This is to certify that the

dissertation entitled

Performance Tradeoffs in the

Hierarchical Design of Regular VLSI Structures

presented by

Yu-Ying Jackson Leung

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Elect. Engr.

Michael Shanblatt
Major professor

Date 12/10/85

PERFORMANCE TRADEOFFS IN THE HIERARCHICAL DESIGN OF REGULAR VLSI STRUCTURES

By

Yu-Ying Jackson Leung

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical Engineering and
Systems Science

1986

ABSTRACT

PERFORMANCE TRADEOFFS IN
THE HIERARCHICAL DESIGN OF
REGULAR VLSI STRUCTURES

By

Yu-Ying Jackson Leung

Hierarchical layout is the prevalent design methodology in the computer-aided generation of VLSI circuits. But, the tradeoff of design-complexity versus circuit performance among the various design paths of this hierarchy has yet to be fully assessed and compared numerically. This research provides a systematic and quantitative investigation of this tradeoff for regular structures.

Systolic array structures, for which regular levels of modularity are well defined, are used as testbed structures. The hierarchical regularity levels in a systolic structure are the transistor, gate, functional device, processing element (PE) and algorithmic levels. The investigation of modularity is pursued along two distinct paths -- the bottom-up and top-down approaches.

Using the bottom-up approach and standard NMOS design techniques, several PE's are independently designed using the various design entry-points and pathways of the design hierarchy. Results provide a set of area-performance versus design-complexity index figures that represent the tradeoffs among the various design pathways. The general

expressions for the complexity, chip area and propagation time are numerically derived based on an infinite array model.

The top-down approach is applied to the algorithmic and PE levels where chip-wise modularity is desired. The various chip-wise decompositions of an array structure are parameterized by an I/O bottlenecking index (BI). Results show the relationship of BI to the number of pinouts, word size in number of bits and chip size.

Results obtained from the bottom-up and top-down approaches provide an analysis of several tradeoff parameters. An appropriate design pathway can thus be chosen with respect to the desired tradeoff parameters of the final design.

To my mother and eldest brother

Mdm. Me Chit and Mr. Wai-Ying Tommy Leung

ACKNOWLEDGEMENTS

The author wishes to express his sincere appreciation to his major advisor, Dr. Michael A. Shanblatt, for his guidance and encouragement in the course of this research.

He also wishes to thank the committee members, Dr. E. Goodman, Dr. P. D. Fisher, Dr. R. A. Schlueter and Dr. D. Yen for giving the guidance, suggestions and comments in this work.

Other thanks are given to Dr. Goodman, Director of A. H. Case Center for Computer-Aided Design at MSU, for allowing the author to use the Computervision CADDS system, and Dr. Stephanie Shanblatt for the proofreading of this dissertation.

Finally, the author owes a special thanks to his wife, Jasmine, for her emotional encouragement and support, and especially, for her patience while the author was in East Lansing pursuing this work.

From alpha to omega, Glory to God for endlessly providing the love, wisdom, hope, faith and forgiveness.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

A methodology for VLSI (Very Large Scale Integration) circuit layout beyond the element level is rapidly emerging. Hierarchical circuit design for VLSI-based structures provides simplicity in both hardware specification and verification. These hierarchical design methodologies have become the prevalent technique in the computer-aided design (CAD) of VLSI circuits [1, 2]. In particular, a hierarchical technique provides designers with the flexibility of approaching a VLSI design via several different pathways, the choice of which corresponds to the final circuit performance. Moreover, circuits designed using different hierarchical levels will entail different design complexities which relate to a measure of design-time.

Specifically at higher levels in the design hierarchy, IC (Integrated Circuit) layout requires a relatively low design-time but tends to produce circuits larger in size and with slower propagation delay. At lower levels, however, layout requires more design-time but can be anticipated to produce more efficient circuits in terms of area and delay time parameters. In other words, there is an obvious tradeoff among design-time, final chip area, and circuit performance. This tradeoff is intimately related to the choice of the specific pathway traversed in the hierarchical design methodology.

This research further investigates and contributes to aspects of hierarchical circuit design techniques for regular structures and is aimed at enabling ultra-dense circuit design, layout and verification to become an efficient and routine task. Broadly, two major types of VLSI-based structures can be defined. They are regular structures, such as memory, logic-array and systolic array structures, and random (non-regular) structures such as microprocessor architectures. Since multi-level regularity and modularity are well defined in regular structures, the hierarchical design of this type of circuit is of particular interest in this work.

Regular structures contain arrays or meshes of interconnected modules which possess the properties of modularity and local connectivity. These regular modules can be subdivided into sub-modules, the sub-modules can further be subdivided and so on, such that the multi-level modularity of the design hierarchy is distinctly defined. Within this hierarchy, modules from the preceding level may be tessellated to form modules in the next level. For example, transistors can be tessellated to form gates, gates tessellated to form functional devices, and so on. Finally, high level functional devices, called processing elements (PE's) are tessellated to implement a particular algorithm. Tessellation implies that modules are designed and placed on the chip surface so as to "tile the plane" or fill the chip surface with minimum null interstitial space. In the literature, modules such as processing elements are depicted as circles or hexagons, but in actual circuit implementation, these devices usually do not possess such ideal geometric shapes.

The ultimate goal of this work is to define a set of general expressions using several defined parameters such as complexity, utilization of chip real estate, circuit performance, pin number and global input/output (I/O). Using this set of expressions, tradeoffs among the multiple levels of the design hierarchy can be observed. Therefore, a designer can choose an appropriate design pathway with respect to the desired tradeoff parameters of the final design.

1.1 Problem Statement

All current design methodologies tend to tradeoff design turnaround time for circuit performance. In general, fully custom designs produce IC's exhibiting the best performance and density but require the longest design time. On the other hand, standard cell and gate-array approaches provide much faster turnaround time but greatly degrade circuit performance and chip area utilization. Therefore, a methodology aimed at producing IC designs with optimal tradeoffs among design-time, circuit performance and chip area utilization is desired. Ideally, a well defined methodology must provide flexibility in design tasks and verification such that it is useful at any level of IC design.

A hierarchical methodology appears to be the best approach to achieve such flexibility. In addition, a methodology of hierarchical modularity corresponds to the current trend that requires increasing hardware regularity at successive levels to enable cost- and time-effective layout and verification. However, the tradeoff of

circuit performance versus design complexity among the various entry points of this hierarchy has yet to be fully assessed and compared numerically.

The testbeds for the hierarchical design of regular structures explored in this research are systolic array computing structures. Systolic structures consist of arrays of interconnected processing elements (PE's) which possess the properties of regularity (modularity), local communication, and parallelism and pipelining. The multi-level regularity of these structures makes them an ideal candidate for studying hierarchical design modularity. Regularity in systolic structures is present at the following levels of increasing complexity:

-- transistor modules,

-- gate modules,

-- functional device modules,

-- PE modules.

Systolic structures have already been shown to hold great promise in engineering systems and other applications including signal and image processing, matrix arithmetic and graph theoretic algorithms [3-8]. However, systolic arrays have yet to be widely accepted in industry due to a lack of sound and well defined design methodologies and a feasible solution to the chip level I/O limitation [9, 10].

The goals of this research are twofold as follows:

1. This research is to provide a systematic and quantitative investigation of the tradeoffs for several levels of

entry-points and paths within the hierarchical design of regular structures. In other words, design tradeoffs of systolic array or other similar structures can be parameterized in terms of design time or complexity, system performance and chip area utilization. This parameterization will vary depending on the levels at which the design is performed. Thus, the design-complexity/performance tradeoff of this design methodology depends on the level or levels that are chosen.

2. The second goal is to parameterize the highest level of regularity possible -- modular algorithmic decomposition onto chips. This is specifically to be examined with respect to I/O limitation parameters. Structural decomposition of systolic arrays presents the possibility of infinite dimension problems (bandwidth and/or word size) to be implemented by a set of chips. Decomposition of this type is physically constrained by near term IC lithographic linewidth and chip size limitations. In addition, the I/O bottleneck caused by limitation of the number of pinouts per chip, may result in degradation of overall system performance. Thus the goal of this phase of the research is to study the effect of I/O limits on the geometric decomposition of regular arrays.

As a whole, this research aims to define the tradeoffs among the various levels in hierarchical layout design for regular structures. Two approaches will be presented -- the bottom-up approach (Chapters 3 and 4) and the top-down approach (Chapter 5). In general, a top-down

approach is first used to design the functional circuit and then a bottom-up approach is applied to the design of the final layout [11].

1.2 Approach

In order to investigate and further define design tradeoffs and methodology for hierarchical modularity in VLSI circuits, structures possessing multi-level regularity and modularity are used as testbed or benchmarking structures. Of particular interest is the specification of systolic computing structures for matrix calculation. The levels of this hierarchy to be examined are:

1. The transistor level at which basic transistor modules (pull-up and pull-down) are defined and tessellated to form basic gates;
2. The gate level at which basic gates (e.g. NAND, NOR, etc.) are defined and tessellated to form functional modules (e.g. functional devices);
3. The functional device level at which basic functional devices (e.g. full adders) are defined and tessellated to form processing elements (e.g. multiply-add cells);
4. The processing element (PE) level at which PE's are defined and tessellated to form numerical computing structures;
5. The algorithmic level at which numerical decompositions provide chip-wise modularity.

Levels 1, 2, 3 and 4 may be lumped together and considered as four on-chip device levels of increasing complexity and viewed externally as computational modules. Thus, the investigation of modularity can be pursued along two paths -- the bottom-up and top-down approaches described in the following sections.

1.2.1 The Bottom-Up Approach

The bottom-up approach involves the specification of increasingly complex, tessellated functional modules. This implies designing modules at the transistor, gate, functional device and PE levels. These design tasks were performed on the Computervision CADDS 2/VLSI design system at Michigan State University. However, it will be shown later that the final results of this research do not depend on any particular CAD system.

Circuit simulation programs are developed to parameterize net propagation delays and chip area requirements. These results are used to specify the performance and chip area utilization of the structure when designed at specified levels. A heuristic assessment of the design task complexity with respect to the relative time required for the different hierarchical design paths are developed and justified. This assessment is based on a parameter of circuit complexity determined as a function of the number of building blocks or modules requiring placement, the number of interconnecting lines or wires requiring routing, or the compactness of the target circuit (active region/module size).

The final goal is to provide circuit designers with a choice of the best balance between the cost or design complexity versus the performance specifications for a given circuit.

Specifically, the tasks in the bottom-up approach are as follows:

1. As a basis to the study of tradeoffs among the various design levels, modules at several levels are designed and simulated using standard NMOS design rules. Transistor modules and gate level modules are first designed and stored in the cell library of the CADDS 2/VLSI design system. Next, independent full adders (FA's) are carefully crafted and tessellated based on the predefined transistor or gate modules. Then, five independent multiply-add cells (MAC's) are likewise designed corresponding to five different design paths of mixed levels. Finally, VLSI systolic array structures are tessellated from MAC's and dynamic latches.

2. Overall performance and total chip area of the conglomerate systolic array structures built from the combination of the various modules are obtained. Since the complete systolic array structure is merely a tessellation of MAC's and dynamic latches, its total chip area is simply the sum of the areas of all modules. The propagation time of the array is calculated by using a charging-discharging model. This model, taking communication path and effective load capacitance into account, provides an estimation of time delay accurate enough for comparative purposes [12]. As a result, five different sets of area and delay time data, corresponding to each different

design path, are obtained. These data are then normalized to provide index figures of the performance of the various ultimate systolic structures.

3. A heuristic measure assessing design cost is examined in terms of design complexity. The complexity of the structures is represented by the number of devices (modules) to be placed, the number of interconnecting lines to be routed or the compactness of circuit (active region/module size). A selected parameter is normalized in order to provide an index of the design complexity for the different hierarchical paths. Finally, a complete set of performance versus design complexity index figures is obtained.

4. An automated testing methodology for systolic array structures is investigated with respect to its specific effect on design complexity. Scan design used in level-sensitive scan design (LSSD) [13] and scan path techniques [14], which of course affects the chip area utilization and performance, is considered.

1.2.2 The Top-Down Approach

The top-down approach is applied to the algorithmic and PE levels where chip-wise modularity is desired. Chip level modularity, which is

necessary to alleviate the pinout limited I/O problem, is attacked by algorithmic decompositions. These decompositions are studied by evaluating the effect of various physical partitions to the array structure. The ideal size of a partition matches, in the sense of throughput, the maximum feasible I/O bandwidth.

The specific tasks performed here are as follows:

1. To study the decomposition of a final target systolic array structure; an optimal number of PE's that will fit on a single chip is determined. Too few PE's on a chip implies that the I/O bandwidth capacity is underutilized. On the other hand, too many PE's on a chip will cause an I/O bottleneck of operands. Next, MUX/DMUX schemes that affect the I/O bandwidth and the number of pinouts are examined. Finally, general parameters are developed for evaluating the effect of the structural decomposition of regular arrays with respect to I/O limitations.

2. Using the testbed systolic array, the general parameters are depicted graphically. As a result, the tradeoffs among the I/O bottleneck, the number of pinouts and the chip size limitation are explicitly shown.

CHAPTER II

BACKGROUND

During the 1950's, the photolithographic process for the fabrication of transistors on crystalline silicon was first developed. In the 1960's, the integrated circuit (IC) fabrication process including design and testing was largely manual. Process parameters, such as diffusion temperature and time, metal line widths and spacing were characterized primarily through trial-and-error. As the technology progressed through the 1970's, the number of devices per chip just about doubled every year (Moore's law) and design costs grew nearly as fast as complexity [15]. This increase in complexity and cost is largely responsible for the proliferation of design automation (DA) facilities and computer-aided design (CAD) for circuit layout and verification.

In the early 1980's, a new type of computerized tool, the computer-aided engineering (CAE) workstation with general IC layout editing capability was developed to help cope with the growth of IC complexity [16]. But, there are still many problems dealing with ultra-dense circuit design and verification. For example, circuit simulation or testing may run for weeks or months in order to obtain accurate results [17]. This and other reasons have promoted hierarchical design to rapidly become the prevalent approach in VLSI design [1, 2].

Today, numerous design systems or tools (either fully automated or computer-assisted) using various design methodologies are available commercially. In the following sections, aspects relating to the state-of-the-art in today's VLSI technology are discussed.

2.1 VLSI Design Methodology

VLSI design practices vary from the fully integrated, highly automatic gate array design capabilities of the large systems manufacturers to the computer-assisted but still somewhat manual methodologies of the designers of high-density custom MOS or bipolar microprocessors. A typical composite state-of-the-art design system includes interactive and graphics terminals, host mainframe computer, control and release system, multimode hierarchical data base and an automated verification system for design rule checking.

Modern design automation systems are powerful tools for the synthesis and analysis of VLSI circuits. Logic entry is an interactive task which is supported by intelligent engineering workstations. The verification of specifications of system behavior is accomplished through design reviews, emulation on existing hardware, and simulation using general- or special-purpose simulations. Simulators are used to verify a system design in terms of functional components. Ideally, a mixed-mode simulator capable of combining behavioral, unit logic and switch level, and analog circuit level models is desired [18, 19].

A mixed-mode simulator, which has proven to be flexible and cost-effective, is especially crucial to the hierarchical design methodology [18, 20]. It allows different elements of a complete circuit to be modeled and simulated at different levels of detail. In addition, it allows for digital macromodeling and mixing logic and timing simulation with transient analysis at device or macromodel levels. Most mixed-mode simulators can handle primitives ranging from high level devices (op-amps, registers, etc.) down to the transistor level. This is made possible by the use of the same program from logic verification down to transistor circuit design.

In mixed-mode simulation, a circuit can be simulated and evaluated in timing, unit delay or multiple delay mode [18, 19]. The rise and fall delays of a normal gate are computed automatically from the transistor current-voltage curves and load capacitances. For logic gate and functional primitives, the rise and fall delays (unit or multiple) are computed from user specified delay coefficients, gate types, routing capacitances and the number of fanouts. The mixed-mode simulation also allows the mode to be changed dynamically (i.e., a mode switching during simulation) providing a more cost-effective simulation [18-20].

Hierarchical design methods, utilizing the "divide and conquer" principle, are considered a means of managing the VLSI design and verification problems [1, 21]. The primary concepts are abstraction, repetition and the use of a database library. Abstraction is a method of replacing an object by a simplified version that only defines the interactions of the object with its environment while neglecting the internal organization of the object. The virtue of abstraction is data

reduction, often by one or more orders of magnitude [1]. Several levels of abstraction may be required for VLSI systems. Repetition is an easy and often applied method of simplifying design. It is most useful for regular (array type) architectures (RAM's, ROM's, PLA's and systolic arrays). The use of a database library avoids redesign.

2.1.1 IC Design Systems

Today, the practice of the VLSI circuit design is so broad that there is no single IC design system which universally meets the needs of every IC designer or engineer. Current IC design systems include computer-aided design (CAD) systems, design automation (DA) systems and computer-aided engineering (CAE) workstations. The differences and similarities among CAD, DA and CAE are described in the following paragraphs.

CAD systems range from the use of simple, interactive graphics and digitizing systems to individual programs used for circuit or logic simulation, mask layout, and data manipulation or reformatting [22]. CAD can also be characterized as a batch-oriented process with a designer guiding each design step such as functional, logic, test and layout design. In other words, CAD is a collection of hardware and software tools to provide the designer with assistance during each design step. Although there is often a lack of smooth links among those different tools used during the design cycle, a good CAD system provides a designer with a rapid and orderly method for consolidating and evaluating design ideas. In addition, it relieves the designer of

numerous mechanistic and sometimes very repetitive design steps.

Unlike the CAD system, all aspects or steps of a design automation (DA) system are interrelated. Every step in DA shares the same database and draws information from every other step [23]. Thus, DA smoothes the CAD design process by insuring that the completed work of each step properly relates to that of other steps and can be passed efficiently to the next. The tools used by a DA system depend on the design methodology. Basically, they are

-- text editors and documentation systems for design specification;

-- functional and logic simulators for initial checking;

-- test analyzer, generator, simulator and grader for design testing;

-- timing analyzer and waveform displays for timing analysis;

-- layout graphic editors, routing and placement algorithms, layout compacter and design rule checker for circuit layout;

-- tooling and art work programs for mask or pattern generation.

Computer-aided engineering (CAE) workstations are a combination of the CAD and DA system philosophies. A CAE workstation must be able to handle the tasks of schematic capture, logic simulation, timing analysis, data reformatting and manipulation, documentation and interfacing with other computers. Additionally, CAE systems contain other tools for analysis and automated layout [16]. Following the procedure of the schematic capture, where netlists having a special syntax are produced, the CAE station can immediately simulate and analyze the design automatically. The interfacing with another computer (mainframe) allows detailed simulation or other time-consuming tasks to

be handled more efficiently.

Ideally, a complete IC design system is composed of a host mainframe computer interconnecting with numerous standalone CAE workstations, high resolution color graphic terminals, technology independent DA tools or software packages, a silicon compiler for custom IC designs, mixed-mode simulators, and sufficient memory for database and cell libraries. Software packages must include aids for design specification and partitioning, system and circuit synthesis, system partitioning, simulation at various levels, IC mask layout, design verification, testability evaluation, test sequence generation and design documentation. In addition, hierarchical design must be fully supported and the design system should support functional (logic translation), testability (simulation) and physical (layout) designs to be done in parallel throughout the design process.

2.1.2 Application-Specific IC Layout

Application-specific IC design can be categorized by three approaches from the standpoint of design complexity and circuit performance. Arranged in decreasing order of design complexity and circuit performance, they are the fully custom, semi-custom and gate-array approaches.

The fully custom approach involves the placement and interconnection of devices at the transistor level so as to achieve maximum packing density and circuit performance. This approach, without using any predefined cell library, requires a complete set of unique

mask layouts to define the user-specific design. As a result, the fully custom approach represents the best method in terms of highest performance and chip area utilization. On the other hand, the high design costs coupled with long design and development time tend to limit the practicality of this approach.

Semi-custom design, which is also known as standard cell, building block or masterimage approach [24, 25], involves the use of fully characterized standard circuit cells extracted from a cell library to implement a desired design. Elements or cells, characterized in electrical and performance terms, are extracted from this library and arranged manually, or under automatic CAD control for placement and routing.

The different types of standard basic cells that are prevalent today include random logic cells (e.g. gates such as inverters, NAND, NOR and XNOR), storage cells (e.g. latches and flip-flops), I/O pads (e.g. tri-state and push-pull drivers) and analog cells (e.g. op-amps, A/D and D/A converters). Larger cells called macros are also commonly found in the library. These include, for example, full adders, shift registers, 1-bit ALU's and decoders. Ideally, all cells should be technology independent and standardized in all design systems. But, due to the individual interests of the designers and manufacturers, such a library has yet to be realized.

Usually, once all the required cells have been specified, layout, simulation and mask generation are handled automatically by the CAD system and the prototype of the design can be obtained within several weeks or months depending on the chip complexity [24]. Thus, in

general, the design turnaround time of the semi-custom approach is faster than that of the fully custom approach due to the previously defined and characterized cells which allow faster layout and verification.

An unfortunate consequence, however, is that since the dimensions and I/O connections of all the individual cells are fixed, layouts using the semi-custom approach exhibit longer interconnection lines and lower packing density when compared to the fully custom approach. In addition, long lines and large sizes imply longer propagation delays and increased power requirements. Additional costs and time are required to update and maintain a complete standard cell library database since the predesigned cells and macros must keep up with the fast changing IC technology. Finally, although standard cell design uses predesigned library modules, it also requires a complete unique set of mask layouts delaying manufacturing turnaround time.

A gate-array (also known as uncommitted logic array, masterslice, cell array or universal array) consists of a prefabricated and fixed array of identical transistors or commonly used logic gates. These arrays of cells can be interconnected as required by appropriate customization of one or a few layers of metalization [26, 27]. In other words, the designer has only to specify an interconnection pattern to wire a sea of uncommitted transistors and/or gates arrays to form desired circuits. Therefore, the gate-array approach has the fastest turnaround time and lowest costs among the other custom IC designs since all the transistors and gates are made in mass production and it requires only a few mask layers to complete the chip design. In

addition, the regular characteristics of the gate-array approach allow most of the basic elements to be prechecked and the final testing is simplified.

But, like the semi-custom approach, the gate-array approach suffers from slow speed and poor chip area utilization due to long interconnection lines, large gate size and wasted (unused) gates. The large gate size is due to the requirement of more contacts on each gate for flexible I/O connections.

As a whole, the three approaches to custom IC design described in this section present tradeoffs among design complexity, design time, circuit size and performance. The fully custom approach provides a design with the best circuit performance and chip area utilization but suffers from the longest design-time and highest costs. The gate-array approach has fastest turnaround time and lowest costs but produces designs with slower speeds and larger chip areas with the semi-custom approach falling somewhere in between.

2.1.3 Silicon Compilation

Silicon compilation is defined as the process of translating a design description (behavioral or structural) into a geometric description (physical) [28, 29]. The behavioral description of a design is used to describe the relationship of the inputs and outputs in a common form such as timing diagrams or finite state machine descriptions. The structural design can be described in terms of nets (nodes) and components, where components can be decomposed into

primitive nets and devices such as transistors. The physical description of a design is a series of patterns on a set of masks, which may be organized in a hierarchy of geometrical descriptions. Therefore, a silicon compiler translates a high level language description of a circuit into layout information that allows an IC to be fabricated immediately [28, 29].

Current silicon compilers are broadly divided into "front-end" and "back-end" compilation processes [28]. The "front-end" process is the translation of a behavioral or functional description into a more precise intermediate description that is still implementation (technology) independent. The "back-end" process, which is implementation dependent, is the automatic generation of a chip layout from the intermediate description.

There are two major goals in the development of a good silicon compiler. The first is to overcome the drawbacks of fully custom, standard cell and gate-array approaches. That is, the intelligent software routines of the compiler should provide simple descriptions of the circuit behavior and the ability to manage design complexity while producing efficient layouts. Secondly, the same software routines should carry out logic and timing simulation to check overall function and performance. The compiler should check the layout against design rules and then generate the detailed silicon representation. Such a representation is generated in the Caltech Intermediate Form (CIF) [6] or Electronic Design Interchange Format (EDIF) [30] which are the prevalent formats for describing the IC layouts.

Like the standard cell approach, silicon compilation requires the database as well as software routines be kept up to date with IC fabrication technology. With this technique, the entire compilation process is transparent to the designer who has no direct control over the layout process. Therefore, in terms of chip area and performance, layouts designed by the compilation technique still tend to be less efficient than those done by handcrafted methods and in general are 10%-20% larger [31].

2.1.4 Hierarchical Design Methodology

Hierarchical design methods are considered as a means of managing the complexity of the VLSI design problem. Three systematic approaches to hierarchical systems design have been discussed in the literature [32]. They are the top-down and bottom-up approaches and schemes which combine the two.

Currently, top-down system design through silicon compilation is capable of producing a wide variety of VLSI circuits that are of market quality [28, 33]. This approach allows designers to work above the circuit element and layout levels. Once the behavioral and high-level architectural description of an integrated system are specified, a designer will be provided with physical layout, simulation, verification and documentation by the compiler. However, this generally implies bounds on the density of a structure because the designer has no direct control over the details below the circuit level.

In contrast to the top-down approach, bottom-up system design starts from the basic level of circuit layout [11]. Every circuit element (e.g. transistors or gates) within a circuit block or module is carefully designed, usually by trial-and-error, and placed so that the most efficient area-speed layout is obtained. However, as the number of circuit elements increases, the circuit complexity and thus the design time increase rapidly.

In practice a combination of the top-down and bottom-up approach is generally used. At one end, the complex system can be specified by high-level descriptions and by functional specifications from which lower level descriptions and ultimately parts of the layout will be compiled in an automatic or semiautomatic manner. At the other end, layout aided by graph theoretic techniques alleviates the problems of wasted chip area and long channel routing by closely clustering or tessellating modules. By combining both approaches, this design methodology provides an acceptably rapid, accurate development of VLSI systems with a reasonably small sacrifice of silicon and performance.

2.2 VLSI Technologies

Both bipolar junction transistor (BJT) and metal-oxide-semiconductor (MOS) IC technologies are used for the fabrication of VLSI circuits. Broadly, BJT technology is preferred for high speed digital applications and high gain, high bandwidth, low offset and low noise analog designs. MOS technology is favored for low power, high

complexity and high density digital designs. It is also most useful for high input impedance requirements and low power analog applications.

MOS technology is preferred over BJT technology for high performance, dynamic circuit implementation because the MOS transistor incorporates a near perfect input capacitor which permits input signals to be stored [26]. Inherently, MOS transistor structure is much simpler and smaller than the BJT structure. More essential, the scaling down of the device's surface dimension results in better time performance for MOS, but no corresponding change in BJT circuits. This is because the transit time in a MOS transistor is determined by the channel surface width whereas in a BJT it is proportional to the thickness of the base region instead of the base surface [6]. As a result, MOS technology has many advantages over BJT technology for digital VLSI applications requiring low power dissipation and high circuit packing density.

The two most widely used MOS technologies are n-channel MOS technology (NMOS) and complementary MOS technology (CMOS). Currently, CMOS tends to replace NMOS in high-density memory design due to the low standby power, high noise immunity, reliability and superior temperature characteristics of CMOS [34]. However, CMOS chips are generally 10%-20% larger than functionally identical NMOS chips [35]. Also, CMOS structures require one or two more mask layouts, which increase production costs and time. They also are more susceptible to encounter latchup problems when scaling down [36]. The latchup is caused by the extraneous currents in the gate channel which forward-bias the junction between the p-channel and n-channel devices. Pass transistor networks using NMOS have been found to have simplest topology, highest density

and best performance among MOS random logic designs [6, 37, 38].

2.3 VLSI Systolic Array Structures

A promising outgrowth of VLSI capabilities is the prospect of designing high-speed dedicated computing structures using both parallel and pipelined processing concepts. These structures can be implemented on a single chip or a matched chip set. A number of VLSI systolic array algorithms and architectures have been proposed which show great promise in engineering problem applications such as signal and image processing, matrix arithmetic and graph theoretic algorithms [3-8].

A systolic array processor is a dedicated (non-programmable) computing subsystem having a one- or two-dimensional configuration of repeated processing elements (PE's) arranged in a parallel-pipeline fashion. The processor synchronously "pumps" data between levels of PE's performing part of an overall computation at each time step, thus the name "systolic". Latches, which are simple dynamic storage elements, are placed amid rows of PE's to provide synchronization and to assure that a regular flow of data is maintained.

The majority PE commonly found inside many systolic array structures is the multiply-add cell (MAC), which is also known as the inner-product step processor. Two types of ideal geometries for MAC are shown in Figure 2.1 [3, 6]. Both types of MAC's perform the operation C=AB+C and transfer A=A and B=B. However, the type-1 MAC, shown on the left, is used for matrix-vector multiplication and backward substitution of triangular matrix system, whereas a type-2 MAC is used for matrix

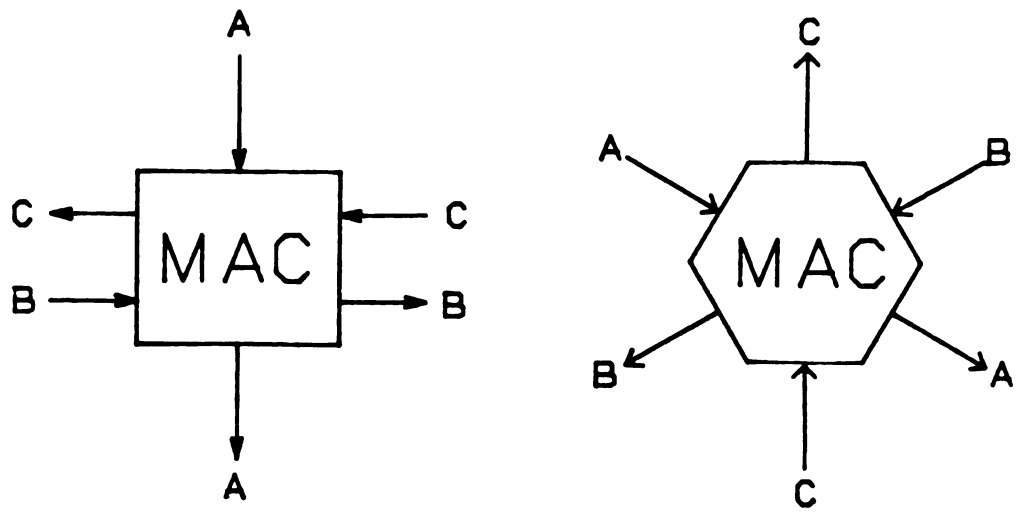

Figure 2.1 Two types of MAC's.

inversion, multiplication, triangulation and L-U decomposition. Thus, one-dimensional (1-D) or two-dimensional (2-D) systolic array structures can be configurated by using these MAC's depending on the application.

For example, consider a 2-D systolic array structure used for band matrix multiplication [3, 6]. The multiplication of two NxN matrices $\underline{A}$ and $\underline{B}$ with bandwidth of $w_1=p_1+q_1-1$ and $w_2=p_2+q_2-1$, respectively, can be represented by

$$\begin{array}{c} \overset{p_1}{|\leftarrow\!-\!\rightarrow|} \\ \overset{\top}{q_1}\overset{\perp}{} \end{array} \begin{pmatrix} a_{11} & a_{12} & & 0 \\ a_{21} & a_{22} & a_{23} & \\ a_{31} & a_{32} & a_{33} & a_{34} \\ & a_{42} & & \\ & & \ddots & \\ 0 & & & \ddots \end{pmatrix} \times \begin{array}{c} \overset{p_2}{|\leftarrow\!-\!-\!-\!\rightarrow|} \\ \overset{\top}{q_2}\overset{\perp}{} \end{array} \begin{pmatrix} b_{11} & b_{12} & b_{13} & & 0 \\ b_{21} & b_{22} & b_{23} & b_{24} & \\ & b_{32} & b_{33} & b_{34} & \\ & & b_{43} & & \\ & & & \ddots & \\ 0 & & & & \ddots \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} & 0 \\ c_{21} & c_{22} & c_{23} & c_{24} & \\ c_{31} & c_{32} & c_{33} & c_{34} & \\ c_{41} & c_{42} & & & \\ & & \ddots & & \\ 0 & & & \ddots & \end{pmatrix} \quad (2\text{-}1)$$

In this example, both w_1 and w_2 are equal to 4. Figure 2.2 shows the corresponding mesh-connected network of type-2 MAC's with the appropriate intermediate latches depicted by rectangular blocks. Elements of matrices $\underline{A}$ and $\underline{B}$ are pumped into the array network synchronously until all elements of $\underline{C}$ are obtained. In general, the numbers of MAC's, N_{MAC}, and time steps, T_{MAC}, required for the multiplication of any two NxN matrices of bandwidth w_1 and w_2 are expressed as follows [3, 6],

$$N_{MAC} = w_1 w_2, \tag{2-2}$$

and

$$T_{MAC} = 3N + \min\{w_1, w_2\}. \tag{2-3}$$

2.4 Design for Testability

The testability of a digital network is directly related to the difficulty of controlling and observing the logical values of internal nodes from external circuit ports [39, 40]. Thus, controllability and observability are the key concepts in the design and implementation of testability. In addition, accessibility, which is the measure of the ease of controllability and observability, is also important in testing complex VLSI circuits. The two basic approaches prevalent today in VLSI design for testability are the ad hoc and structured approaches [39].

The ad hoc approaches use techniques which can be applied to a given product, but are not directed at solving the general problem. Examples of these techniques, some of which evolved from MSI and LSI techniques, are partitioning, test points and signature analysis [39].

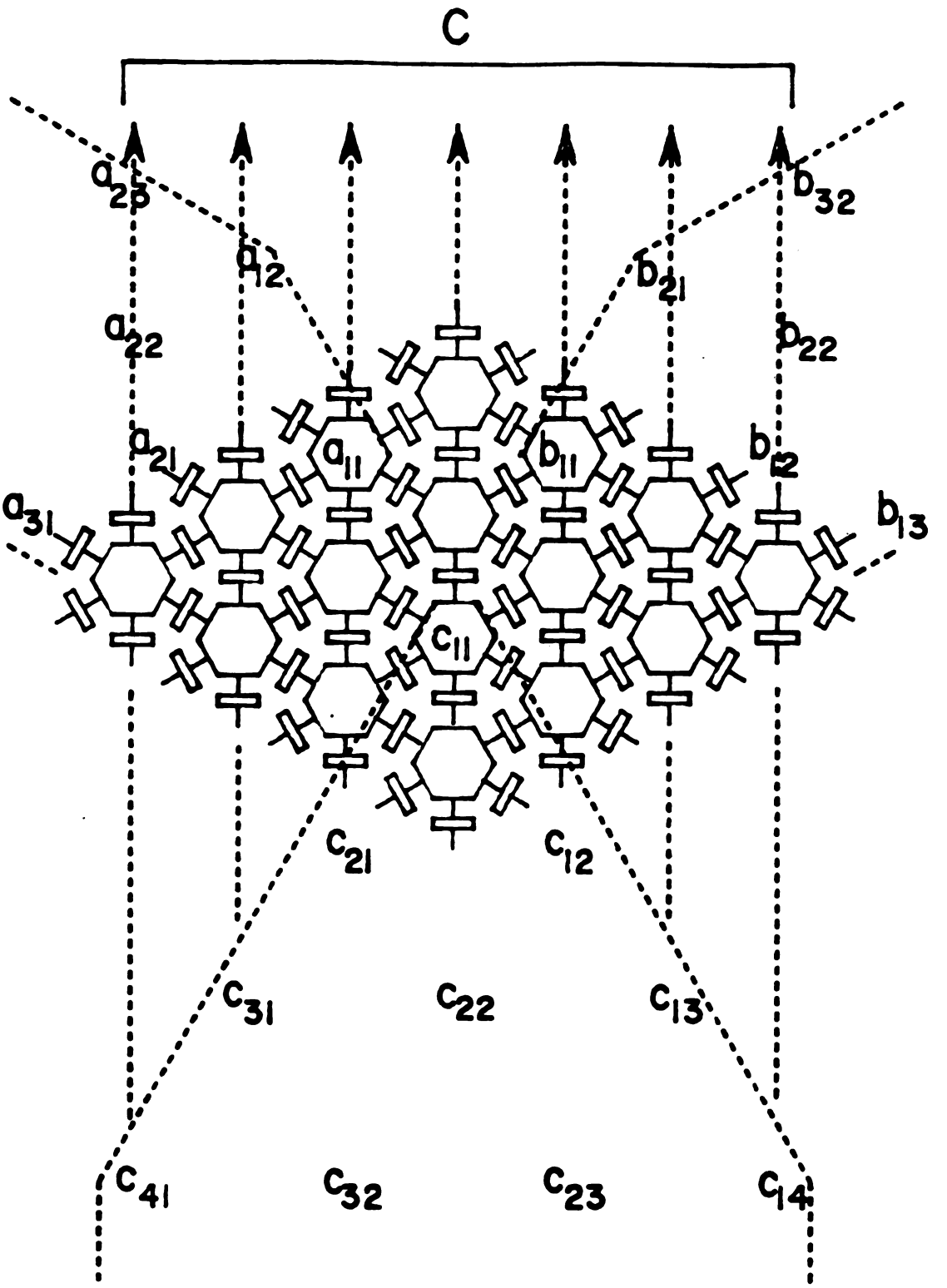

Figure 2.2 Mesh-connected MAC's for matrix multiplication.

Most structured testing approaches are built upon the concept that if the values in all the latches can be controlled and observed in a straightforward operation, then the test generation and simulation of a sequential network can be reduced to that of doing test generation and fault simulation for a combinational logic network. A control signal can switch the memory elements from their normal mode of operation to a mode that makes them controllable and observable. Techniques that use the structured approach are level-sensitive scan design (LSSD), scan path, scan/set logic, random-access scan, self-testing and built-in tests [39]. Among these, LSSD [13] and scan path [14] techniques have been the most popular approaches to VLSI testing.

In LSSD structures the memory elements or latches are threaded together to form a serial-in/serial-out shift register. The shift register latches (SRL's) are controlled by shift clocks for loading and unloading and by system clocks for latching logic values present at their data inputs. In testing the logic networks, test inputs are applied through primary inputs as well as SRL's through a "scan-in" operation. Outputs of the networks are observable at both primary outputs and SRL's through a "scan-out" operation.

The scan path technique has the same objectives as the LSSD approach. The memory elements used in the scan path approach are raceless D-type flip-flops. The scan-in and scan-out operations are controlled by the master/slave operations of the flip-flops. With proper adjustment to the delays of the flip-flops, race conditions can be avoided [14]. Therefore, only a single system clock is required in this technique.

In general, all designs for testing require additional hardware and I/O pinouts so as to achieve controllability and observability. Therefore, accessibility tends to trade off with hardware size and/or performance.

Design for testability of systolic arrays and other structures having similar regular and pipelined characteristics are relatively straightforward in comparison with other VLSI circuits. The functional and logic tests of these structures are directly related to algorithm implementation. Therefore, once the systolic or other similar algorithm has been correctly translated and implemented into a hardware algorithm, the remaining test required is just the physical testing of the chip. In addition, systolic structures use dynamic latches for pipelining and each pipelined segment is composed of combinational logic. Thus, scan design used in LSSD and scan path techniques can be applied to these structures without a significant increase in hardware or signal delay time.

2.5 Y-Chart Representation of VLSI Design Methodologies

A tripartite representation on a Y-chart has been used to define VLSI design methodologies [8, 41]. A Y-chart is a descriptive model using three axes. In general designs, the three axes of the Y-chart are associated with functional or behavioral representation (e.g., a Boolean expression), structural representation (e.g., the functional realization of the Boolean expression), and geometrical representation (i.e., the

physical implementation of a design) [41]. In the design of systolic algorithms, the three axes are related to algorithm representation (specified forms or levels of systolic algorithm), algorithm model (abstraction of the features of the algorithm) and architecture specification (physical description of the systolic array) [8].

Alternative Y-charts will be minimally related to each other if the axes among these Y-charts are not consistent in representation. However, all Y-charts utilize directed or feedback arcs to represent the transformations on the same axis or among the axes. The means for these transformations vary with the design methodologies. The means can be CAD tools, sets of parameters or even a direct mapping or transfer.

Design representations using Y-charts have many advantageous features. First, the three-axes as well as arcs are all user-defined so that the Y-chart can be used to clarify the explanation of a design approach. Secondly, the tradeoffs of the different design methodologies can be initially assessed by comparing the information gleaned from the Y-chart. (Of course, the axes of the compared Y-charts must be consistently defined.) Thirdly, Y-charts can be further subdivided into more detailed Y-charts such that informative details can be additionally depicted. This may be considered as a "hierarchical" Y-chart representation of VLSI designs.

As an example, a modified Y-chart describing the design of several VLSI structures mentioned in this chapter is shown in Figure 2.3 [8, 41]. In this Y-chart, the axis of STRUCTURE REPRESENTATION broadly shows two major types, random and regular, VLSI-based structures. Six typical random and regular structures are shown on this axis but,

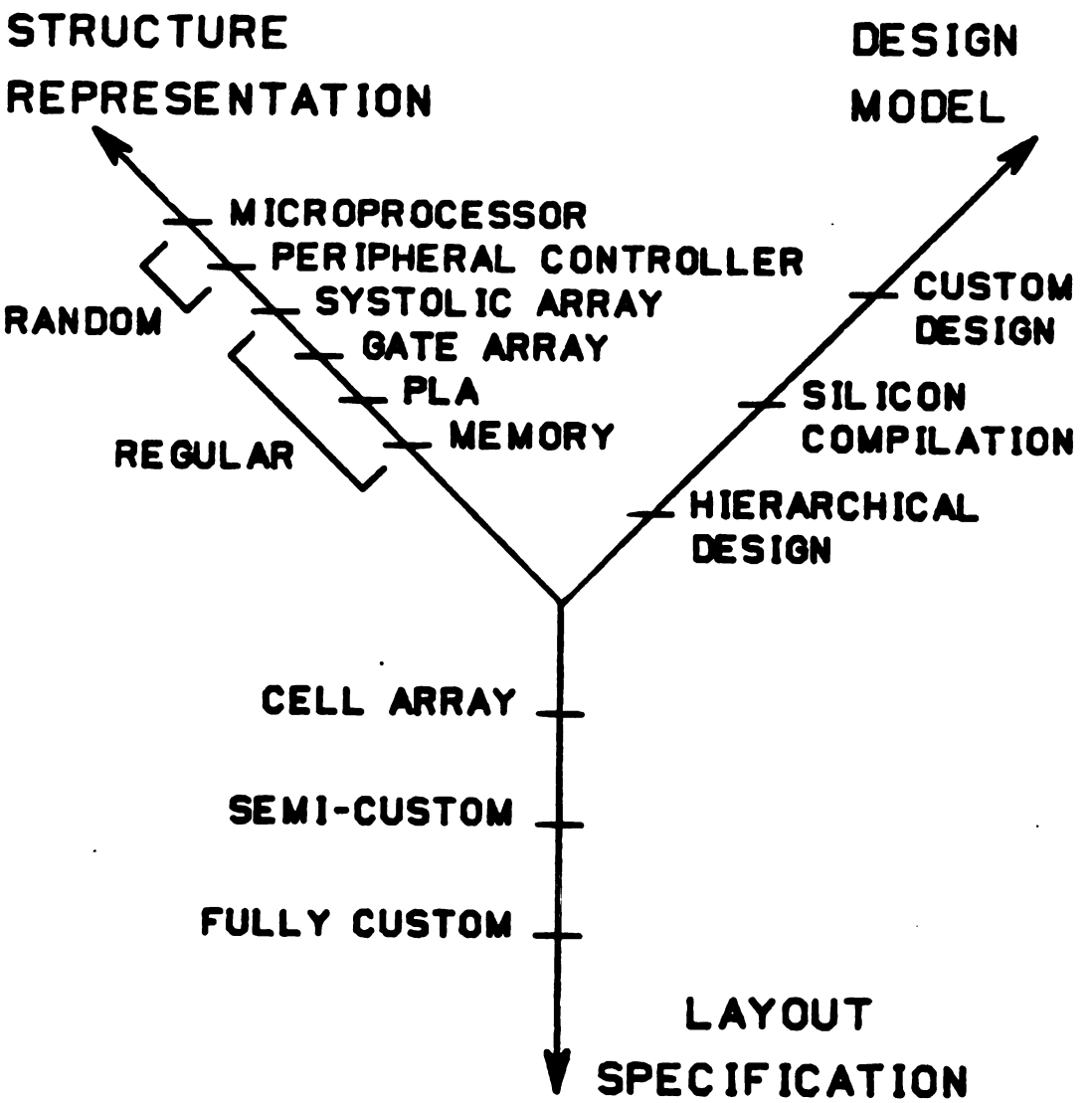

Figure 2.3 A modified Y-chart representation of typical VLSI-based design [8, 41].

obviously, VLSI-based structures are not restricted to these six. This axis can also indicate the degree of either regularity or non-regularity of the structures. Regularity increases toward the origin while non-regularity or randomness increases with the direction of the axis. Note that the scales on this axis are not necessary in equal proportion.

The methodologies of custom design, silicon compilation technique and hierarchical design are depicted in the axis of DESIGN MODEL. The custom design model includes all other traditional design methodologies such as symbolic layout approaches, building-block techniques, etc. [17]. Design techniques must not be constrained to a single design model; therefore, a mixed design model can also be described. For example, silicon compilation techniques can also be used in some of the procedures in the hierarchical design model [29, 33].

The third axis for LAYOUT SPECIFICATION represents IC layouts in order of increasing complexity. The cell array approaches such as the gate-array and logic array, have the least complex layout while the fully custom layout has the highest complexity.

Any directed or self-looped arcs denoting a transformation through certain means such as simulators, compilers, sets of parameters, etc., can be placed on this Y-chart. Finally, a detailed Y-chart can be further developed for a particular structure or design model. For example, a detailed Y-chart can be developed for the description of the design of regular structures using a custom design approach.

CHAPTER III

BOTTOM-UP APPROACH

An overview of the bottom-up approach to the hierarchical design of systolic array structures is shown in Figure 3.1. The layout design starts from a selected design entry-point according to a set of desired parameters such as circuit complexity, size and performance. Design pathways can then traverse among the module levels of transistor, gate, functional device or processing element (PE) as desired. Finally, a "target" structure can be formed by modular tessellation.

Tradeoffs existing among the design hierarchy have yet to be fully assessed and compared numerically. In order to do so, the bottom-up approach presented here involves the layout and simulation of different target systolic array structures designed through different hierarchical design entry-points and paths. The thread of this hierarchy in the bottom-up approach is further shown in Figure 3.2.

The primary target systolic structure is the multiply-add cell (MAC), which is a commonly used and often the majority PE inside systolic array structures. Examples of systolic structures in which the MAC is the major component include arrays for matrix triangulation, multiplication and inversion. As an example in hierarchical design of layout, a MAC module considered at the PE level can be designed starting from transistor, gate or functional device level. Likewise, a full

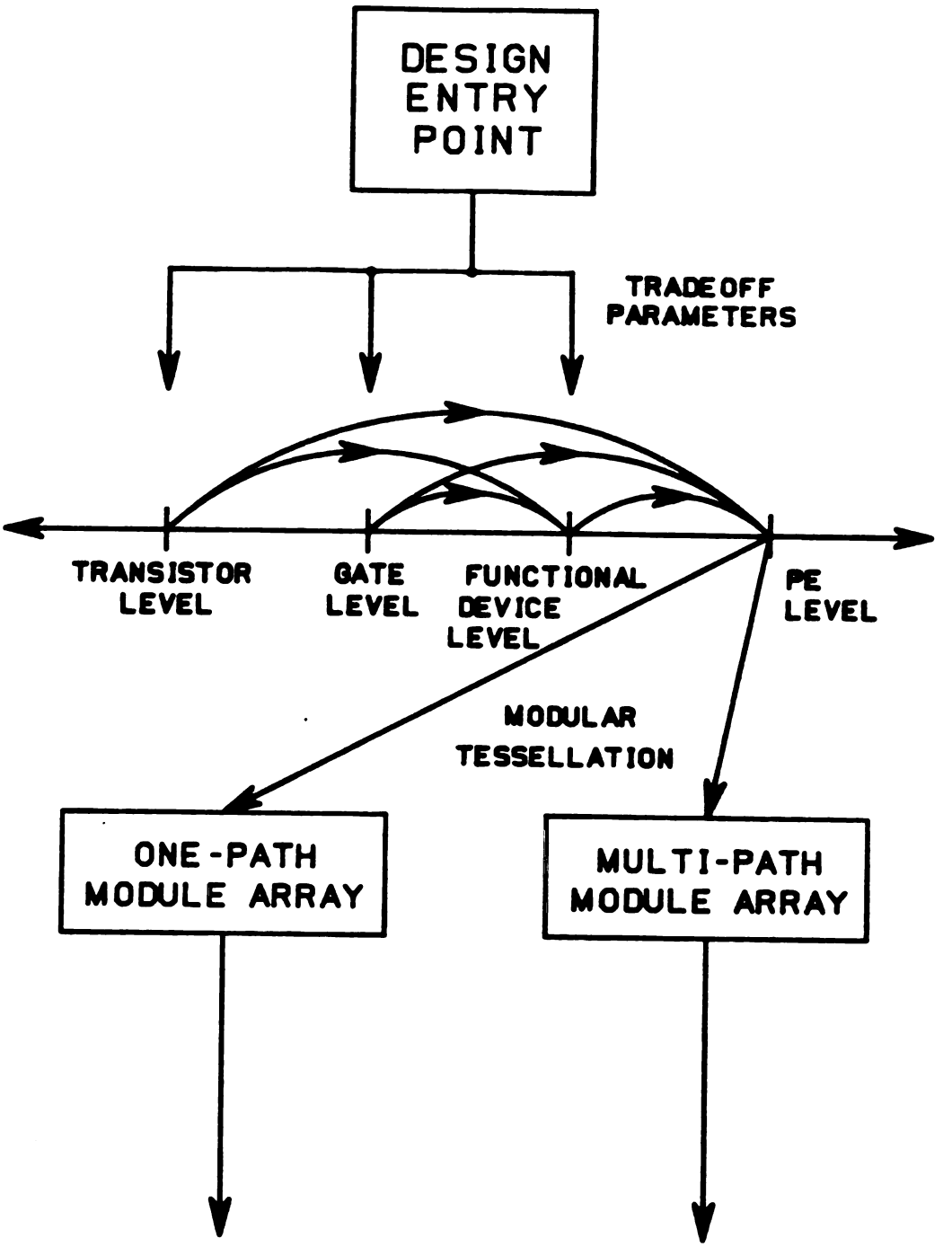

Figure 3.1 An overview of the bottom-up approach.

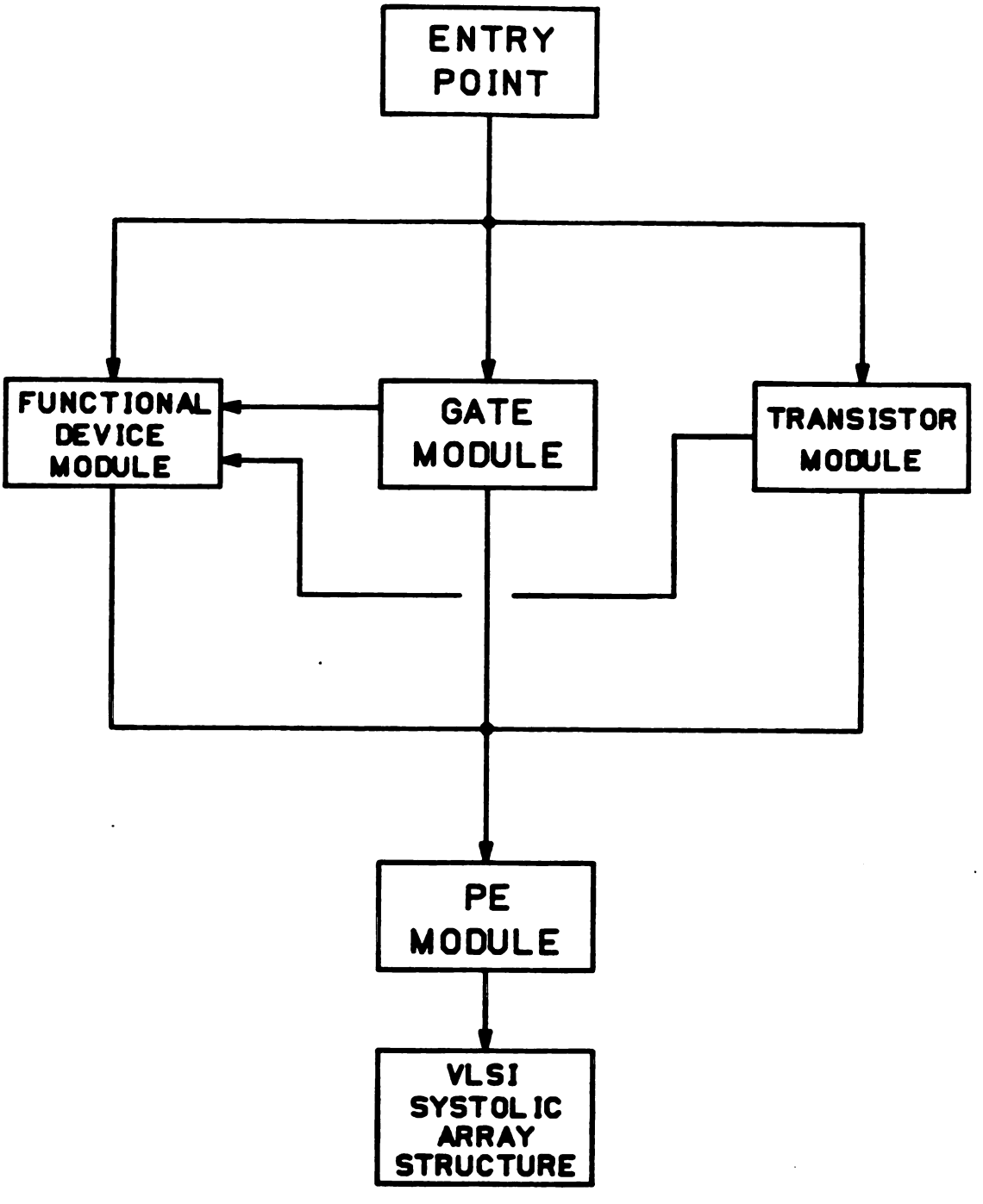

Figure 3.2 Design hierarchy in the bottom-up approach.

adder (FA) module at the functional device level can be built of either gate modules or transistor modules.

As an initial approach to study the tradeoff among the various design levels, modules at several levels is designed and simulated using standard NMOS design rules. The choice of these design rules is due mainly to the widespread use and knowledge base of NMOS technology [6, 27, 37]. Other factors making NMOS attractive include high circuitry density, richness of available circuit functions and simple topological properties of circuit layout and simulation.

3.1 Transistor and Gate Level Design

The boundary of a circuit module at any level is simply defined as the rectangle which contains the complete planar layout of the functional circuit. Therefore, the area of a circuit module is simply the product of the length and width of the boundary of the module. Figure 3.3 illustrates the cell boundary of a depletion mode (pull-up) and an enhancement mode (pull-down) transistor modules. Figure 3.4 shows the four typical gate-level modules of inverter, 2-input NOR, 2-input NAND and XNOR. Both of these figures represent standard NMOS design [6, 27, 37].

In addition, due to the assumption of module tessellation, modification of predefined lower level modules is not considered when they are used to form higher level modules or circuits. Otherwise, there will be an increase in design complexity and design-time and the design parameters cannot be generalized. As an example, consider a

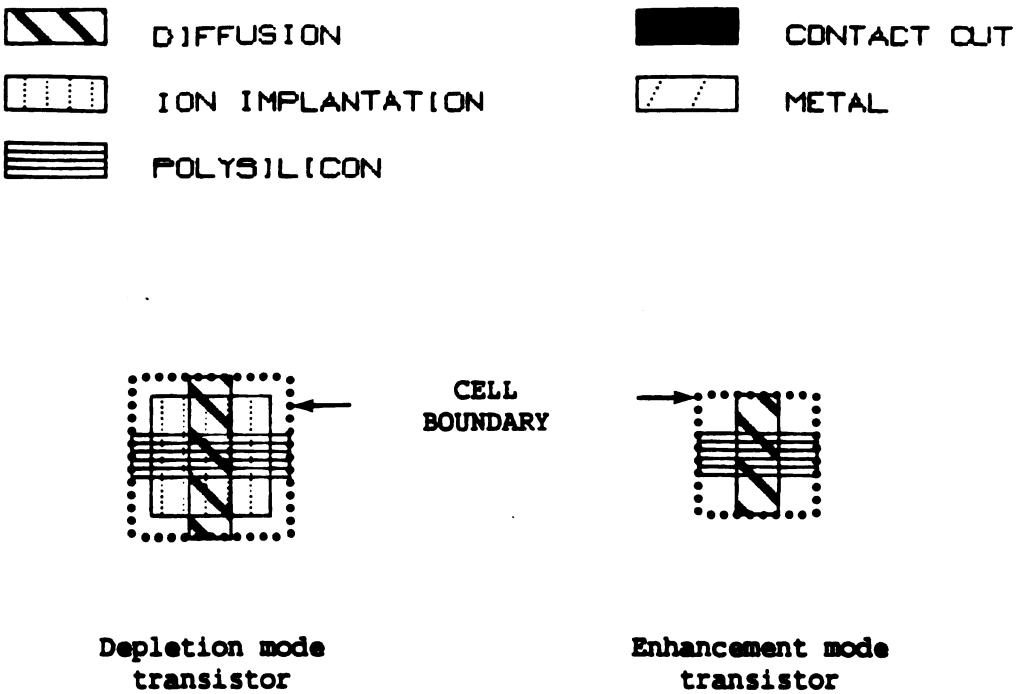

Figure 3.3 Cell boundary of two transistor modules.

gate-level FA which are tessellated by using a 2-input NAND gate as shown in Figure 3.4. During the tessellation, the pull-up and pull-down transistors inside the NAND gate can be shifted, rotated or/and reflected provided that all the transistors stay within the predefined gate boundary. Otherwise, the FA is considered to be a transistor-level FA ("X.FA") instead of a pure gate-level FA ("G.FA"). This is because the NAND gate is actually broken down into transistor modules which implies an increase in the number of modules requiring placement and lines requiring routing.

Transistor modules and gate level modules are defined and then "programmed", via a compilation technique, into a VLSI design oriented CAD system in order to establish a flexible cell library. Then, the

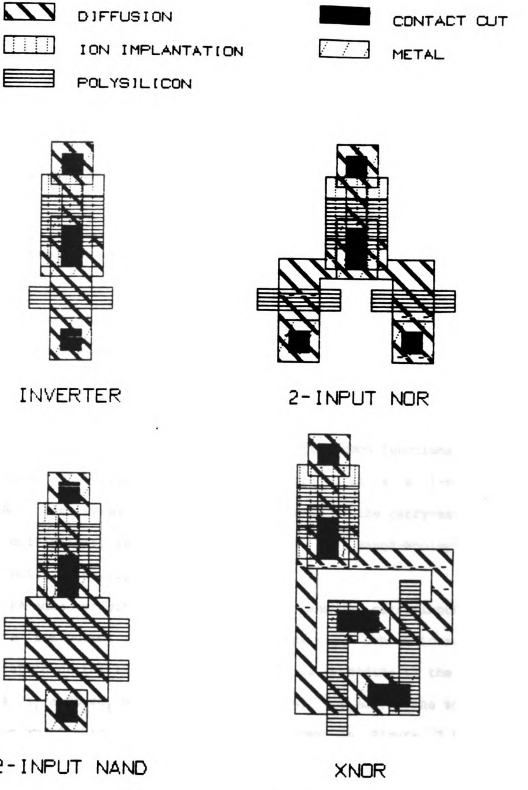

Figure 3.4 Typical gate-level modules.

layout of any transistor (pull-up, pull-down) or gate (NOR, NAND, XNOR, etc.) module can be generated automatically once the required specifications of the module, such as size, pull-up to pull-down transistor ratio or I/O and geometrical orientation, have been defined. Therefore, high-level modules can be designed by tessellating the predefined lower-level transistor modules or gate modules.

3.2 Functional Device Level Design

Common circuits that can be considered as functional device modules include the half adder, full adder, comparator, complementer and counter cells. In general, the functional device module is composed of a certain number of transistors or gates and is the majority device module inside the higher level PE module. The most common functional device module found inside regular VLSI computing structures is a 1-bit full adder (FA). High speed arithmetic circuits such as the carry-save adder and fast multipliers, such as the Braun array and Baugh-Wooley array, are also built primarily of FA's [42, 43].

The FA adds two binary digits a_i and b_i, and a carry-input c_i to produce a sum-output s_i and a carry-output c_{i+1}. Conventionally, the implementation of the 1-bit binary FA module is according to the Boolean equations $s_i=a_i \oplus b_i \oplus c_i$ and $c_{i+1}=(a_i \wedge b_i) \vee (b_i \wedge c_i) \vee (a_i \wedge c_i)$. The schematic logic diagram of this implementation is shown in Figure 3.5. The reasons for choosing this implementation are summarized as follows:

1. It contains various discrete and conventional gates such that transistor and gate modules can be easily distinguished.

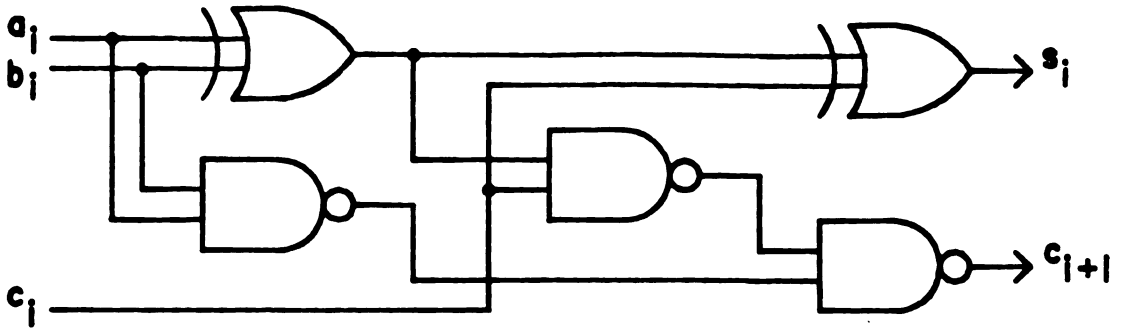

Figure 3.5 Logic diagram for the implementation of FA.

2. Its simple interconnection and small number of fan-ins and fan-outs per gate allow flexible layout design, predictable circuit behavior, straightforward circuit simulation, and speed calculation.

3. It is not particularly biased against or toward any IC technology. (Note that NMOS is biased to NOR while CMOS and TTL (Transistor-Transistor Logic) are biased to NAND [26, 27].)

In designing an FA as a target structure, there are three distinct design paths as noted in Figure 3.2. Arranged in decreasing order of design complexity and circuit performance, they are "X.FA", "G.FA" and "FA", where 'X', 'G', and 'FA' represent the transistor, gate and functional device module level, respectively, and '.' denotes the design path between two levels. The terms of "functional device module level" and "FA module level" will be used interchangeably since the majority functional device is the FA in this research. Likewise, "PE level" and "MAC level" will be interchangeable because the majority PE is the MAC.

Therefore, "X.FA" represents a transistor-level FA module which is designed via the pathway of "entry point", "transistor module" and

"functional device module" as shown in Figure 3.2. Using only the transistor modules, this FA module is carefully crafted according to the logic diagram shown in Figure 3.5. In order to utilize the standard NMOS gates as shown in Figure 3.4, the XOR gate in Figure 3.5 is realized by an XNOR concatenated with an inverter as shown in Figure 3.6.

Since the layout positions of global I/O can affect overall module size and speed, different layouts of the FA with different I/O orientations are generated and compared. The I/O orientation is determined by the direction of data flow such that the module is best fit for tessellation. Finally, a preferred "X.FA", which is best in performance and chip area utilization, will be used as the building blocks for higher level modules.

Likewise, "G.FA" is a gate-level FA module designed through the path of "entry point", "gate module" and "functional device module" again as illustrated in Figure 3.2. This module is designed by placing gate modules only and routing the interconnection lines for I/O, power

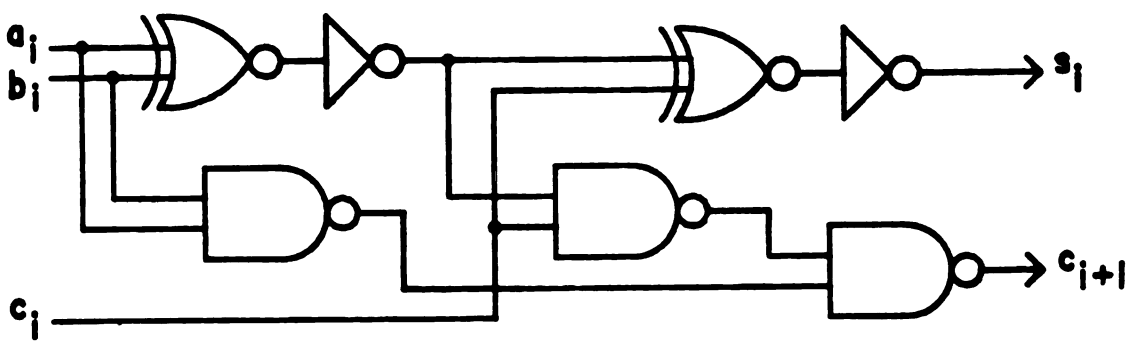

Figure 3.6 Alternate version of Figure 3.5.

and ground. In the same manner as designing the "X.FA", layouts of "G.FA" with different global I/O positions are obtained and compared. Thus, "G.FA" modules heuristically optimal in both speed and area are obtained.

"FA", an functional device-level FA module, requires the least design time, thus representing the FA module with lowest design complexity. This module, mimicking a standard FA cell stored in the cell library, can be extracted and directly placed for tessellation. The actual design of this FA module uses the idea of the gate-array approach as discussed in Section 2.1.2. That is, the gates required to form an FA are simply packed as close as possible together in an array form and then I/O and power/GND lines are connected directly. Therefore, this design is completed in the shortest time compared to the other target FA's.

Once the three major types of FA's ("X.FA", "G.FA" and "FA") designed according to the above criteria have been obtained, they become part of the database of the CAD system as FA cells. Although the Computervision CADD 2/VLSI design system is used as a design aid in this work, any other basic CAD system or even paper and pencil can be used instead. This is because the final results and comparison will not be expressed in terms of absolute design time such as CPU seconds or man-hours but rather in relative ratios in terms of the number of modules or density of the modules. What's more, none of the basic layout designs utilized any placement or routing algorithms because of the assumed properties of regularity and local connectivity. As a result, these designs are machine independent.

3.3 PE Level Design

The primary target structure in the design hierarchy is the multiply-add cell (MAC), which is the majority PE found in many systolic structures. The implementation of a MAC is based on the Baugh-Wooley array multiplier which allows fast and direct two's complement array multiplication [9, 42]. In addition, the Baugh-Wooley array is constructed entirely from the conventional 1-bit FA shown in Figure 3.5. For illustration, Figure 3.7 shows a 5-bit by 5-bit MAC which is merely a Baugh-Wooley array multiplier (enclosed by dotted lines) with an extra row of FA's at its bottom edge.

The design complexity and final performance of a MAC are again quite dependent on the design entry-point and pathway. As shown in Figure 3.2, five distinct MAC's of different complexity corresponding to five design paths of mixed levels are possible. They are the "X.MAC", "X.FA.MAC", "G.MAC", "G.FA.MAC" and "FA.MAC". For example, "X.MAC" is a transistor-level MAC which is laid out using only transistor modules and "FA.MAC" is tessellated only by "FA" modules. Obviously, the design of a "FA.MAC" is less complex than that of "X.MAC" simply because the former is built using larger blocks without any concern given to details inside each block while the latter one is crafted from basic transistor modules. However, the chip area and propagation time of the "FA.MAC" are expected to be greater than that of "X.MAC" due to the relatively smaller interstitial space and shorter signal path inside the "X.MAC".

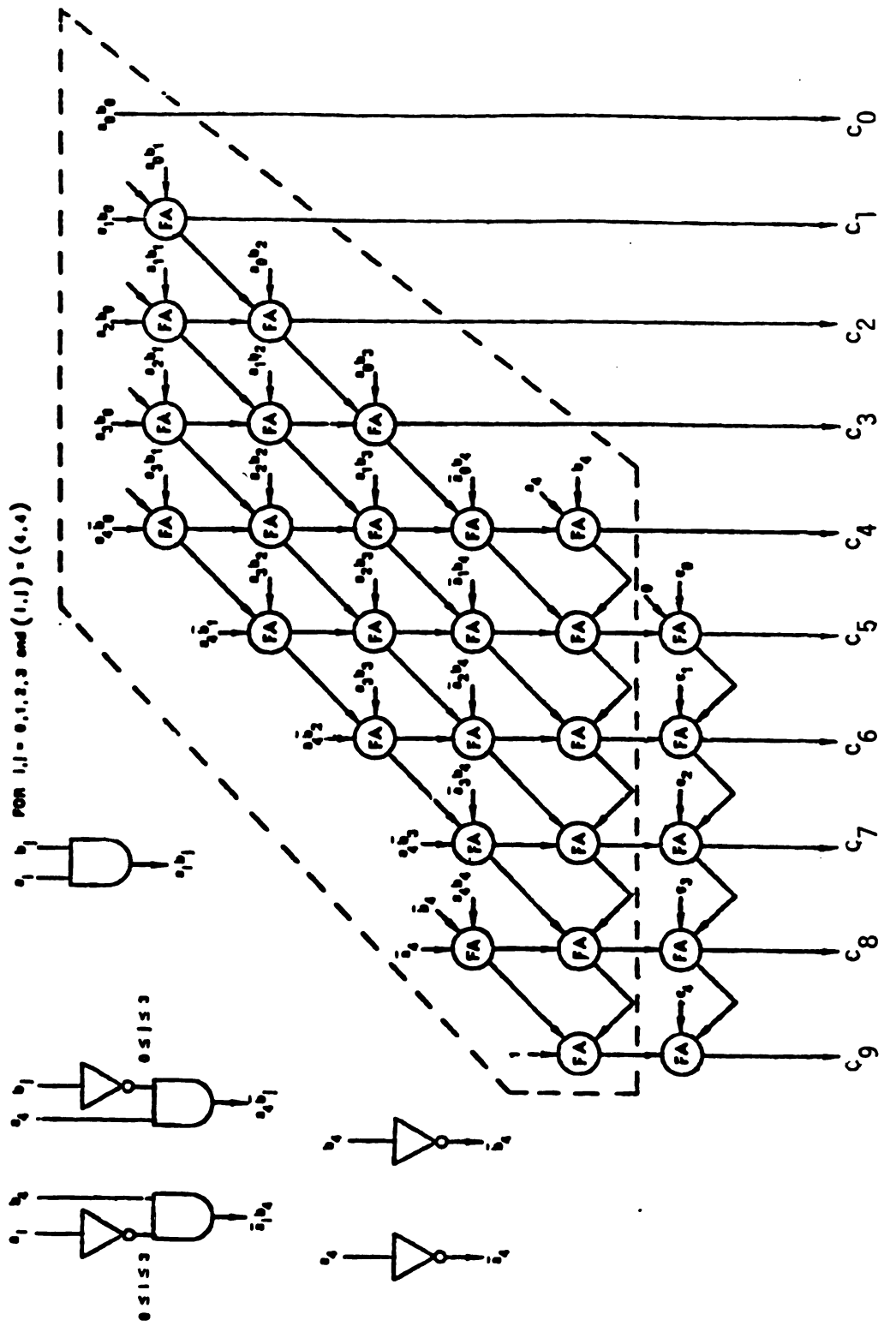

Figure 3.7 Logic diagram of a 5-by-5 Baugh-Wooley based MAC [9, 42].

Among the five different MAC's, "X.MAC", "G.MAC" and "FA.MAC" represent the transistor-level MAC, gate-level MAC and FA-level MAC, respectively. However, closer inspection of Figure 3.2 reveals two points-of-view with respect to the design entry-point and target-point as observed for "X.FA.MAC" and "G.FA.MAC". From the standpoint of design entry-point, or a straight bottom-up approach along the arrows shown in the figure, "X.FA.MAC" and "G.FA.MAC" are considered to be transistor-level MAC and gate-level MAC, respectively. However, from the viewpoint of design target-point (PE module), or a top-down approach reversely following the arrows shown in the figure, both "X.FA.MAC" and "G.FA.MAC" can be alternately considered as FA-level MAC's. In addition to these two points-of-view, both "X.FA.MAC" and "G.FA.MAC" can be simply treated as mixed-level MAC's. Therefore, the defined entry-level of "X.FA.MAC" and "G.FA.MAC" depends on the point-of-view of the observer.

As a whole, five independent MAC's corresponding to five distinct design paths of different complexity are constructed. They are stored in the CAD system and will be used to form the final target VLSI systolic array structures whose majority PE is an MAC.

3.4 Systolic Arrays for Matrix Multiplication

A VLSI systolic array structure having the MAC as its majority PE can be designed directly by tessellating the required MAC's and other minority modules. As a testbed for benchmarking, the matrix-matrix

multiplier, shown in Figure 2.2 in Section 2.3, is considered. By tessellating arrays of any library MAC designed from the various combinations of module levels and the external latches, a matrix-matrix multiplier of any matrix dimension and word size can be constructed. Of course, it has been assumed that there is no limitation on single chip size. If such limitation exists, a top-down approach such as algorithm decomposition, must be considered. The top-down approach will be discussed in Chapter 5.

The rhombic shape of the overall connected network illustrated in Figure 2.2 is impractical for actual layout. Therefore, the layout of the network is "sheared" to a rectangular shape and the hex-shaped MAC's are changed to rectangular MAC's for better chip area utilization. This modified network is shown in Figure 3.8. All the small rectangular blocks shown in this figure are arrays of identical latches for data synchronization. The number of latches in each block is equal to the word size (the number of bits per word). These latches, however, are comparatively much smaller than the MAC's and thus have an insignificant effect on overall area and speed. The design of these latches will be presented in next section. Finally, five different designs of a VLSI matrix-matrix array multiplier are obtained with respect to the three distinct entry-points and five different design paths as shown in Figure 3.2. Specifically, they are "X.MAC.VLSI", "X.FA.MAC.VLSI", "G.MAC.VLSI", "G.FA.MAC.VLSI" and "FA.MAC.VLSI".

With all the target structures discussed in this chapter, the tradeoffs among chip area, circuit performance and design complexity of the proposed design hierarchy can be explored in next chapter. In

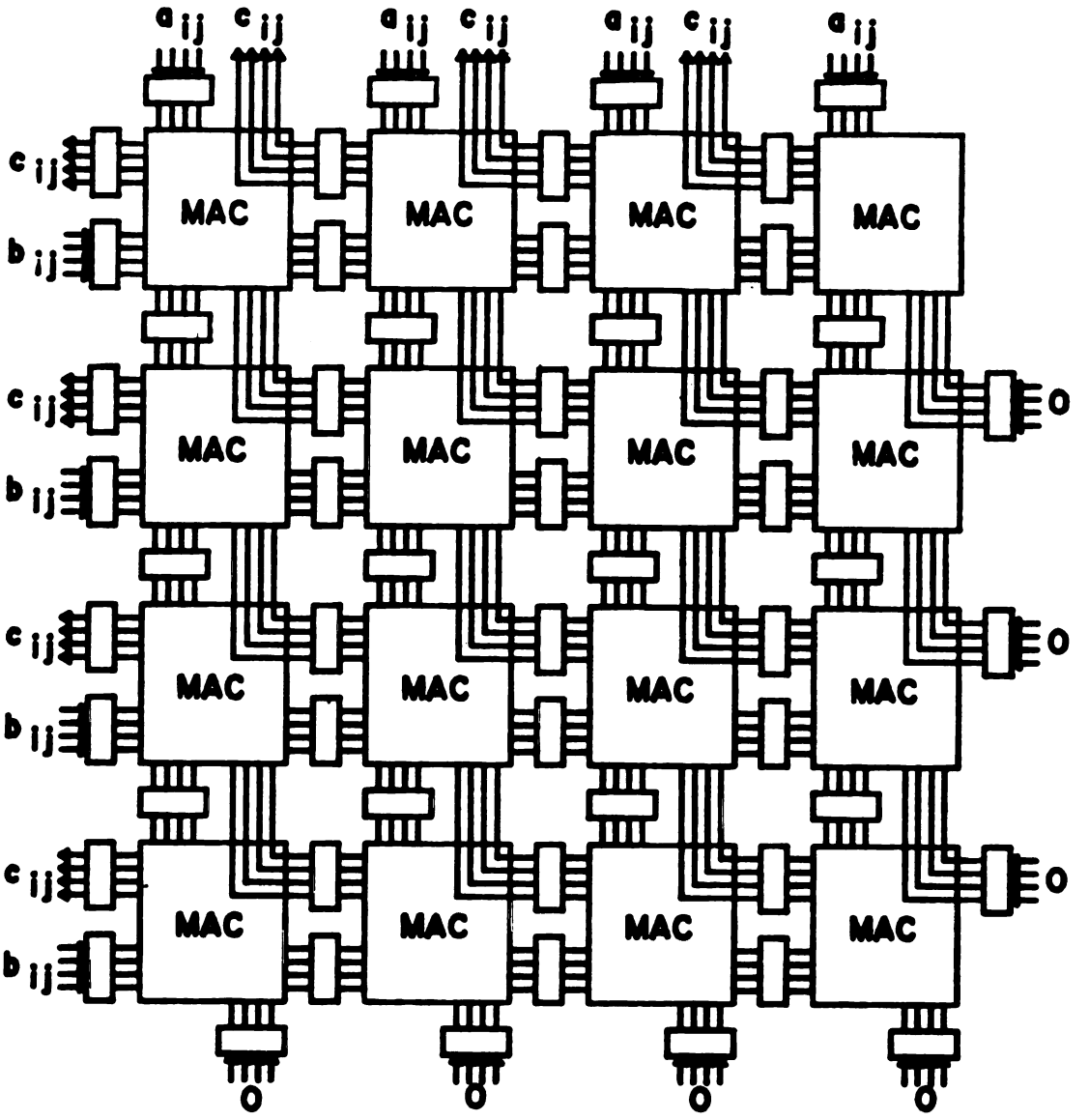

Figure 3.8 Array network for $\underline{C}=\underline{A}\times\underline{B}$ shown in Figure 2.2.

summary, utilizing the bottom-up approach, the target structures designed previously are 1-bit full adder (FA), multiply-add cell (MAC) and VLSI matrix-matrix multiplier (VLSI) as shown in Figure 3.9.

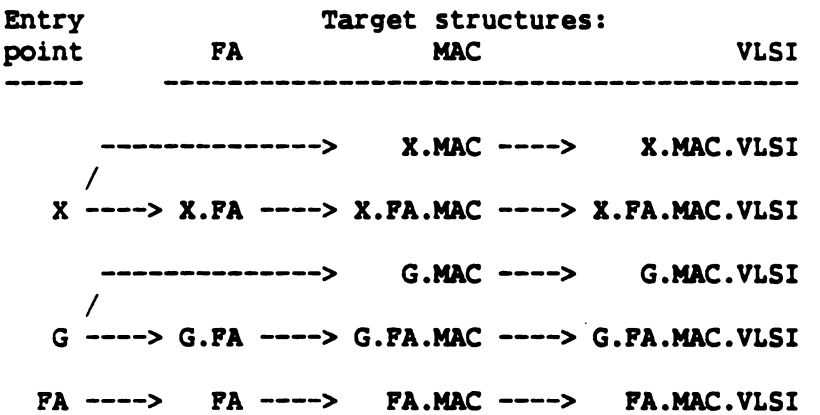

Figure 3.9 Design entry-point and pathway of various target structures.

3.5 Latch Design for Systolic Array Structures

Dynamic registers or latches are required for high speed, synchronous processor chip designs, especially using VLSI [6, 13, 39]. Therefore, dynamic latches using a feedback path as shown in Figure 3.10 are utilized in the design of the systolic array structure presented in this work [6]. These latches are designed as shown in the figure so that they can be directly tessellated either vertically or horizontally amid the PE's. An individual latch, enclosed by the dotted lines in the figure, uses a two-phase nonoverlapping clock to load (ϕ_1) and refresh (ϕ_2) the input data. As mentioned in the previous section, the number

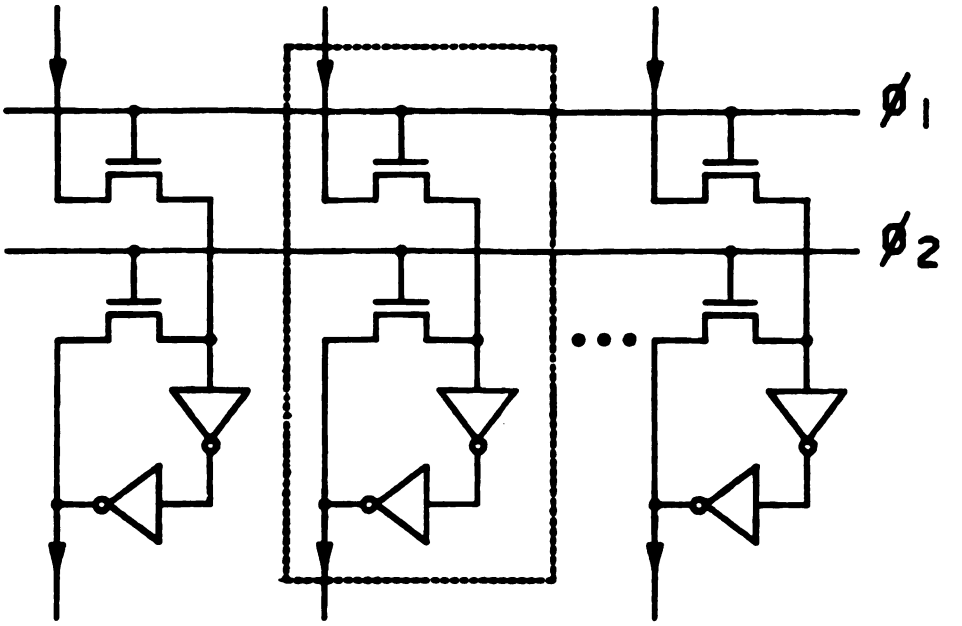

Figure 3.10 Dynamic latches.

of individual latches inside each row or column of an array is dependent on the word size.

Scan design for testability of systolic arrays and other structures having similar regular and pipelined characteristics uses special latches to facilitate the scan-in and scan-out operation. Therefore, the latches shown in Figure 3.10 are modified in order to insure a proper scanning operation. These modified latches for scan design are shown in Figure 3.11. In this figure, the signals of ϕ_1 and ϕ_1' are exclusive to each another during normal and scanning operations. During a scan-in or scan-out operation, ϕ_1' and ϕ_2 are used to load the testing vectors through the line marked S-IN or unload the results through S-OUT while ϕ_1 is disabled. In normal operation, ϕ_1 and ϕ_2 provide the

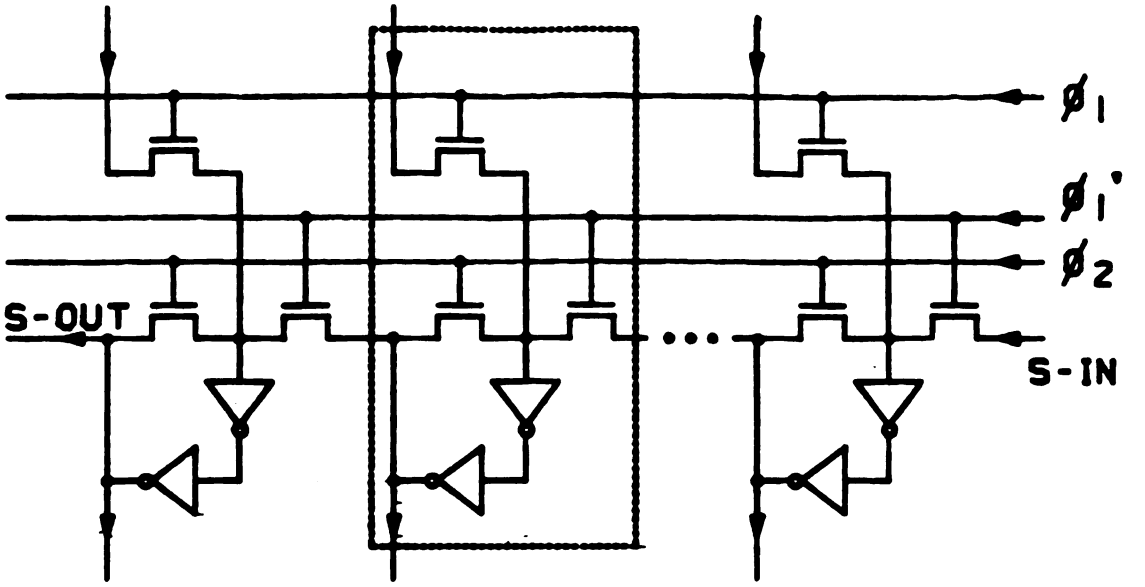

Figure 3.11 Dynamic latches with scan design.

synchronism for the data pumping through the computing arrays while ϕ_1' is off.

The obvious difference between the latches with and without scan design is the addition of an extra signal line, ϕ_1', and an extra pass transistor per latch. Thus, the increase in overall chip area and propagation time due to the additional scanning function is insignificant. What's more, the effect on the design complexity of a latch is so small that the overall increase in the complexity of the systolic arrays is negligible.

CHAPTER IV

TRADEOFF PARAMETERS EVALUATION AND JUSTIFICATION

The main purpose of the bottom-up approach in this work is to study the tradeoff of design complexity versus chip area and propagation time among the various design paths. This chapter first presents the measures for chip area and propagation time and design complexity. Then, the tradeoffs of the design complexity versus area-time based on a generalized layout model are developed.

The design complexity can be expressed by actual design time in terms of CPU seconds or man-hours [44, 45]. Unfortunately, these measures of design time are undesirable due to their dependence on the CAD or CAE system or the designer's expertise. Alternative approaches to parameterization of the design complexity will be discussed and justified in this chapter.

The design of input/output (I/O) circuits for the overall systolic array structure has been neglected in the bottom-up approach by assuming the area of I/O circuits is small and potential I/O bottlenecks are avoided. However, it is noted that if certain conditions exist, such as the domination of I/O time over the pipeline segment time of PE, an I/O bottleneck may be encountered which will eventually degrade the overall performance of the circuit [5, 9]. This I/O problem will be further addressed in the top-down design approach presented in Chapter 5.

4.1 Chip Area and Propagation Time Computation

Since the layout of any higher level module is done solely by tessellation of lower level modules which include communication lines, module areas are determined simply by the sum of the areas of all required lower level modules. In addition, the layout of each module uses standard NMOS design rules as per the technique of Mead and Conway [6]. Thus, the total chip area of a structure is the product of the width and length measured from the final layout of the structure in terms of the minimum lithographic linewidth, λ.

The fundamental limit on the switching speed of a MOSFET (MOS Field-Effect Transistor) is the transit time of a carrier between source and drain. In practical circuits, however, the switching speed is limited by the capacitance charge and discharge times. The capacitance of main concern is the effective load capacitance of an active gate, C_L, which is due to the capacitance of the active gate and the input capacitance of the next gate(s), plus intercommunication lines, parasitic, fringing and Miller-effect capacitance [6, 12, 26].

As an example, Figure 4.1 shows an NMOS N_i-input NOR gate driving N_o identical outputs. By assuming one input of this gate is switched at a given time, the total effective load capacitance is expressed as

$$C_L = C_{stray} + (N_i+1)C_{GDe} + N_i C_{DSe} + C_{GDd} + C_{DSd} + N_o(C_{GSe} + 2C_{GDe}), \quad (4\text{-}1)$$

where G, D, S, e and d refer to gate, drain, source, enhancement mode

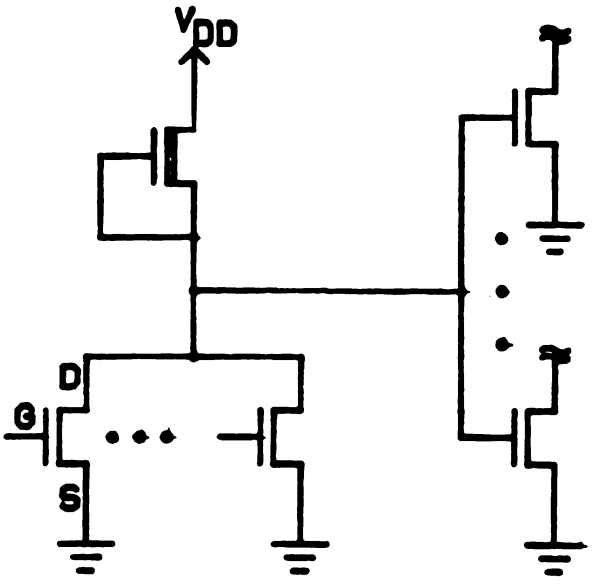

Figure 4.1 An N_i-input NMOS NOR gate driving N_o identical outputs [26].

device and depletion mode device, respectively [26]. The stray capacitance, C_{stray}, is due to the communication lines and fringing capacitance. C_{GDe} and C_{DSe} are the gate-drain and drain-source capacitance of the enhancement mode (pull-down) transistor of the NOR gate. C_{GDd} and C_{DSd} are the gate-drain and drain-source capacitance of the depletion mode (pull-up) transistor. The last term of the equation refers to the total capacitance of the pull-down transistors of N_o fan-out gates. Note that both C_{GDe} terms include a factor of 2 for the Miller effect. This is because the gate-drain capacitance is charged in one direction for one polarity of input and in the opposite direction for the opposite polarity input [6, 26].

C_{stray} is the smallest, per unit area, among all the C terms shown in Equation 4-1 [6]. However, the long length of the communication or interconnecting lines among all the gates makes the total effect of C_{stray} significantly large, especially when gate channels shrink down into the sub-micron region in the near future [46]. C_{stray} will dominate over other capacitance in future ultra dense VLSI circuits [6].

In the hierarchical design of VLSI structures, the length of the interconnecting lines and spacing among modules generally varies with different module levels. The average spacing between two modules tends to increase as the design entry-point and pathways move upward along the design hierarchy [6, 27, 37]. Therefore, since the length of the interconnecting lines between two modules is simply proportional to the spacing between the two modules, C_{stray} is greater among high-level modules than among low-level modules.

Once the total effective load capacitance of an active gate has been found, its delay time can be estimated based on the charging or discharging time of this capacitance [6, 12]. The pull-up time (t_{pu}) and pull-down time (t_{pd}), which correspond to the charging (high-going transition) and discharging (low-going transition) time, respectively, are estimated by

$$t_{pu} = C_L(V_H - V_L)/I_{pu} \qquad (4\text{-}2)$$

and

$$t_{pd} = C_L(V_H - V_L)/I_{pd} \qquad (4\text{-}3)$$

where V_H and V_L are the output voltages of the high and low states, respectively [47, 48]. I_{pu} and I_{pd} are the averaged pull-up and pull-down currents, respectively. I_{pu} and I_{pd} are approximated by the expressions as follows [12].

$$I_{pu} = u_n C_{ox} V_{tu}^2 (V_{DD}+V_{tu}/3)/2Z_{pu}V_{DD} \quad (4\text{-}4)$$

and $$I_{pd} = u_n C_{ox} (V_{DD}-V_{td})^2 (2V_{DD}+V_{td})/6Z_{pd}V_{DD} \quad (4\text{-}5)$$

where u_n = the mobility factor for n-type charge carriers,

C_{ox} = the gate oxide capacitance per unit area of the pull-up or pull-down transistor,

V_{DD} = the supply voltage,

V_{tu}, V_{td} = the threshold voltage of the pull-up or pull-down transistor,

Z_{pu}, Z_{pd} = the length-width ratio of the pull-up or pull-down transistor.

The length-width ratio, Z, refers to the ratio of length to width of the planar gate region (channel) of a MOS transistor [6, 27].

Assuming $V_H=V_{DD}$ and $V_L=0$, and substituting I_{pu} and I_{pd} into Equations 4-2 and 4-3, respectively, yields

$$t_{pu} = 2Z_{pu}V_{DD}^2 C_L/[u_n C_{ox} V_{tu}^2 (V_{DD}+V_{tu}/3)], \quad (4\text{-}6)$$

and $$t_{pd} = 6Z_{pd}V_{DD}^2 C_L/[u_n C_{ox} (V_{DD}-V_{td})^2 (2V_{DD}+V_{td})]. \quad (4\text{-}7)$$

As a result, Equations 4-6 and 4-7 estimate the time required for switching the output of an active gate from low to high and high to low, respectively. Finally, the propagation time of a gate is taken as the worst case of pull-down and pull-up time, i.e., $\max\{t_{pd}, t_{pu}\}$. Note also that since all the terms of capacitance are dependent on the size of lines or gate channels, t_{pd} or t_{pu} can be expressed in terms of λ.

Based on the above capacitance charging-discharging model, the propagation time of each module or target structure is found by summing the worst case delay time of all active gates located in the most critical path. The most critical path is the signal path with the longest propagation delay among all possible signal paths. Since the

propagation delay of all possible paths must be known before the longest delay path can be selected, a computer search of some sort is usually required, especially for random circuits [49]. However, finding the longest path in a highly regular structure is much easier because of the relatively low number of possible signal paths. If there are p possible signal paths inside a module or target structure, the propagation time, T, of this module or target structure can be expressed numerically by

$$T = \max\{ \sum_{i_1=1}^{g_1} \max\{t_{pu}(i_1), t_{pd}(i_1)\}, \sum_{i_2=1}^{g_2} \max\{t_{pu}(i_2), t_{pd}(i_2)\}, \ldots, \sum_{i_p=1}^{g_p} \max\{t_{pu}(i_p), t_{pd}(i_p)\} \}, \quad (4\text{-}8)$$

where g_1, g_2, ..., g_p are the total number of active gates inside the 1^{st}, 2^{nd}, ..., p^{th} possible signal paths, respectively. The index i is used to indicate the active gates in a particular signal path.

4.2 Design Complexity of Regular VLSI Structures

In general, there is no unique measure for VLSI design complexity [50]. As an example, compare an 8-bit microprocessor chip with a 256K memory chip. In terms of design time, the microprocessor chip is more complex than the memory chip because the memory chip has a high degree of regularity and thus requires less design time. However, in terms of

the number of transistors, the memory chip, crowded with hundreds of thousands of transistors is far more complex than the microprocessor chip which contains only a few thousand of transistors. Therefore, a comparison of the design complexity of VLSI circuits can only be made if all the circuits are of same nature. For example, from all points of view the 32-bit microprocessor and 256K memory chips are more complex than their 8-bit and 128K counterparts. The measure for design complexity is thus generally expressed on a relative scale.

The design complexity, or simply the complexity of a circuit is loosely defined as the degree of difficulty encountered in designing that particular circuit. The design-time is said to be proportional to the complexity of the circuit since it requires more time to layout a more complex circuit [44, 45]. As a crude estimation, the worst case (upper bound) time or number of trials needed to obtain an "optimal" placement of M modules among M points is M! time steps or trials. The design time in this example is considered to be factorially proportional to M. A circuit having a larger number M can thus be said to be more complex than the same circuit having a smaller M. Similarly, the time required to obtain an optimal routing of L interconnecting lines can also be expressed in terms of L depending on the routing requirement. In reality, however, there are more practical limitations which relate the design-time of a circuit to the circuit complexity. These are in terms of modules to be placed and/or the number of interconnecting lines to be routed.

As an alternate approach, the relative complexity of a circuit can be directly related to the density of the circuit module layout, i.e.,

active region/module area. The active region of a module is defined as the total (logical-OR) surface area of covered silicon regions, such as lines, contact cuts, gate channels, etc., within that module. This approach is based on the fact that the layout area of any functional circuit designed by higher level modules (e.g., gates) is larger than that of the same circuit packed by lower level modules (e.g., transistors). This is because the predefined boundary of the higher level module prohibits them from being flexible at the upper levels of the design layout. As a result, a target circuit layout tessellated by high-level modules is larger and contains more null interstitial space (unused silicon surface area) than the same target circuit designed using lower level modules. Also, as a consequence of designing the same target circuits using different level modules, the target circuit having the larger layout generally has longer propagation delay due to longer interconnecting paths.

Since this research concentrates on regular structures and MOS IC design technology, the following sections are aimed at justifying the application of the above approach in defining the design complexity of this class of structures.

4.2.1 Module Placement Approach to Design Complexity

For a highly regular structure, such as a systolic array, the placement of modules and routing of lines are plainly defined due to the properties of local connectivity and regularity. What's more, routing is not even an issue once the required modules have been placed. The

interconnecting lines are assumed to abut since the array structure utilizes local communication (i.e., nearest neighbors). Thus, as an initial measure the design complexity of a regular structure can be directly related solely to the number of modules to be laid out. Therefore, the design complexity can be expressed as a function of the number of modules to be placed (f(M)). For example, M=19 transistor modules (3 transistors for each XNOR or 2-input NAND and 2 transistors for each inverter as shown in Figure 3.6) must be placed in order to form a transistor-level FA ("X.FA") in NMOS. However, M=7 gate modules (2 XNOR's, 2 inverters and 3 2-input NAND's) must be placed to design a gate-level FA ("G.FA").

In a CAD/CAE design environment f(M) depends on the placement algorithm utilized by the design system. In the worst case f(M)=M! as discussed previously. In general, since heuristic placement algorithms are not optimal for all designs, f(M) is determined by the specified placement algorithm that is chosen to best suit a particular design [51].

In a non-CAD environment, on the other hand, f(M) is dependent on the expertise of the designer. What's more, the designer's sophistication in IC technology and graph theory can also effectively influence f(M). However, as M increases, f(M) tends to increase much more rapidly due to the limited capability of human memory. Therefore, intuitively, f(M) is linearly proportional to M if M is small but can be exponentially proportional to M as the number of modules increases.

As a result, there is no general expression to relate f(M) and M for all designs. It is possible, however, that an order of magnitude

bound can be found for certain classes of designs. For example, with highly regular structures, which have the ideal locality and regularity properties to allow straight tessellation and simple module placement, f(M) can be assumed to be O(M). Still, the approach using f(M) is undesirable since there is no exact bound on f(M). A better approach to the design complexity is described next.

4.2.2 Compactness Ratio Approach to Design Complexity

An alternative approach to the assessment of the design complexity of regular structures is to relate the complexity of a circuit to the density of the layout of the circuit module. The density of the layout of a circuit module is also called the compactness ratio (CR) and relates to the design complexity. The compactness ratio of a circuit is given by

$$CR = \frac{\text{active region inside a circuit module}}{\text{area of the circuit module}} . \qquad (4\text{-}9)$$

For example, the CR's of the NMOS depletion mode and enhancement mode transistor module shown in Figure 3.3 are 11/16 and 5/9, respectively. These ratios are found by measuring the net covered area per unit square and then dividing by the overall module area per unit square.

In general, the lower bound CR of a circuit module can be expressed as

$$CR_{LB} = \frac{w}{(w+s)} = \frac{w}{P}, \qquad (4\text{-}10)$$

where w and s are the average line width and spacing between two lines, respectively. The pitch, P, is simply the sum of the line width and spacing. If w=s, the lower bound CR of a circuit module is 1/2. However, the field of random or custom circuit design is so broad that the general expression for the CR of any random IC module varies with the application of the circuit and the IC technology used. A general expression for the CR of any MOS circuit module or device is derived next.

The most primitive element in MOS integrated systems is the MOSFET [6, 26, 27]. A MOS transistor is produced whenever a polysilicon path crosses a diffusion path (see Figure 3.3). Any functional circuit can be formed by using such a transistor. For example, the target structures of "X.FA" and "X.MAC" as described in Chapter 3 are tessellated from transistor modules. These transistor modules are packed together, restricted by a set of layout design rules such as the minimum linewidth of polysilicon, diffusion or metal, the spacing between any two lines, size of contact cuts, etc.

Assuming that w_X and s_X are the average linewidth and spacing, respectively, Figure 4.2 shows a layout model of a transistor-level structure built by an infinite number of transistors in a plane unbounded on the sides. The w_X and s_X distances are not necessarily equal in scale and the dotted lines indicate the boundary of a transistor module. Let k be the unbounded number of transistor modules in each row or column. The expression for the CR of this

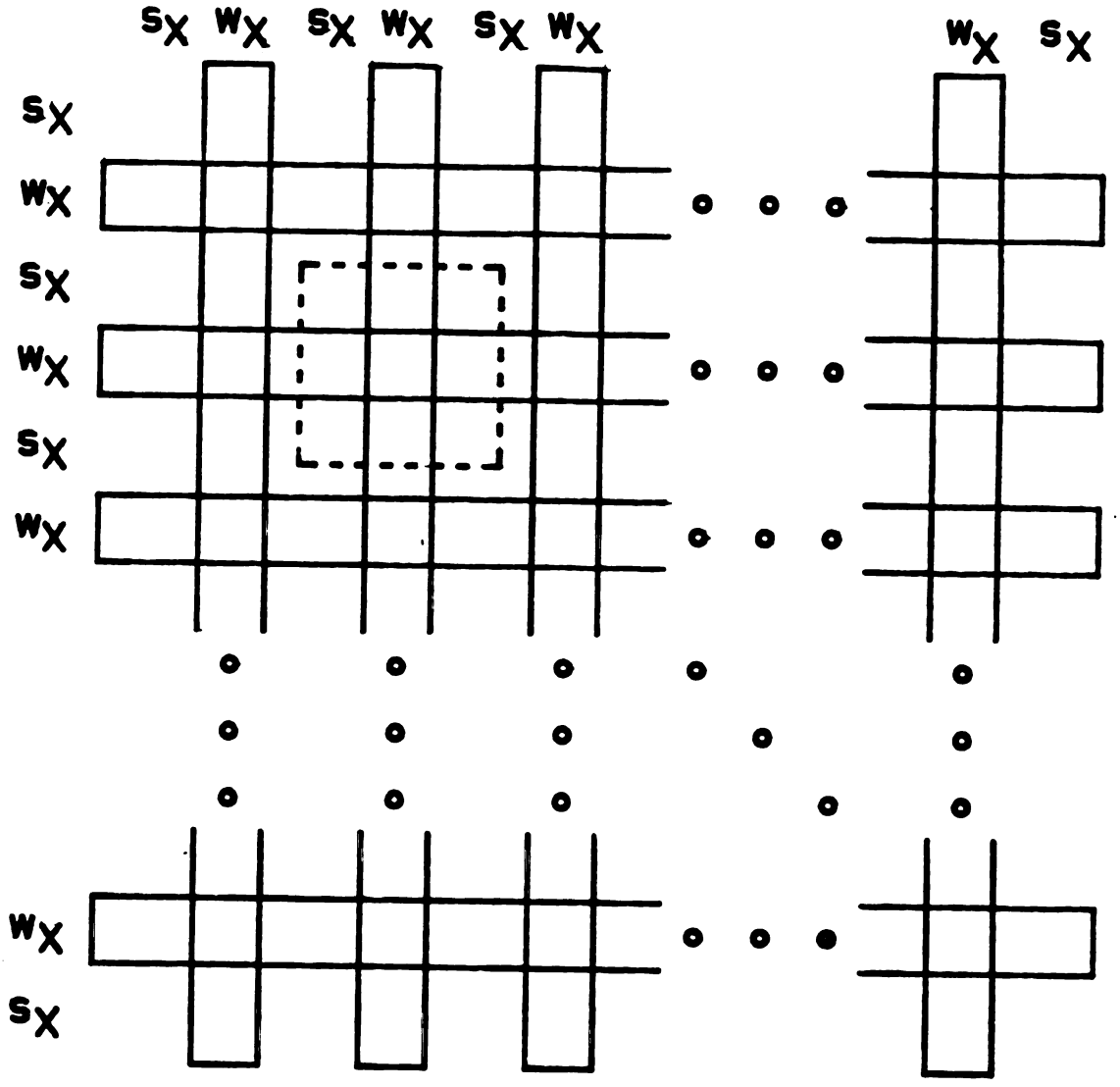

Figure 4.2 A transistor-level module.

transistor-level module is

$$CR_X = \frac{kw_X[k(w_X+2s_X)+2s_X]}{[k(w_X+s_X)+s_X]^2}$$

$$= \frac{kw_X[k(w_X+2s_X)+2s_X]}{L_X^2} , \qquad (4\text{-}11)$$

where L_X is defined as the length of the overall module. By letting $k\rightarrow\infty$ representing a generalization of an unbounded module, Equation 4-11 gives

$$\lim_{k\rightarrow\infty} CR_X = \frac{w_X(w_X+2s_X)}{(w_X+s_X)^2} = \frac{w_X(P_X+s_X)}{P_X^2} , \qquad (4\text{-}12)$$

where P_X is the average pitch as defined previously. Further, if $w_X=s_X$, which is the usual case for MOS design, Equation 4-12 gives the expected $CR_X=3/4=0.75$.

Likewise, a gate-level module or device, such as the "G.FA" or "G.MAC" as described in the last chapter, is tessellated only by gate modules. A basic gate module can be assumed to be formed by a finite array of $k_X x k_X$ transistor modules as in Figure 4.3. This figure illustrates a layout model of a general gate-level structure built by an unbounded number of gate modules. s_G is taken as the average spacing between any two gate modules and the dotted lines are used to clarify the boundary of a gate module. Usually, s_G is greater than s_X or w_X. Note that if $s_G=s_X$ then Figure 4.3 will be exactly the same as Figure 4.2.

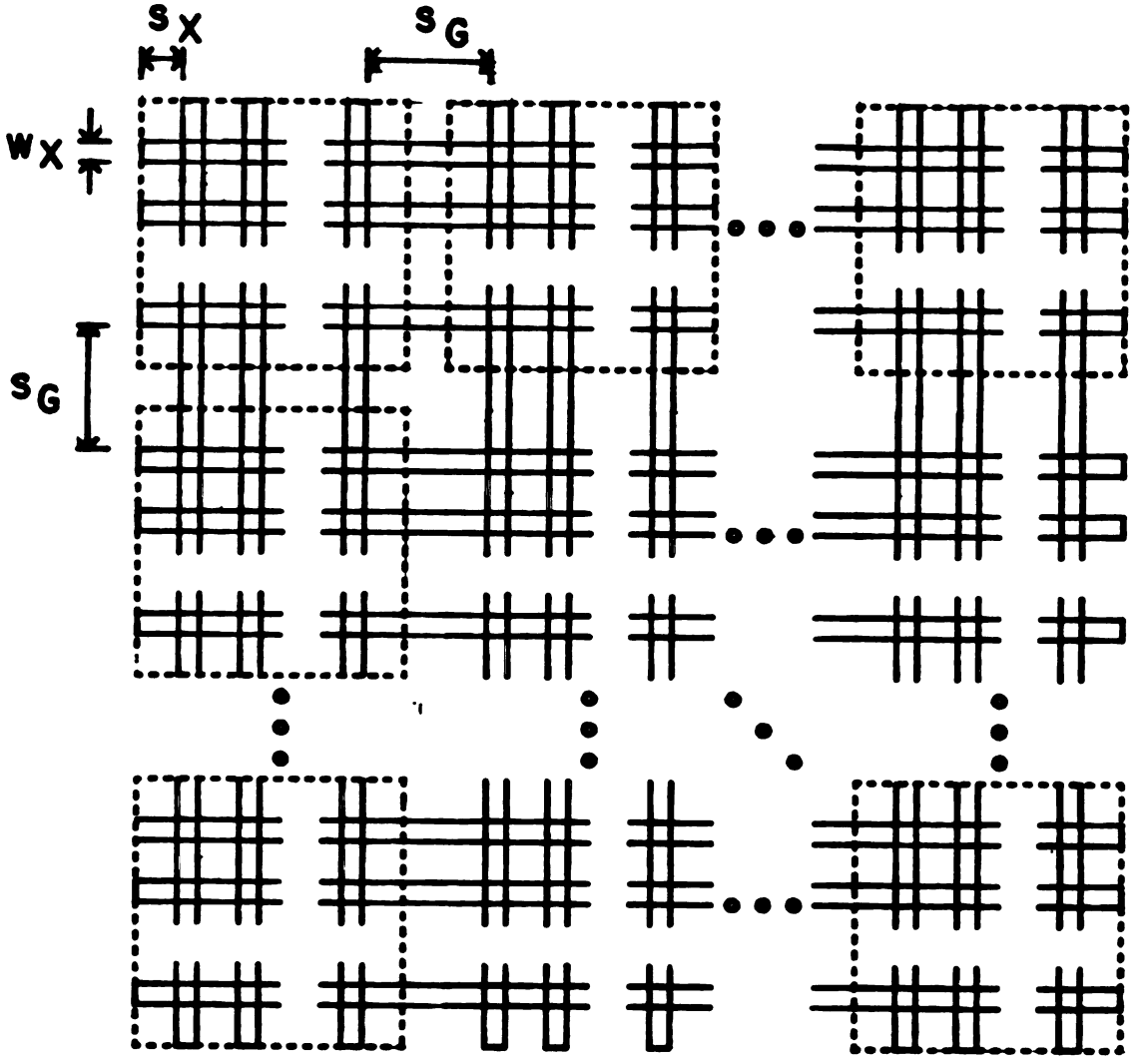

Figure 4.3 A gate-level module.

Again, let k be the unbounded number of gate modules in each row or column. The expression for the CR of any gate-level structure is thus given by

$$CR_G = \frac{kk_X w_X\{k[k_X(w_X+2s_X)-2s_X+2s_G]+4s_X-2s_G\}}{\{k[k_X(w_X+s_X)-s_X+s_G]+2s_X-s_G\}^2} \quad . \tag{4-13}$$

By letting k-->∞, Equation 4-13 yields

$$\lim_{k\to\infty} CR_G = \frac{k_X w_X[k_X(w_X+2s_X)-2s_X+2s_G]}{[k_X(w_X+s_X)-s_X+s_G]^2} \quad . \tag{4-14}$$

For the example of MOS design where $w_X=s_X$, the above equation becomes,

$$\lim_{k\to\infty} CR_G = \frac{k_X w_X(3k_X w_X-2w_X+2s_G)}{(2k_X w_X-w_X+s_G)^2} \quad . \tag{4-15}$$

Therefore, the CR of a MOS gate-level structure is determined by the number of transistors, $k_X xk_X$, inside a gate module, the average spacing between any two gate modules, s_G, and the average line width and spacing between two lines, w_X and s_G, respectively. For example, for some typical value $k_X=2$ and $w_X=s_X=(2/3)s_G$ [6, 37], then $CR_G=56/81=0.691$.

As a result, Equations 4-12 and 4-14 represent the general expressions for the CR's of the MOS transistor-level circuit module (CR_X) and the gate-level circuit module (CR_G). It is obvious that CR_X is greater than CR_G if $s_G>s_X$. Based on the layout models and equations derived above, a more general model can be further developed and applied to all structures that possess multi-level regularity and

modularity. This model is derived in next section and can provide a general comparison of CR versus module area and propagation delay.

4.2.3 Tradeoff of Compactness Ratio Versus Area-Time

A generalized model for the comparison of compactness ratio (CR), chip area and propagation time of a circuit module is presented in this section and can be applied to any structure possessing multi-level regularity. The model is expanded and generalized from the models that have been shown in Figures 4.2 and 4.3. This model represents a bottom-up approach to the design task by assuming that all the high level modules and final target structure are basically tessellated by the lowest level modules such as the MOSFET in MOS design.

Figure 4.4 shows a hierarchical design model of n+1 arbitrary level modules assuming all the high-level modules and the final target structure are formed by tessellation. The lower numbers correspond to lower level modules and the m_{n+1} level module represents a final target structure. Note that any one of these levels can also be considered as the target structure of a sub-module or building block. It is assumed that the design entry-point starts from the lowest level, m_1. Of course design entry-points other than m_1 can also be selected. But, it will be clear that the general expression will become the same no matter which level is chosen as design entry-point.

As a practical example of this model Figure 4.5 illustrates the corresponding hierarchical design of the systolic structure for matrix multiplication as described before. In terms of the notation that was

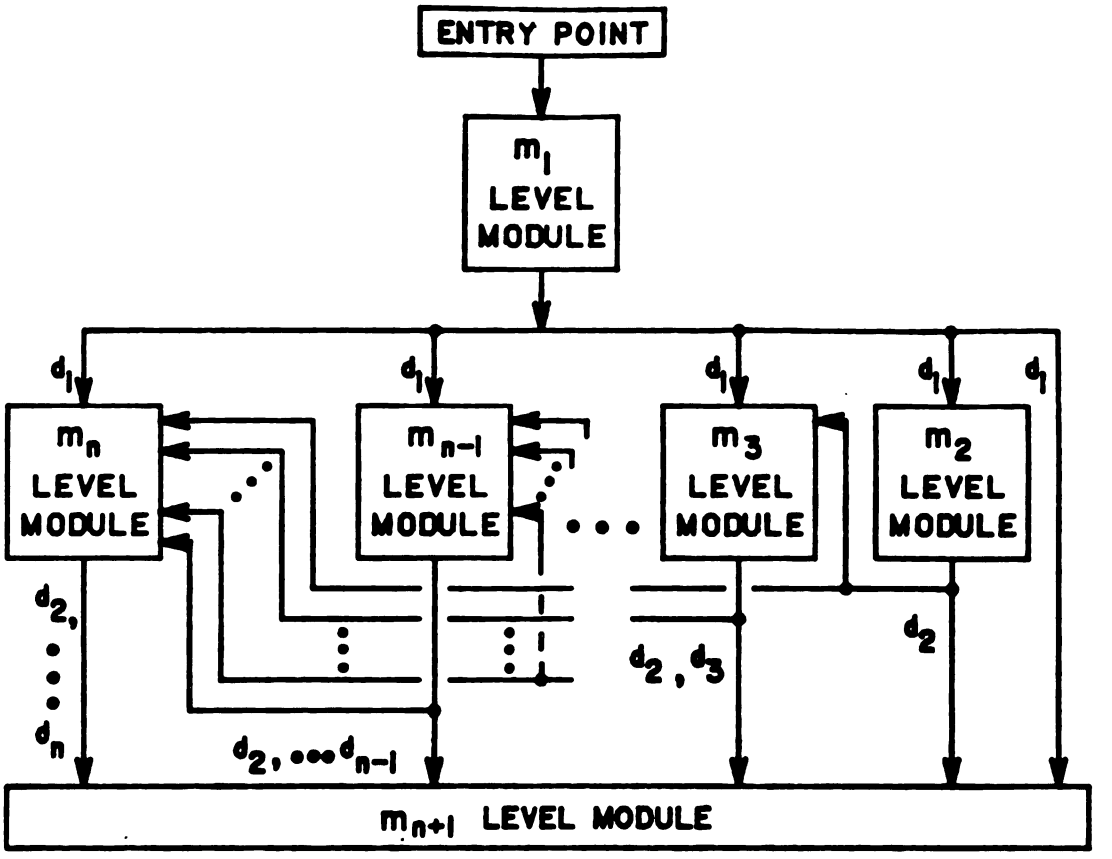

Figure 4.4 Hierarchical design model of an n+1 level module.

used to represent a multi-level module in Chapter 3, $m_1.m_2$, $m_1.m_3$, ..., $m_1.m_{(n+1)}$ denote the "one-path" modules or target structures. These one-path modules are designed via pathways marked as d_1 in Figure 4.4. Specifically, $m_1.m_n$ is called a one path m_n level module. Likewise, $m_1.m_i.m_j$ for $(n+1) \geq j > i$, are defined as two-path m_j level modules designed via pathways chained by d_1 and d_2 as shown in the same figure. An n-path m_{n+1} level module is denoted as $m_1.m_2....m_{n+1}$. For example, an m_3 level module, considered as a target structure, can be formed by either m_1 level modules or m_2 level modules. An m_2 level module in turn

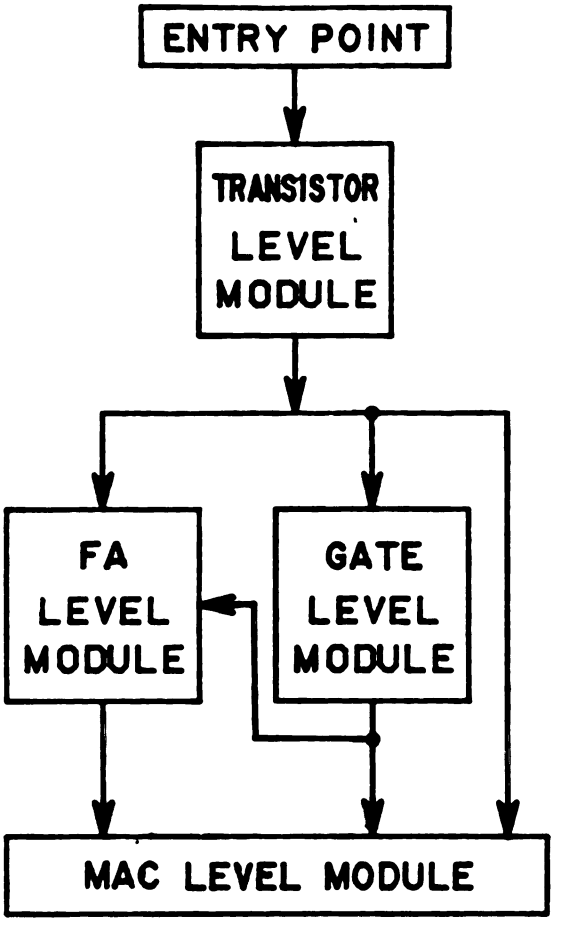

Figure 4.5 Hierarchical design model of a MAC level module.

is tessellated by m_1 level modules. Therefore, $m_1.m_3$ and $m_1.m_2.m_3$ are one-path and two-path m_3 level modules, respectively.

To relate these design path transitions to the design hierarchy described in this work, consider that m_1 is the transistor module, m_2 is the gate module and m_3 is an FA module. The figure implies that to design this particular device two options exist. These options are:

$m_1.m_3$ - implying the construction of the device from transistor modules;

$m_1.m_2.m_3$ - implying the construction of the device from new gate modules specifically designed for this task from transistor modules.

However, if a design entry-point other than m_1 is considered, two more options included are:

$m_2.m_3$ - implying the construction of the device from the gates previously designed and extracted from a cell library;

m_3 - implying the extraction of the device itself from a previously stored library of defined device level modules.

It is clear that the design times of these four options are quite different. It will obviously take a much longer time to design an FA from transistors than it will from gates while the last option has a null design time.

Selecting m_1 as the design entry-point implies that there are $2^{(n-2)}$ different combinational paths for the design of an m_n level module. In other words, the final target structure, an m_{n+1} level module, can be obtained through any one of the $2^{(n-1)}$ distinct design paths as shown in Figure 4.4. However, if all other levels are also considered as possible design entry-points, there are $1+2^{(n-2)}+2^{(n-3)}+\ldots$ different paths for the design of an m_n level module.

Based on the model as illustrated in Figure 4.2, the expression for the CR of any one-path module that is built from an array of $k_1 x k_1$ m_1 level modules is

$$CR_1 = \frac{k_1 w_1 [k_1(w_1+2s_1)+2s_1]}{[k_1(w_1+s_1)+s_1]^2}$$

$$= \frac{k_1 w_1 [k_1(w_1+2s_1)+2s_1]}{L_1^2} , \qquad (4\text{-}16)$$

where w_1 is the average width of the active device within an m_1 level module (e.g., the line width of a MOS transistor) and s_1 is the average spacing between any two active devices. L_1 is simply the length of the overall one-path module. Indeed, this equation is the same as Equation 4-11 if k_1 and the subscript of 1 are replaced by k and the subscript of X, respectively.

Likewise, based on Figure 4.3 and corresponding to the design paths chained by d_1 and d_2 as shown in Figure 4.4, the CR of any two-path module that is built from $k_2 x k_2$ one-path modules is

$$CR_2 = \frac{k_2 k_1 w_1 \{k_2[k_1(w_1+2s_1)-2s_1+2s_2]+4s_1-2s_2\}}{\{k_2[k_1(w_1+s_1)-s_1+s_2]+2s_1-s_2\}^2} , \qquad (4\text{-}17)$$

where s_2 is the average spacing between any two one-path modules. Again, the above equation is same as Equation 4-13 if k_2, k_1 and the subscripts of 2 and 1 are substituted by k, k_X and subscripts of G and X, respectively.

Further, the CR of any three-path module tessellated by $k_3 x k_3$ two-path modules with s_3 as the average spacing between any two two-path module is

$$CR_3 = \frac{k_3k_2k_1w_1(k_3\{k_2[k_1(w_1+2s_1)-2s_1+2s_2]-2s_2+2s_3\}+4s_1-2s_3)}{(k_3\{k_2[k_1(w_1+s_1)-s_1+s_2]-s_2+s_3\}+2s_1-s_3)^2} \qquad (4\text{-}18)$$

In order to check the validity of the above equation, let $k_3=1$. Then $CR_3=CR_2$ because a 1x1 two-path module is simply an array of k_2xk_2 one-path modules. Further, if $k_2=k_3=1$, then $CR_3=CR_2=CR_1$.

For the derivation of CR for a general n-path module formed by k_nxk_n (n-1)-path modules with spacing of s_n, Equations 4-16 to 4-18 are modified as follows. First, let $W_0=w_1$, $R_0=w_1+4s_1$ and $L_0=w_1+2s_1$ be the initial values for the general equation. Then Equation 4-16 is rewritten and becomes

$$CR_1 = \frac{k_1w_1[k_1(w_1+4s_1-4s_1+2s_1)+4s_1-2s_1]}{[k_1(w_1+2s_1-2s_1+s_1)+2s_1-s_1]^2}$$

$$= \frac{k_1W_0[k_1(R_0-4s_1+2s_1)+4s_1-2s_1]}{[k_1(L_0-2s_1+s_1)+2s_1-s_1]^2} \qquad (4\text{-}19a)$$

Physically, W_0 is the average width of the active region of a basic device (e.g., the line width of a MOS transistor) and R_0 is the total lump-sum length of the active region of the same device. Therefore, W_0R_0 is simply the the logical-OR area of a basic device module. Also, L_0 is the length of the basic module.

Further, let $W_1 = k_1W_0$,

$$R_1 = k_1(R_0-4s_1+2s_1)+4s_1-2s_1,$$

and

$$L_1 = k_1(L_0-2s_1+s_1)+2s_1-s_1,$$

then,

$$CR_1 = \frac{W_1 R_1}{L_1^2} \quad . \tag{4-19b}$$

Next, Equation 4-17 is rewritten as

$$CR_2 = \frac{k_2 k_1 W_0 \{k_2[k_1(R_0-4s_1+2s_1)+4s_1-2s_1-4s_1+2s_2]+4s_1-2s_2\}}{\{k_2[k_1(L_0-2s_1+s_1)+2s_1-s_1-2s_1+s_2]+2s_1-s_2\}^2}$$

$$= \frac{k_2 W_1 [k_2(R_1-4s_1+2s_2)+4s_1-2s_2]}{[k_2(L_1-2s_1+s_2)+2s_1-s_2]^2} \quad . \tag{4-20a}$$

Again, let $W_2 = k_2 W_1$,

$R_2 = k_2(R_1-4s_1+2s_2)+4s_1-2s_2$,

and

$L_2 = k_2(L_1-2s_1+s_2)+2s_1-s_2$,

then,

$$CR_2 = \frac{W_2 R_2}{L_2^2} \quad . \tag{4-20b}$$

Likewise, Equation 4-18 is rewritten as

$$CR_3 = \frac{k_3 k_2 W_1 \{k_3[k_2(R_1-4s_1+2s_2)+4s_1-2s_2-4s_1+2s_3]+4s_1-2s_3\}}{\{k_3[k_2(L_1-2s_1+s_2)+2s_1-s_2-2s_1+s_3]+2s_1-s_3\}^2}$$

$$= \frac{k_3 W_2 [k_3(R_2-4s_1+2s_3)+4s_1-2s_3]}{[k_3(L_2-2s_1+s_3)+2s_1-s_3]^2} \quad . \tag{4-21a}$$

Finally, let $W_3 = k_3 W_2$,

$R_3 = k_3(R_2-4s_1+2s_3)+4s_1-2s_3$,

and

$$L_3 = k_3(L_2-2s_1+s_3)+2s_1-s_3,$$

then,

$$CR_3 = \frac{W_3R_3}{L_3^2} \quad . \tag{4-21b}$$

A general recursive equation of CR for the n-path module derived from above equations is

$$CR_n = \frac{W_nR_n}{L_n^2} \quad , \tag{4-22}$$

where,

$$W_n = k_nW_{n-1},\ W_0=w_1;$$

$$R_n = k_n(R_{n-1}-4s_1+2s_n)+4s_1-2s_n,\ R_0=w_1+4s_1;$$

and

$$L_n = k_n(L_{n-1}-2s_1+s_n)+2s_1-s_n,\ L_0=w_1+2s_1.$$

Physically, W_n and R_n are the lump-sum width and length of the active region, respectively, and L_n is simply the length of the overall module.

By letting $k_n \dashrightarrow \infty$ as in Equations 4-12 and 4-14, Equation 4-22 becomes

$$\lim_{k_n \dashrightarrow \infty} CR_n = \frac{W_{n-1}(R_{n-1}-4s_1+2s_n)}{(L_{n-1}-2s_1+s_n)^2} \quad . \tag{4-23}$$

If the m_1 level module is assumed to be a transistor module (i.e., n=1=X), Equations 4-22 and 4-23 are exactly the same as Equations 4-11 and 4-12, respectively.

As a result, Equation 4-23 provides a general recursive expression for the compactness ratio of any multi-level module. What's more, the denominator of Equation 4-22 is actually the area of the n-path module.

Therefore, the general recursive expression of the overall area of a multi-level module is given by

$$A_n = (L_n)^2 = [k_n(L_{n-1}-2s_1+s_n)+2s_1-s_n]^2, \qquad (4\text{-}24)$$

where $L_0=w_1+2s_1$.

In order to find a general expression for the propagation delay of a multi-level module, the delay time of the intercommunication lines among the modules is calculated independently of the internal module delay. Based on the model illustrated in Figure 4.2, the propagation time of a one-path module formed by $k_1 x k_1$ m_1 level modules is found by

$$T_1 = f_1(k_1)(t_{w1}+t_{s1})+t_{s1}, \qquad (4\text{-}25)$$

where t_{w1} is the average propagation time due to the active transit time of a sub-module or the m_1 module (e.g., channel delay of a MOS transistor), and t_{s1} is the delay of the intercommunication lines connecting the sub-modules. $f_1(k_1)$, a function of k_1, is the number of sub-modules through which the critical signal passes.

Obviously, $f_1(k_1)$ varies with the nature of different circuits. For example, the MAC built by approximately k_1^2 FA's in Figure 3.7 has $f_1(k_1)$ approximately equal to $2k_1+1$. In general, $f_1(k_1)$ can be either an analytical distribution function based on probabilistic assumptions or a statistical function based on stochastic programming [52, 53].

Similarly, based on Figure 4.3, the propagation time of a two-path module, built from $k_2 x k_2$ one-path modules, is

$$T_2 = f_2(k_2)(T_1-2t_{s1}+t_{s2})+2t_{s1}-t_{s2}. \qquad (4\text{-}26)$$

Again, $f_2(k_2)$ is a function of k_2 depending on the nature of the circuit. Likewise, the propagation time of a three-path module built from $k_3 x k_3$ two-path modules is given by

$$T_3 = f_3(k_3)(T_2 - 2t_{s1} + t_{s3}) + 2t_{s1} - t_{s3}. \quad (4\text{-}27)$$

Based on Equations 4-25 to 4-27, a recursive expression for the propagation time of an n-path module is derived as

$$T_n = k_n^2(T_{n-1} - 2t_{s1} + t_{sn}) + 2t_{s1} - t_{sn}, \quad (4\text{-}28)$$

where $T_0 = t_{w1} + 2t_{s1}$. Note that the key comparison of different T_n is based on the propagation delay of the intercommunication lines among the sub-modules, t_{sn}.

In summary, Equations 4-23, 4-24 and 4-28 provide the general expressions for the compactness ratio, chip area and propagation time of a general multi-level module. Once the parameters of w_1, t_{w1}, s_i, t_{si}, k_i and $f_i(k_i)$, for all i=1,2...,n, have been estimated, the tradeoffs of compactness ratio versus area-time can be obtained. In fact, in designing a particular target module, if the spacing among higher level modules is increasingly greater than that among lower level modules, i.e., $s_n > \ldots > s_2 > s_1$, the expressions for the CR_n is a decreasing function, i.e., $CR_n < \ldots < CR_2 < CR_1$, while those for A_n and T_n are increasing functions, i.e., $A_n > \ldots > A_2 > A_1$ and $T_n > \ldots > T_2 > T_1$. For example, if $s_2 > s_1$, the compactness ratio of a two-path target structure (e.g. "X.G.FA") is less than that of a one-path target structure (e.g. "X.FA") while the area-time parameters of the two-path structure are greater than those of the one-path structure. As a result, CR_n is inversely proportional to A_n and T_n, corresponding to the parameters of the number, width, and spacing of the sub-modules of different levels.

CHAPTER V

TOP-DOWN APPROACH IN ARRAY DECOMPOSITION

The investigation of hierarchical modularity pursued along the top-down approach starts from the algorithmic level at which a systolic, or other algorithm possessing similar regularity, is physically decomposed providing chip-wise modularity. The algorithmic level structures, such as the matrix-matrix systolic multipliers, are constructed from processing elements (PE's) as defined by the design hierarchy. Therefore, the purpose of the top-down approach is mainly to evaluate the effect of various physical partitions to the tessellated array of PE's. An overview of the top-down approach to the design of systolic structures is shown in Figure 5.1.

The top-down approach is especially important due to the current IC lithographic and chip size limitations. In addition, it can provide the flexibility to customize a modularized array structure with a "best fit" size and/or performance for any system with variable dimensions. In other words, designs using a set of chips (decomposed arrays) can be more flexible and efficient if the dimensions of the problem are variable or not known a priori.

In the following sections, parameters are first defined and then a general expression relating the I/O bottlenecking of operands and pin limitation is derived. Finally, the general expression is depicted graphically.

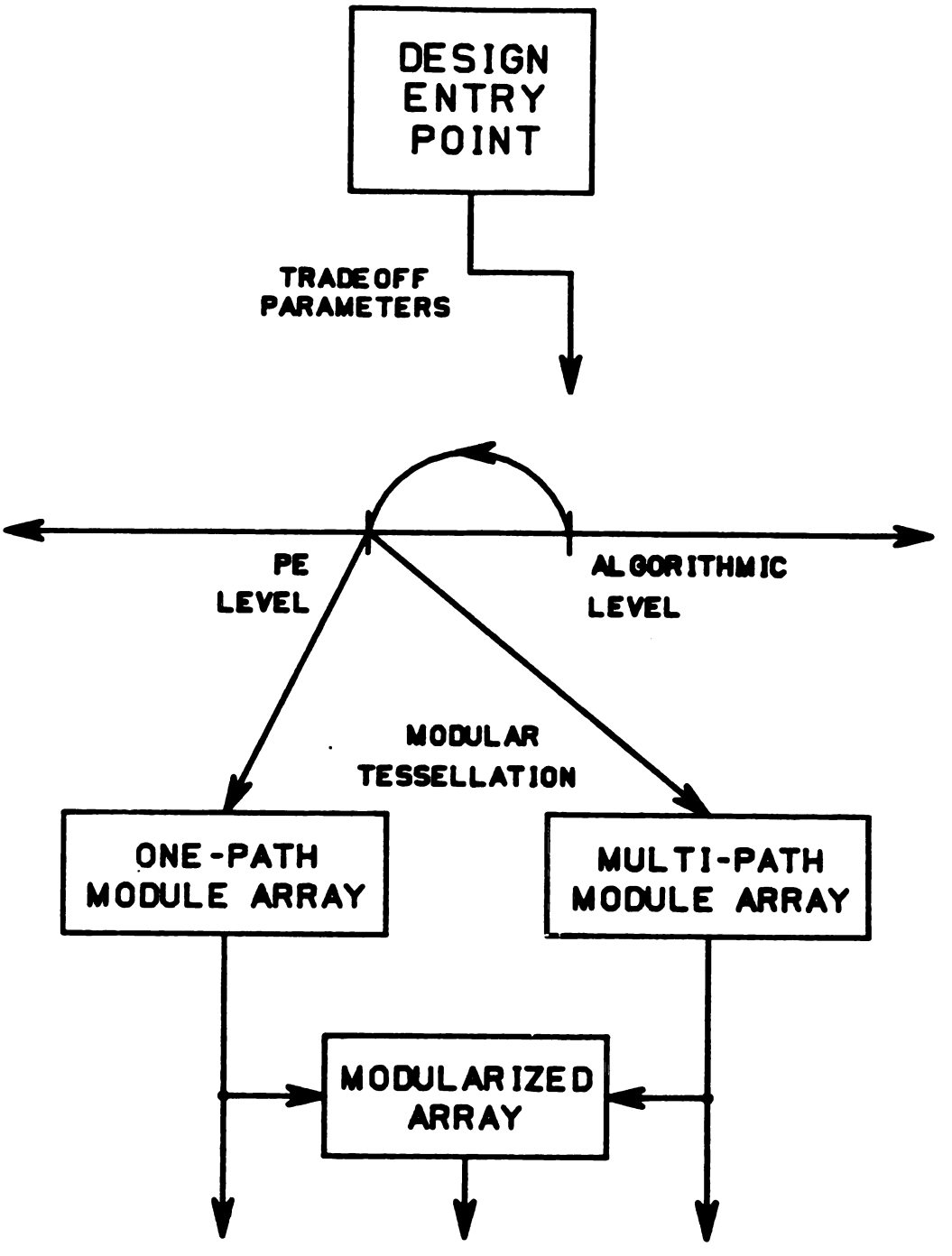

Figure 5.1 An overview of the top-down approach.

5.1 Chip-Wise Partitioning of Systolic Structures

The physical decomposition, which is also known as modularization [4], of a systolic structure is simply the partitioning of groups of interconnected PE's. For similar regular structures, the array of elements at the next lower level below the algorithmic level is to be partitioned. Each group of partitioned PE's is assumed to be implementable and fabricated on a single IC chip. The maximum number of PE's inside each chip depends, of course, on their size. Thus, assuming an unbounded number of chips, a systolic structure of arbitrarily large dimension can be implemented on a set of IC chips.

To maintain regularity, all the PE's within a partitioned group or chip must be identical. Let i and j be the numbers of input and output operands of each PE (local I/O), respectively, and n be the number of bits per operand. Then the total number of local I/O lines per PE is (i+j)n. Usually I/O symmetry is a "by-product", characteristic of a structure possessing regularity and local connectivity properties. Symmetrical I/O lines mean i=j, which will be assumed throughout this section. For example, the MAC's shown in Figures 2.1 and 2.2 have i=j=3, respectively. PE's which are commonly found in systolic structures have i=j=2 or 3 [3-6].

Consider an R-by-C partitioned array of symmetrically interconnected PE's fabricated on an IC chip where R and C are the number of elements in a row and column, respectively. The maximum

number of pinouts (for operands only) per modularized chip is then given by

$$P_{op}(R,C,i=j,n) = 2[(R+C)(i-1)-(i-2)]n. \qquad (5\text{-}1)$$

As an illustration, Figure 5.2 shows two sample arrays with RxC=3x3 and 4x2. PE's in the 3x3 array have local I/O lines of i=j=2 and those

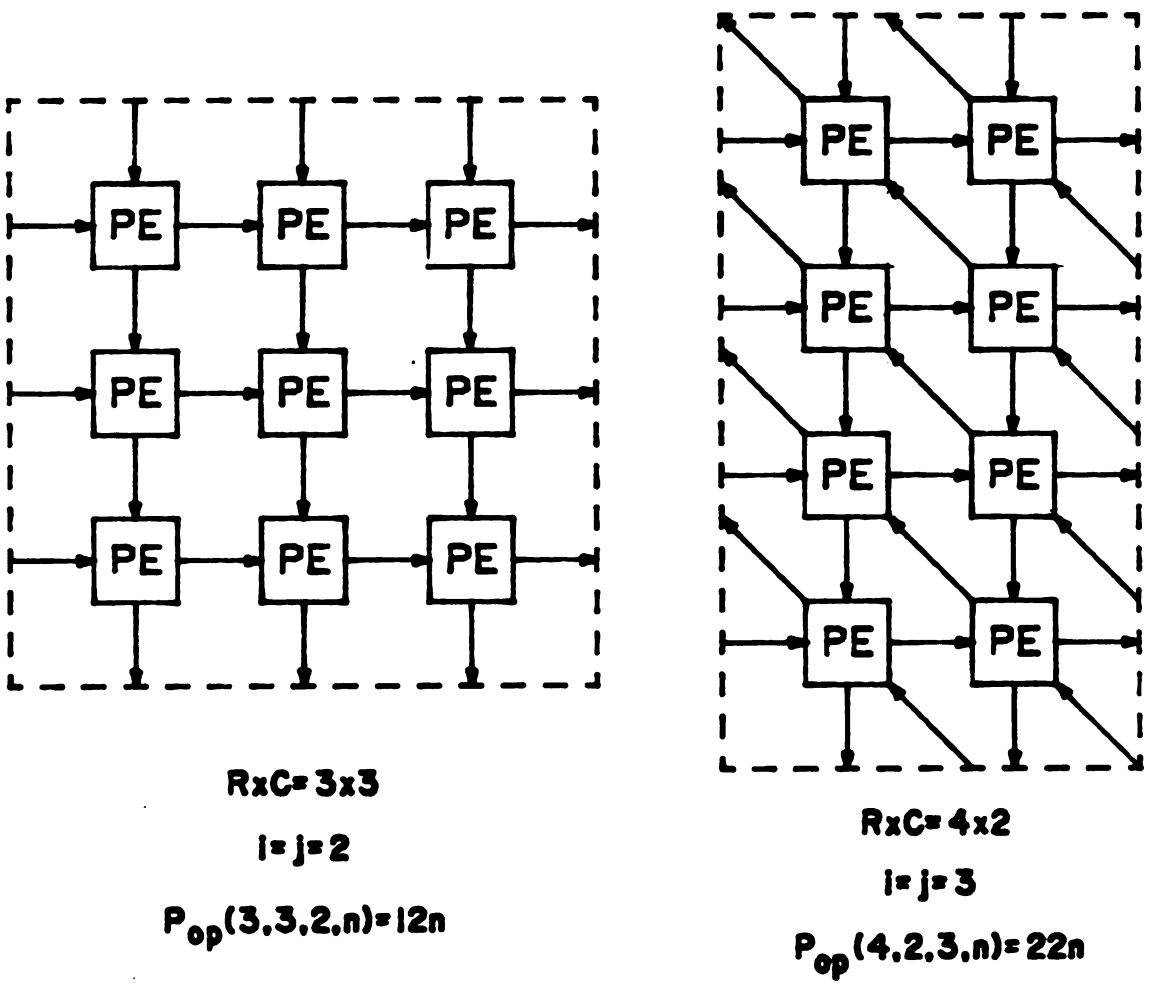

Figure 5.2 Examples of partitioned arrays.

in the 4x2 array have i=j=3. Note that the depiction of latch placement is neglected to simplify this illustration. Also, the direction of the data flow is assumed to be symmetric but not restricted to the fixed directions as shown in the figure.

Assuming all I/O buses shown in Figure 5.2 contain n lines, the maximum numbers of pinouts of the partitioned arrays are then equal to $P_{op}(3,3,2,n)=12n$ and $P_{op}(4,2,3,n)=22n$. This points out an immediate constraint to the practical implementation of these examples in that the numbers of pinouts for n=32 bits are 384 and 704. This is far beyond the current IC pin limitation and thus requires the use of an appropriate I/O multiplexing scheme.

Assuming two separate sets of pinouts are used for the input fan-in demultiplexing (DMUX) and output fan-out multiplexing (MUX) of I/O operands, the number of pinouts is reduced to $P_{op}=2n$. Further, if a combined I/O DMUX/MUX scheme is used then $P_{op}=n$, but this is not cost-effective as will be explained in the next paragraph. Thus, the number of pinouts for operands can be expressed in terms of n depending on the multiplexing schemes used. Multiplexing schemes using $P_{op}=2n$ and 3n have been proposed for computing arrays for matrix multiplication and covariance matrix inversion [10].

As mentioned, it is not cost-effective to use a single set of pinouts for a combined I/O DMUX/MUX scheme. This is due to many reasons. First, the sharing of the same set of pinouts by both input and output operands complicates the scheduling of systolic operations. Secondly, extra logic circuits will be required inside each chip for the control of bidirectional flow of input and output operands. Thirdly,

since all data pinouts of all chips are connected to the same data buses, there is virtually no chip-wise parallel data transmission. Lastly, the combined scheme has the largest potential for exhibiting an I/O bottleneck because it has longest multiplexing time in comparison with other schemes using separate sets of pinouts. An I/O bottleneck occurs when the I/O operand multiplexing time dominates over the pipeline segment time [5, 9]. Based on these reasons, multiplexing schemes utilizing at least two sets (a pair of n bits) of data pinouts are considered best, i.e., $P_{op} \geq 2n$. The upper bound $P_{op}(n)$ is simply the maximum number of pinouts for operands without I/O multiplexing as found in Equation 5-1.

Unfortunately, no single multiplexing scheme is universally applicable to every modularized systolic algorithm [4, 10]. Different systolic algorithms use different schemes as well as different numbers of pairs of n data pinouts.

For example, two different schemes for modularizing systolic structures for L-U decomposition and matrix inversion have been reported by Hwang and Cheng in [4]. The square-shaped multiply and divide partitioned array chips for the construction of the L-U decomposition network utilize 5 and 2.5 pairs of n data pinouts, respectively. But, the triangular multiply and square-shaped multiply modularized array chips for the formation of the matrix inversion network use 3 and 4.5 pairs of n pinouts, respectively. All of the modularized multiply arrays in [4] contain MAC's having i=j=3 and the division arrays are composed of division cells (DC's) having i=j=2. Since 5 pairs of n data pinouts is obviously equal to 10n pinouts, n is limited to a word size

of less than 32 bits. This limitation is very undesirable because many engineering problems require high precision calculations [5]. Also, the consideration of I/O bottlenecking was not mentioned in [4].

5.2 Tradeoff Between I/O Bottleneck and Pinout Limitations

Although no universal multiplexing scheme can be applied to all modularized systolic algorithms, a general expression can be derived relating R, C, i, j, and N_s. This expression will be used to further study the details of I/O bottlenecks between partitions. N_s is defined as the number of pairs of n data pinouts. The others have already been defined in Equation 5-1. As discussed previously, at least one pair of n data pinouts must be used for a cost-effective multiplexing scheme; therefore, $N_s \geq 1$.

Interestingly, if N_s is less than i=j, arrays of external demultiplexers (DMUX's) and multiplexers (MUX's) must be placed amid the modularized chips in order to route the data operands properly. As an example, consider the array of 2x2 modularized chips consisting of RxC=2x2 PE's as shown in Figure 5.3. Each chip in this array has N_s=1 and i=j=2. Assuming each bus is n bits wide, an array of n 1:2 DMUX's is required to demultiplex the output operands to the either right or lower neighboring chips. Similarly, n 2:1 MUX's are necessary to multiplex the input operands from either the left or the top adjacent chips. However, if $N_s \geq i=j$, no external DMUX's or MUX's are needed. This can be illustrated by using the same example except having N_s=i=j=2 as shown in Figure 5.4.

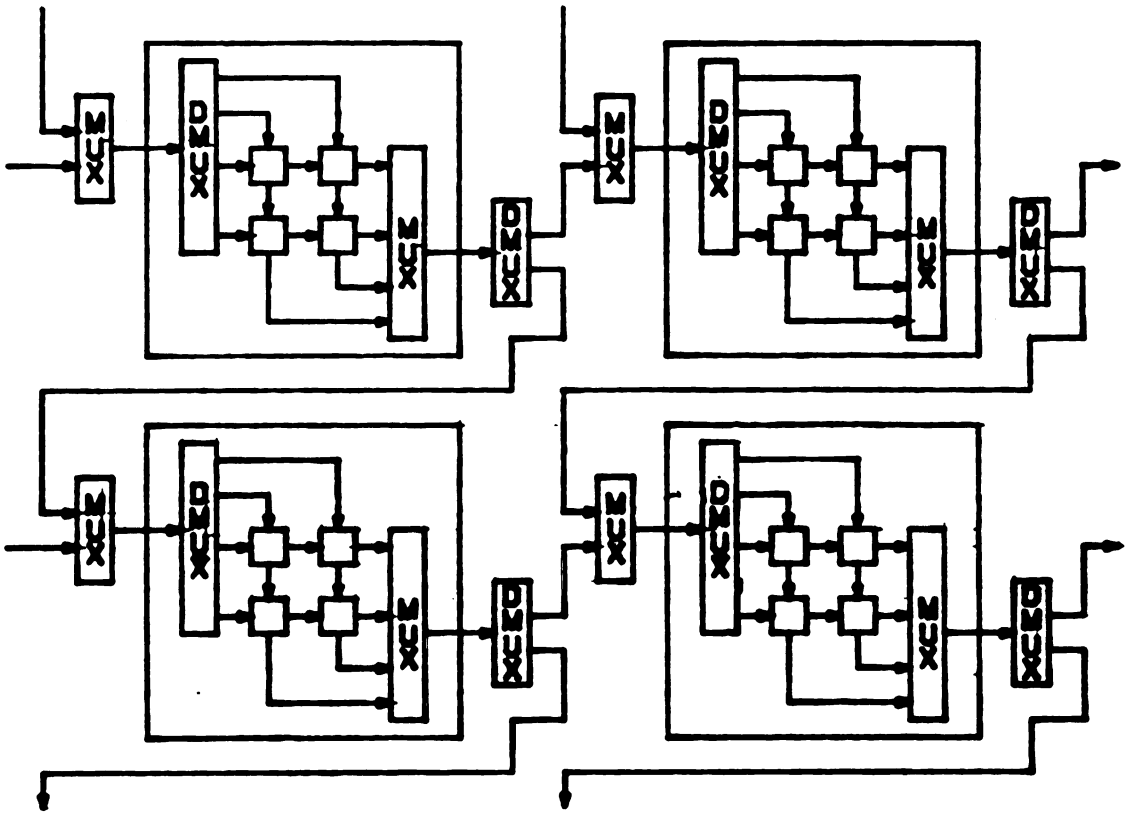

Figure 5.3 Array of modularized chips having N_s<i=j.

Close inspection of both Figures 5.3 and 5.4 reveals that the fan-in (FI) DMUX and fan-out (FO) MUX time inside the chip are the most critical factors of inducing a dynamic I/O bottlenecking of operands. On the other hand, the external DMUX/MUX time, if any, outside the chip is comparatively smaller and thus can be neglected when R and C are large. Note that the maximum possible number of pairs of FI DMUX and FO MUX circuits inside a chip is exactly equal to N_s. Also, since all modularized chips are to operate in a parallel/pipeline fashion, the FI demultiplexing is synchronous with the FO multiplexing. That is, the

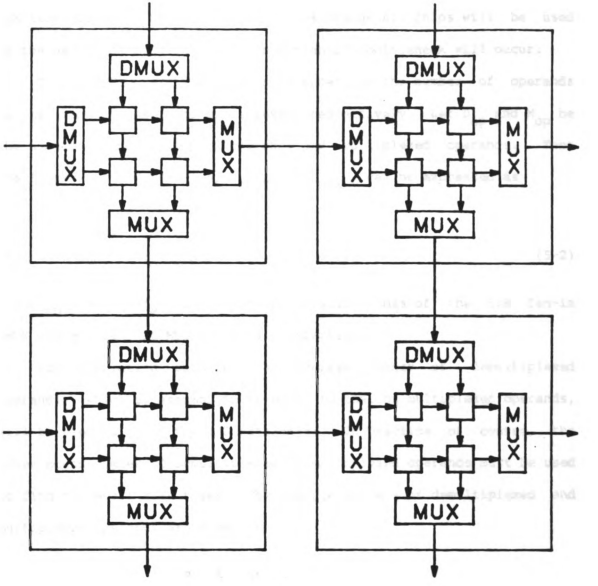

Figure 5.4 Array of modularized chips having N_s=i=j.

loading of input operands overlaps with the unloading of output operands. Thus, the most critical (worst) case of either FI DMUX or FO MUX time among all DMUX and MUX circuits inside all chips will be used in the determination of whether or not an I/O bottleneck will occur.

The FI DMUX or FO MUX time is dependent on the number of operands to be demultiplexed or multiplexed, respectively. Let D_{op} and M_{op} be the numbers of demultiplexed operands and multiplexed operands. Then the FI DMUX and FO MUX time, t_{DMUX} and t_{MUX} , can be expressed as

$$t_{DMUX} = D_{op} t_D(1{:}M),$$

$$\text{and} \qquad t_{MUX} = M_{op} t_M(M{:}1), \tag{5-2}$$

where $t_D(1{:}M)$ and $t_M(M{:}1)$ are the propagation times of the 1:M fan-in DMUX and M:1 fan-out MUX circuits, respectively.

For simulation purposes, an average number of demultiplexed operands, which is assumed to be equal to that of multiplexed operands, will be used to study the I/O bottleneck. In practice, of course, the upper bound number of demultiplexed or multiplexed operands must be used to find the worst case times. The average numbers of demultiplexed and multiplexed operands are given by

$$D_{op} = M_{op} = \frac{P_{op}(R,C,i=j,n)}{2N_s n}, \tag{5-3}$$

where $P_{op}(R,C,i=j,n)$ is the maximum number of pinouts for data operands per modularized chip as defined in Equation 5-1, and $2N_s n$ is the actual number of pinouts per modularized chip used for data I/O. Thus, substituting Equation 5-1 into Equation 5-3 gives

$$D_{op} = M_{op} = \frac{(R+C)(i-1)-(i-2)}{N_s} \quad . \tag{5-4}$$

As a result, the average fan-in DMUX and fan-out MUX times are represented by

$$t_{DMUX} = \frac{(R+C)(i-1)-(i-2)}{N_s} t_D(1:M),$$

and

$$t_{MUX} = \frac{(R+C)(j-1)-(j-2)}{N_s} t_M(M:1), \tag{5-5}$$

where i=j if the inputs and outputs of PE's are equal (symmetry property).

Finally, let the worst case PE segment time be t_{PE}. An I/O bottlenecking index, BI, can be defined as

$$BI = \frac{\max \{ t_{DMUX}, t_{MUX} \}}{t_{PE}} \quad . \tag{5-6}$$

If BI>1 an I/O bottleneck exists. What's more, the greater the BI, the more severe the degradation of overall performance of the circuit will be. Assuming i=j and substituting Equation 5-5 into Equation 5-6 produces

$$BI = \frac{(R+C)(i-1)-(i-2)}{N_s t_{PE}} \max\{t_D(1:M), t_M(M:1)\}. \tag{5-7}$$

To depict the above equation graphically, consider a systolic array built merely by MAC's. Since MAC has 3 pairs of symmetry input and

output lines, i.e., i=j=3, Equation 5-7 becomes

$$BI = \frac{2(R+C)-1}{N_s t_{PE}} \max\{t_D(1:M), t_M(M:1)\}. \qquad (5\text{-}8)$$

For simplicity in simulation, R=C and $t_D(1:M)=t_M(M:1)$ are assumed. Note that R=C implies that the partitioned array has R^2 or C^2 MAC's. Interestingly, since $BI \propto (R+C)$, BI is the minimum if R=C. Further, let $t_B=t_D/t_{PE}$, then Equation 5-8 gives

$$BI = \frac{(4R-1)}{N_s} t_B. \qquad (5\text{-}9)$$

Based on the above equation, Figure 5.5 graphically depicts the relationship between BI and the number of pinouts in terms of t_B and n. Each curve in the figure corresponds to the number of R=C, which can also be used to represent the chip area of a modularized array.

To illustrate the application of this graph, consider that R=10 (i.e., 10x10 MAC's), N_s=1 (i.e., number of pinouts=2n) and t_B=0.05 (i.e., $t_D(1:M)$=5 nsec and t_{PE}=100 nsec); then BI=1.95. This indicates that an I/O problem exists. In other words, the I/O time is almost twice as long as the pipeline segment time which degrades the overall performance of the circuit by one-half. The solution of this I/O problem is to either design faster DMUX and MUX circuits or to reduce the number R. Another feasible solution is to increase the number of pinouts which, however, is dictated by the I/O multiplexing scheme as well as the technological pinout limitation.

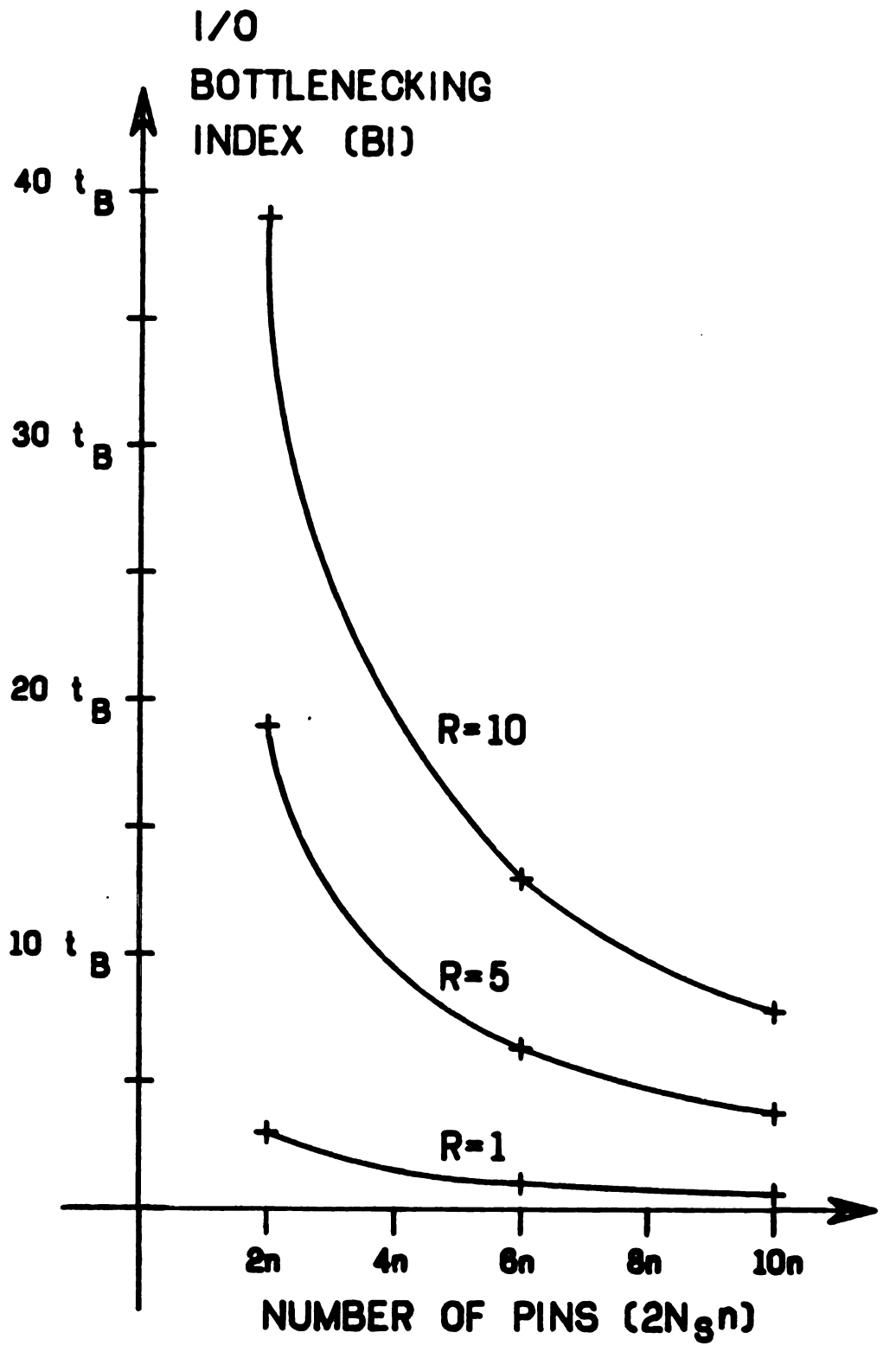

Figure 5.5 I/O bottlenecking index versus number of pins.

The I/O bottlenecking index, BI, defined in Equation 5-7 provides an indication of the occurrence of an I/O bottleneck of operands during chip-wise partitioning of a systolic or other similar structure. By examining BI and the number of pinouts allowed per modularized array or chip, one can evaluate the tradeoff of various physical partitions to the tessellated arrays of PE's versus the I/O pin limitation.

CHAPTER VI

RESULTS AND CONCLUSIONS

The bottom-up and top-down approaches to the design methodologies and the study of tradeoffs for hierarchical modularity in regular VLSI-based circuits have been presented in previous chapters. In this chapter, the results obtained from the design of a systolic array structure as a testbed using these approaches are presented and discussed. These results numerically verify the tradeoffs among circuit complexity, chip area, propagation time, number of pinouts and the condition for I/O bottleneck existence in the design of multi-level structures. A design of a target structure can be started according to an analysis of these tradeoff parameters. The choice of the design entry-point and pathway will eventually determine the final circuit performance. Finally, future trends in the design of VLSI-based structures related to this research are discussed.

6.1 Compactness Versus Area-Time Ratios

Using the bottom-up approach discussed in Chapter 3, the physical layouts of several FA's as target structures are shown in Figures A.1-A.3 in the Appendix. These represent "X.FA", "G.FA" and "FA". In each figure there are two different versions, type-A and type-B. These

two types were designed so as to enable tessellation into two types of nearest neighbor communication patterns.

A 5-bit MAC tessellated by both types of FA's is illustrated in Figure 6.1. The schematic layout of this target MAC is actually sheared and flipped vertically from the rhombic shaped MAC shown in Figure 3.7. Note that the summand generation takes place in the lower left corner of the FA's. This is depicted by the solid geometric symbols shown in Figure 6.1. The required number of type-A and type-B FA's to form an n-bit MAC are n^2-2n+2 and $2n+1$, respectively. Type-A is the majority FA and will dominate over type-B FA's if n is large. For example, if n=32, the number of type-B FA's is just 6.76% of that of type-A FA's.

Table 6.1 shows the chip area, propagation time and compactness ratio parameters corresponding to the different entry-points and pathways traversed in designing the various target FA modules. Since the relative area and time ratios are used for comparison only, the parameter of the lithographic linewidth, λ, has been removed. For simplicity, the figures of area and time in Table 6.1, and all other tables presented in this chapter, are given assuming λ=1 μm. The chip areas of the various modules were measured directly from their physical layouts. The compactness ratio (CR) is a ratio of the logical-OR area to the overall area of the module. The logical-OR area is the total surface area of covered silicon regions. This was found from the physical layout automatically by the CAD system utilizing a command called "LOG OR ACTIVE" which uses the coordinates of the layout to find the logical-OR area.

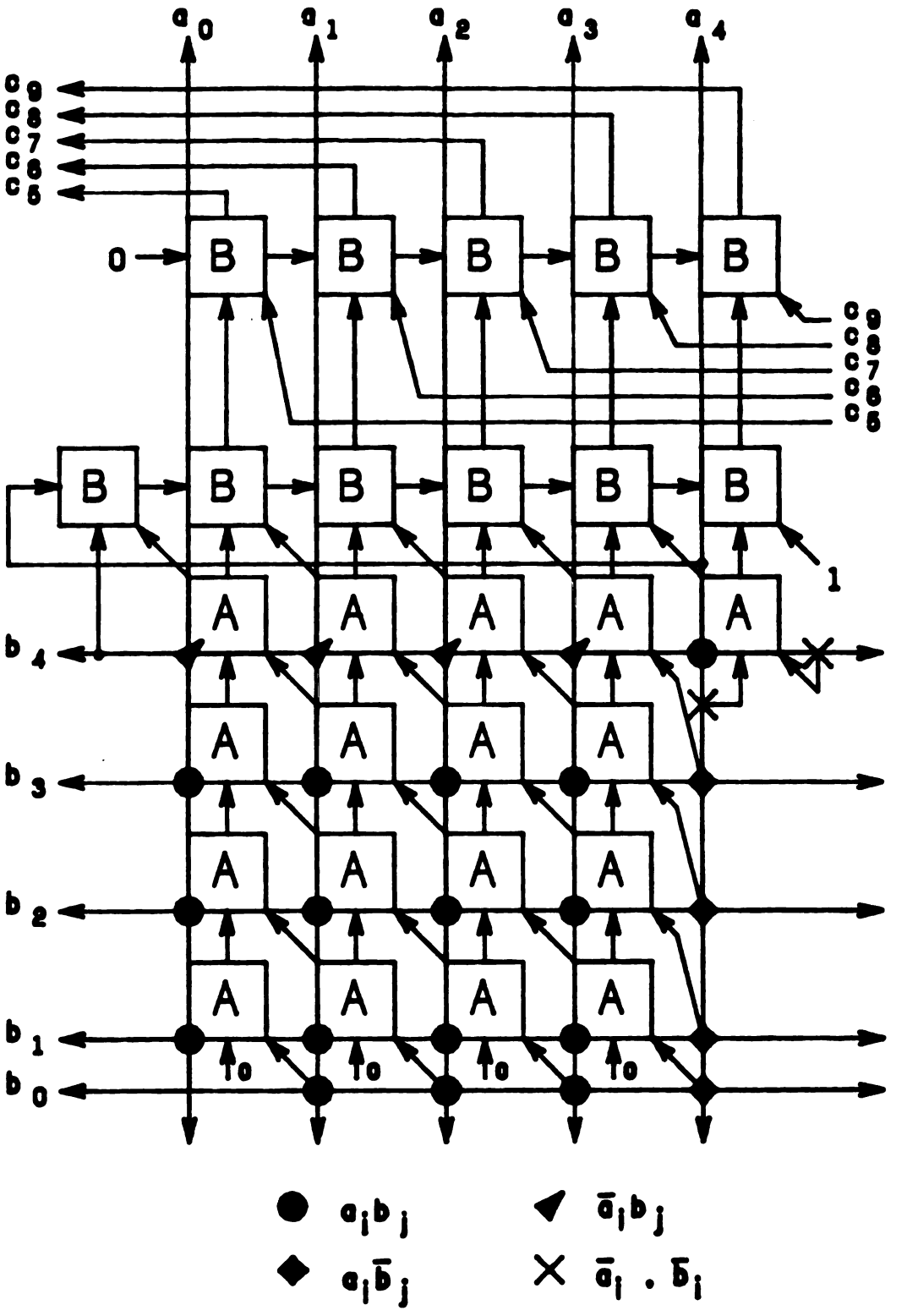

Figure 6.1 Schematic layout of a 5-bit MAC.

Table 6.1 Area-time and compactness ratios of FA.

Target module	Entry point	Design path	Area um^2	Area NR	Time nsec	Time NR	ATR	CR
Type-A FA	X	X.FA	90x34	1.00[a]	2.04	1.00[a]	1.00	0.755
	G	G.FA	94x36	1.11[a]	2.33	1.23[a]	1.17	0.686
	FA	FA	103x40	1.35[a]	2.56	1.37[a]	1.36	0.644
Type-B FA	X	X.FA	87x30	1.00[a]	2.06	1.00[a]	1.00	0.751
	G	G.FA	92x36	1.27[a]	2.19	1.06[a]	1.17	0.684
	FA	FA	103x37	1.46[a]	2.36	1.15[a]	1.31	0.671

NR=Normalized Ratio
ATR=Area-Time Ratio
CR=Compactness Ratio
[a]Ratio based on X.FA

The propagation time results were obtained by using the charging-discharging model presented in Section 4.1. Since the model requires physical parameters of the MOS transistor, such as threshold voltages and doping concentration, typical values of these parameters were obtained from references and are listed in Table 6.2 [6, 26, 27].

The normalized ratios (NR's) of both area and time shown in Table 6.1 are based on the corresponding values of either type of "X.FA". The area-time ratio (ATR) shown in the second from last column of the table is the mean of NR's of the corresponding design path. The table shows that CR increases, representing an increase in complexity, as ATR decreases, indicating a better area utilization and performance.

Table 6.2 The typical physical parameters of MOSFET technology [6,26,27].

Threshold voltages:	
depletion mode transistor (V_{tu})	-4 V
enhancement mode transistor (V_{td})	1 V
Voltage supply (V_{DD})	5 V
High level gate output (V_H)	5 V
Low level gate output (V_L)	0 V
Oxide thickness between transistor gate and channel	250 A-1000 A
Permittivity of silicon dioxide	3.45×10^{-11} F/m
n-type impurity doping concentration	10^{16} cm^{-3}
Electron mobility at 300°K (u_n)	1000 cm^2/V-sec

A circuit with a higher complexity has several important implications as follows.

1. The high density of the circuit layout complicates the process of mask generation because masks must be perfectly aligned over each other. There is virtually no tolerance of any misalignment.

2. Pinch-through between two devices is most likely to happen in a more complex circuit because of the closeness of the devices. Pinch-through simply causes two devices to short together.

3. Power dissipation per unit area is much higher in a denser circuit requiring special treatment in heat removal.

4. The difficulty of testing a circuit tends to increase with the complexity.

The average CR's of the transistor-level FA, "X.FA", and the gate-level FA, "G.FA", are 0.75 and 0.69 as shown in Table 6.1. According to Equations 4-12 and 4-15, this implies that $w_X=s_X=(2/3)s_G$ and $k_X=2$. This is because the average spacing is equal to the average line width (an NMOS design rule as per Mead and Conway). Also, the average spacing between two gate modules, again by design rule, is 3/2 of that between two transistors [6, 37].

The average CR of both types of "FA" is found to be 0.66. This indicates that the spacings among the submodules inside the "FA" are larger than those of the other target FA's. Interestingly, the design of a "FA", which used an approach similar to a gate-array technique in order to provide a fast design time, has the smallest CR but the largest ATR among the three different target FA modules.

Using the above various FA modules as basic cells, three target MAC's, "X.FA.MAC", "G.FA.MAC" and "FA.MAC", are tessellated. In addition to these MAC's, two more MAC's designed using transistor or gate modules only are the "X.MAC" and "G.MAC". The geometrical layouts of these five MAC's are presented in Figures A.4-A.8 in Appendix. All MAC's shown in these figures are 4 bits in size for illustration. In each of these figures, a high percentage of wasted (null) areas appears in the far left and right sides of the MAC layout. In a practical application, however, MAC designs are at least 16 bits rendering the null area insignificant. As an example, a 16-bit "G.FA.MAC" is depicted in Figure A.9. In this figure, the null area as mentioned above is less than 5% of the overall module area.

Table 6.3 presents the results obtained from the designs of five different target MAC's again with λ=1 μm. The area LxW in μm^2, and time t in nsec, of the corresponding target MAC's are expressed as

$$\begin{aligned}
&\text{X.MAC} &&: L_X xW_X = (104n+84)x(44n+29), && t_X = 4.07n+2.11;\\
&\text{X.FA.MAC} &&: L_{XF} xW_{XF} = (114n+95)x(48n+31), && t_{XF} = 4.14n+2.18;\\
&\text{G.MAC} &&: L_G xW_G = (111n+94)x(46n+40), && t_G = 4.67n+1.91;\\
&\text{G.FA.MAC} &&: L_{GF} xW_{GF} = (118n+101)x(50n+37), && t_{GF} = 4.77n+1.90;\\
&\text{FA.MAC} &&: L_F xW_F = (127n+114)x(54n+38), && t_F = 5.23n+1.98. \qquad (6\text{-}1)
\end{aligned}$$

Table 6.3 Area-time and compactness ratios of MAC.

Target module	Entry point	Design path	Area um^2	NR	Time nsec	NR	ATR	CR
MAC	X	X.MAC	$L_X xW_X$	1.00[a]	t_X	1.00[a]	1.00	0.69[b]
		X.FA.MAC	$L_{XF} xW_{XF}$	1.20[a]	t_{XF}	1.02[a]	1.11	0.62[b]
	G	G.MAC	$L_G xW_G$	1.12[a]	t_G	1.15[a]	1.14	0.65[b]
		G.FA.MAC	$L_{GF} xW_{GF}$	1.29[a]	t_{GF}	1.17[a]	1.23	0.59[b]
	FA	FA.MAC	$L_F xW_F$	1.50[a]	t_F	1.29[a]	1.40	0.57[b]

NR=Normalized Ratio
ATR=Area-Time Ratio
CR=Compactness Ratio
[a]Ratio based on X.MAC & n-->∞
[b]n-->∞

$L_X xW_X = (104n+84)x(44n+29)$ $\quad t_X = 4.07n+2.11$
$L_{XF} xW_{XF} = (114n+95)x(48n+31)$ $\quad t_{XF} = 4.14n+2.18$
$L_G xW_G = (111n+94)x(46n+40)$ $\quad t_G = 4.67n+1.91$
$L_{GF} xW_{GF} = (118n+101)x(50n+37)$ $\quad t_{GF} = 4.77n+1.90$
$L_F xW_F = (127n+114)x(54n+38)$ $\quad t_F = 5.23n+1.98$
n=Number of bits per word

By letting n-->∞ for comparison, the normalized ratio (NR) and compactness ratio (CR) in Table 6.3 are made independent of n. In fact, if $n \geq 32$, new NR's and CR's will vary by less than 1% from those in Table 6.3. This table further shows the inverse relationship between the ATR and CR of the target MAC's designed by different design entry-points and pathways.

Interestingly, the set of CR's for the MAC as a target module is relatively lower than that of the FA as the target module (compare Tables 6.1 and 6.3). This is because of the effect of the greatly increased number of signal communication paths connecting and routing around the FA's inside the MAC. Since the average lines in NMOS design are separated by an equal width, there will be a factor of $w_X/(w_X+s_X)=w_X/2w_X=0.5$ effect on the overall CR if the number of signal paths is significantly large. But, if multiple metal layers are considered for global communication lines, the overall CR will not be significantly affected. This, however, increases the number of fabrication steps and decreases the yield of good IC chips due partially to an increase in the number of contact cuts [54].

In the design of the final target structure, a matrix-matrix systolic multiplier, the design of latches to be placed amid the MAC's must be considered. Figures A.10 and A.11 in Appendix present the layouts of two dynamic latches with and without scan design. These layouts are based on the logic diagrams presented in Figures 3.10 and 3.11. These latches are designed to tessellate with the MAC's, either vertically or horizontally, without any routing. The area and time delay of the latch without scan design are 33x38 μm^2 and 0.48 nsec; and

those of the latch with scan design are 46x38 μm^2 and 0.52 nsec, respectively, for λ=1 μm. Therefore, the increases of area and time for scan design are 39% and 8.3%, respectively. The CR of the latch with scan design is 0.698 and is lower than that of the latch without scanning, which is 0.755. This further reflects the effect on the overall CR if the number of communication signal paths increases.

Figures A.12 and A.13 illustrate a 4x4 matrix-matrix multiplier built by the 4-bit "X.MAC's" and the latches without scan design. Some of the details inside the MAC's shown in Figure A.12 have been omitted for a clearer view. Figure A.13 is a zoom view of Figure A.12 which shows the detailed tessellation of the MAC's and latches.

Let L, W, t are the length, width and propagation time of the n-bit MAC. Then, based on Figure A.12 and Equations 2-2 and 2-3, the area, A(L,W), and time, T(t), of a w_1-by-w_2 matrix-matrix multiplier for two NxN matrices having bandwidth of w_1 and w_2, are derived as

$$A(L,W) = [w_1(L+33)+33][w_2(W+33)+33]\ \mu m^2, \qquad (6\text{-}2)$$

and

$$T(t) = [3N+\min(w_1,w_2)][t+0.48]\ \text{nsec}. \qquad (6\text{-}3)$$

Only the length of the latch (33 μm) was used in the calculation of the total area in the above equation. This is because of the latches placement shown in Figure A.13.

The above equation also implies that the design performance of the matrix-matrix multiplier depends mainly on the MAC since the latch arrays are comparatively insignificant in both area and time if n is large. For example, if a 32-bit "X.MAC" is used, the length of the latch (33 μm) is 1.0% and 2.3% of the length and width of the "X.MAC" (3412 μm and 1437 μm). Likewise, the CR contributed by the latch will

not have a significant effect on the CR of the MAC. The area-time parameter and CR of the matrix-matrix multiplier are thus assumed to be equal to those of the corresponding MAC, for n-->∞.

Table 6.4 and Figure 6.2 summarize the results obtained from the bottom-up approach to the design of various target modules.

Table 6.4 Area-time and compactness ratios of various target modules.

Target module	Entry point	Design path	Area-time ratio (ATR)	compactness ratio (CR)
Type-A FA	X	X.FA	1.00	0.755
	G	G.FA	1.17	0.686
	FA	FA	1.36	0.644
Type-B FA	X	X.FA	1.00	0.751
	G	G.FA	1.17	0.684
	FA	FA	1.31	0.671
MAC	X	X.MAC	1.00	0.69
		X.FA.MAC	1.11	0.62
	G	G.MAC	1.14	0.65
		G.FA.MAC	1.23	0.59
	FA	FA.MAC	1.40	0.57
VLSI	X	X.MAC.VLSI	1.00	0.69
		X.FA.MAC.VLSI	1.11	0.62
	G	G.MAC.VLSI	1.14	0.65
		G.FA.MAC.VLSI	1.23	0.59
	FA	FA.MAC.VLSI	1.40	0.57

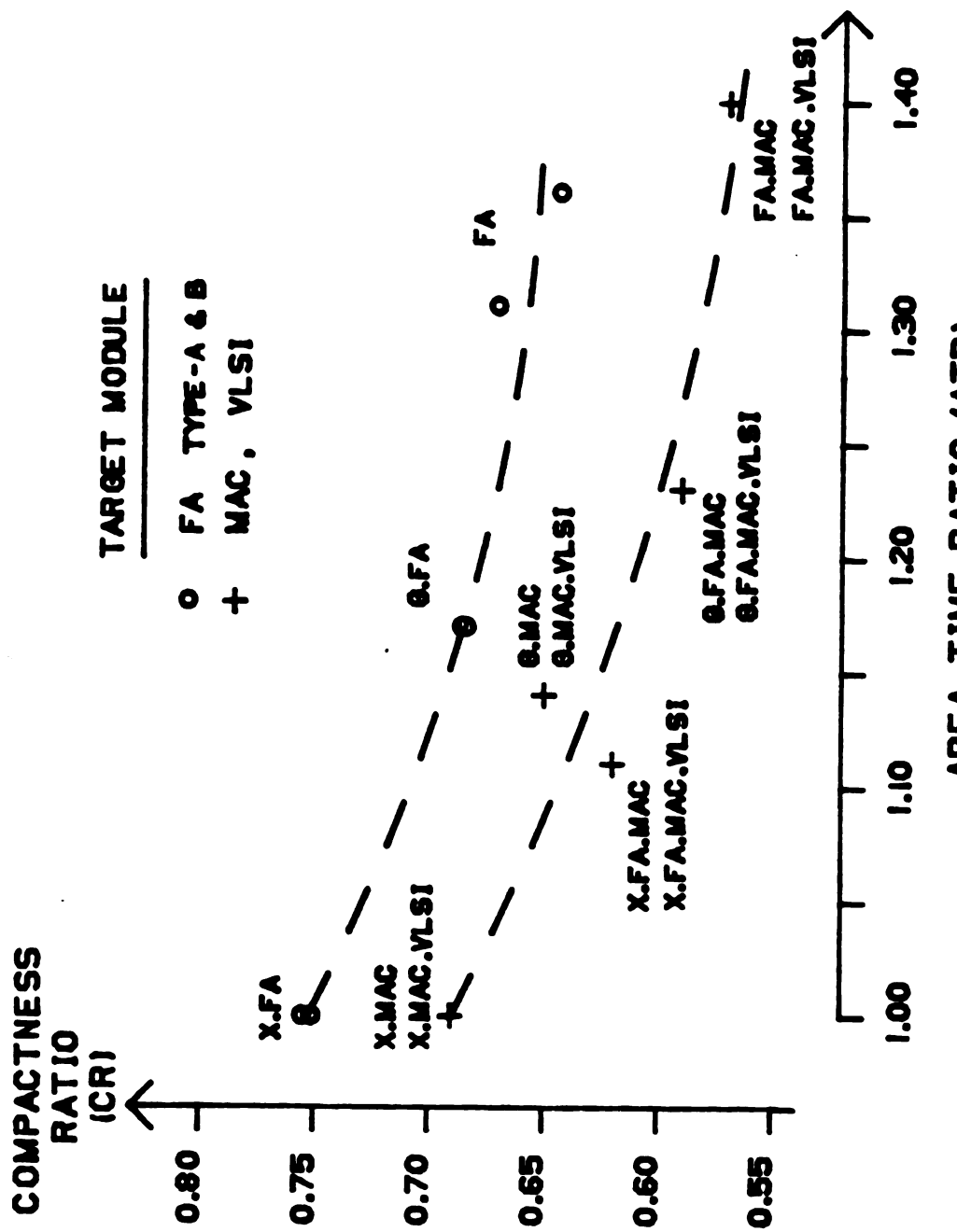

Figure 6.2 Compactness ratio versus area-time ratio.

Significantly, the graph in Figure 6.2 clarifies that the CR's of the target modules using low-level design entry-point are greater than those of the modules using high-level design entry-point. However, the ATR of the target modules using higher design entry-points are larger, representing poorer area utilization and circuit performance. The figure shows that these tradeoffs are also true for the one-path and multi-path modules. For example, selecting transistor level as design entry-point, the CR of the "X.MAC" is greater than that of the "X.FA.MAC" but the ATR of the "X.MAC" is the smaller of the two.

The CR and ATR of the target MAC are same as those of the final VLSI multiplier. This implies that the average spacing among the submodules inside the MAC's is same as that among the MAC modules. That is, using the notation as derived in Section 4.2.2, s_{MAC} may be equal to s_{FA}, $s_{G.FA}$, s_{G}, $s_{X.FA}$, or s_{X} depending on the target MAC used.

6.2 I/O Bottlenecking Index

Using the top-down approach presented in Chapter 5, the tradeoff of various physical partitions to the array structure and pin limitations can be observed. The I/O bottlenecking index, BI, as defined in Equation 5-7 and the number of pinouts per chip for data I/O, $2N_s n$, are used to evaluate the effect of various decompositions to the array of MAC's and latches.

The total chip area of a matrix-matrix multiplier is a function of R as shown in Equation 5-7. Near term IC lithographic and chip size limitations cap the size of R per chip. Therefore, a parameter of the

chip size limitation, in terms of the lithographic linewidth, λ, can be incorporated into the graph used in Figure 5.5 in Section 5.2. This is done as follows.

Let λ=1 μm and n=32 bits, then Equations 6-1 to 6-3 imply that the total area and pipeline segment time of the five different matrix-matrix multipliers are expressed as

$$\begin{aligned}
&A(X.MAC) = (3445R+33)(1470R+33)\ \mu m^2,\\
&t_{PE}(X.MAC) = 132.83\ nsec;\\
&A(X.FA.MAC) = (3776R+33)(1600R+33)\ \mu m^2,\\
&t_{PE}(X.FA.MAC) = 135.14\ nsec;\\
&A(G.MAC) = (3679R+33)(1545R+33)\ \mu m^2,\\
&t_{PE}(G.MAC) = 151.83\ nsec;\\
&A(G.FA.MAC) = (3910R+33)(1670R+33)\ \mu m^2,\\
&t_{PE}(G.FA.MAC) = 155.02\ nsec;\\
&A(FA.MAC) = (4211R+33)(1799R+33)\ \mu m^2,\\
&t_{PE}(FA.MAC) = 169.80\ nsec. \qquad (6\text{-}4)
\end{aligned}$$

Note that the propagation time of the latch, 0.48 nsec, is included in t_{PE}.

The propagation time of the DMUX and MUX I/O circuits must also be determined. Three I/O circuit candidates specific for single chip VLSI systolic arrays have been presented and evaluated in [9]. A set of input and output circuits, known as a "data controlled" input circuit and a "shift-register control sequence" output circuit, are selected from these candidates to apply to this work. Both of the selected I/O circuits are pipelined in nature and are thus well suited for the design

of systolic structures. Through simulation, the worst cast propagation time of these I/O circuits is found to equal $t_D(1:M)=3.6$ nsec, for $\lambda=1$ μm [5, 9]. The total area of the I/O circuits is significantly small (less than 0.1% of the overall chip area) and is neglected [5, 9].

Table 6.5 and Figure 6.3 present the comparison among the I/O bottlenecking index, pinout and chip size limitations. In Table 6.5, t_B was calculated by dividing $t_D(1:M)=3.6$ nsec by the corresponding t_{PE} as listed in Equation 6-4. The chip area and BI were found by using Equations 5-9 and 6-4.

All values of BI in Table 6.5 are well under 1.0 indicating no I/O bottleneck. However, some of target arrays have a chip size greater than the current standard chip size of 1 cm^2 [55]. This means that the number of MAC's on a single chip is more seriously constrained by the chip size limitation than the I/O bottleneck.

The index of t_B can provide a quick insight into the potential for encountering an I/O bottleneck. A ratio of less than 0.05 can be reasonably assumed as an index threshold unlikely to have an I/O problem.

The number of bits per word (n) also affects the chip size as well as the propagation time (See Equation 6-1). If n decreases, the overall chip size and t_{PE} decrease implying that R may increase and t_B will increase as well. Although BI does not directly depend on n as shown in Equation 5-9, BI is directly proportional to R and t_B. Therefore, there is a chained relationship among BI, R, t_B and n.

It must be emphasized that Figure 6.3 does not provide the direct relationship between BI and n. It indicates that BI is inversely

Table 6.5 Comparison of I/O bottlenecking index, pin limitation and chip area.

Modularized array	Parameter* t_B			R = 3	R = 4	R = 5
X.MAC.VLSI	0.027		Area(cm^2)	0.461	0.817	1.274
		BI	N_s=1	0.297	0.405	0.513
			N_s=2	0.149	0.203	0.257
			N_s=3	0.099	0.135	0.171
X.FA.MAC.VLSI	0.027		Area(cm^2)	0.549	0.974	1.519
		BI	N_s=1	0.297	0.405	0.513
			N_s=2	0.149	0.203	0.257
			N_s=3	0.099	0.135	0.171
G.MAC.VLSI	0.024		Area(cm^2)	0.517	0.916	1.430
		BI	N_s=1	0.264	0.360	0.456
			N_s=2	0.132	0.180	0.228
			N_s=3	0.088	0.120	0.152
G.FA.MAC.VLSI	0.023		Area(cm^2)	0.593	1.052	1.642
		BI	N_s=1	0.253	0.345	0.437
			N_s=2	0.127	0.173	0.219
			N_s=3	0.084	0.115	0.146
FA.MAC.VLSI	0.021		Area(cm^2)	0.688	1.220	1.904
		BI	N_s=1	0.231	0.315	0.399
			N_s=2	0.116	0.158	0.200
			N_s=3	0.077	0.105	0.133

* Based on n=32 bits and λ=1 μm.

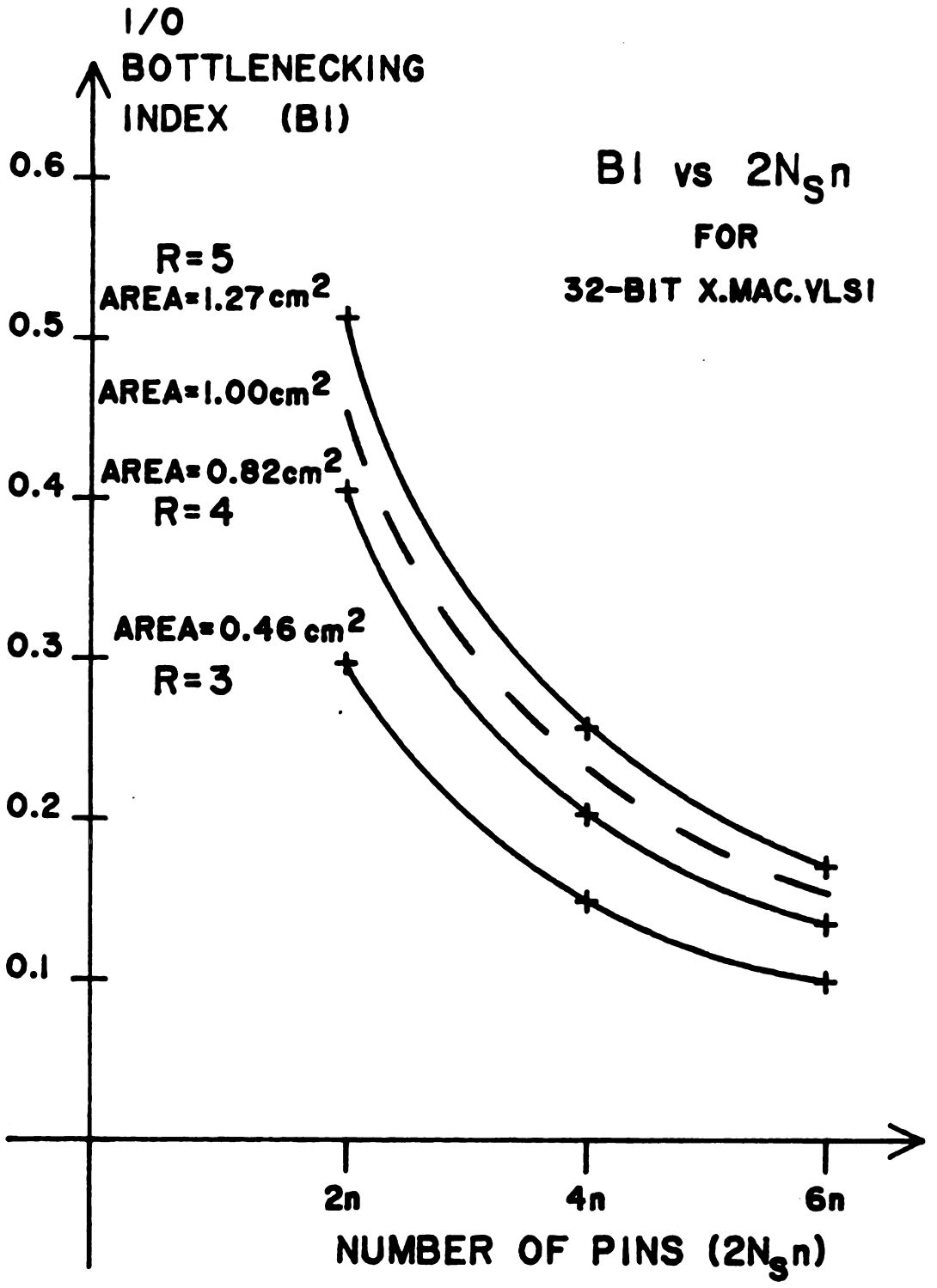

Figure 6.3 BI versus $2N_s n$ for 32-bit "X.MAC.VLSI".

related to the number of pairs of n pinouts for data I/O (N_s). Note that the graph in the figure uses the data of "X.MAC.VLSI" from Table 6.5 for illustration. The limitation of current standard chip size of 1 μm^2 is projected on the graph by a dotted curve between the curves of R=4 and 5. This shows, for instance, that the maximum number of 32-bit "X.MAC's" that can be implemented on a single 1 cm^2 chip is about 20.

To further show directly the relationship between BI and n consider Figure 6.4. All the parameters shown in Figure 6.4 are the same as those in Figure 6.3 except that n=16 bits. In comparing both figures, BI is seen to be inversely proportional to n if all other parameters remain unchanged. Figure 6.4 also shows an example of a minor I/O bottleneck where BI=1.026 for R=5 and N_s=1. To resolve this problem, select R to be less than 5 or N_s to be more than 1. Alternatively, designing faster I/O DMUX/MUX circuits can also alleviate such a problem.

In summary, the tradeoffs among the I/O bottlenecking potential, numbers of pinouts, word size and chip size limitation have been illustrated in the graphs developed in the top-down approach. Specifically, the I/O bottlenecking index, BI, is inversely proportional to the number of pairs of n pinouts, N_s, which is dictated by the I/O multiplexing scheme and current pin limitation. On the other hand, BI directly relates to the number of PE's on a single chip which is constrained by the word size and chip size limitation in terms of λ.

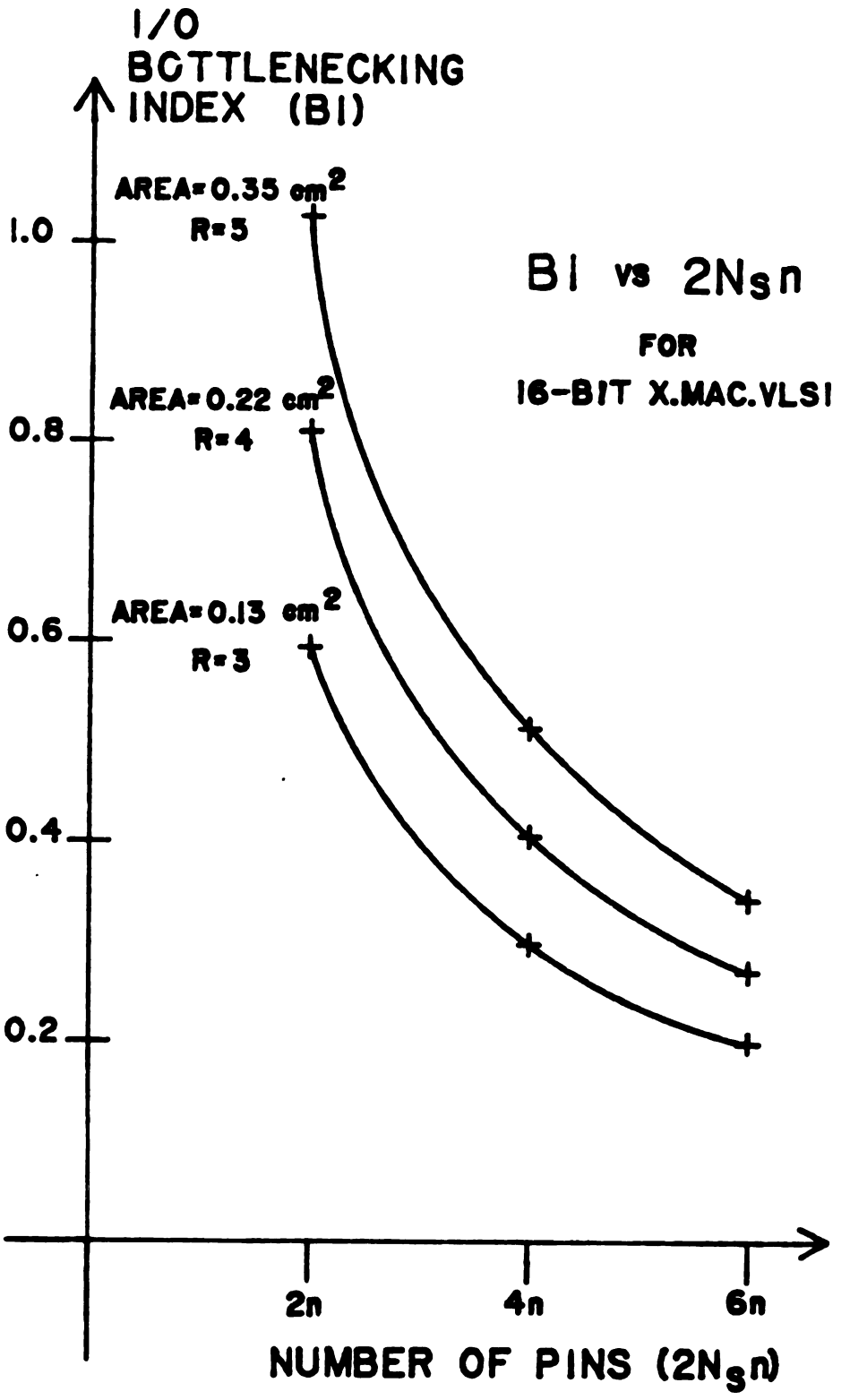

Figure 6.4 BI versus $2N_s n$ for 16-bit "X.MAC.VLSI".

6.3 Contributions

Tradeoffs involved in the bottom-up approach to the hierarchical design of regular VLSI-based structures are parameterized by the design complexity, chip area and circuit performance. A generalized model using infinite (unbounded) modular arrays was presented in Section 4.2.3. Based on this model, the general recursive expressions for the compactness ratio (a measure of complexity), chip area and propagation time were derived. These general expressions reveal that a tradeoff among the compactness ratio and area-time parameter exists within the multi-levels of the design hierarchy. The compactness ratio was found to vary among the one-path and multi-path modules. The area-time parameter was shown to be inversely proportional to the compactness ratio. This is specifically shown in Equations 4-23, 4-24 and 4-28. The use of a systolic array structure as a testbed further verifies the inverse relationship between the compactness ratio and the area-time parameter as the design entry-point and pathways move upward along the design hierarchy.

Testability of VLSI circuits using scan design has been of special interest. A dynamic latch with scanning ability was designed and specifically applied to systolic testbed structure. This latch required 39% more chip area compared to a standard latch without scan design. The propagation time of the latch with scan design had an increase of only 8.3% which had negligible effect on the overall performance of the

array structure.

The top-down approach to the various decompositions of regular array structures were parameterized by an I/O bottlenecking index. The I/O bottlenecking of operands tends to increase as the number of sets of pinouts for data I/O and the word size in number of bits decrease. Results from the testbed systolic structure indicate that the current limitation of the number of PE's on a single standard chip is constrained more by the chip size and lithographic linewidth than by the I/O bottleneck problem.

As a whole, the bottom-up and top-down approaches to the hierarchical design of regular structures presented in this research are summarized in Figure 6.5. According to this figure, a designer can choose an appropriate design entry-point with respect to the desired tradeoff parameters of the final design. For example, the tradeoff parameters derived in this work are area-time versus circuit complexity and I/O bottlenecking index versus pin limitation. Once the design entry-point has been selected, the design path can proceed either back or forth along the design hierarchy, representing a top-down or bottom-up approach, as dictated by desired performance parameters.

Finally, by modular tessellation, the IC layout of the regular structure is obtained. This layout can be implemented on either a single chip or a set of multiple chips depending on the size of the system as well as the I/O limitation. However, if the layout requires modification or redesign, for reasons such as being less optimal than expected or having an intolerable I/O bottleneck, a new top-down or bottom-up approach may be required.

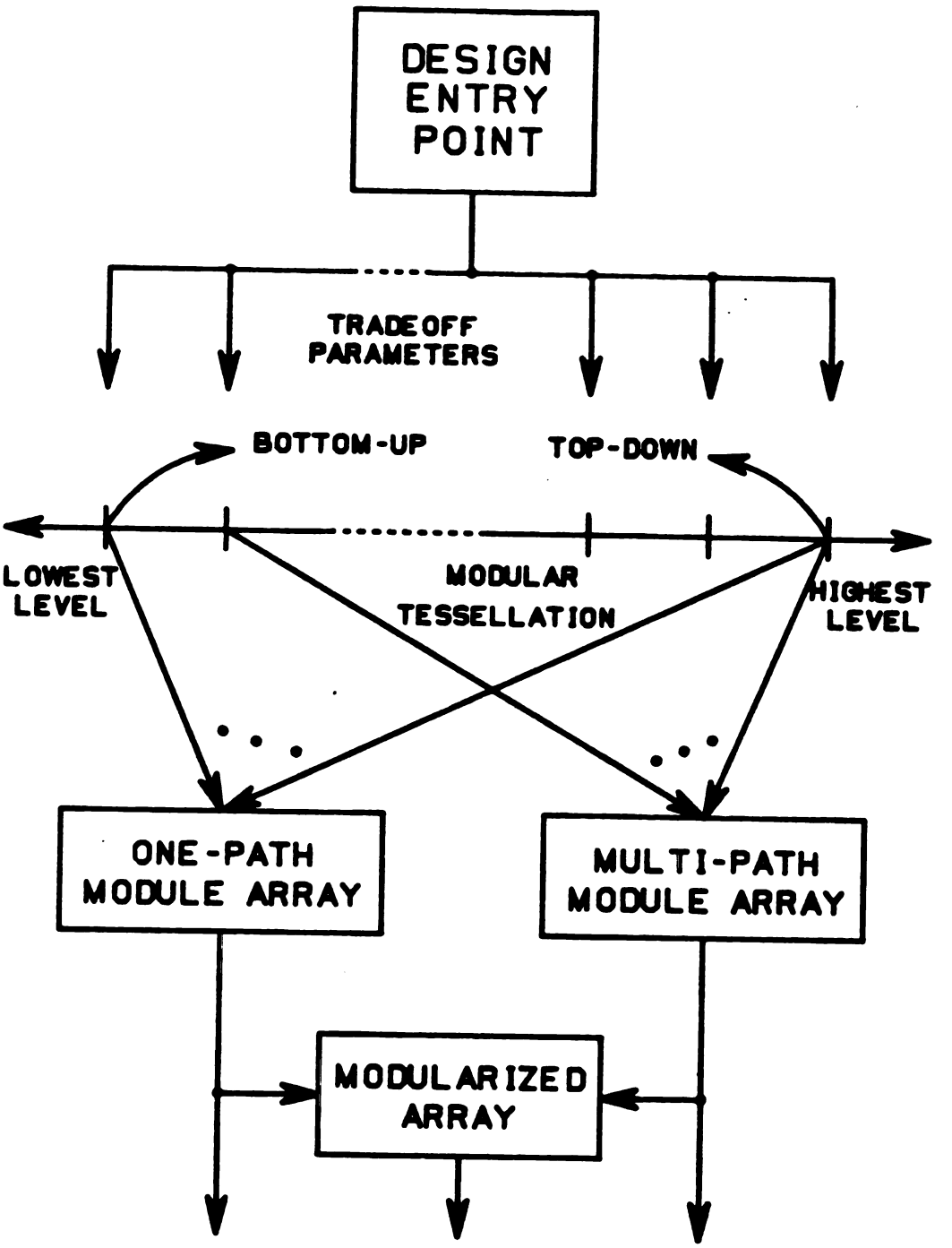

Figure 6.5 A summary of bottom-up and top-dwon approaches.

6.4 Trends in VLSI and Future Study

As the density of digital IC circuits grows, the design-time, complexity, difficulties in simulation and testing, mask generation and fabrication, and percentage of yield loss tend to increase tremendously. Designs using silicon compilation, semi-custom approaches, symbolic layout techniques or hierarchical design currently provide a fast design turnaround for highly complex and dense circuits.

The performance tradeoffs in the hierarchical design of regular structures have studied in this dissertation. However, tradeoffs in the design of random structures can also be examined based on the principles and techniques derived in this work. The average compactness ratio of a random circuit can be obtained by proportionally averaging the compactness ratios of all regular and non-regular sub-modules. If the compactness ratio is used to relate to the design time, a "regularity factor" must be found and incorporated into the compactness ratio of the regular sub-modules [44, 45]. This regularity factor can be determined either experimentally or statistically [45, 53]. Likewise, the area-time parameters of a random circuit can also be derived using heuristic or statistical models [53, 56]. The evaluation and derivation of these regularity factors and models for all VLSI structures are thus essential next steps.

The recent development of design frames shows great promise in simplifying the design of VLSI-based systems [57, 58]. A design frame is

a hardware frame that contains all the details of I/O circuits, drivers, power supplies or other interface counterparts. A standard design frame can be used to interface with any new circuit design. Whenever a new circuit is designed, it can be directly inserted into a design frame for fabrication.

As an approach to the design of regular structures as studied in this dissertation, a designer can select a design frame to match the needs of the array. For the bottom-up approach, the designer can concentrate more on the design of the array structure giving less concern to the I/O or other interfacing problems. For the top-down approach, the designer can use the data of the design frame, such as the bandwidth size and propagation delay of the DMUX and MUX circuits, to find a suitable way to decompose the array structure.

However, the design of a standard design frame for systolic array structures is not a simple matter because the required I/O bandwidth is greatly dependent on the size of the array. The size of the drivers for I/O also varies with the number of PE's. What's more, the design frame must be compatible with standard IC package constraints [59, 60].

Another recent development in VLSI is wafer-scale integration (WSI) circuits. The idea behind a WSI approach is to assemble an entire system or circuit structure on a single wafer [61-63]. There is virtually no pin limitation inside the wafer. Therefore, WSI is ideally suited for the implementation of regular structures, especially systolic array structures. For example, all MAC's and latches can simply be laid out on a single wafer and joined directly by interconnecting lines. Assuming a silicon wafer with 4-inch diameter is used to implement the

matrix-matrix systolic multiplier designed in this research, as many as 1600 32-bit MAC's with latches can be put on this wafer. This implies that the multiplication of any matrix having a dimension up to 40 can possibly be done on one wafer.

The current major problem with WSI is that some of the cells, such as MAC's, are likely to be defective. A high percentage of defective cells will render the wafer useless. It is very costly to discard the wafer if some of the cells are bad. To overcome this problem, current research is aimed at designing restructurable wafer-scale arrays or using redundancy [63].

Future work relating the WSI approach to the design of regular structure described in this research has two trends. The first one is to study the tradeoff between the circuit complexity and the yield of defective cells [64-66]. That is, designs using different design entry-points and pathways will have different percentages of defective cells. Secondly, the multiple levels of regular structure can be further extended to the wafer-wise modularity. This extended level will become the wafer module level in between the PE and algorithmic levels.

On the whole, current trends project that design methodologies using design frames and wafer-scale integration show a great promise in VLSI circuit design. And, these new frontiers will eventually allow any VLSI algorithmic structure to be directly implemented on a silicon or GaAs wafer without any concern of density or I/O constraints.

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Niessen, C., "Hierarchical Design Methodologies and Tools for VLSI Chips," *Proc. IEEE*, Vol. 71, No. 1 (January 1983), pp. 66-75.

2. Wallich, P., "The One-Month Chip: Design," *Spectrum*, Vol. 21, No. 9 (September 1984), pp. 30-34.

3. Kung, H. T., "The Structure of Parallel Algorithms," *Advances in Computers*, Vol. 19, Academic Press, New York (1980), pp. 65-112.

4. Hwang, K. and Cheng, Y. H., *VLSI Arithmetic Arrays and Modular Networks for solving Large-Scale Linear System of Equations*, Tech. Report No. TR-EE 80-4, Purdue University, W. Lafayette, Indiana (March 1980).

5. Leung, Y.-Y. J. and Shanblatt, M. A., *A VLSI Systolic Array for Matrix Triangulation in Load Flow Analysis*, Tech. Report No. MSU-ENGR-83-003, Michigan State Univ., East Lansing, MI (January 1983).

6. Mead, C. and Conway, L., *Introduction to VLSI Systems*, Addison-Wesley Pub. Co., Reading, Massachusetts (1980).

7. Zhang, C. N. and Yun, D. Y. Y., "Multi-Dimensional Systolic Networks for Discrete Fourier Transform," *Proc. IEEE 11th Annual Int'l Sym. on Computer Architecture* (June 1984), pp. 215-222.

8. Fortes, J. A. B., Fu, K. S. and Wah, B.W., "Systematic Approaches to the design of Algorithmically Specified Systolic Arrays," *Proc. 1985 Int'l Conf. on Acoustics, Speech, and Signal Processing* (March 1985), pp. 300-303.

9. Hsu, W. C. and Shanblatt, M. A., *Evaluation of a Single VLSI Chip Algorithm for Triangulating Large Band Form Matrices*, Tech. Report No. MSU-ENGR 82-015, Michigan State University, East Lansing, Michigan (August 1982).

10. Liu, P. S. and Young, T.Y., "VLSI Array Design Under Constraint of Limited I/O Bandwidth," *IEEE Trans. on Computers*, Vol. C-32, No. 2 (December 1983), pp. 1160-1170.

11. Rose, F., "Application of a Hierarchical Layout System to VLSI," *Tech. Digest IEEE 1984 Int'l Conf. on CAD* (November 1984), pp. 125-127.

12. Reinhard, D. K., *Integrated Circuits Engineering*, Course Notes, Michigan State Univ., E. Lansing, MI (Winter 1985).

13. Eichelberger, E. B. and Williams, T. W., "A Logic Design Structure for LSI Testing," *Proc. 14th Design Automation Conf.* (June 1977), pp. 462-468.

14. Masui, T., Niimi, F. and Iwase, M., "A New Approach to Design for Testability in an LSI Logic Synthesis System," *Tech. Digest IEEE Int'l Conf. on CAD* (November 1985), pp.105-107.

15. Moore, G. E., "Progress in Digital Integrated Electronics," *Tech. Digest IEEE 1975 Int'l Electron Devices Meeting* (December 1975), pp. 11-13.

16. Wallich, P. "A Review of Engineering Workstations," *IEEE Spectrum*, Vol. 21, No. 10 (October 1984), pp. 48-53.

17. Avenier, J. P., "Digitizing, Layout, Rule Checking - The Everyday Tasks of Chip Designers," *Proc. IEEE*, Vol. 71, No. 1 (January 1983), pp. 49-56.

18. Chen, C. F. et al, "The Second Generation MOTIS Mixed-Mode Simulator," *Proc. 21st Design Automation Conf.* (June 1984), pp. 10-17.

19. Butt, H. H. and El-zig, Y. M., "Impact of Mixed-Mode Self-Test on Life Cycle Cost of VLSI Based Designs," *Proc. IEEE 1984 Int'l Test Conf.* (October 1984), pp. 338-347.

20. De Man, H. et al, "A Debugging and Guided Simulation System for MOSVLSI Design," *Tech. Digest IEEE 1983 Int'l Conf. on CAD* (September 1983), pp. 137-138.

21. Wong, Y., "Hierarchical Circuit Verification," *Proc. 22nd Design Automation Conf.* (June 1985), pp. 695-701.

22. Daniel, M. E. and Gwyn, C. W., "CAD Systems for IC Design," *IEEE Trans. on CAD of IC and Sys.*, Vol. CAD-1, No. 1 (January 1982), pp. 2-12.

23. Hughes, S. C., Lewis, D. B. and Rimkus, C. J., "A Technique for Distributed Execution of Design Automation Tools," *Proc. 22nd*

Design Automation Conf. (June 1985), pp. 23-30.

24. Collett, R., "Designer's Guide to Semi-Custom IC's," *Digital Design* (June 1984), pp. 64-85.

25. Kessler, A. J. and Ganesan A., "Standard Cell VLSI Design: A Tutorial," *IEEE Circ. and Devices Magazine*, Vol. 1, No. 1 (January 1985), pp. 17-34.

26. Barna, A., *VHSIC Technologies and Tradeoffs*, Wiley-Interscience, New York (1981), pp. 19-106.

27. Mavor, J., Jack, M. A. and Denyer, P. B., *Introduction to MOS LSI Design*, Addison-Wesley Pub. Co., Reading, Massachusetts (1983), pp. 18-162.

28. Werner, J., "Progress Toward the 'Ideal' Silicon Compiler," *VLSI Design*, Vol. 4, No. 6 (September 1983), pp. 38-41.

29. Southard, J. R., "MacPitts: An Approach to Silicon compilation," *Computer*, Vol. 16, No. 12 (December 1983), pp. 74-82.

30. Crawford, J. D., "EDIF: A Mechanism for the Exchange of Design Information," *Proc. IEEE 1984 Custom IC Conf.* (May 1984), pp. 446-449.

31. Duyn, B. and Trimberger, S., "Structured-Design System Takes Over the Complexities in VLSI Circuits," *Electronics*, Vol. 56 (August 25, 1983), pp. 127-130.

32. vanCleemput, W. M., "An Hierarchical Language for the Structural Description of Digital Systems," *Proc. 14th Design Automation Conf.* (June 1977), pp. 377-385.

33. Curtis, W., "Re-Implementing the MicroVAX I on the GENESIL," *Digest IEEE 1985 Compcon Spring* (February 1985), pp. 132-136.

34. Salsbury, P. J., "Nonvolatile Memories," *Tech. Digest IEEE 1984 Int'l Solid-State Circ. Conf.* (February 1984), p. 135.

35. Fiebiger, J., "C-MOS: A Designer's Dream with the Best Yet to Come," *Electronics*, Vol. 57 (April 5, 1984), pp. 113-115.

36. Sangiorgi, E. C. et al, "Two-Dimensional Numerical Analysis of Latchup in a VLSI COMS Technology," *IEEE Trans. on CAD of IC and Sys.*, Vol. CAD-4, No. 4 (October 1985), pp. 561-574.

37. Newkirk, J. A. and Mathews, R., *The VLSI Designer's Library*, Addison-Wesley Pub. Co., Reading, Massachusetts (1983).

38. Whitaker, S., "Pass-Transistor Networks Optimize n-MOS logic,"

Electronics, Vol. 56 (September 22, 1983), pp. 144-148.

39. Williams, T. W. and Parker, K. P., "Design for Testability - A Survey," *Proc. IEEE*, Vol. 71, No. 1 (January 1983), pp. 98-112.

40. McCluskey, E. J., "VLSI Design for Testability," *Tech. Digest 1984 Sym. on VLSI Technology* (September 1984), pp. 2-5.

41. Gajski, D. D. and Kuhn, R. H., "Guest Editor's Introduction: New VLSI Tools," *Computer*, Vol. 16, No. 12 (December 1983), pp. 11-14.

42. Baugh, C. R. and Wooley, B. A., "A Two's Complement Parallel Array Multiplication Algorithm," *IEEE Trans. on Computers*, Vol. C-22, No. 12 (December 1973), pp. 1045-1047.

43. Hwang, K., *Computer Arithmetic*, John Wiley and Sons, New York (1979), pp. 85-254.

44. Fey, C. F., "Custom LSI/VLSI Design Manpower Model," *Proc. IEEE 1984 Custom IC Conf.* (May 1984), pp. 246-250.

45. Fey, C. F., "Custom LSI/VLSI Chip Design Productivity," *IEEE Journal of Solid-State Circ.*, Vol. SC-20, No. 2 (April 1985), pp. 555-561.

46. Broers, A. N., "Practical Methods for Sub-Micron Lithography," *Microcircuit Engineering 84*, edited by Heuberger, A. and Beneking, H., Academic Press, New York (1985), pp. 3-20.

47. Hamilton, D. J. and Howard, W. G., *Basic Integrated Circuits Engineering*, McGraw-Hill, New York (1975), pp. 515-541.

48. Wagner, K. D. and McCluskey, E. J., "Effect of Supply Voltage on Circuit Propagation Delay and Test Applications," *Tech. Digest IEEE 1985 Int'l Conf. on CAD* (November 1985), pp. 42-44.

49. Jouppi, N. P., "Timing Analysis for nMOS VLSI," *Proc. 20th Design Automation Conf.* (June 1983), pp. 411-418.

50. Sequin, C. H., "Managing VLSI Complexity: An Outlook," *Proc. IEEE*, Vol. 71, No. 1 (January 1983), pp. 149-166.

51. Blanks, J. P., "Near-Optimal Quadratic-Based Placement for a Class of IC Layout Problems," *IEEE Circ. and Device Magazine*, Vol. 1, No. 5 (September 1985), pp. 31-37.

52. El Gamal, A. and Syed, Z., "A New Statistical Model for Gate Array Routing," *Proc. IEEE 20th Design Automation Conf.* (June 1983), pp. 671-674.

53. Shragowtiz, E. and Desmonds, D., "Statistical Methods in

Hierarchical Design of VLSI," *Proc. IEEE Int'l Conf. on Computer Design: VLSI in Computers* (October 1984), pp. 468-473.

54. Okabayashi, H., "Multilevel Metallization Technology for VLSIs," *Tech. Digest IEEE 1984 Sym. on VLSI Technology* (September 1984), pp. 20-23.

55. Keyes, R. W., "The Evolution of Digital Electronics Towards VLSI," *IEEE Journal of Solid-State Circ.*, Vol. SC-14, No. 2 (April 1979), pp. 193-201.

56. Feuer, M., "Connectivity of Random Logic," *IEEE Trans. on Computers*, Vol. C-31, No. 1 (January 1982), pp. 29-33.

57. Katz, R. H., Bell, A. G. and Conway, L., "VLSI System Design by the Numbers," *IEEE Spectrum*, Vol. 22, NO. 2 (February 1985), pp. 44-50.

58. Bell, A. G., "Using System Kits and Design Frames to Enhance VLSI Prototyping," *Digest IEEE 1985 Compcon Spring* (February 1985), pp. 182-183.

59. Balde, J. W., "New Packaging Strategy to Reduce System Costs," *Proc. IEEE 1984 Int'l Conf. on Computer Design: VLSI in Computers* (October 1984), pp. 579-584.

60. Mahalingam, M., "Thermal Management in Semiconductor Device Packaging," *Proc. IEEE*, Vol. 73, No. 9 (September 1985), pp. 1396-1404.

61. McDonald, J. F. et al, "The Trials of Wafer-Scale Integration," *IEEE Spectrum*, Vol. 21, No. 10 (October 1984), pp. 32-39.

62. Johnson, R. R., "The Significance of Wafer Scale Integration in Computer Design," *Proc. IEEE 1984 Int'l Conf. on Computer Design: VLSI in Computers* (October 1984), pp. 101-105.

63. Raffel, J. I. et al, "A Wafer-Scale Digital Integrator Using Restructurable VLSI," *IEEE Journal of Solid-State Circ.*, Vol. SC-20, No. 1 (February 1985), pp. 399-406.

64. Ferris-Prabhu, A. V., "Modeling the Critical Area in Yield Forecasts," *IEEE Journal of Solid-State Circ.*, Vol. SC-20, No.4 (August 1985), pp. 874-878.

65. Perry, S., Mitchell, M. and Pilling D., "Yield Analysis Modeling," *Proc. 22nd Design Automation Conf.* (June 1985), pp. 425-428.

66. Pichler, P. et al, "Simulation of Critical IC-Fabrication Steps," *IEEE Trans. on CAD of IC and Sys.*, CAD-4, No. 4 (October 1985), pp. 384-397.

APPENDIX

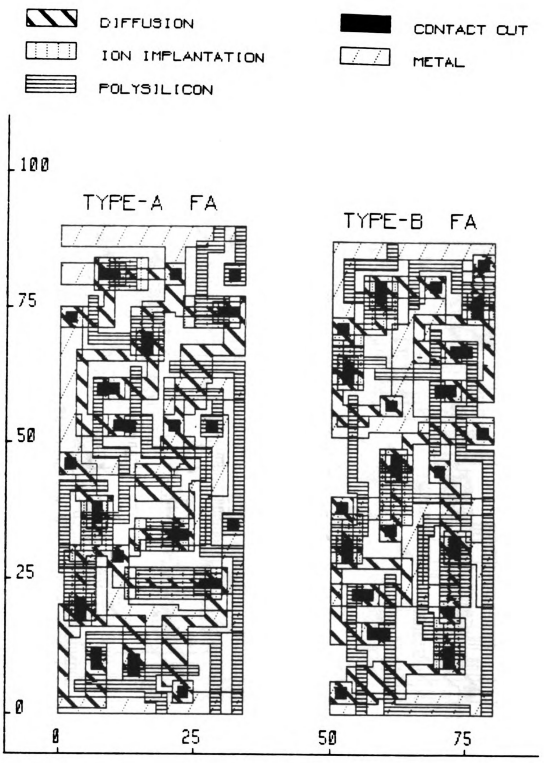

Figure A.1 Physical layouts of "X.FA".

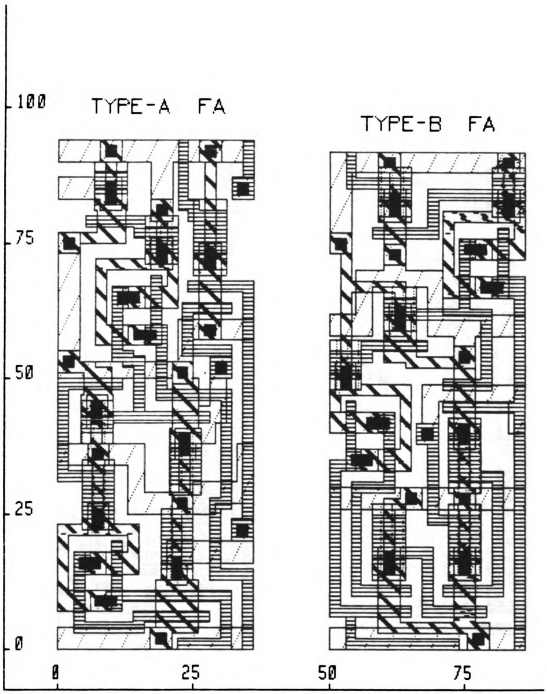

Figure A.2 Physical layouts of "G.FA".

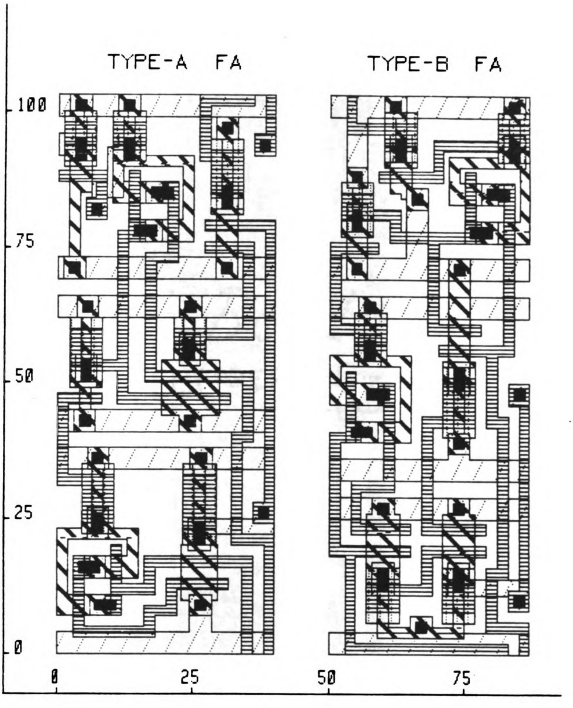

Figure A.3 Physical layouts of "FA".

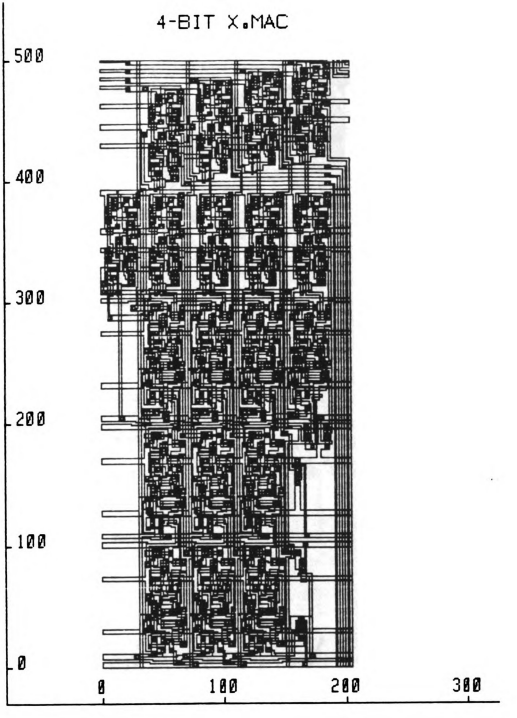

Figure A.4 Physical layout of a 4-bit "X.MAC".

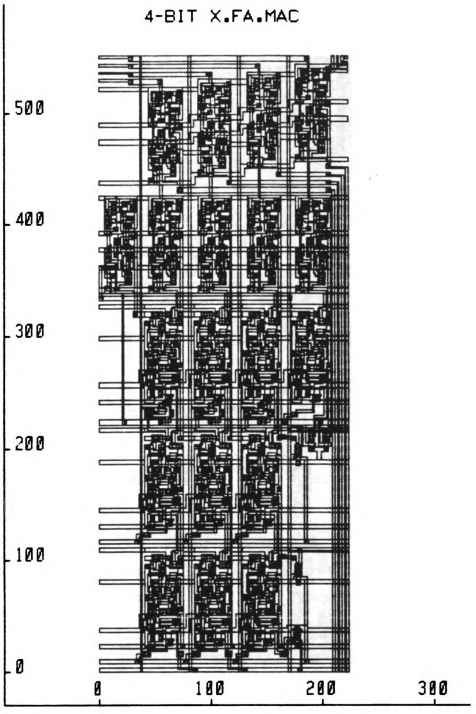

Figure A.5 Physical layout of a 4-bit "X.FA.MAC".

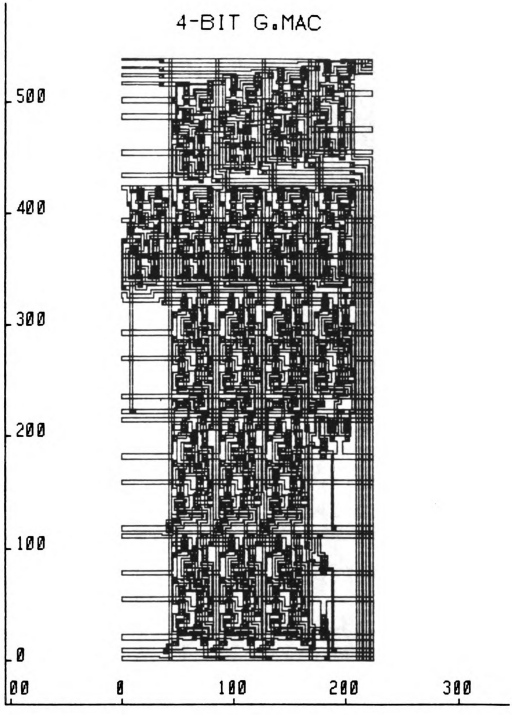

Figure A.6 Physical layout of a 4-bit "G.MAC".

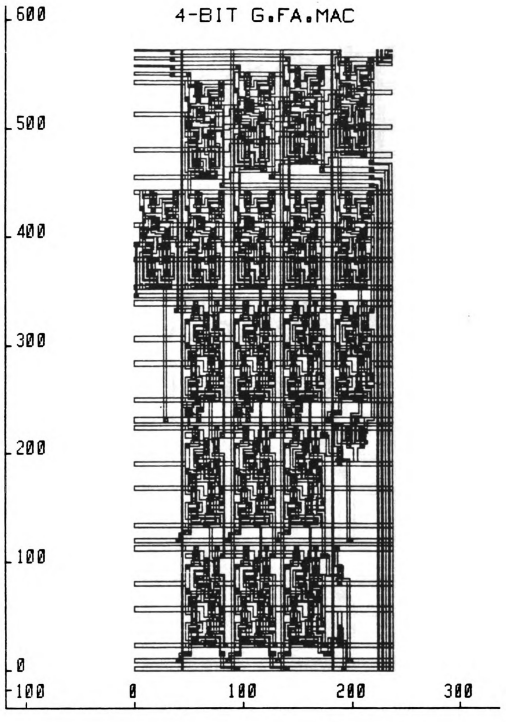

Figure A.7 Physical layout of a 4-bit "G.FA.MAC".

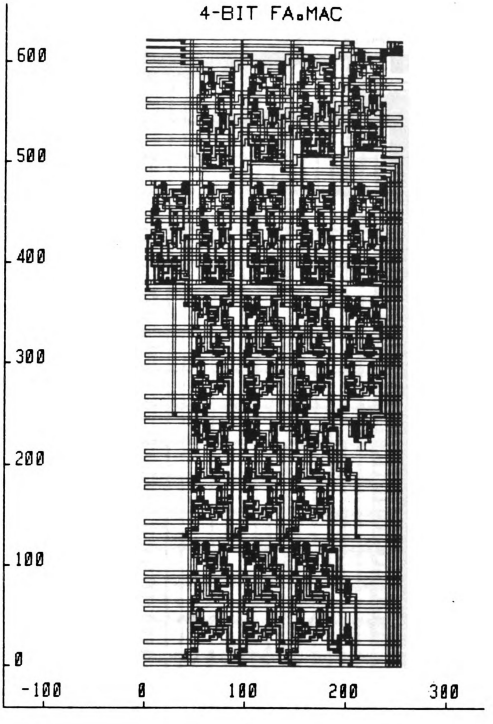

Figure A.8 Physical layout of a 4-bit "FA.MAC".

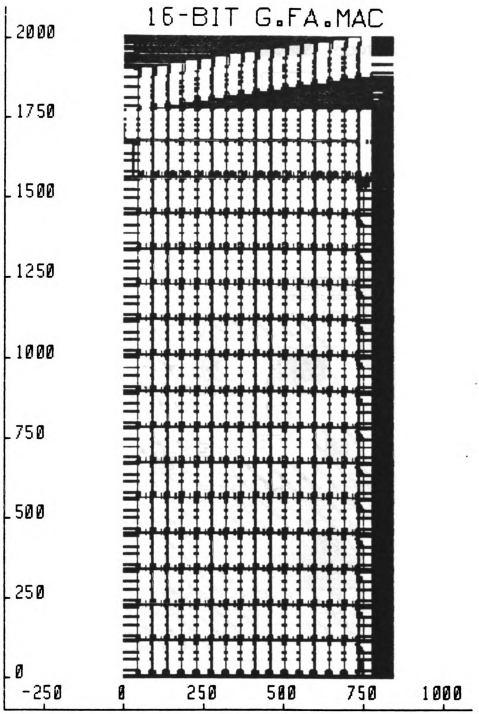

Figure A.9 Physical layout of a 16-bit "G.FA.MAC" without "G.FA's".

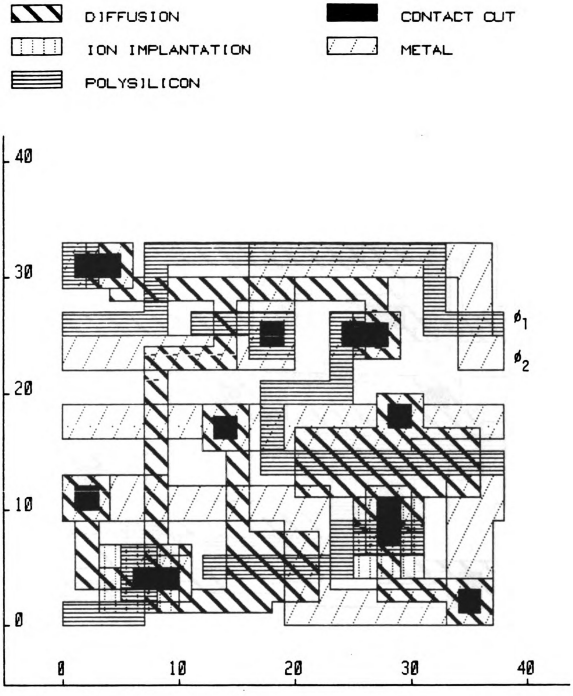

Figure A.10 Physical layout of a latch without scanning ability.

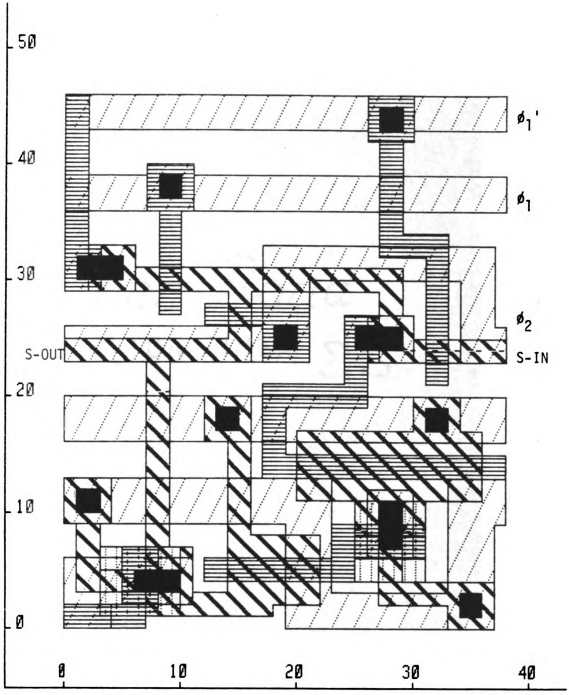

Figure A.11 Physical layout of a latch with scanning ability.

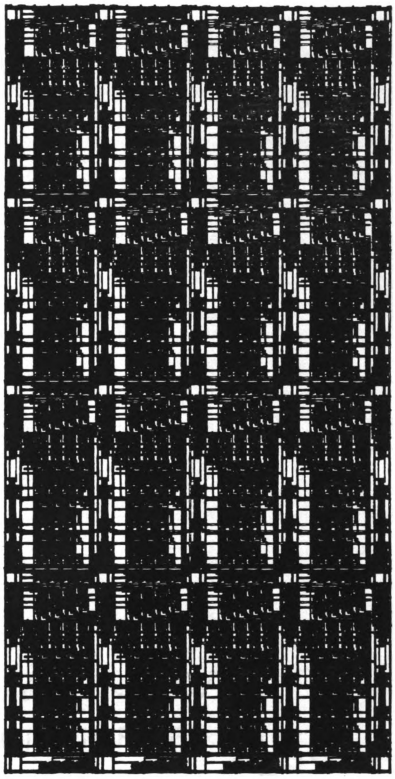

Figure A.12 Physical layout of a 4x4 matrix-matrix multiplier.

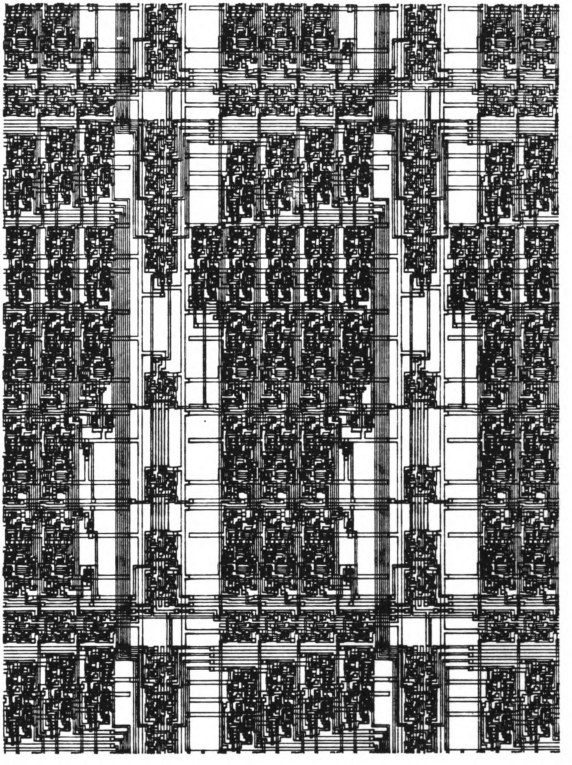

Figure A.13 A zoom view of Figure A.12.