

Design of Modular Digital Circuits for Testability

Andrew Kusiak, *Member, IEEE*, and Chun-Che Huang

Abstract—Modular products and reconfigurable testing processes are crucial in modern manufacturing. This paper discusses the concept of product modularity, test modules of increased reusability and exchangeability, and some aspects of design for testability. A methodology for design of modular products for testability in the presence of testing modules is developed. An integrated approach to design of modular products and test processes is discussed. The relationship between product modularity and design for testability is explored. This paper focuses on digital circuits which may be difficult to decompose into modules. The impact of testability on module sizes is considered. The testability points are identified at the circuit level while modular testing is considered at the board and system level. The main contribution of this paper is in the development of a formal approach to modularize products, and to assess the impact of modularity on the design process and testability.

Index Terms—Controllability points, modular products, modularity, observability points, testability.

I. INTRODUCTION

MODULAR products and reconfigurable testing processes with exchangeable test modules are crucial in agile manufacturing [1]. Modular product design allows to generate a large number of different products by creating combinations of modules and components providing each product with distinctive functionality, features, and performance levels [2]–[3]. Thus modular product design enhances *strategic flexibility* [4] by creating flexible designs allowing a company to respond to the changing markets and technologies by rapidly and inexpensively creating product variants derived from different combinations of the existing or new modules. The modular design principle supports also one of the most important rules of design for environment. The increasing complexity of new products and the proliferation of new fabrication and packaging technologies have results in more complex test software and hardware. Costs can no longer continue to be added to products through higher than necessary test programming time, test time, troubleshooting time, and high capital equipment cost. The partition of the test hardware/software into modular functions to increase the test resource reusability becomes a necessity.

Potential benefits of modularity include [5]: economy of scale; increased feasibility of product/component change; increased product variety; reduced order leadtime; decoupled risks; easier product diagnosis, maintenance; repair and test; and environmental friendliness.

Manuscript received May 1996; revised January 1997. This work was supported in part by National Science Foundation Grant DDM-9215259.

The authors are with the Intelligent Systems Laboratory, Department of Industrial Engineering, The University of Iowa, Iowa City, IA 52242-1527 USA.

Publisher Item Identifier S 1083-4400(97)04321-0.

Modular products refer to the products, assemblies and components that fulfill various functions through the combination of distinct building blocks (modules) [6]. *Basic components* refer to the components, subsystems, and mechanisms that interact with distinct modules resulting in different product variants. Based on the interactions within a product, three categories of modularity have been defined [5].

- 1) *Component-swapping modularity* occurs when two or more different *basic components* are paired with a module creating different product variants belonging to the same product family.
- 2) *Component-sharing modularity* is complementary to the component-swapping modularity. Various modules sharing the same *basic component* create different product variants belonging to different product families.
- 3) *Bus modularity* occurs when a module can be matched with any number of *basic components*. Bus modularity allows for varying the number and location of basic components in a product while component-swapping and component-sharing modularity allows only for the basic components to vary.

Numerous successful cases of launching modular designs have been reported in the literature. Over 160 creations of SONY Walkman products were leveraged by “mixing and matching” modular components of a few basic product designs [7]. Several upgraded models of SONY HandyCam video cameras were leveraged from an initial system design by successive introduction of improved modular components [7]. The principles of modular design have been applied to personal computers, e.g., Steffens [8], Gilder [9], and Langlois and Robertson [2]. However, the literature on product modularity has not received sufficient attention. Approaches to determine modules, represent modularity, optimize modular designs, and assess the impact of modularity on the design process, manufacturing, and management practices need to be explored.

To date, the growth in size and complexity have made testing of circuits difficult and expensive. Indeed, the test cost of electrical products is growing more rapidly than other components of the overall product cost [10]. Furthermore, minimizing the product development time requires the integration of design and test at early stages of the product development. In quantitative terms, testability is defined as a measure of the ease with which comprehensive test programs can be written and executed, as well as the ease with which faults can be isolated in defective components, subassemblies, and systems.

Benefits of design for testability are as follows [11], [12]:

- 1) improved product quality, availability, and acceptance;
- 2) decreased time to market;
- 3) decreased capital equipment investments for automatic test equipment;

- 4) decreased production test, field service, and fault isolation;
- 5) eliminated test bottlenecks;
- 6) reduced design verification time;
- 7) reduced organizational strife between design, test, and management;
- 8) increased liability of an electronics manufacturer;
- 9) increased market share and profitability.

The design for testability at the circuit level has been discussed in numerous papers, e.g., combinational circuits [13] and sequential circuits [14]. Methods of adding inputs, outputs, and gates to a general circuit to make it completely testable by a small number of tests were presented in Hayes [15] and Hayes and Friedman [16]. However, the design of modular products for testability at a circuit level has not been discussed in the literature.

This paper focuses on digital circuits, e.g., logic chain circuits. The test level here refers to the circuit level at the identification phase of testability points, and to the board and system level of the modular testing phase.

In this paper, the concept of modularity is applied to the development of products and testing. The relationship between the design of modular products, and testing the products in modular tests is explored. A methodology for the design of modular products for testability is developed. An integrated system for design of modular products and test processes is presented. The remainder of the paper is organized as follows: Section II considers modeling the product modularity and presents a methodology for determining product modularity with the consideration of testability. Section III discusses modular testing and an integrated system for design of products. Section IV considers the interaction between product modularity and design for testability. Section V concludes the paper.

II. PRODUCT MODULARITY

In this section, modular products are represented with matrices. The matrix formulation has been broadly applied in manufacturing [17]. A methodology for the determination of modules with the consideration of testability is developed.

A. Modeling the Product Modularity

1) *Representation of Modular Products:* Modularity is viewed by Ulrich and Tung [18] as depending on two characteristics of the design:

- 1) similarity between the physical and functional architecture of the design;
- 2) minimization of incidental interactions between physical components.

Based on the two relationships, the interaction and modularity matrices are used to represent modular products.

Let the row set and column set I corresponds to the component set $C = \{1 \dots m\}$ in the matrix. The *interaction matrix*, $A = [a_{ij}]_{m \times m}$, is a component-component incidence matrix, where a_{ij} represents the interaction between component i and component j ; i and $j \in C$. Rearranging the columns and rows of the interaction matrix results in a matrix with block-diagonal

Fig. 1. Modularity matrix A' .

TABLE I
INTERPRETATION OF THE MODULARITY MATRIX

		Modularity matrix		
Design phase	Product type	Entry interpretation	Column name	Entry value: value interpretation
Conceptual phase	All	Function flow	Mechanism/Subsystem	Integer number: frequency of application
Detailed phase	Electrical	Electrical flow	Component	
	Mechanical	Force flow, thermal, function, etc.		
	Mixed	Force flow, thermal, function, etc.		

submatrices. The transformed matrix, $A' = [a'_{ij}]_{m \times m}$, is defined as a *modularity matrix*. The rearrangement of rows and columns is performed by the decomposition discussed in Section II-B.

A *module* corresponds a block-diagonal submatrix of A' (see Fig. 1). The components that do not belong to any module are referred to as *basic components*. The basic components jointly with the modular components (included in modules) result in different types of modular products.

In the modularity matrix in Fig. 1, columns and rows correspond to the component numbers. An entry “1” represents to the interaction between two components and “blank” indicates no interaction. Three modules are visible in the matrix in Fig. 1.

The entries “1” in the interaction matrix may be generalized to integer numbers representing, e.g., the frequency of using any two functions at the conceptual design phase, or two components at the detailed design phase. The summary of entry values, their interpretation, and the meaning of the column and row labels in the modularity matrix applied to different types of products and in different design phases are summarized in Table I.

2) *Interpretation of Different Types of Modularity:* Three axioms for interpreting different types of modularity are presented next.

Axiom 1 interprets the component-swapping modularity.

Axiom 1:

Let C_i be the set of columns corresponding to entries “1” of row i [e.g., $C_7 = \{1, 2, 3\}$ in Fig. 2(a)].

If

- 1) row i corresponds to a module, and
- 2) columns $j \in C_i$ do not correspond to any other modules,

then the modularity is referred to as *component-swapping modularity*, e.g., module M and the set $C_7 = \{1, 2, 3\}$ in Fig. 2(a) form the component-swapping modularity.

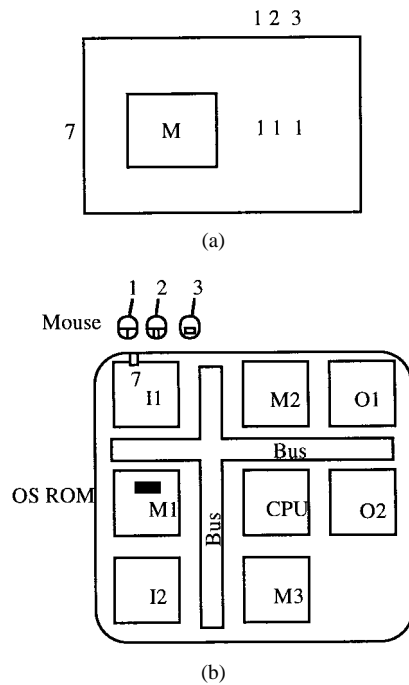


Fig. 2. Example of component-swapping modularity.

Example 1: An example of product variants generated through component-swapping is provided in Fig. 2(b). The various computers in Fig. 2(b) are assembled on a frame board (M). Different product variants are produced by changing the mice (basic components 1, 2, and 3).

In computer industry, the component-swapping modularity is accomplished by matching different hard disk types, monitor types, and keyboards with the same frame board.

Axiom 2 is used to interpret the component-sharing modularity.

Axiom 2:

Let C_i be the set of columns corresponding to entries "1" of row i [e.g., $C_i = \{1, 2\}$ in Fig. 3(a)].

If

- 1) row i corresponds to a basic component, and
- 2) each column in C_i corresponds to a module,

then the modularity is referred to as *component-sharing modularity*, e.g., modules M1 and M2, and component 3 in Fig. 3(a) form the component-sharing modularity.

Example 2: An example of product variants generated through component-sharing is provided in Fig. 3. The different types of frame boards and monitors (M1, M2) in Fig. 3(b) sharing the same microprocessor (component 3) make up different types of computers.

In consumer electronics, component-sharing arises when a common power cord or a common tape transport mechanism is used in different product families.

Axiom 3 interprets the bus modularity.

Axiom 3:

Let C_i be the set of columns corresponding to entries "1" [e.g., $C_i = \{1, 2, 3\}$ in Fig. 4(a)] and R_i be the set of rows corresponding to entries "1" [e.g., $R_i = \{4, 5\}$ in Fig. 4(a)].

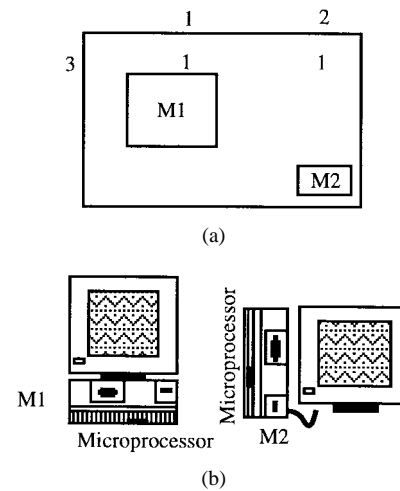


Fig. 3. Example of component-sharing modularity.

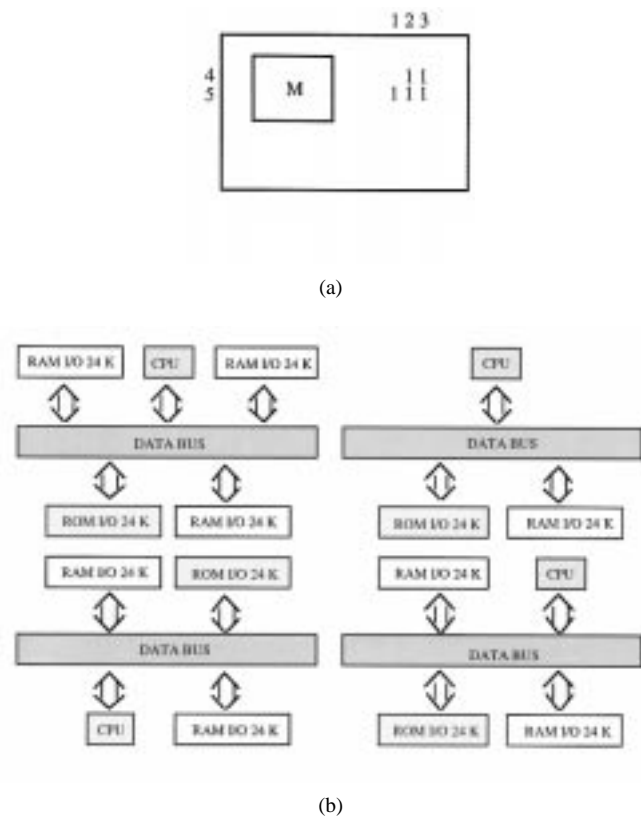


Fig. 4. Example of bus modularity.

If

- 1) the set of rows R_i corresponds to a module, and
- 2) all columns $j \in C_i$ do not correspond to any other modules,

then the modularity as referred to is *bus modularity*, e.g., module M and basic components 1, 2, and 3 in Fig. 4(a) form the bus modularity.

Example 3: An example of product variants generated through the bus modularity is provided in Fig. 4(b). The same type of data bus (M), and different types of CPU and memory units (components 1, 2, and 3) in Fig. 4(b) form different types of data processors with RAM/ROM of different capacity.

Other examples of bus modularity are a computer, or a circuit breaker.

B. Solving the Modularity Problem With Testability Consideration

In this section, the modularity problem represented by the interaction matrix is solved with a decomposition approach. Some research on decomposition in engineering design has been reported in [19] and [20]. As the interaction matrix is square, the decomposition algorithm [21] is applied to identify modules, also with the testability constraints imposed based on the design for testability guidelines.

In order to make complex product designs testable, three key testability principles based on the definition of testability in technical terms must be considered [11].

1) *Partitioning*: Partitioning aims at splitting functions into logically separable units. Functions should be designed as logically complete units rather than being split among different assemblies. Partitioning simplifies testing and troubleshooting and facilitates built-in test equipment (BITE) designs as the fault isolation can be easily achieved. Modular design of functions simplifies testing by reduced amount of I/O simulation required to test these functions.

2) *Controllability*: Controllability is the ability to externally [typically by automatic test equipment (ATE)] alter the internal status of a unit under test (UUT). Control is imperative for the board to be functionally testable. It is especially needed over the processor lines such as READY, RESET, HOLD, TRAP, and NMI (nonmaskable interrupt).

3) *Observability*: Observability points (OP's) are an easy way to provide improved testability. Candidate points for OP's are:

- 1) points of instruments connected to a circuit node (or added an extra pseudo-primary output to the design file of the circuit) to make it easier to verify the design/debug the prototype circuit;
- 2) points of reconvergent fanout, or high fanin points;
- 3) points of the output of logic driving displays or other output devices. *Intelligent* observability points placed between device packages of the modules are much more valuable than test points placed at intermediate stages in a single package.

Abramovici [12] suggested the following basic rules to improve testability:

- 1) employ test points to enhance controllability and observability;
- 2) design circuits to be easily initializable;
- 3) disable internal one-shots during test;
- 4) disable internal oscillators and clocks during test;
- 5) partition large counters and shift registers into small units;
- 6) avoid the use of redundant logic;
- 7) provide logic to break global feedback paths.

For the studies in the controllability enhancement refer to Chickermane and Agrawal [22], Goldstein and Thigen [23], Chickermane *et al.* [24], and the observability enhancement to Goldstein [25].

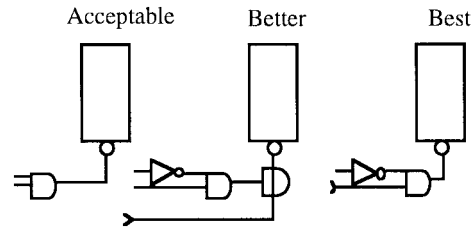


Fig. 5. Added logic elements for initialization.

4) *Design Guidelines for Testability*: Each entry a_{ij} in the interaction matrix represents to the relationship between components i and j . A test point is imposed between two components i and j (entry a_{ij} of the interaction matrix). The IEEE standard 1149 and the testability guidelines [11], [12], [26], [27] for a circuit board, general digital circuit, built-in test, and adding testability points are used to check the suitability of adding extra testing points for the key entries in the interaction matrix. The guidelines can be embedded into an expert system that can assist in decomposing a product architecture into modules with the consideration of testability. The testability guidelines and IEEE standard 1149 are used at the circuit level to identify testability points. The decomposition can be also perform at the system level in the spirit of concurrent engineering (integrated design).

For example, for the entry of the interaction matrix corresponding to a *buried* memory element, the guideline suggests extra traces, extra inputs to gates, and extra logic elements or functions for initialization to be added to the memory element in order to initialize it without a difficult to develop “homing sequence” (see Fig. 5).

5) *Problem Formulation*: The problem of forming modules is stated next.

Decompose a component-component interaction matrix into mutually separable submatrices (modules) with:

- 1) the minimum number of nonempty high value entries outside the block-diagonal interaction matrix;
- 2) subject to the following constraints:

Constraint C1 Empty modules of components are allowed, and

Constraint C2 The number of components in a module can not exceed upper bound N_u , and the total cost of the components duplicated can not exceed B .

Constraint C3 The testability guidelines are satisfied for each entry (accomplished by the expert system).

Decomposition allows one to explore potential modules among components and to analyze various types of modularity. The challenge is to group components into modules that are of acceptable size, cost, and test requirements.

6) *Decomposition Approach*: The design for testability methodology developed in this paper aims at

- 1) modularizing circuits;
- 2) circuit controllability;
- 3) circuit observability.

The decomposition approach presented in this section transforms the interaction matrix into modularity matrix A' (defined

in Section II), analyzes the modularity matrix, and detects modularity across different products. Two characteristic conditions are considered to include/exclude a component from a module.

Condition 1: Exclude a component if the following conditions are satisfied:

- 1) Control points are needed.
- 2) None of the submatrices violates constraints C1 and C2.

Condition 2: Include the component if the following conditions are satisfied:

- 1) a component is used in two modules simultaneously;
- 2) visible points are needed;
- 3) none of the submatrices violates constraints C1 and C2.

Algorithm:

Step 0 (Initialization)

Initialize the interaction matrix. Specify an upper bound N_U on the number of components in a module and the budget B .

Step 1 (Triangularization)

Triangularize the interaction matrix A into matrix A' with the algorithm presented in Kusiak *et al.* [21].

Step 2 (Testability)

Determine the controllability and observability points based on the IEEE standards and testability guidelines [11]–[12], [26]–[27].

Step 3 (Exclusion)

Exclude a component from a module that satisfies Condition 1, and place it in the last column of the modularity matrix. Repeat this step until no more components can be removed.

Step 4 (Inclusion)

Include a component that satisfies Condition 2, and repeat this step until no more components can be duplicated.

Step 5 (Identification)

Identify modules corresponding to the groups in A' .

Step 6 (Classification)

Classify the modules in the modularity matrix based on the three axioms presented in Section II.

Step 7 (Termination)

Stop and output the results.

Note that in Step 4 of the algorithm, the components that include control points are removed from the modules in order to be connect to the inputs of test modules. Step 5 produces a solution of better quality by including some components and making the module points more visible for testing.

7) Illustrative Example:

Example 4: Electrical circuits.

Consider the set of electrical components C1, C2, C3, C4, C5, C6, C7, and C8 in Fig. 6. An electrical product, whether it is a simple radio or a complex supercomputer, consists of two basic elements—electronic components and interconnections between the components. This example focuses on the modularization of components and the analysis of modules.

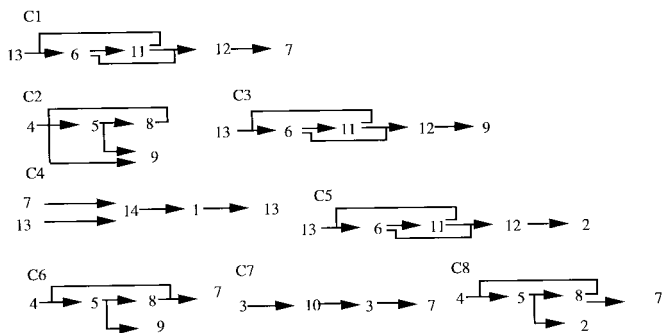


Fig. 6. The set of electrical components with inputs and outputs.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	1	2	3	4									
2		+												
3			+			1			1					
4				+	1				1	1				
5	1													
6						+						1	1	
7							+							
8			1				1	+						
9	1								+					
10		1								+				
11						1					+	1		
12	1						1		1				+	
13										1				+
14	1													+

Fig. 7. The interaction matrix for the eight components in Fig. 6.

	13	6	11	12	13'	14	1	7	2	9	10	3	4	5	8
13	+	1													
6		+	1	1											
11			+	1											
12				+			1	1	1						
14					+	1									
1						+	1								
13'							1								
7								+							
2									+						
9										+					
10											+	1	1		
3												+	1		
4													+	1	
5														+	1
8															+

Fig. 8. The transformed modularity matrix.

The interaction matrix of the component set are defined in Fig. 7.

The “entry $a_{5,1} = 1$ ” in Fig. 7 means that the electrical signal flows from output 5 to input 1.

Applying the decomposition algorithm to the interaction matrix in Fig. 7 results in matrix $[A']$ in Fig. 8.

Four modules have been identified in the matrix in Fig. 8: $(13, 6, 11, 12)$, $(13', 14, 1)$, $(10, 3)$, and $(4, 5, 8)$. Based on the definitions presented in Section II, module $(6, 11, 12, 13)$ and components 7, 9, 2 form component-swapping modularity. Module $(13, 14, 1)$, $(10, 3)$, and component 7 form component-sharing modularity. Module $(4, 5, 8)$ and components 2, 7, 9 form bus modularity.

Note that in order to obtain a solution of better quality, input 13' duplicating 13 is introduced. The modularity matrix without duplicating 13 is presented in Fig. 9. Three modules are identified: $(13, 6, 11, 12, 14, 1)$, $(10, 3)$, and $(4, 5, 8)$, where the submatrix corresponding to module $(13, 6, 11, 12, 14, 1)$ is sparse. The group efficacy of module $(13, 6, 11, 12, 14, 1) = 8/36$ is lower than the group efficacy

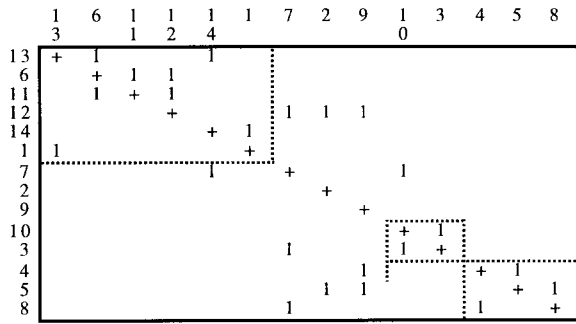


Fig. 9. The transformed modularity matrix without duplicating 13.

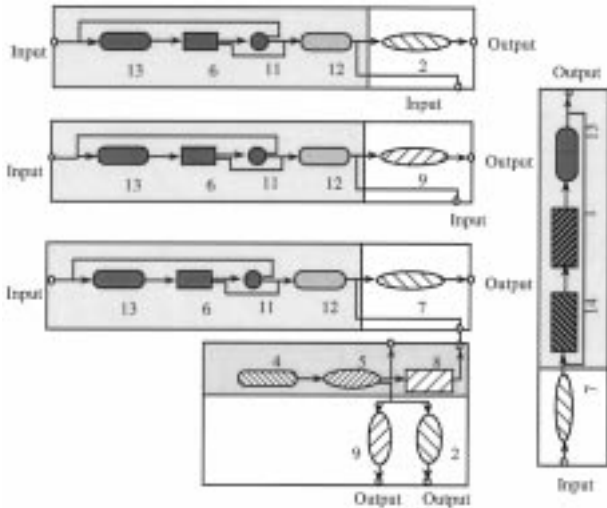


Fig. 10. Example of five circuits.

of modules (13,6,11,12) and (14,1,13') = 5/16 + 3/9 = 93/144 in Fig. 9 (see [17] for the definition of efficacy measure).

Examples of the designs based on the modules identified in Fig. 6 are shown in Fig. 10. The shadowed components form the modules.

Example 5: Design for testability the six level logic chain in Fig. 11.

In this example, the components from Example 4 are used in the six level logic chain. The logic chain includes 20 IC's, some VLSI IC's and some ASIC's, along with regular "glue logic." This example signifies the testability in the modularization process.

Step 1 The triangularized interaction matrix is presented in Fig. 12.

Steps 2-4 Identify testability points.

The circuit in Fig. 11 with some "glue logic" may contain feedback loops and may be initialized. In Step 1, the block-diagonal submatrices are formed. In Step 2, e.g., CP1 is suggested by the testability guideline regarding the buried memory element (U16) in order to initialize it without a "homing sequence." Other CP's and OP's may be identified based on the guidelines presented in Turino [11]. The result of partitioning and adding CP's to the six level logic chain is illustrated in Fig. 13.

Adding more CP's and OP's for testability leads to the result shown in Fig. 14 [11].

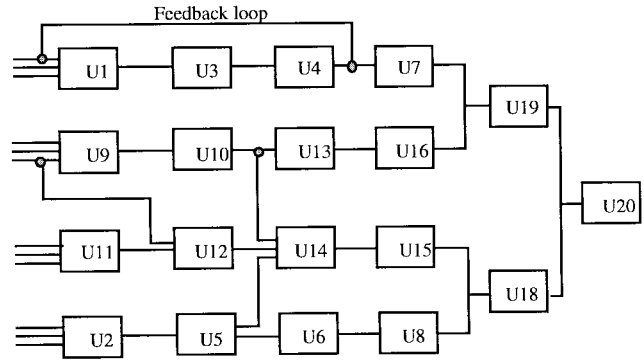


Fig. 11. Six-level logic chain.

Step 5 The decomposition approach results in three modules: {U1, U3, U4, CP4, OP1}, {U9, U10, U13, CP1}, and {U11, U12, U14, U15, U2, U5, CP2, CP3, CP5, OP2, OP3}.

III. INTEGRATED SYSTEM FOR DESIGN OF MODULAR PRODUCTS AND TESTS

In this section, an integrated system for design of modular products and tests is developed. The modular testing is presented in Section III-A. In Section III-B, an integrated approach for design of modular products in a modular test environment is presented. The board and system test level of modular products is considered.

A. Test Modularity

In this section, test modules of increased reusability and exchangeability are discussed. An increased testability is achieved by adding controllable and observable points in structures called *testing modules* of products. A *testing module* consists of test generation software, gates, inputs, and outputs.

Based on the interactions between the test equipment and products tested, three categories of modularity have been defined (analogous to product modularity in Section II).

- 1) *Test-swapping modularity* occurs when two or more different *basic test modules* are paired with a product module creating different test function variants belonging to the same testing resource family.
- 2) *Test-sharing modularity* is complementary to the component-swapping modularity. Various product modules sharing the same *basic test module* create different test function variants for various product families.
- 3) *Bus modularity* occurs when a product module can be matched with any number of *basic test modules*. Bus modularity allows for variation in the number and location of basic test modules while test-swapping and test-sharing modularity allows only the basic components to vary.

Examples 6-8 illustrate the test modularity [12], [28].

Example 6: Test-swapping modularity.

Fig. 15 shows a portion of the TMS 32010 signal-processing chip. In Fig. 16, four modules of built-in logic

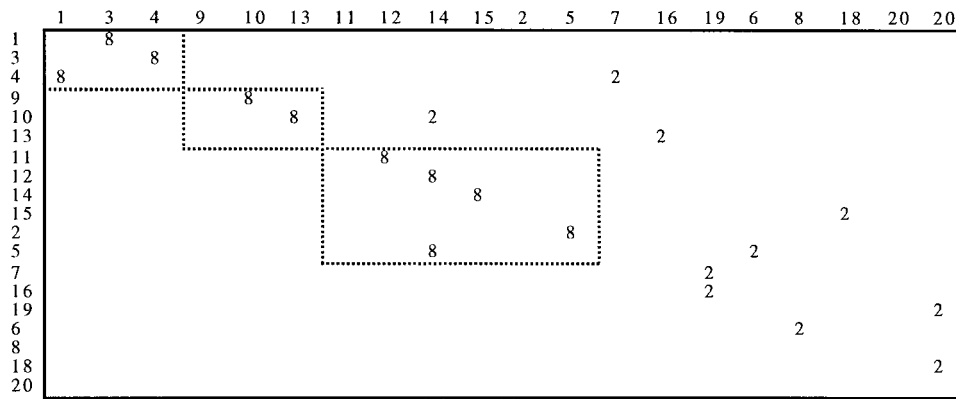


Fig. 12. The triangularized interaction matrix of the six-level logic chain.

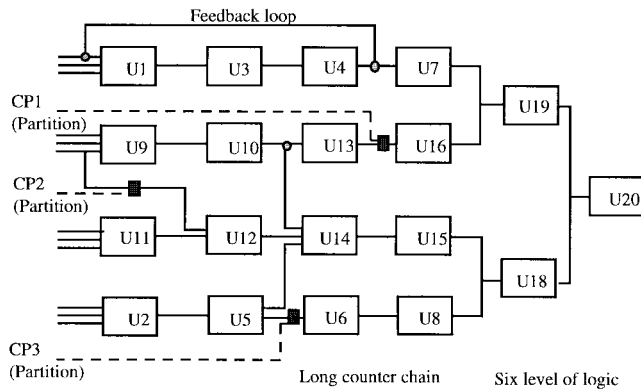


Fig. 13. The partitioned six-level logic circuit with OP's added.

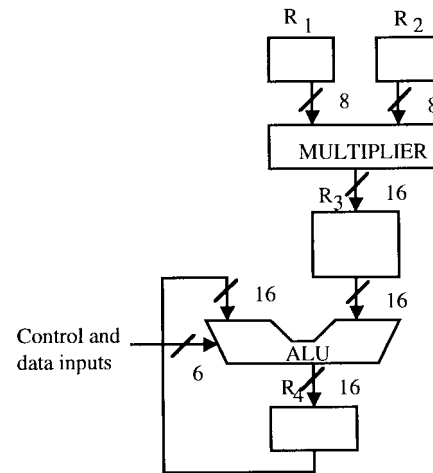


Fig. 15. Portion of the TMS 32010 signal-processing chip.

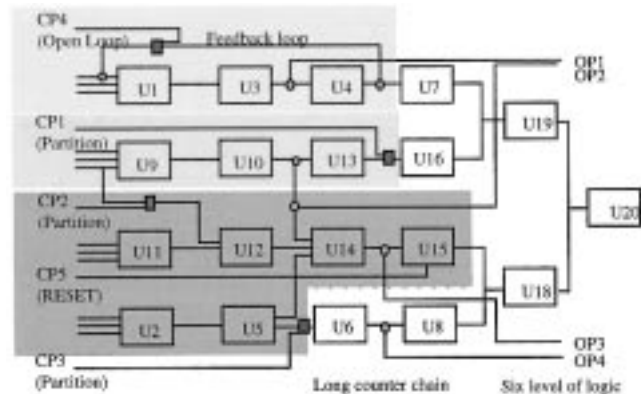


Fig. 14. The result with CP's and OP's added.

block observation (BILBO) are presented to test this chip [28].

Example 7: Test-sharing modularity.

Fig. 17 presents how the concept of boundary scan can be extended to the board and system level according to the test-sharing modularity. The system here is a collection of modules, such as PCB's, each consisting of IC's. In this example, the boundary scan path of each chip on a board is interconnected in a daisy chain fashion to create one long scan path on each board. All boards share common S_{in} , \bar{N}/T , and CK inputs. Their S_{out} lines are wired-ORed together.

Example 8: Test bus modularity.

This design for testability (DFT) approach makes use of a module's or system's functional bus to control and observe sig-

nals during functional level testing. A test and/or maintenance processor appears as another element attached to the system buses. The system bus in Fig. 18 controls the testing of this system according to the test bus modularity. This is often done by emulating the system's processing engine. The automated test equipment can also emulate the various units attached to the bus, monitor bus activity, and test the processing engine. In general, complex, manually generated functional tests are used.

1) *Test Library:* Based on the concept of test modularity, the test "library" consists of modular sets of software and hardware. The software includes the test generation (TG) algorithms for different types of chips. The examples of the TG algorithms include: D-algorithm [29], 9-V algorithm [30], Path-Oriented Decision Making (PODEM) [31], Fanout-Oriented (FAN) [32], fault-oriented algorithm for sensitized-path testing (FAST) [33], and so on. Each algorithm combined with a set of hardware is able to test the testability points in a module. The hardware consists of gates, inputs, and outputs incorporated to logic gate level description of the circuit.

B. The Integrated Design Approach

With product and test modules determined, a matching approach, e.g., technology mapping [34], is used to select the desired modular set of software and hardware from the test

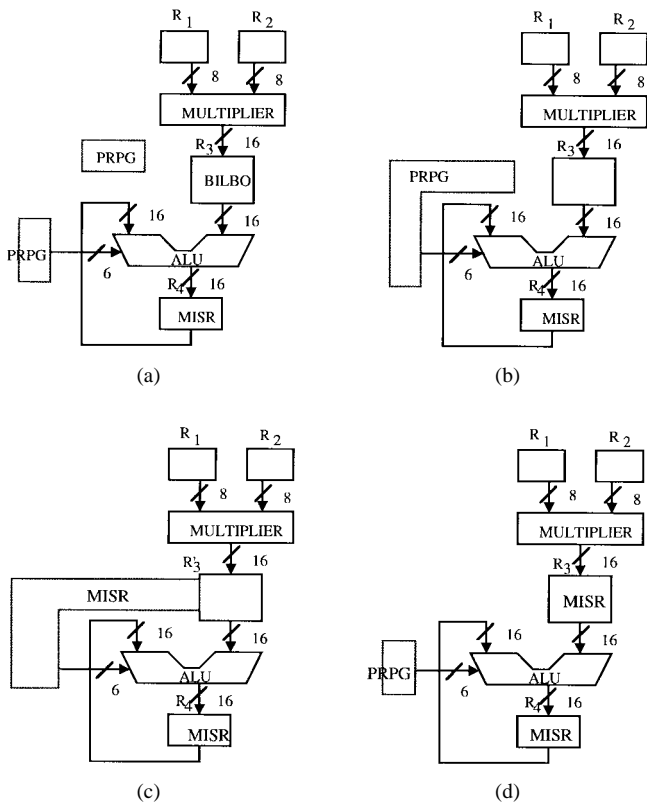


Fig. 16. Test-swapping modularity: Four test modules for the TMS 32010: (a) BILBO solution, (b) solution with R_3 remains unmodified, (c) solution with ALU tested by separate MISR's, and (d) solution with ALU tested by sharing an MISR.

library to apply to the testability points in a test object. It is assumed that each point is testable with an existing testing module.

The integrated design approach is summarized in Fig. 19. The design object is modularized with the algorithm discussed in Section II-B. In the process of modularization, testability points are added by the expert system which supplies testability guidelines according to the IEEE standards, e.g., IEEE Std. 1149. With the design products modularized, possible CP's (controllable points) and OP's (observable points) are used to determine test modules. Matching the test requirements of CP's and OP's of design products with proper test modules is crucial and needs to be completed through the negotiation and mapping between the design for testability and modular testing. The trade-off analysis of the test modules is necessary.

As illustrated in Fig. 19, the testability points CP1, OP1, and OP2 are located and added to the modular product with three modules M1, M2, and M3, thus improving its testability. The test modules $\{2, b, c\}$, $\{1, a, b\}$, and $\{1, d\}$ are used to test OP1, CP1, OP2, respectively.

IV. THE INTERACTION BETWEEN PRODUCT MODULARITY AND DESIGN FOR TESTABILITY

The activities related to the design of modular products and design for testability interact with each other. This interaction is discussed next.

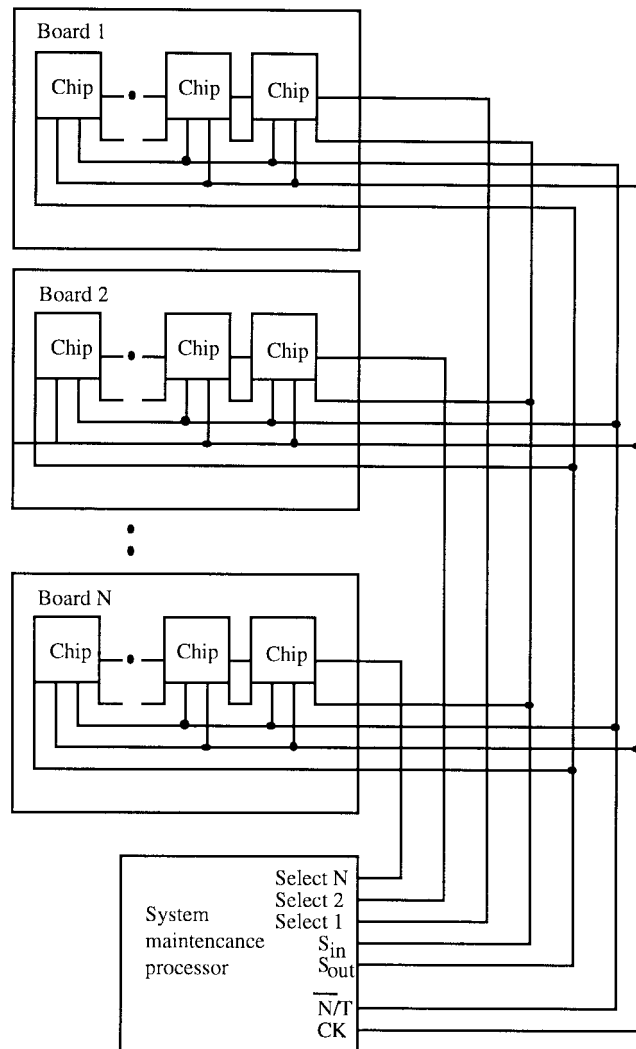


Fig. 17. Test-sharing modularity.

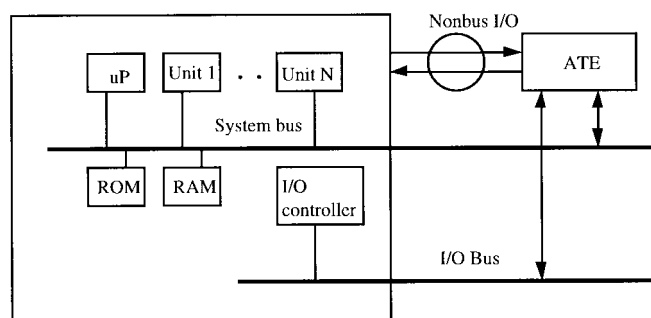


Fig. 18. Test bus modularity.

A. The Impact of Product Modularity on Design for Testability

The impact of product modularity on the design for assembly and the product life cycle are presented in He and Kusiak [35] and Newcomb *et al.* [36].

Decomposition, standardization, and exchangeability are the attributes of a modular product. Decomposition enhances the controllability and observability of testing, repetitively, and hence reduces the complexity of the test process. For example, assume test generation requires n^2 steps for a circuit C having

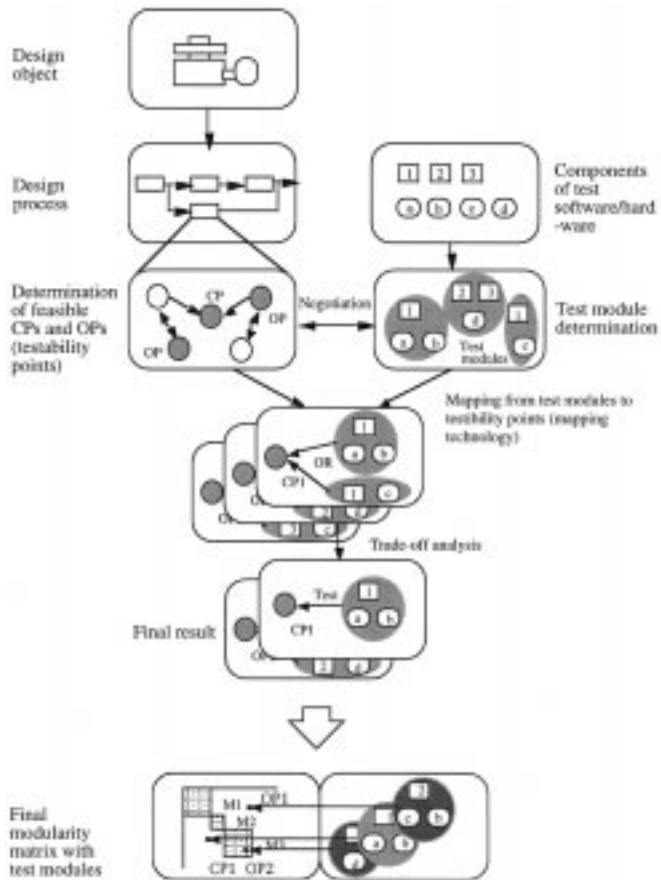


Fig. 19. The paradigm of design for testability.

n gates, and that C has 10 000 gates and can be partitioned into two circuits C_1 and C_2 of 5000 gates each. Then, the test generation for the unpartitioned version of C requires $(10^4)^2 = 10^8$ steps, while the test generation for C_1 and C_2 requires only $2 * 25 * 10^6 = 5 * 10^7$ steps, or about half of the testing time. Standardization simplifies test adapter designs and also reduces the number of different test adapters required for any system. Similar to using standard pin configurations, standardization of connector types reduces the number of types of test adapters required and improves manufacturing and logistics conditions.

B. The Impact of Design for Testability on Product Modularity

The size of a module is often compromised due to several factors, e.g., panel size, performance requirements, or cost. Khan and Madisetti [37] presented a systematic approach to partitioning the multichip-based system for low power considering the area of panel, yield, reuse of cell libraries, and routing constraints.

Design for testability may also be another crucial factor in partitioning a circuit and determining the boundary of a module. Conventionally, a circuit is partitioned into reasonably small functional blocks (clusters). In the modular design approach, a block (cluster) corresponds to module. The partitioning problem is concerned with breaking a circuit system into modules making the circuit system easier to understand, easier to control by including reasonably direct paths from the

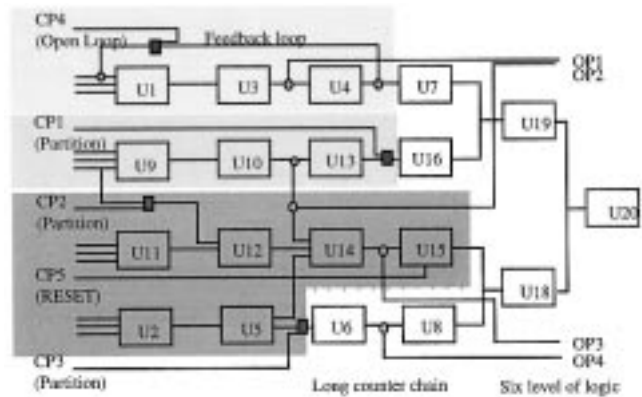


Fig. 20. The result with CP's and OP's added.

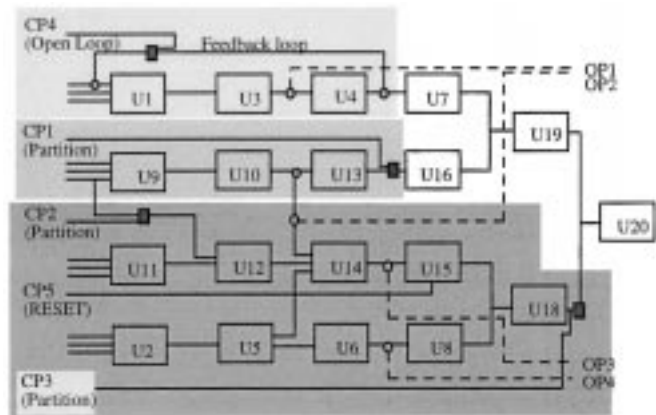


Fig. 21. The alternative result with more CP's and OP's added.

test resources (either automatic test equipment or built-in-test circuitry) to critical internal nodes required for initialization of the circuitry under test, partitioning that circuitry, and control for fault activation. The procedure for forming modules with the consideration of testability may include partitioning, and adding control points and visible points. Testability may become a constraint of the module formation problem.

Example 5 of Section II is used to illustrate the impact of testability on the formation of modules.

Example 9: Design for testability for the six-level logic chain in Fig. 6.

The result of partitioning, of six level logic chain with added gates and control points to break long chains and cut programming times is illustrated in Fig. 20.

Step 5 Three modules are determined: $\{U1, U3, U4, CP4, OP1\}$, $\{U9, U10, U13, CP1\}$, and $\{U11, U12, U14, U15, U2, U5, CP2, CP3, CP5, OP2, OP3\}$.

The partitioning with control and visible points added may alternate, thus resulting in modules of different sizes. An alternative partition is presented in Fig. 21, where the placement of CP3 has changed.

The decomposition approach with the different guidelines considered in Step 2 provides three modules: $\{U1, U3, U4, CP4, OP1\}$, $\{U9, U10, U13, CP1\}$, and $\{U11, U12, U14, U15, U2, U5, U6, U8, U18, OP2, OP3, OP4\}$. Note that the third module is larger than in the previous partition.

V. CONCLUSION

Methodologies for the design of modular products for testability and the design of test modules were developed. Test modules of increased reusability and exchangeability were discussed. An integrated system for design of modular products and tests was presented. The interaction between product modularity and the design for testability was discussed. The main contribution of this paper is in exploring the relationship between the design of modular products and testing the products in modular tests.

The following four issues require further studies.

- 1) Test libraries for different types of electronic products need to be developed.
- 2) An efficient matching approach between the testability points and test modules in test library is needed.
- 3) An expert system supporting the identification of testability points need to be developed.
- 4) The trade-off analysis of the design guidelines for testability should be developed.

REFERENCES

- [1] P. T. Kidd, *Agile Manufacturing: Forging New Frontiers*. New York: Addison-Wesley, 1994.
- [2] R. N. Langlois and P. L. Robertson, "Networks and innovation in a modular system: Lessons from the microcomputer and stereo component industries," *Research Policy*, vol. 21, no. 4, pp. 297–313, 1992.
- [3] W. Ward, J. F. Liker, J. J. Cristiano, and D. K. Sobek, "The second Toyota paradox: Delaying decisions can make better cars faster," *Sloan Manage. Rev.*, pp. 43–61, 1995.
- [4] R. Sanchez, "Strategic flexibility, firm organization, and managerial work in dynamic markets: A strategic options perspective," *Adv. Strategic Manage.*, vol. 9, pp. 251–291, 1993.
- [5] G. Pahl and W. Beitz, *Engineering Design*, London, U.K.: Design Council, 1988.
- [6] A. Kusiak and C. C. Huang, "Development of modular products," *IEEE Trans. Comp., Packag., Manufact. Technol.*, vol. 19, pp. 523–538, 1996.
- [7] R. Sanchez, "Strategic product creation: Managing new interactions of technology, markets, and organizations," *Europ. Manage. J.*, vol. 14, no. 2, pp. 121–138, 1996.
- [8] J. Steffens, "AT&T adopts a global manufacturing architecture," *Manuf. Syst.*, vol. 13, no. 1, pp. 34–39, 1994.
- [9] G. Gilder, *Microcosm: The Quantum Revolution in Economics and Technology*. New York: Simon and Schuster, 1989.
- [10] E. Trischler and M. Johansson, "Ten: A concurrent test engineering environment," *IEEE Design Test Computers*, vol. 11, pp. 6–16, 1994.
- [11] J. Turino, *Design to Test*. New York: Van Nostrand Reinhold, 1990.
- [12] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*. New York: IEEE Press, 1990, pp. 347–358.
- [13] S. K. Gupta, "Synthesis of testable combinational circuits: An overview of university activities," in *Proc. Int. Test Conf. 1994, ITC94*, Altoona, PA, 1994, Paper 1.2.
- [14] I. Pomeranz and S. M. Reddy, "Synthesis of testable sequential circuits: An overview of university activities," in *Proc. Dig. Paper, Test Synthesis Seminar, International Test Conference, ITC94*, Altoona, PA, 1994, Paper 1.3.
- [15] J. P. Hays, "On modification logic networks to improve their diagnosability," *IEEE Trans. Comput.*, vol. C-23, pp. 56–62, 1974.
- [16] J. P. Hays and A. D. Friedman, "Test point placement to simplify fault detection," *IEEE Trans. Comput.*, vol. C-23, pp. 727–735, 1974.
- [17] A. Kusiak, *Intelligent Manufacturing Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1990.
- [18] K. Ulrich and K. Tung, "Fundamentals of product modularity," *Issues in Design/Manufacture Integration 1991*, A. Sharon, Ed. New York: ASME, 1991, vol. DE-39, pp. 73–79.
- [19] D. V. Steward, "Partition and testing systems of equations," *J. Soc. Ind. Appl. Math.: Numer. Anal.*, vol. 2, no. 2, pp. 345–365, 1965.
- [20] A. Kusiak and K. Park, "Concurrent engineering: decomposition and scheduling of design activities," *Int. J. Prod. Res.*, vol. 28, no. 10, pp. 1883–1900, 1990.
- [21] A. Kusiak, T. Larson, and J. Wang, "Reengineering of design and manufacturing processes," *Comput. Ind. Eng.*, vol. 26, no. 3, pp. 521–536, 1994.
- [22] V. Chickermane and V. D. Agrawal, "An optimization based approach to the partial scan design problem," in *Proc. Int. Test Conf.*, Altoona, PA, 1990, pp. 28–35.
- [23] L. M. Goldstein and E. L. Thigen, "SCOAP: Sandia controllability/observability analysis program," in *Proc. 17th Design Automation Conf.*, 1980, pp. 190–196.
- [24] V. Chickermane, E. M. Rudnick, P. Banerjee, and J. H. Patel, "Non-scan design-for-testability techniques for sequential circuits," in *Proc. 30th Design Automation Conf.*, 1993, pp. 236–241.
- [25] L. H. Goldstein, "Controllability/observability analysis of digital circuits," *IEEE Trans. Circuits Syst.*, vol. CAS-26, pp. 685–693, 1979.
- [26] R. C. Aitken, "An overview of test synthesis tools," *IEEE Design Test Computers*, vol. 12, pp. 8–15, 1995.
- [27] J. P. Parker, *The Boundary-Scan Handbook*. Boston, MA: Kluwer.
- [28] K. Kim, D. S. Ha, and J. G. Tront, "On using signature registers as pseudorandom pattern generators in built-in self-testing," *IEEE Trans. Computer-Aided Design*, vol. 7, pp. 919–928, 1988.
- [29] J. P. Roth, Bouricius, and P. R. Schneider, "Programmable algorithm to compute tests to detect distinguish between failures in logic circuits," *IEEE Trans. Electron. Comput.*, vol. EC-16, pp. 567–579, 1967.
- [30] C. W. Cha, W. E. Donath, and F. Ozguner, "9-V algorithm for test pattern generation of combinational digital circuits," *IEEE Trans. Comput.*, vol. C-27, pp. 193–200, 1978.
- [31] P. Goel, "An implicit enumeration algorithm to generate tests for combinational logic circuits," *IEEE Trans. Comput.*, vol. C-30, pp. 215–222, 1978.
- [32] H. Fujiwara and T. Shiono, "On the acceleration of test generation algorithms," *IEEE Trans. Comput.*, vol. C-32, pp. 1137–1144, 1983.
- [33] M. Abramovici, P. R. Kulikowski, P. R. Menon, and D. T. Miller, "SMART and FAST: Test generation for VLSI scan-design circuits," *IEEE Design Test Computers*, vol. 3, pp. 43–54, 1986.
- [34] F. Mailhot and G. D. Micheli, "Algorithm for technology mapping based on binary decision diagrams and on Boolean operations," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 599–620, 1993.
- [35] D. W. He and A. Kusiak, "Performance analysis of modular products," *Int. J. Prod. Res.*, vol. 34, pp. 253–272, 1996.
- [36] P. J. Newcomb, B. Bras, and D. W. Rosen, "Implications of modularity on product design for the life cycle," *ASME Design Theory Methodology Conf. '96*, 1996.
- [37] A. K. Khan and V. K. Madiseti, "System partitioning of MCM's for low power," *IEEE Design Test Computers*, vol. 12, pp. 41–52, 1995.



Andrew Kusiak (M'90) is Professor in the Department of Industrial Engineering, University of Iowa, Iowa City.

His interests are in product development, manufacturing, and applications of artificial intelligence and optimization. He has published research papers in journals sponsored by AAAI, ASME, IEEE, IIE, INFORMS, ESOR, IFIP, IFAC, IPE, ISPE, and SME. He speaks frequently on international meetings, conducts professional seminars, and consults for industrial corporations.

Dr. Kusiak serves on the editorial boards of 16 journals, edits book series, and is the Editor-in-Chief of *The Journal of Intelligent Manufacturing*.



Chun-Che Huang received the M.S. degree in operations research from Columbia University, New York and is currently pursuing the Ph.D. degree in the Department of Industrial Engineering, University of Iowa, Iowa City.

His interests are in the design of modular systems and intelligent manufacturing systems.