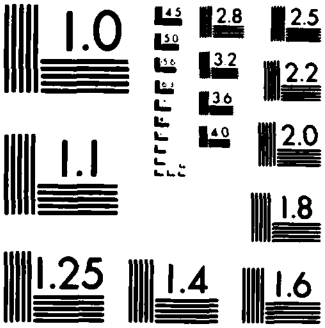
AD-A151 288 CONVERSION OF ALGORITHMS TO CUSTOM INTEGRATED CIRCUIT 1/1
DEVICES(U) MASSACHUSETTS INST OF TECH CAMBRIDGE
RESEARCH LAB OF ELECTRONICS J ALLEN 30 JAN 85
UNCLASSIFIED AFOSR-TR-85-0175 F49620-84-C-0004 F/G 9/5 NL

END
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

FINAL REPORT

Conversion of Algorithms to Custom Integrated Circuit Devices

AFOSR Contract F49620-84-C-0004

covering the period
1 November 1983 - 31 October 1984

submitted by
Jonathan Allen

Massachusetts Institute of Technology
Research Laboratory of Electronics
Cambridge, MA 02139

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
Unclassified

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release;
distribution limited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)
AFOSR-TR- 85-0175

6a. NAME OF PERFORMING ORGANIZATION
Research Laboratory of Electronics

6b. OFFICE SYMBOL *(If applicable)*

7a. NAME OF MONITORING ORGANIZATION
AFOSR

6c. ADDRESS *(City, State and ZIP Code)*
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

7b. ADDRESS *(City, State and ZIP Code)*
Bolling Air Force Base
Washington, D.C. 20332

8a. NAME OF FUNDING/SPONSORING ORGANIZATION
AEOSR

8b. OFFICE SYMBOL *(If applicable)*
NE

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
F49620-84C-0004

8c. ADDRESS *(City, State and ZIP Code)*
Bolling Air Force Base
Washington, D.C. 20332

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
61102F	F08671- 8500175 2305	B3	

11. TITLE *(Include Security Classification)* Conversion of Algorithms to Custom Integrated Circuits

12. PERSONAL AUTHOR(S) Jonathan Allen

13a. TYPE OF REPORT
FINAL REPORT

13b. TIME COVERED
FROM 1/11/83 TO 31/10/84

14. DATE OF REPORT *(Yr., Mo., Day)*
January 30, 1985

15. PAGE COUNT
33

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
VLSI Design, Custom Integrated Circuit Design

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

This project is devoted to the development of computer-aided design techniques at a fundamental and basic level for the creation of high-performance custom integrated circuits to be used for digital signal processing. The project takes a fundamental point of view in considering the design process to consist of a set of transformations between abstract representations at various levels. There are five major of the work.

First, research on specification languages that characterize the functionality of the algorithm to be performed by the chip are considered. These languages must be capable of providing an unambiguous specification of the task but also provide the means to reveal all possible parallelism in the algorithm. A signal representation language has been developed, and concepts from modern data flow languages that are functional but are capable of being compiled into data dependence graphs that reveal all possible

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. □ DTIC USERS □

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL
Robert W Carter, Lt Col, USAF

22b. TELEPHONE NUMBER *(Include Area Code)*
(202) 767-4931

22c. OFFICE SYMBOL
NE

EDITION OF 1 JAN 73 IS OBSOLETE.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

parallelism are being exploited. The second focus is on architectural exploration, whereby architectures derived from the input functional specification can be modified without any possibility of changing the desired input functionality. Emphasis has been placed on the use of the retiming transformation, that can produce optimal throughput for signal processing tasks. In particular, array multipliers have been studied in terms of a trade off between the degree of pipelining and the underlying circuit technology performance. In the third area, we have focused on techniques for generation and composition of cells. We have developed a new regular structure generator which provides a powerful software framework for the generation of high performance regular cells such as register arrays, array multipliers, arithmetic-logic units, program logic arrays, and other important structures. We have also focused on the placement and interconnection of rectangular cells with interconnect on all four sides, as well as means to specify logic so that circuit forms can be composed without any possibility of races in the logic. The fourth area of emphasis is the characterization and generation of optimum circuit performance. This work has included techniques for optimum sizing of transistors to minimize delay, waveform bounding techniques that provide for very efficient and highly accurate circuit simulation, and a high performance circuit extractor from layout that provides the necessary bridge between the mask artwork and the circuit representation needed for simulation. Finally, attention has been focused on the design and construction of special architectures for computer aided design. A special machine for design rule checking has been designed and constructed offering at least two orders of magnitude in speed-up over commercially available design rule checking programs.

The report describes these activities in detail and unifies them together in an overall perspective of computer aided design for high performance digital signal processing applications.

TABLE OF CONTENTS

Overview

In order to place the research done under this contract in perspective we start by describing the view of integrated circuit design that motivates our work. Our goal is to provide design tools that enable the production of high performance chips quickly, correctly, and economically. At the outset, it is important to emphasize what we mean by high performance. We see that there are two aspects of performance associated with integrated circuit design. First there is the performance associated directly at the circuit level which is characterized in terms of speed, power, and area. Thus, for instance, the clock rate at which a synchronous circuit can be run is in large measure dependent on the technology and circuit forms utilized. But there is another form of performance associated with architecture, which is most readily parameterized in terms of the degree of parallelism provided. Thus, in addition to circuit performance, parallelism can help provide greater computational throughput by means of such techniques as pipelining, multiprocessing, multi-ported memories, cache memories, and the separation of instruction store from data store. In many cases, one or the other of these forms of parallelism is utilized to achieve the desired performance. Often designers will choose to maximize the performance available from a given technology and set of circuit forms before utilizing architectural parallelism which inevitably implies a more complicated level of control. There are situations, however, when parallelism will be exploited in a modest technology in order to achieve great throughput. For example, high performance FFT systems have been designed and built using 5 micron CMOS technology but a maximum degree of parallelism is made realistic by the use of wafer scale integration. Despite the virtues of both forms of performance enhancement that we have discussed, there are still many cases where we need both maximal circuit performance and substantial architectural performance through parallelism. These examples are particularly prevalent in digital signal processing, and it is in this arena that we choose to focus most of our attention. It is a safe statement that there are significant digital signal processing tasks, particularly those with military application, that can utilize all of the performance available at any given time. For these reasons, we have focused our efforts on the development and cohesive integration of techniques that provide for effective design utilizing both kinds of performance. In order to exploit architectural performance, we have focused on the development of functional languages that can be compiled to reveal the latent parallelism in the algorithms of interest. Following this phase, techniques for architectural exploration are introduced that allow the designer to systematically move within the space of architectural alternatives, all of which are unified by having the same functional specification. Once a particular architecture of interest has been determined, then it remains to generate the constituent cells or blocks of that architecture in high performance circuit modules. It is convenient to think of the generation of these modules as taking place in one of three ways. First of all, it may be that the module is already available from previous design efforts. This is often the case for input/output pads, as well as a variety of standard cells. It is this technique, of course, that is used in the so called standard cell or polycell approach, where all of the cells utilized have been previously designed. The next technique that may be utilized to generate these cells can be characterized as procedural generation. This approach relies on a technique, given great emphasis here at MIT, that represents a layout design in terms of a parameterizable procedure which when executed will produce the desired mask artwork. Such techniques have been widely used to generate structures such as program logic arrays, but they can be used for much more complicated structures, and we believe that there is still a great deal of useful work that can be done to enhance this method of cell generation. The major problem that we see in this area is that there is a tension in procedural design between specificity and efficiency. On the one hand, it is possible to devise procedural techniques for design that produce highly efficient layouts, but of a very specific sort. The best known example of this approach would be, once more, the program logic array, but there are many other procedural generators in which the functional specification is related to a highly specific target architecture. In our work at MIT, for example, we have developed specialized function generators for array multipliers with modified Booth's encoding, and are currently designing similar techniques for floating point addition and multiplication. Other examples would include register arrays, add/subtract units, and many different kinds of memory arrays. On the other extreme, techniques have been generated that are very general, but do not give high efficiency. Here we can mention the McPitts systems (the principal investigator was a consultant during the design of this system) where an entire processor complete with special purpose registers is generated from a high level functional specification characterized in a LISP-like language. Great generality is afforded in this system, but there is still only one major target architecture. For this reason, some designs will be relatively efficient in McPitts, but many others will not provide adequate circuit performance. We must contrast this situation with the special purpose compilers which provide excellent circuit performance, but have very little generality. Clearly a major task must be to combine the best of both worlds. Specifically, we must try to find a way to retain the efficiency of the special purpose compilers while introducing a greater degree of generality. It is precisely this goal that is attacked by our efforts in designing a regular structure generator, which we describe in the sequel. We have tried to abstract upon those elements of special purpose compilers that are common in order to provide a general framework for special purpose compilers.

The third approach to module generation is of course full custom design, where all details are specified directly by the designer. In a good design system, relatively little of this kind of design will be done, since it is very difficult for humans to provide the needed degree of optimal design and control, and since this approach is exceedingly time consuming. Nevertheless, it is important to be able to translate circuit ideas into layout quickly and characterize the design in terms of an equivalent circuit which can be tested or simulated to ascertain the achieved level of performance. Tools must be provided to verify all aspects of the resultant layout. This includes techniques for design rule checking, topological extraction from layout, unit delay logic simulation, circuit simulation, testability analysis, and test vector generation. All of these tools have been generated in novel and highly effective forms, and have been widely utilized both within the university and industrial environments.

Finally we mention the need to place and route the modules, the creation of which we have described above, in an efficient layout. We have had a long standing research project going on within this overall project to efficiently place and route rectangular modules orthogonally related with interconnect on all four sides. Out of this project has come highly efficient channel routers, as well as novel approaches to module packing and hierarchical routing. More recently, a major theme of this activity has been cell compaction, which is important in order to allow modifications in the basic fabrication technology as well as providing for convenient pitch matching between busses on a overall chip.

At this point it is appropriate to summarize the five major themes of our research. First there is the development of specification languages for algorithms at a functional level that allow for the compilation of the task into a two dimensional graph of modules that reveals all latent parallelism. Here we exploit advances that have been made in functional languages, particularly those activities centering in the data flow area. We are also focusing on characterizations of algorithms in terms of linear algebra operations, both because many modern signal processing algorithms are expressed in a linear algebra formalism, but also because of the need to optimize algebraic operations in high performance circuit simulation, which is a major emphasis of our activity. The second major theme is what we call "architectural exploration" whereby we seek to modify the architecture obtained from the initial functional specification through all possible variations while retaining the initial functionality as an invariant. We have developed several different techniques for this purpose, and have been particularly successful in utilizing one particular technique, called retiming, to carefully study possibilities in multiplier design. We have also shown how to translate any given signal flow graph characterization of a digital signal processing task into an equivalent systolic array form. We believe that these techniques are essential for high quality integrated circuit design, but can also serve as a scientific basis for computer architecture in the large. The third phase of our activities centers around the generation and composition of the modules specified in a particular architecture. Here our activities have centered around specialized compilers for multipliers and floating point units, as well as the generalization on these activities in the form of a regular structure generator. We continue to work on the PI project which provides for comprehensive placement and routing capability, and note that both the regular structure generator and the PI system utilize highly novel and efficient compaction techniques for altering the size of cells in a way that is consistent with the underlying design rules constrained by the fabrication technology. The fourth area of our activities centers around the characterization of circuit performance.

Here we have been focusing on the optimal sizing of devices to provide maximal throughput or minimal power, as well as exceedingly powerful and novel techniques for waveform bounding, which serve to provide accurate estimates of waveform shape without the extreme expense of a full blown circuit simulation. We have also developed under this contract a high performance circuit extractor which can also extract the topological forms of networks. It is important to point out that we see this area of circuit performance optimization as increasingly finding a strong fundamental mathematical base, thus avoiding the mere collection of ad hoc techniques. Lastly, we have focused on the development of special purpose architectures. Under this contract we have completed the design and fabrication of a special purpose architecture for design rule checking which provides two orders of magnitude speed up in design rule checking. This kind of capability fits well with the notion of a high performance interactive work station, where the designer may ascertain the correctness of his or her design very quickly. We are also turning increasing attention to the design of novel architectures for linear algebra tasks, since we recognize the fundamental nature of these algorithms for signal processing purposes as well as high performance circuit simulation.

All of these five main emphases can be seen as a cohesive collection of tasks aimed at high performance custom integrated circuits obtained through a series of transformations from an initial functional specification, most importantly of digital signal processing tasks. In the sequel, we spell out the detailed statement of work, the status of our research, and the publications that have derived from this work.

Work Statement

In this section, we characterize the specific individual projects undertaken under this contract. Keeping the previous overview in mind, it is easy to see how the individual projects fit into the overall plan for addressing the major problems of high performance custom integrated circuit design for digital signal processing applications.

We continue to attack the problem of devising high level functional languages for VLSI design. We know that many hardware design languages capture many performance aspects which we would rather bring out through techniques of architectural exploration. Instead, we seek to develop functional languages that characterize the competence of the algorithms to be performed, while retaining, possibly in latent form, all of the inherent parallelism of the task. That is to say, we do not wish to have an initial formalization of an algorithm that makes it very difficult to find parallelism which could lead to very high performance. In order to help us, we have performed a systematic exploration of techniques developed in computer science for this purpose. Both hardware and software approaches have been used for the automatic extraction of parallelism from algorithmic descriptions, but perhaps the most fundamental approach has been that adopted within the field of data flow architecture. It is too early to be certain if the data flow approach to computer architecture will be useful over a broad class of tasks, but study of this approach has dealt with a wide variety of interesting architectural problems including the semantic characterization of algorithms with a specific intent of revealing latent parallelism. It also turns out that work in the data flow field has led to compilation techniques taking as input the functional language specification and producing as output a two dimensional data dependence graph that illustrates all inherently semantic dependencies, but without introducing additional dependencies that are dependent solely on a particular performance bias. In order to study this area, we are explicitly collaborating with the data flow research group here at MIT, and the student involved in this area works both with the data flow project and our integrated circuit design project. We believe this is a good example of cooperation between projects that can provide additional benefit to all concerned.

The compilation of a functional specification into a data dependence graph using data flow techniques generates one particular architecture which may or may not be appropriate for the final VLSI design. In order to explore all other possible architectures that can be derived from the particular functional specification, an explicit phase of architectural exploration must be provided for. We are committed to developing these new techniques and unifying them into a cohesive design module. Earlier, Miranker, working under this contract, developed the theoretical base for space-time tradeoffs in the context of a general hardware design language schema. These techniques have also been used at a very practical level for gate array design by Darringer and his associates at IBM in terms of manipulations of designs based on multi input NAND gates. At MIT our influence has been directed in two areas.

Firstly, we have introduced a new level of abstraction that characterizes an overall design in terms of blocks of combinational logic and interposed registers. We have introduced basic techniques for moving registers around in these designs to modify throughput without changing the functionality. Furthermore, we have provided constructive proofs for the generation of optimal throughput in any particular design, thus guaranteeing that the best possible allocation of register placement is utilized. For example, in the design of finite impulse response filters, we have shown that it is possible through relatively easy but non-obvious transformations to achieve designs with throughput improvements of a factor of 4 over the initial naive design. We continue to exploit the retiming techniques in a wide variety of applications motivated by signal processing needs. In this report we show as one of our publications a particular example of the use of retiming to study optimized pipelining in the context of a given technology for array multipliers. We believe that retiming will continue to bear many useful results, and that its basic principles must be part of an overall system for architectural exploration. In addition to retiming, we have worked out (as reported in our publications) systematic techniques for transforming signal processing signal flow graphs to equivalent systolic formulations. We also show how to transform between many equivalent signal flow graphs in an insightful way, once again retaining the initial functionality as a invariant in the process. It is also known how to transform systolic array formulations to basic canonical signal processing formulations where noise sensitivity and word length considerations can be examined effectively in terms of the well known results in the literature. Clearly these techniques are also of great utility. Undoubtedly there are other exploration techniques that we continue to look for. We do feel, however, that we have succeeded in demonstrating the great utility of this approach, and further investigation will attempt to reveal the utility of these techniques at the matrix linear algebra level.

The next major area of our emphasis has been on compilation techniques for cell generation over broad classes of circuits. We have commented earlier that compilation techniques have been successfully developed for specialized circuits that give highly efficient results, but that those approaches intended for a more general class of circuits have been very inefficient both in terms of area utilization and speed. Our approach has been to generalize on our highly successful PLA generator which is capable of generating a very large variety of PLAs with many optional features. Essentially what we are doing is to modularize that part of the PLA generator program that deals with specific knowledge of what constitutes a well-formed PLA in terms of its constituent modules and replace this element with a new formal file that characterizes the overall structure to be created. In this way, the example structure which illustrates all possible cell connectivity, the logic or parameter file that is needed to characterize the functional description, and a third file which encapsulates the basic architecture of the device are all brought together to provide a flexible but highly efficient means for generating regular structures. We call the resulting software structure the regular structure generator, or RSG. By means of this tool, we can generate array multipliers, register files, arithmetic logic units, and a wide variety of structures, including the PLA forms that were the original motivation for this work. This is a strong piece of work with a well developed mathematical basis.

Another form of layout compilation is related to McPitts, a general purpose silicon compiler developed at Lincoln Laboratory where the principal investigator consults. The McPitts system as originally developed has its own cell library, but as part of overall compilation process an intermediate level representation is generated which is technology independent. One of our students is currently exploring the interfacing of a commercially available gate array placement and routing software package so as to investigate the efficiencies that may be obtained through the introduction of various levels of modulation and hierarchy in the overall design specification. We believe that this study will provide new insight into the levels of specificity that must be provided to a silicon compiler in order to be able to utilize the best features of a particular layout strategy.

A major focus of our work has been the systematic investigation, using fundamental mathematical approaches, of circuit performance. In particular, we have developed under this contract a novel technique called waveform bounding which provides estimates of upper and lower bounds on a circuit waveform that are guaranteed within a particular specified percentage accuracy. This technique is important since it provides useful performance information without the cost of a full blown circuit simulation. The original introduction of this technique was performed under this contract in the past, and we have now gone on to optimize these results, placing them in a much more insightful perspective in terms of optimal control theory. We have also used Hamiltonian techniques for optimal device sizing in order to both maximize speed and minimize power within a particular circuit change. This has led to a recent doctoral thesis and is of fundamental importance due avoidance of ad hoc formalisms. Just as we have put emphasis on architectural exploration, we are also interested in the exploration of alternate circuit forms that may have performance benefits. For example, even at the simple level of two input NON to NOR gates in CMOS, layout variation can have substantial impact on performance, particularly with respect to the size of side wall drain capacitances located some distance from the ground and 5 volt supply rails. If it is a matter of providing a cell library, then we can certainly avoid undesirable forms of this sought through the initial designs. But in many cases in compilation, due to a variety of compositional processes overall circuit forms may be realized that have undesirable performance characteristics. In this case it is desirable to transform them, once again keeping the basic function invariant, in such a way that the nodal capacitances internal to the circuit and away from the rails are minimized leading to high speed results. We have not yet formalized these exploration techniques but in one of our studies of compilation of floating point adder and multiplier units, a great deal of the effort was focused on the formal means of specifying cells which are guaranteed to be correct without any hazards, and which can be composed safely to form larger units which once again are guaranteed to be correct. A continuing emphasis of this project is to complement these aspects of circuit performance with exploration techniques to guarantee optimal circuit performance utilizing the ideas that we have described above. This is a powerful set of new techniques which we are just beginning to understand.

Since the design of custom integrated circuits is beset with problems of complexity due to the large number of artifacts which must be examined, it is important to look for powerful means to provide the needed generation and verification techniques in an efficient manner. In particular, we have focused on the introduction of special purpose architectures for the performance of several of these tasks, particularly at the artwork level. Our first example of this effort has been the design and construction of special hardware for design rule checking. The goal here has been to design and build a single printed circuit board which could be plugged into a work station as an accelerator for design rule checking. This design utilizes four novel custom integrated circuits, and provides a speed order of two orders of magnitude. We are also focusing increased attention on the design of special purpose architectures for linear algebra operations. We feel that there will be two payoffs from this investigation. First of all, we have noted that many operations in modern signal processing are formulated in terms of linear algebra. Hence a linear algebra engine would be highly useful for digital signal processing. In addition, however, we also note that linear algebra techniques are central to circuit simulation, and hence the development of a highly efficient linear algebra machine would be useful for high performance circuit simulation, once again functioning as an accelerator in the context of an interactive work station for integrated circuit design.

One of the major problems facing the design of integrated circuits together with their fabrication is the generation of test vectors that are comprehensive in coverage and yet efficient in execution. We have undertaken two investigations in this area. Firstly we have investigated the possibility of using high parallel architectures with concurrent algorithms for test vector generation. In particular, the Connection machine developed at the MIT Artificial Intelligence Laboratory has been used as a basis for developing algorithms for test vector generation. This machine provides nearest neighbor communication between its computing elements, and thus provides a base for concurrent algorithms for test vector generation. Simulation studies of this process have been completed, and although speed ups have been obtained of roughly two orders of magnitude, we have been disappointed in the amount of performance improvement we could achieve by means of such massive parallelism. Another aspect of testing that has proved to be a big problem is the presence of what is called "reconvergent fan-out" in circuits. In this situation, a particular signal is fanned out in at least two different ways but then brought together later on in the circuit at a particular point. It turns out that the presence of reconvergent fan-out leads to many difficulties in the generation of test vectors, and we have undertaken to find methods to automatically detect the presence of this phenomenon. In the process, we have revealed many difficulties in the coverage of supposedly good test vector generation schemes, and our goal is to not only discover areas of reconvergent fan-out, but to transform the circuit in such a way that the reconvergent fanout is made to disappear while of course preserving the overall circuit function at a functional level.

A long standing emphasis of this project has been the development of placement and interconnect techniques for the automatic connection of rectangular modules orthogonally related with interconnect on all four sides. The collection of programs that achieves this goal is called the PI system, and it has served as a vehicle for a number of fundamental research projects in computer science theory. This has led to techniques in optimal placement, highly efficient channel routers, hierarchical routing schemes, power routing methodologies, and most recently techniques for automatic compaction for the minimization of area and matching of bus pitches. In this final report we give new results that have been achieved in this project and indicate the overall direction of continued activities.

Finally, in order to provide continuing feedback to our research, we continue to provide an overall design environment for the generation of custom integrated circuits focused on digital signal processing applications. In this way, tools, algorithms, and other techniques that are being generated can be continually utilized and tested, so that none of the fundamental research is carried out in a vacuum. The principal investigator is in charge of the basic MIT course in integrated circuit design, and hence is continually involved with a large number of student projects and the folding in of new research results into this teaching environment. We have also been active in responding to numerous requests from outside universities and industries for our research reports as well as particular programs that have proved to be highly useful.

Status of Research

In this section we indicate the progress achieved in the various areas of focus detailed in the previous section. We give brief insightful remarks as to the nature of these results, but the interested reader may refer to the references given in this section to the publications listed in section four of this report. Copies of any of these publications may be obtained by contacting the principle investigator.

As mentioned before, we continue our efforts to adapt high level functional data flow languages to the needs of VLSI design. Here the problems which we address include the determination of which hardware site a computation is performed at, when is it performed, what is the communication to other hardware computational sites, and how are these computations related to physical packaging constraints. At present, we do not feel that the form of results available in the data flow area is appropriate for our needs in VLSI design, so we are continuing to work with this formalism in order to best utilize these techniques. In earlier work, we did address the semantic nature of signal processing algorithms and developed the language SRL, or signal representation language. This language has been implemented on LISP machines and is currently being used both here at MIT and at the Fairchild Artificial Intelligence laboratory.

In the area of architectural exploration, we have focused most of our attention on the use of retiming transformations and also on the systematic specification of systolic wave algorithms. In the area of retiming, we have been concerned about the interaction of architectural parallelism and circuit performance. In reference (1), we study the way in which various levels of pipelining, obtained through retiming transformations, interact with the underlying circuit technology performance. We also utilized the regular structure generator to produce all of the layouts needed to provide the base for studying the detailed circuit performance of these systems. We believe that this is the first time that the space of pipelining alternatives in such an important module as an array multiplier has been systematically studied thanks to the availability of the retiming transformation and the regular structure generator layout tool. Previous investigators have not been able to address this many alternatives since the tools for generating and evaluating all these tradeoffs were simply not available. We have also been concerned about the specification of systolic algorithms. In the literature, it is common to find systolic algorithms specified by graphical means, which lack formality and preciseness although they are certainly insightful. In reference (2) we show how systolic algorithms can be specified formally, and also how their correctness can be verified in terms of underlying signal flow representations. Once again, this work may be seen as a fundamental attack on the specific kinds of architectural transformations needed in order to study high performance signal processing systems. Although no publication has yet been developed, we have also shown how to transform between a variety of functionally equivalent signal flow representations. A publication in this area will be available shortly.

We now turn to the area of cell generation and composition. One of the most seminal works developed in our laboratory has been a "design by example" PLA generator. (Reference 3) In this approach, a PLA is specified completely by showing the interconnections between its constituent cells in a simple example. Thus it is possible to specify all of the necessary interconnections within a PLA by demonstrating a two-input, two-output, and two-product-term PLA. The HPLA program is capable of generating a PLA of any size from a description of its logical characteristics and the minimal example just cited. This program has been exceedingly well received and is widely used both here at MIT and elsewhere. In the course of the design of HPLA, we realized that it was not necessary for the example to show a minimal version of the circuit to be designed, but that it was sufficient if all of the intercell interconnections were demonstrated by the example, even if the example itself never occurred in the final design. This led us to the idea of a design by example regular structure generator which can generate a wide variety of structures from a set of constituent cells, the functional specification of the overall circuit, and the example which demonstrates all needed cell interconnections. This program has also been exceedingly successful and has been used to generate array multipliers, register files, arithmatic logic units, as well as the PLA's originally generated by HPLA. Reference (4) gives a detailed description of this approach. As a result of our work in the procedural generation of floating point units, we have developed a new set of procedural rules for the composition of digital circuits so that the logical performance can be realized by the composed circuits without any danger of circuit glitches such as hazards. We feel that this is an important contribution and that techniques of this sort that guarantee that the result obtained by composing a circuit from constituent cells is in fact a well behaved circuit from the point of view of correctly representing the logical function desired, one essential for silicon compilation. A thesis in this area will be forthcoming shortly.

We now turn to the area of circuit performance and performance constraints. Here our attention has been focused both on optimization tools for circuit performance, but also on the development of computationally attractive means to specify and characterize circuit performance without the need for comprehensive circuit simulation. References (5,6,7, & 8) give a view of our work in this area relating to the waveform bounding technique. We have been able to show that it is possible to bound a waveform both above and below to any specified degree of accuracy while guaranteeing that the true waveform is constrained within these bounds. It also turns out that the computational means to achieve these bounds is very efficient. Much of the early work was focused on RC networks, but we are now attacking the use of bounds on active elements. Reference (7) is a very significant result since it characterizes the waveform bounding problem as a optimal control problem and shows how optimal bounds can be found for any desired degree of accuracy in RC tree networks. We are continuing to investigate and utilize the waveform bounding approach, particularly as we combine it with the relaxation type of simulation. Waveform bounding can be used to development an initial starting point for relaxation simulation, thus achieving increased efficiency while retaining the high accuracy obtainable with relaxation techniques. Reference (9) is an important paper showing how delay and power optimization in VLSI circuits can be achieved. These techniques have been further refined in a doctoral thesis, the content of which is described in references (10 and 11). First a macro-modeling technique is used to characterize the time performance of MOS VLSI circuits to approximately a 5% accuracy, while saving a large amount of computation. Then Hamiltonian techniques are used to optimize these resultant models, again providing very accurate results at an attractive computational budget. References (12 and 13) characterize the EXCL circuit extractor which has been developed in our laboratory for the very high accuracy extraction of circuit parameters from integrated circuit layout. This program has proved to be very popular both within MIT and outside. Resistances are automatically determined by either counting squares, using form factors, or direct solution of Poisson's equation. The capacitances are also calculated very accurately, including internodal capacitances as well as the traditional capacitances calculated to the substrate. This program is also modular with respect to technology and can be used to extract the typology of the circuit in a form suitable for unit delay logic simulation by a simple modification of a single program parameter. EXCL also has the virtue of providing the circuit output in the form of a SPICE input deck, thus saving the user a large amount of tedious work in analyzing the performance of the circuit.

We have commented that a strong interest of ours has been to speed up many of the CAD algorithms, particularly those associated with artwork, since the complexity of these calculations is very high. We have now completed the design and construction of a single board for computing design rule checking. This technique utilizes four new custom integrated circuits for pattern matching, and is complete with a UNIBUS interface so that it may be conveniently used within an interactive work station. This is the first such hardware excelerator for design rule checking that has been achieved, and it has had substantial impact both within the university and industrial environments.

We have devoted a great deal of attention to the study of test vector generation. In one study, described by reference (14), we investigated the way in which test vectors could be generated using a highly concurrent machine, namely the Connection machine developed at the MIT Artificial Intelligence laboratory. This machine provides nearest neighbor communication between its several modules, and we have been able to show that substantial speed-up in test vector generation can be accommodated in such an architecture. Nevertheless, although a speed-up of about a factor of 50 can be obtained, the presence of additional processors does not provide additional speed-up due to communication difficulties in the architecture being utilized. It may be, of course, that other special purpose architectures more appropriate for test vector generation may provide greater performance, and we are continuing to investigate this possibility. On another front, we are systematically investigating the impact of reconvergent fanout on the problem of test vector generation. In a thesis that should be available shortly, we show the difficulties introduced by reconvergent fanout in many currently available test vector generation schemes, but also how this phenomenon can be detected in practical circuits. This is an area where we hope to use our techniques of architectural exploration and modification to transform the difficult presence of reconvergent fanout to a more benign architecture where the test vector generation problem is much easier. We believe that this is the first time that any investigator has attacked this problem in a straightforward way and attempted to find alternate means to represent logic without these difficulties. Test vector generation is a very large problem, and unless ways can be found to minimize its computational complexity, radical new architectural partitioning techniques will have to be used in order to make testing feasible.

The PI project, which has served as a framework for many important complexity level investigations in applied computer science theory, is continuing to develop into a highly optimized system. Reference (15) shows how power and ground wires can be routed in the PI system on a VLSI chip. We expect shortly to show how new compaction techniques can be utilized with the PI system to provide attractive packing densities. In particular, it appears that a new representational technique for the constraints used in compaction may allow us to speed up the compaction process by as much as two orders of magnitude with minimal change to the algorithms that are utilized.

Finally we come to a discussion of the applications of our techniques in signal processing. Reference (16) gives an overview for how these techniques can be used to design high performance signal processing chips in the speech area. Reference (17) is an invited paper soon to appear in the proceedings of the IEEE which gives a comprehensive view of computer architecture for digital signal processing. We believe that this is the most comprehensive review of this field available, but which also gives an appropriate view of the direction of future research.

During the period of this contract, the principle investigator has been actively involved in the CAD professional field. He has served on the program committees of both the 1984 Custom Integrated Circuit Conference and the 1984 Integrated Circuit CAD Conference. He has also served as a session chairman at both of these conferences. In addition, he was the organizer of a special panel at the 1984 conference on VLSI and modern signal processing which dealt with the relationship between algorithms and VLSI architectures. It is our belief that the results achieved under this contract are significant advances in the development of fundamental theoretically-based techniques for the optimal determination of signal processing architectures and the underlying circuit technology, as well as the way in which these two interact. Our additional focus on the utilization of functional languages for the specification of signal processing algorithms, starting with the SRL language and now extending to our new work in the data flow area is unique, and in fact is the only rigorous language work available for the expression of signal processing algorithms in a form amenable to VLSI design. In the future, we intend to maintain our emphasis on the development of fundamental techniques based on solid mathematical theory and the introduction of novel and appropriate abstractions that have great impact on the VLSI design process for signal processing applications. We welcome discussions on any of the topics that we have discussed in this report.

References

1. Hauck, C.E.; Bamji, C.S.; and Allen, J. "The Systematic Exploration of Pipelined Array Multiplier Performance" Proceedings 1985 International Conference on Acoustics, Speech, and Signal Processing. March, 1985.

2. Kuo, C.J.; Levy, B.C.; and Musicus, B.R. "The Specification and Verif- ication of Systolic Wave Algorithms" VLSI Signal Processing, IEEE Press, 1984, pages 271-281.

3. Bamji, C.S. "HPLA: A Design by Example PLA Generator" MIT VLSI memo no. 84-164. March 1984. 19 pages.

4. Bamji, C.S.; Hauck, C.E.; and Allen, J. "A Design-by-Example Regular Structure Generator" submitted to 22nd Design Automation Conference, June 1985. 14 pages.

5. Wyatt, J.L.; and Yu, Q. "Signal Delay in RC Meshes, Trees and Lines" Proceedings of the IEEE 1984 International Conference on Computer-Aided Design (ICCAD-84), November, 1984. 3 pages.

6. Wyatt, J.L. "Signal Delay in RC Mesh Networks" MIT VLSI Memo No. 84-196, August, 1984. 7 pages.

7. Tan, H.-N.; and Wyatt, J.L. "Time Optimal Trajectories Associated with Voltage Bounds in RC Tree Networks" MIT VLSI Memo No. 84-205, October, 1984. 18 pages.

8. Wyatt, J.L. "Monotone Sensitivity of Nonlinear Nonuniform RC Transmission Lines, with Application to Timing Analysis of Digital MOS Integrated Circuits" IEEE Transactions on Circuits and Systems, Vol. CAS-32, No. 1, January 1985. Pages 28-33.

9. Glasser, L.A.; and Hoyte, L.P.J. "Delay and Power Optimization in VLSI Circuits" Proceedings 21st Design Automation Conference, June, 1984. 7 pages.

10.Matson, M.D. "Macromodeling of Digital MOS VLSI Circuits" submitted to 22nd Design Automation Conference, June 1985. 19 pages.

11.Matson, M.D. "Optimization of Digital MOS VLSI Circuits" submitted to 22nd Design Automation Conference, June 1985. 16 pages.

12.McCormick, S.P. "EXCL: A Circuit Extractor for IC Designs" Proceedings of the 21st Design Automation Conference, June 1984. 9 pages.

13.McCormick, S.P. "Automated Circuit Extraction from Mask Descriptions of MOS Networks" MS thesis, MIT Dept of EECS, February, 1984. 123 pages. Also Research Laboratory of Electronics Technical Report 503, February, 1984.

14.Seiler. L.D. "A Hardware Assisted Methodology for VLSI Design Rule Checking" MIT Dept. of EECS, Ph.D. thesis, February, 1985. 274 pages.

15.Kramer, G.A. "Test Generation Algorithms: Overcoming the Von Neumann Bottleneck" Proceedings 1983 IEEE Automatic Test Generation Workshop March, 1983. 10 pages.

16.Moulton, A.S. "Routing the Power and Ground Wires on a VLSI Chip" MS Thesis, Dept of EECS, MIT, May, 1984. Also MIT VLSI Memo No. 84-184. 65 pages.

17.Allen, J. "Computer Architecture for VLSI Speech Processing" Proc- eedings of Speech Tech '84, April 1984. 5 pages.

18.Allen, J. "Computer Architecture for Digital Signal Processing". accepted for publication in the IEEE Proceedings.

PROFESSIONAL PERSONNEL

J. Allen

L.A. Glasser

B. Musicus

R. Rivest

J.L. Wyatt