

25. Custom Integrated Circuits

Academic and Research Staff

Prof. J. Allen, Prof. L.A. Glasser, Prof. P. Penfield, Prof. R.L. Rivest, Prof. G.J. Sussman, Dr. H. Shrobe, Jr.

Graduate Students

R. Armstrong, I. Bain, Y. Brown, W.H. Evans, L.P.J. Hoyte, A. Fujimura, M.D. Matson, S.P. McCormick, J.J. Paulos, J. Pineda, L.D. Seiler, J. Siskind, B.C. Williams, J.T. Wroclawski

25.1 Conversion of Algorithms to Custom Integrated Circuits

U.S. Air Force (Contract F49620-81-C-0054)

Jonathan Allen, Paul Penfield, Jr., Ronald Rivest, Gerald Sussman, Howard Shrobe

The design of custom integrated circuits is an instance of the more general problem of the design of complex systems. Certainly if circuits are to be designed that contain between half a million and a million transistors, then techniques for the design of complex systems must be introduced that will assure quick, correct, and economical construction of the fabricated circuits. The established means of dealing with this complexity is to break up the problem into a number of subproblems, each of which can be tackled relatively easily. We also look for regularity in each of these components, so that basic circuits can be repeated many times in a regular array in order to achieve the desired functionality either in terms of arithmetic and logic function or memory. This breaking up of the overall task corresponds to the introduction of structural hierarchy, and visual inspection of contemporary large integrated circuits readily reveals the constituent blocks of this hierarchy. Use of regular structures such as memory, programmed logic arrays, register files, and iterative multiplier arrays makes the number of distinct circuits present in the overall circuit small enough so that substantial design effort can be devoted to each such circuit. There is an additional hierarchy, however, that is orthogonal to the structural hierarchy. This second hierarchy deals with the various qualitatively distinct kinds of representation that must be maintained over all levels of the design. Thus the design representations must range from a functional specification, through architectural representation to logic, circuits, devices, and a detailed geometrical layout specification which is suitable for mask making. It is striking that these different levels of representation are qualitatively different, and yet they must all be aligned to the same basic circuit since they describe in each case some limited aspect of the same overall design. The designer's job is complicated by the need to fully specify all these different levels of description, and yet keep them consistent with each other. For this reason, much of our emphasis in this project is focussed on careful choice of representations at all

these different levels together with the algorithmic means to transform between them.

At the functional level, much of our effort has been devoted to a variety of ad hoc languages for various applications. Thus we have devised high level languages for microprocessor type architectures, as well as languages intended to capture the semantic content of digital signal processing algorithms. As yet, little effort has been expended on a more generalized language, but as the other design tools under investigation become more mature, we are abstracting on the individual languages to design a powerful high level functional specification language. A major emphasis of our research is to introduce within such high level languages the means to perform architectural exploration. We have known for some time the formal requirements necessary to perform space/time trade-offs in a way that preserves the original functional specification. We are now exploring means to introduce into high level languages the linguistic means to perform these trade-offs. In the literature, despite the presence of several high level languages, no such means is currently available, yet the means to increase or decrease the amount of parallelism in a design is clearly an important degree of freedom present to the designer of custom integrated circuits. There exist many different algorithms, especially in the signal processing field, where the amount of parallelism available in the algorithm has not been completely exploited in hardware at this time, so we expect that this capability will be very useful in many applications.

The design of specific circuits has given rise to a number of regular designs such as programmed logic arrays, multipliers, and register arrays. We have devised a number of techniques for abstracting on this experience in order to lessen the overall design burden of optimized circuits. Many approaches to silicon compilation have focussed on general techniques to cover a broad class of architectures. It has become clear to us, however, that there is a strong need for specialized compilers that can provide a degree of efficiency in space, time and power that is simply not attainable in the general case. One example of a specialized compiler is a PLA generator, and several of these have been developed by numerous investigators. Nevertheless, most of these PLA generators have only been able to deal with one particular technology in a particular set of design rules. We have implemented a new style of PLA generator which is capable of learning from example in terms of different technologies, circuit styles, and design rules. In order to use such a technique, the designer uses an interactive graphic system to specify a two-input, two-product term, and two-output programmed logic array which serves as an exemplar for all other possible programmed logic arrays corresponding to this style. Thus the designer may wish to specify NMOS or CMOS technology, two-phase or three-phase clocking, input encoding, output multiplexing, precharging, and other aspects of the design. This is easily accomplished by specifying these features in the exemplar design. These designs are built up readily from a number of cells contained in a library, although additional cells can easily be added to this library at any time. The designer then instructs the system to create a compiler for this style of programmed logic array, and this is quickly accomplished by the computer system. From then on, the designer may specify the logic for any size PLA in the style of the exemplar and this will be appropriately created. It will, of course, enjoy the

efficient design provided by the individual cells, and lead to a highly packed dense design. It is interesting that the technique just described amounts to the development of a compiler compiler, and we believe that such techniques are exceedingly powerful because they provide both highly efficient designs but also a great deal of flexibility from the point of view of the designer. We are currently working on the introduction of more flexible control structures within these compiling strategies that will provide even further increases in flexibility while preserving the optimal properties of these circuits.

Another example of our development of specialized compilers is in the area of multipliers. As the result of a substantial effort in the design of a custom integrated circuit for signal processing, a large parallel multiplier using modified Booth's recoding was designed. This was a highly regular design, and suitable for composition from a small number of distinct cells except for the final carry propagation stage which must be generated procedurally. Once again we have developed a compiler for this style of circuit. The designer merely specifies the size of the multiplier and multiplicand, and the overall layout is immediately generated. In this way a huge amount of circuit design time is saved, and yet the design is very compact and fast. This illustrates the fact that specialized compilers can yield designs that are far more efficient than those which would be generated by any more general capability. Under development is a third type of compiler that would provide for the generation of arbitrary size floating point units. The designer will be able to specify both the kinds of floating point functions that are required as well as the field sizes for both the exponent and mantissa of the floating point representation. Additionally, this design will utilize asynchronous timing techniques so that the completion time of the arithmetic operation can be readily detected. We view these techniques as exceedingly valuable, since they also provide the designer with the means to explore space/time trade-offs in terms of complexity of the arithmetic logic unit vs. the amount of memory needed on chip. When one compares fixed point representations with floating point representations, it is readily apparent that while fixed point arithmetic is relatively simple, the amount of memory needed to store such numbers is greater than for floating point numbers for the same signal to noise ratio, although the latter must utilize more complicated arithmetic units.

We have continued to develop a highly accurate circuit extraction scheme which operates on layout parameters and provides the input specification for circuit simulation. The circuit extraction techniques used contemplate modern high density designs, in that they allow for internodal capacitances as well as capacitances to the substrate. Also resistances of any shape can be readily extracted, and no less than three different techniques are employed for computing the resistances of varying size conductors. The program that accomplishes this task is written in a way that is technology independent, and a default mode readily obtains the topology of the circuit in a form that is useful for unit delay logic simulation. When complete circuit simulation is not possible, then it is appropriate to try to bound the delay through various parts of the circuit through a variety of modeling techniques. We have explored the use of fundamental characterization of nonlinear circuits for this purpose, and have already obtained very tight bounds for the propagation time through tree-like

interconnect networks. Current modeling activities are being applied to circuits involving active devices.

For some years now, a major emphasis of our work has been the development of a general capability for placement and routing of arbitrarily sized rectangular components with interconnect on all four sides. General routing strategies have been developed using a two-pass process. In the first pass, the global routing strategy is developed without concern for track utilization within a particular channel. This design is largely topological in nature, but in the second pass the detailed way in which the wiring nets are assigned to channel tracks is determined in optimal fashion. These techniques have been developed to a very high degree, and high performance channel routers have been one of the by-products of this research. Current emphasis is focussed on the development of placement strategies, although there is a great deal of understanding of this problem from our previous theoretical investigations. This approach to the placement and routing of rectangular components is completely algorithmic in nature, and does not require interaction on the part of the designer. We recognize, however, that in many cases it may be appropriate to use specialized routing procedures, just as specialized compilers are used for the layout of the individual cells. For example, a specialized river router can be rather easily constructed which can be used interactively by the designer to produce highly efficient layouts. A variety of these routers are under construction and are utilized in a way that is complementary to the overall placement and routing strategy.

While the integrated circuits under design are in themselves highly complex, another aspect of complexity in integrated circuit design is the nature of the tools and algorithms themselves. Thus, almost all of the artwork analysis tools that are used have performances dependent on the number of rectangles that must be manipulated, and in a large design several million rectangles would have to be contemplated. Thus it is important to be able to focus effectively on techniques for dealing with this amount of data, and two techniques have emerged. On the one hand, it has been possible to improve the algorithms themselves by decomposition techniques into modular cells which can be composed later to form the overall design. We have also been interested in the introduction of special purpose hardware for integrated circuit design. Our first example of these procedures has been the design of special purpose hardware for design rule checking. A single printed circuit board has been designed which utilizes a conventional microprocessor together with four distinct custom integrated circuits for performing course grid type design rule checking of rectangular and 45 degree oriented line designs. This special hardware provides a speed-up of over two orders of magnitude, and is the first such design to be demonstrated in the field. It can be plugged into a work station and thus provide very aggressive performance for interactive use by a designer, particularly on very large designs. We intend to extend our study of the use of special purpose hardware, particularly for such applications as the operations of linear algebra, which are of use both for signal processing and for circuit simulation. We believe that circuit simulation is an exceedingly useful capability, but it is often not used because it is so expensive computationally. Thus the introduction of special purpose hardware would provide this highly useful capability in a much more attractive form.

It is clear from the above discussion that we have attacked a number of the constituent problems of the overall integrated circuit design problem. We are also concerned with the integration of these techniques into a cohesive framework. New work focuses on the utilization of the aforementioned special purpose compilers in a format that maintains technological independence but also the capability for constraining designs to given levels of electrical circuit performance. Optimal design is certainly the result of simultaneous satisfaction of a number of different constraints at all the qualitatively distinct levels of the design hierarchy. The control structures under which this can be accomplished are currently under investigation in the context of high level hardware design languages. We feel that we have benefited a great deal from the implementation of a variety of special purpose solutions to these problems and are now ready to incorporate them in a more general structure.

Publications

- Rivest, R.L., "Benchmark Channel-Routing Problems," M.I.T. VLSI memo No. 82-77, February 1982.
- Rivest, R.L. and C.M. Fiduccia, "A 'Greedy' Channel Router," Proceedings 1982 Design Automation Conference, June 1982.
- Rivest, R.L., "The 'PI' (Placement and Interconnect) System," Proceedings 1982 Design Automation Conference, June 1982.
- Evans, W. H. and J. Allen, "MOS Implementations of TTL Architectures: A Case Study," Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing, May 1982.
- Seiler, L., "A Hardware Assisted Design Rule Check Architecture," Proceedings 1982 Design Automation Conference, June 1982.

25.2 A Circuit Theory for Digital VLSI Systems

National Science Foundation (Grant ECS81-18160)

Lance A. Glasser, Paul Penfield, Jr., Isaac Bain, Charles Zukowski

VLSI circuits are complex nonlinear dynamical systems which seem beyond the domain of most classical network theories. The basic laws of electrostatics – Kirchoff's voltage and current laws – hold, but many of the more powerful tools available for linear networks, such as Laplace transforms, do not. The thrust of our effort is the development of new theories specialized to the realm of VLSI circuits.

The explosive growth in the number of devices in a VLSI circuit and the increased complexity of device models have combined to make the simulation of very large scale circuits extremely costly and, in many cases, almost impossible. We have introduced a technique, called waveform bounding, which addresses this problem. The key observation is that it is sometimes adequate to obtain approximate rather than exact results, provided one understands the magnitude of the uncertainty. Often this can allow one to answer exactly the sort of questions one regularly asks of a circuit simulator, such as, "is this critical path fast enough?" Thus, if the bounds on the delay through a

network are such that if the delay is greater than 55 ns and less than 65 ns, then we can say with certainty that, if the path meets a 70 ns specification and fails a 50 ns specification, more accurate and costly simulation techniques must be used.

Such a technique is only worthwhile if the computer time needed to simulate a VLSI system can be reduced. Preliminary results show that several important circuit simulation problems can be vastly simplified given the freedom to obtain approximate rather than exact answers. These problems include linear RC trees, an interconnect circuit commonly found on MOS VLSI chips; and nonlinear MOS logic gates.

Work in this area now has two thrusts. One is to find new and more general classes of networks to which we can apply our theories; and the second is to improve and tighten the bounds on the networks we can solve. When combined with other circuit simulation techniques being developed in the community, this work may one day form the basis for a powerful new generation of circuit simulation programs.

While a VLSI system might contain millions of transistors, these devices are not, and cannot, be put together in an *ad hoc* manner. Design methodologies, which are self imposed restrictions on the design techniques one will use, limit the number of archetypical circuit forms in a VLSI system. One can conceptualize these circuit forms as being the language in which the design is composed. Like any language it has its correct forms of usage and its incorrect forms. We have research efforts in two aspects of this problem. The first is developing a circuit design language suited to MOS VLSI design which captures the designer's intent. The second is a set of tools for checking the syntactic correctness of designs composed in this language. One problem is that there are many different dialects of the language which one would like to use, depending on the capabilities of the technology and the performance objects of the design. Thus, we are developing a set of computer programs to check a circuit, as it is being composed, for the design methodology under use. Part of the task is to include an easy way to specify this design methodology. A methodology checker which was specifically designed for a single specific design methodology was written by Clark Baker. Our objective is to generalize this concept so the methodology checker can continue to be used as technology and circuit techniques mature.

25.3 Very Large Scale Integrated Circuit Research

U.S. Air Force (Contract F49620-81-C-0054)

Lance A. Glasser, Mark Matson

A VLSI system can be optimized in many different domains of abstraction including the domains of machine organization, logic design, circuit design, and layout. This project is specialized to various aspects of optimization in the circuit domain. Two important types of optimization, are considered. In

the first we are interested only in raw speed, without regard to power or area resources. The problem here is to design a circuit which runs as quickly as possible. While this capability is often useful, a more common constraint in nMOS circuit design is to meet a delay specification while using the minimum resources. The mathematical tool used for both of these problems is the LaGrange multiplier.

Preliminary results from this investigation have produced inequality theorems about how the delays of the various stages in an optimized design behave. A program has also been developed, based on simple logic gate models, to optimally size transistors in the critical path of an MOS digital circuit. This work is now being generalized to multiple paths and incorporating more accurate models. Results of modern optimization theory are being used in conjunction with special properties of the domain of MOS logic gates to make the program computationally efficient.

Eventually this work will be incorporated into silicon compiler and polycell systems at M.I.T. in order to test how much improvement these specialized optimization programs will produce compared to present techniques.

