are the key factor for success for systems manufacturers to occupy a favorable position in the market. ASICs have become a powerful tool for systems designers.

It has often been pointed out that there has been a big gap between LSI production technology and LSI application technology. Lack of design support systems, both software and hardware, has often been a bottleneck for developing VLSIs using advanced technologies. The solution has been recognized to be CAD systems that enable systems designers to design LSIs to meet their needs without the help of semiconductor engineers. Research efforts are directed to develop advanced and friendly CAD systems that enable the user to design cost-effective, high-performance LSIs using advanced device technologies in a short time. The efforts include application of AI technologies, development of special-purpose hardware for simulation and layout, and performance enhancement of engineering workstations. In order to achieve these goals, more and more computer power is necessary, which, in turn, requires higher and higher performance LSIs. Progress will continue in LSI technology and sophisticated design automation technology that makes the best use of it. This progress will continue in various technologies, giving rise to new business opportunities.

Acknowledgment

The authors wish to express their gratitude to Mr. Mitsuhashi, Mr. Takada, Mr. Saigo, and Mr. Ushiku for providing the materials for this paper. Finally, they also thank Dr. Takeishi for continual encouragement.

References

[1] H. Ullrich, W. Brackelman, H. Fritzsche, and H. Wieder, "A 100ps 9k gate ECL masterslice," in *Dig. ISSCC 85*, pp. 200–201, 1985.
[2] N. Toyoda, N. Uchitomi, Y. Kitaura, M. Mochizuki, K. Kanazawa, T. Terada, Y. Ikawa, and A. Hojo, "A 42 ps 2k-Gate GaAs gate array," in *Dig. ISSCC 85*, pp. 206–207, 1985.
[3] "Survey of custom and semicustom ICs," *VLSI Des.*, pp. 30–43, Oct. 1984.
[4] Y. Nishio, I. Masuda, T. Ikeda, M. Iwamura, K. Ogiue, and Y. Suzuki, "A subnanosecond low power advanced bipolar-CMOS gate array," in *Proc. ICCD 84*, pp. 428–433, 1984.
[5] S. Tanaka, J. Iwamura, J. Ohno, K. Maeguchi, H. Tango, and K. Doi, "A subnanosecond 8k-gate CMOS/SOS gate array," in *Dig. ISSCC 84*, pp. 260–261, 1984.
[6] T. Moriya, Y. Hazuki, S. Shima, M. Chiba, and M. Kashiwagi, "Aluminum two-level interconnection for VLSI," in *Proc. ECS 82*, pp. 337–338, 1982.
[7] T. Saigo, K. Niwa, S. Kurosawa, and T. Takada, "A triple-level wired 24k gate CMOS gate array," in *Dig. ISSCC 85*, pp. 122–123, Feb. 1985.
[8] N. Cavlan and R. Cline, "Field PLAs simplify logic designs," *Electron. Des.*, vol. 18, pp. 84–90, Sept. 1975.
[9] M. S. Millhollan and C. Sung, "A 3.6 ns ECL programmable array logic IC," in *Dig. ISSCC 85*, pp. 202–203, Feb. 1985.
[10] R. Leung and S. M. J. Lee, "A 50 ns 48-term erasable programmable logic array," in *Dig. ISSCC 85*, pp. 130–131, Feb. 1985.
[11] E. Fong, M. Converse, and P. Denham, "An electrically reconfigurable programmable logic array using a CMOS/DMOS technology," *IEEE J. Solid-State Circuits*, vol. SC-19, no. 6, pp. 1041–1043, Dec. 1984.
[12] Y. Ikawa, K. Urui, M. Wada, T. Takada, M. Kawamura, M. Miyata, N. Amano, and T. Shibata, "A one-day chip: An inno-

vative IC construction approach using electrically reconfigurable logic VLSI," presented at the CICC 85, May 1985.
[13] A. Hui, A. Wong, C. Dell'Oca, D. Wong, and R. Szeto, "A 4.1k gates double metal HCMOS sea of gates array," in *Proc. CICC 85*, pp. 15–17, 1985.
[14] M. Ueda, K. Sakashita, R. Yonezu, T. Fujimura, T. Arakawa, S. Asai, and Y. Kuramitsu, "A 1.5μ CMOS gate array with configurable ROM and RAM," in *Dig. ISSCC 85*, pp. 126–127, 1985.
[15] A. J. Kessler and A. Ganesan, "Standard cell VLSI design," *IEEE Circuits Dev.*, vol. 1, no. 1, pp. 17–33, Jan. 1985.
[16] S. Petrizzo, "2 μm standard cell family builds VLSI semicustom circuits," *Electron. Des.*, vol. 33, no. 14, pp. 123–130, June 13, 1985.
[17] L. F. Parisoe and G. W. Knapp, "2901/2910 core microprocessor cells brings 'system-on-a-chip' to semicustom standard cells," in *Proc. CICC 85*, pp. 264–266, May 1985.
[18] K. Nagano, Y. Shiotari, A. Sueda, T. Saito, K. Uchida, S. Takeda, Y. Fukushima, M. Hirasawa, and T. Yamamoto, "Super Integration," in *Proc. CICC 85*, pp. 267–271, May 1985.
[19] M. R. Barbacci and T. Uehara, Eds., "Special issue on hardware description languages," *Computer*, vol. 18, no. 2, Feb. 1985.
[20] S. Tsunekawa and S. Shimotsuji, "Automatic drawing reader—TOSGRAPH," *Trans. IECE Japan*, vol. J68-D, no. 4, pp. 466–472, Apr. 1985 (in Japanese—English version to be published).
[21] T. Sasaki, A. Yamada, S. Kato, T. Nakazawa, K. Tomita, and N. Nomizu, "MIXS: A mixed level simulator for large digital system logic verification," in *Proc. 17th DAC*, pp. 626–633, 1980.
[22] T. Blank, "A survey of hardware accelerators used in computer-aided design," *IEEE Des. Test Comput.*, vol. 1, pp. 21–39, Aug. 1984.
[23] T. W. Williams and K. P. Parker, "Design for testability—A survey," *IEEE Trans. Comput.*, vol. C-31, no. 1, pp. 2–15, Jan. 1982.
[24] Panel, "Future directions for DA machine research," discussion at the 22nd DAC, 1985.
[25] B. T. Preas and C. W. Gwyn, "Method for hierarchical automatic layout of custom LSI circuit masks," in *Proc. 15th DAC*, pp. 206–212, 1978.
[26] J. Sonkup, "Routing on one layer—An algorithm and its implementation," in *Proc. ICCD 83*, pp. 126–135, 1983.
[27] S. G. Shiva, "Automatic logic synthesis," *Proc. IEEE*, vol. 71, no. 1, pp. 76–87, Jan. 1983.
[28] D. E. Thomas, "The automatic synthesis of digital systems," *Proc. IEEE*, vol. 69, no. 10, pp. 1200–1211, Oct. 1981.
[29] EDIF, "Electronic Design Interchange Format Version 100," EDIF Steering Committee, 1985.
[30] M. Shahdad, R. Lipsett, E. Marschner, K. Sheehan, H. Cohen, R. Waxman, and D. Ackley, "VHSIC hardware description language," *Computer*, vol. 18, no. 2, pp. 94–103, Feb. 1985.
[31] D. D. Gajski and R. H. Kuhn, "New VLSI tools," *Computer*, vol. 12, no. 12, pp. 11–14, Dec. 1983.
[32] J. R. Fox, "The MacPitts silicon compiler: A view from the telecommunications industry," *VLSI Des.*, pp. 30–37, May/June 1983.
[33] S. Sahni and A. Bhatt, "The complexity of design automation problems," in *Proc. 17th DAC*, pp. 402–411, 1980.
[34] E. Shragowitz and S. Keel, "Heuristic algorithms for CAD of VLSI, their justification, characterization and usage," in *Proc. ISCAS 85*, pp. 31–34, 1985.
[35] F. Hayes-Roth, D. A. Waterman, and D. B. Lenat, *Building Expert Systems*. Reading, MA: Addison-Wesley, 1983.
[36] B. G. Buchanan and E. H. Shortliffe, *Rule Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Projects*. Reading, MA: Addison-Wesley, 1984.
[37] R. Lindsay, B. G. Buchanan, E. A. Feigenbaum, and J. Lederberg, *Applications of Artificial Intelligence for Organic Chemistry*. New York, NY: McGraw-Hill, 1980.
[38] T. Fujita and S. Goto, "Knowledge base and algorithm for VLSI design," in *Proc. ISCAS 85*, pp. 877–880, 1985.
[39] T. J. Kowalski, D. J. Geiger, W. H. Wolf, and W. Fichtner, "The VLSI design automation assistant: A birth in industry," in *Proc. ISCAS 85*, pp. 889–892, 1985.
[40] N. Kawato, T. Saito, and H. Sugimoto, "DDL/SX: A rule-based expert system for logic circuit synthesis," in *Proc. ISCAS 85*, pp. 885–888, 1985.
[41] W. W. Cohen, K. Bartlett, and A. J. Geus, "Impact of metarules in a rule based expert system for gate level optimization," in *Proc. ISCAS 85*, pp. 873–876, 1985.
[42] J. H. Kim, J. McDermott, and D. P. Siewiorek, "Exploiting domain knowledge in IC cell layout," *IEEE Des. Test Computers*, vol. 1, pp. 52–64, Aug. 1984.
[43] C. Lob, R. Spickelmier, and A. R. Newton, "Circuit verification using rule-based expert systems," in *Proc. ISCAS 85*, pp. 881–884, 1985.
[44] M. J. Bending, "Hitest: A knowledge-based test generation system," *IEEE Des. Test Comput.*, vol. 1, pp. 83–92, May 1984.

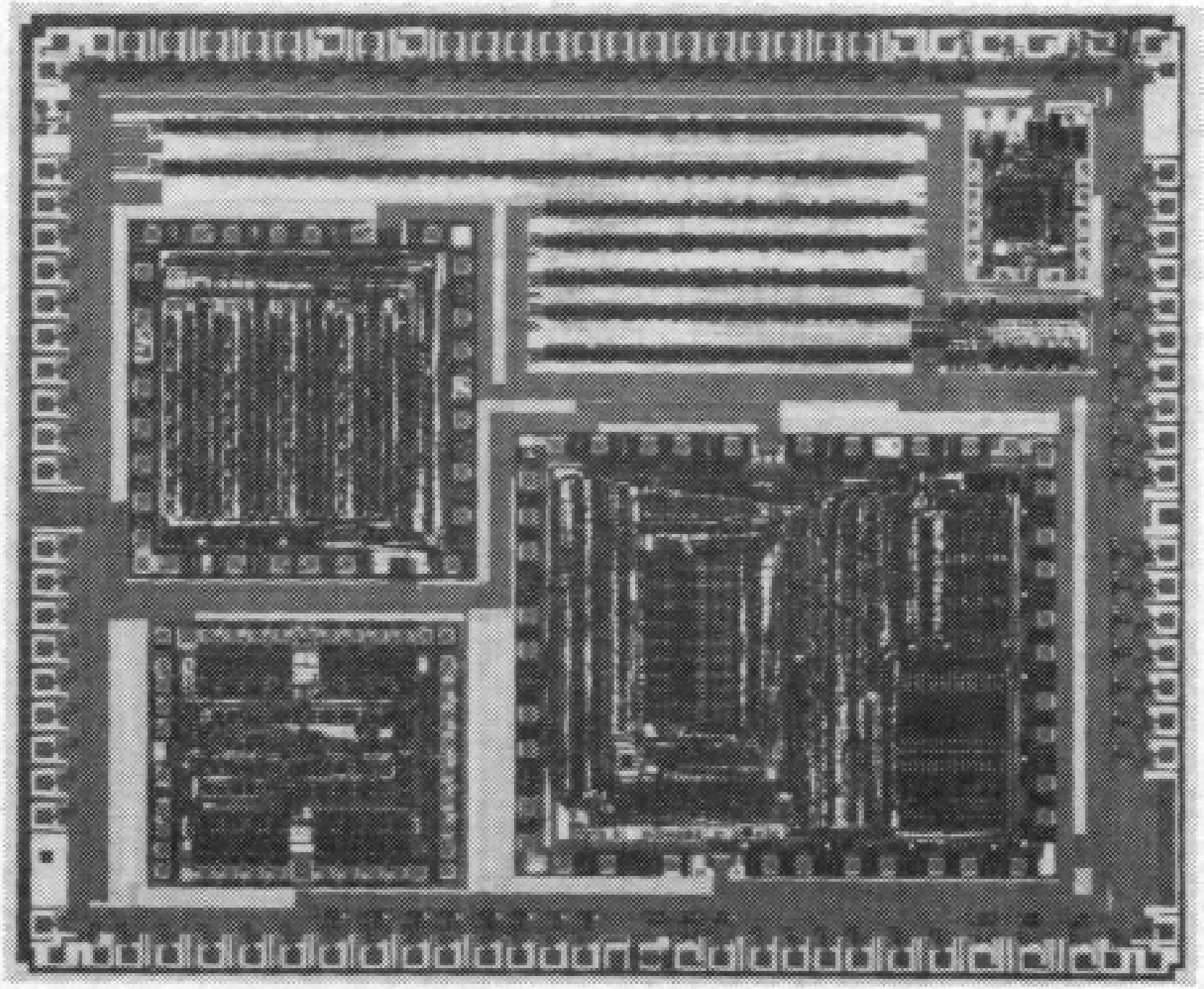

Fig. 6. Super Integration.

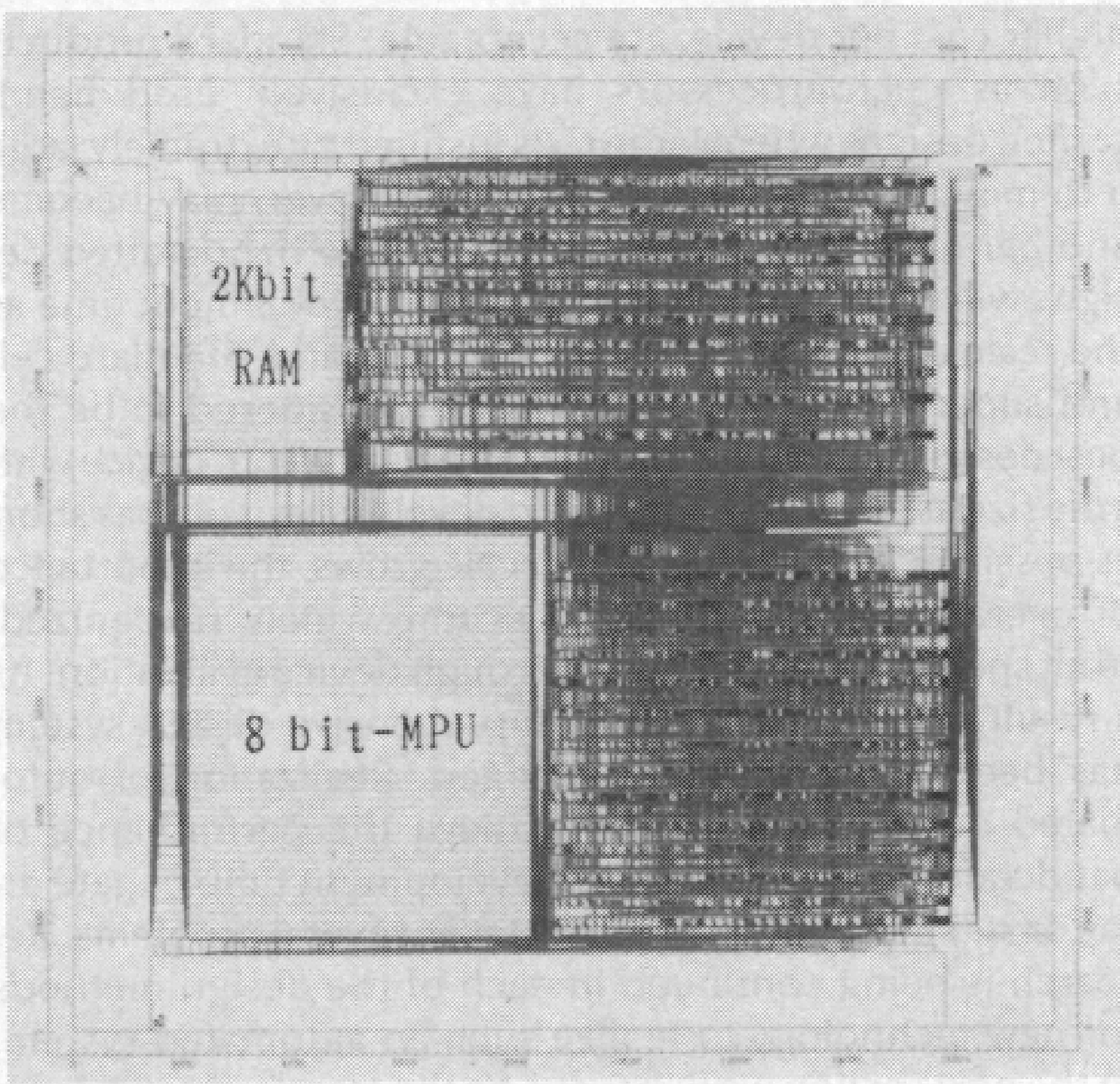

Fig. 8. Example plot of the hierarchical layout system.