RECOGNIZING VLSI CIRCUITS FROM MASK ARTWORK

by

Amir Alon and William Havens

Technical Report 85-1

January 1985

Recognizing VLSI Circuits from Mask Artwork

Amir Alon and William Havens

Laboratory for Computational Vision
Department of Computer Science
University of British Columbia
Vancouver, British Columbia
Canada V6T 1W5

Abstract

The design of Very Large Scale Integrated (VLSI) circuits remains an art despite recent advances in Computer Assisted Design (CAD) techniques. Unfortunately, the sophistication of the design process has not kept pace with the VLSI hardware technology. Very expensive errors proliferate into fabrication despite powerful design rule checkers and circuit simulators. We have developed an alternative approach derived from research in knowledge representation and schema-based computer vision. The system implemented recognizes an abstract logic function description of the VLSI circuit from its mask layout artwork. Our technique reverses the design process thereby recovering the logical function actually fabricated in the chip. No simulation is necessary and conceptually all logical design errors can be detected. The work is a direct application of schema labelling techniques which were developed for the Mapsee2 sketch map understanding system. This prototype system has been tested on a number of logical chip designs with correct results. Some results are presented.

1. Introduction

Very Large Scale Integrated (VLSI) circuits have fostered a revolution in the complexity and sophistication of electronics. The implications of the technology are well appreciated. Unfortunately, the sophistication of the design process for producing VLSI circuitry has not kept pace with this hardware expansion. Progress has been made in utilizing semi-automated layout systems, design rule checkers (DRC), and mask artwork

compaction programs to ensure that the artwork produced is correct and functional. Many of these systems concentrate on the *forward design process* attempting to preclude layout and fabrication mistakes from occurring. As depicted in Figure 1, the design process proceeds from right to left. A *logic level* description of the desired chip is used to generate an interconnected network of active devices (and possibly passive components). This *circuit level* description is then used to create multilayer *layouts*. Photographic masks prepared from these detailed layouts are then used to fabricate the actual silicon chips.

Unfortunately, serious design errors still proliferate from conceptual mistakes made at both the circuit design and layout design levels. Circuit simulation programs can only check the behaviour of the implemented circuit for a subset of its possible states. Furthermore, simulation is inherently inefficient and its use becomes increasingly expensive as the complexity of VLSI circuitry continues to expand. What is needed is the ability to reverse the design process; to recognize a boolean logic level description of the circuit actually implemented directly from the layout artwork. Any logical design errors will then be immediately apparent regardless of their sublety or their origin. Figure 1 shows the recognition process proceeding in the opposite direction from design. The layout level provides the input data from which first a transistor circuit level description can be recognized. This description is interpreted to reproduced a boolean logic description. This description can then easily be compared to the original functional specification for the chip. No simulation is required.

We describe in this paper the theory and implementation of an experimental program for this purpose. The technology employed, called *schema labelling* [Havens,1984],

originated from research in knowledge representations for computer vision. The task of recognizing an abstract circuit description from its realization in mask artwork is directly analogous to recognizing scene descriptions from two-dimensional imagery of the scene. In particular, this work follows as an extension of the Mapsee2 [Havens&Mackworth,1983] sketch-map understanding system. Mapsee2 used a schema-based knowledge representation to encode models of the geographic objects and their spatial relationships that can appear in cartographic sketch maps. The output of the system is an hierarchical structural description of these objects at multiple levels of abstraction.

In the work described here, *schema classes* are used as models for circuit elements and their legitimate interconnections. The classes are organized into a hierarchical knowledge base. At the bottom of this hierarchy the classes represent the devices, resistors, and their connections which can be directly recognized in the input layout artwork. Higher classes in the hierarchy represent digital constructions such as logic gates flip-flops and registers. Rules are used to specify how the classes in the knowledge base can be used to compose a network description for a given input chip layout. We show how to efficiently represent knowledge about the VLSI domain using schema labelling techniques and how to use that knowledge for recognizing abstract VLSI circuit descriptions.

2. Related Research

The work described here follows a long history of representing and applying knowledge to scene analysis. See Mackworth [1977] for a good review of scene analysis research. Of particular importance are the network consistency labelling techniques of

Huffman [1971] and Clowes [1971]. In their work, edges in the scene were labeled as: *convex, concave*, or *occluding*. These edge labels were constrained by a set of relational corner models compiled from *a'priori* knowledge of the *blocks world* task domain. Waltz [1972] subsequently made a large contribution to scene analysis by enlarging the label-sets and by introducing a filtering algorithm in which local inconsistencies were removed before searching for global solutions. Mackworth [1977a] generalized this method further into a class of formal network consistency algorithms. Freuder [1978] extended consistency propagation techniques to synthesize a global n-ary relation over a network of n-variables. Mackworth [1977b] demonstrated that network consistency techniques could be applied to tasks more general than the blocks world. Havens and Mackworth [1983] recently showed how the combination of schema-based knowledge representations and network consistency can overcome a number of current limitations of either methodology alone.

In the VLSI design literature, Takashima et. al. [1982] published algorithms for logic gate recognition in MOS/LSI mask artwork which replaces serial/parallel devices by AND/OR gates respectively. However their algorithm is not a general mechanism. In a similar system, Marvik [1984] allows for verification at the gate level and higher by attaching to the layout higher level descriptions of the modules being implemented. Simulation is used to verify that the intended design behaves identically to this higher level description. This approach requires that the user specifies his design in three different forms: schematics, layout, and high level description attached to the layout. Several other systems developed recently for layout verification [Steven,1984] [Bastian,1983] [hofmann,1983] are basically circuit extractors which are designed to realize a

circuit level description from the layout level. Some of the most advanced systems [hofmann,1983] are technology independent but most others are tailored to specific device technologies.

Luellau et. al.[1984] implemented an extraction program based on effcient network comparison algorithms. Their system allows extraction at any level and is device technology independent but does not provide the ability to recognize more than one level at a time, nor does it provide for recognition of arbitrary boolean logic. No logic function can be extracted unless the expected function was previously coded into a network description. A verification system suggested by Wojcik [1984] was based on a formal reasoning system [Lusk,1982]. Here the verification is done by derivation of correctness proof for circuit level i+1 from axioms and behavioural schemata which describe level i of the design.

In the remainder of this paper we describe our system. We show how schema-based recognition techniques originally developed for computer vision tasks can profitably be applied to VLSI design. The explicit representation of knowledge of the task domain is fundamental to recognition. In particular by representing the structure of integrated circuits at various levels of abstraction, this knowledge can be used to reconstruct a high-level description of the VLSI chip directly from its mask artwork.

3. Representing VLSI Circuits

Knowledge about the VLSI domain is naturally organized into a hierarchy. Levels in this hierarchy correspond loosely to the different levels of abstraction in the integrated circuit design and fabrication processes. Referring again to Figure 1,

integrated circuits can be described (in progressively more concrete terms) at the logic level, circuit level, and finally at an implementation dependent layout level. From this last level, the actual device fabrication masks can be prepared. Each level captures the same function unequivocally but expressed in a description language appropriate to that level. The process of translating from one abstraction level to a lower one is the design process and may depend on the particular device technology employed. In this system we assume a standard *metal oxide semicondutor* (MOS) technology although other technologies are easily adapted.

This same knowledge hierarchy can also be used to reverse the design process and perform recognition. Figure 2 shows the knowledge base used in this system which is called the *composition hierarchy*. Each node in the hierarchy is a *schema class* which is a model for a particular class of objects that can appear in VLSI circuits. Classes are used to represent VLSI components at each level of the design process. For example, the top schema in the composition hierarchy is the *Digital-System* class which represents all possible legitimate VLSI circuits. Schemata lower in the hierarchy represent the components of a VLSI chip at lower levels of abstraction. For example, the *Gate* class represents all legal logical gates known to the system (*NOT, NAND, NOR, AND,* and *OR*). The arcs between classes in the composition hierarchy represent *composition* between the parts (or levels) of a chip design. Classes high in the hierarchy are composed of lower level classes recursively downward to the lowest level transistor layout descriptions which appear directly in the input data.

Each schema class in the composition hierarchy contains specific knowledge:

(1) The class has a finite and discrete *labelset* which provides unique symbolic names for the roles that the class can play in a VLSI circuit design. For example, the labelset for the Gate class is the set: {NOT, NAND, NOR, AND, OR}. Each label in this set names a particular possible interpretation for a digital gate. The labelsets of the other classes in the composition hierarchy also reflect their roles in legal chip designs.

(2) The class contains a list of *components* which can be parts of this class. For example, logic gates are composed of interconnected collections of transistors and other simpler gates. Likewise, latches are composed of gates and recursively other latches. Figure 3 illustrates the composition of a "D-type" latch as a simpler "SR-type" latch and a connected "NOT" gate.

(3) Finally, the class contains a set of *composition rules* which provide constraints on how its components can be interconnected to form valid logical devices. The rules specify which labels remain valid for the class given a particular composition of its components. In this implementation, we rely on the composition rules alone to provide sufficient constraints to disambiguate the labelsets for each class[1]. The result is a unique interpretation for each object recognized in the input data. For example, the composition rule for the "D-type" latch specifies that it must contain an "SR-type" latch with its "set" terminal electrically connected to the "input" terminal of the "NOT" gate and its "reset" terminal connected to the "output" of the "NOT" gate. The rule also assigns names to the input and output terminals of the

[1] See Havens [1984] for reasons to separate constraints on composition from constraints on interpretation.

now completed latch.

The output of the system is a hierarchical structural description of the input data at multiple levels of abstraction. The format of this description is a network of *schema instances* constructed from the schema classes to represent particular occurrences of objects recognized in the input data. The network makes explicit the objects found in the data and their electrical and logical relationships with each other. Typically, the network description contains a large number of instances. However, simple examples can help to illustrate the representation. For example, Figure 4a is a transistor circuit level diagram of a "NAND" gate which contains three transistors arranged in a correct MOS circuit. Figure 4b is a schema network representation of the same gate circuit. The instance, $Gate_1$, contains three transistor instances, T_1, T_2, and T_3. Their particular interconnections in the input data are only consistent with a unique "NAND" label for the gate instance under the constraints defined for its class. A second example is shown in Figure 5a which is a transistor circuit for a random logic function. The network description of the same circuit is shown in Figure 5b. Notice that the composition hierarchy allows the recognition of arbitrary boolean functions from compositions of simple standard gates.

4. Experimental Results

Our system was implemented in compiled Franzlisp on a Vax-11/780 running Unix-4.2BSD. The low-level segmentation of the mask artwork into transistors and their interconnections was simulated although various existing circuit extractor algorithms could have been used [Hofmann,1983] [Steven,1984]. A *Serial 2-Bit Adder* was

used to test the system. This circuit contains over 120 transistors. Figure 6 presents a logic level diagram of the test circuit. Each block in the diagram is composed of a complex circuit. For instance, Figure 7 shows the transistor level circuit for the *1-Bit Full Adder* component. Our experimental system interpreted the circuit correctly. Table 1 summarizes other circuits analyzed correctly. These circuits contains from 3 to 42 devices. The processing time for each circuit is indicated in the table.

5. Conclusion

Knowledge representations developed for computer vision are applicable to other tasks as well. In particular schema labelling techniques are suitable for verifying VLSI chip designs. While previous CAD algorithms were confined to low level recognition, to specific technologies, or to specific design methodologies our schema labelling techniques are more general. In particular, they suggest methods for reversing the VLSI design process to recognize structural descriptions of the chip actually implemented directly from the mask artwork. The experimental results indicate that the system functions correctly and could be extended to production use. In future work, we would like to add electrical constraints to the knowledge base to augment the current logical rules [deKleer&Sussman,1978]. Also it would be desirable to develop a specification language for encoding the knowledge about different VLSI device technologies.

Acknowledgements

We are grateful to the people in the Laboratory for Computational Vision for their assistance in performing this work. This work was supported by NSERC under grants

A5502, and SMI-51, the University of British Columbia.

References

J. D. Bastian et.al. (1983) Symbolic Parasitic Extractor for Circuit Simulation (SPECS) *Proc. 20th Design Automation Conference,* June, 1983, pp 346-352

M. B. Clowes (1971) On Seeing Things, *Artificial Intelligence 2,* pp.79-116.

J. de Kleer & G. J. Sussman (1978) Propagation of Constraints Applied to Circuit Synthesis, AIM-485, MIT AI Lab, Cambridge, Mass.

E. C. Freuder (1978) Synthesizing Constraint Expressions, *CACM 21, no. 11,* November, 1978, pp.958-966.

W. Havens & A. Mackworth (1983) Representing Knowledge of the Visual World, *IEEE Computer 16, no. 10,* October, 1983, pp.90-96.

W. Havens (1984) A Theory of Schema Labelling, TR-84-16, Computer Science Dept., Univ. of British Columbia, Vancouver, Canada.

M. Hofmann & U. Lauther (1983) HEX: An Instruction Driven Approach to Feature extraction, *Proc. 20th Design Automation Conference,* June, 1983, pp 331-336

D. A. Huffman (1971) Impossible Objects as Non-Sense Sentences, in *Machine Intelligence 6,* B. Meltzer & D. Michie (eds.) Edinburgh Univ. Press, Edinburgh, pp.295-323.

F. Luellau, T. Hopken & E. Barke (1984) A Technology independent blovk extraction algorithm, *Proc. 21th Design Automation Conference,* June, 1984, pp 610-615

E. L. Lusk, & R. A. Overbeek (1982) A LMA-Based Theoram Prover, Argonne National Laboratory, Mathematics and Computer Science Division report, 82-75, December, 1982.

A. K. Mackworth (1977a) On Reading Sketch Maps, *Proc. 5-IJCAI,* MIT, Cambridge, Mass.,pp.598-606, August 1977.

A. K. Mackworth (1977b) Consistency in Networks of Relations, *Artificial Intelligence 8, no. 1,* February, 1977.

A. K. Mackworth [1977c] How to See a Simple World, *Machine Intelligence 8,* E.W. Elcock & D. Michie (eds.), Halstead Press, N.Y., pp.510-537.

A. K. Mackworth & E. C. Freuder (1982) The Complexity of Some Polynomial Network Consistency Algorithms for Constraint Satisfaction Problems, TR-82-6, Computer Science Dept., Univ. of British Columbia, Vancouver, Canada.

O. A. Marvik (1984) A Method for IC Layout Verification, *Proc. 21th Design Automation Conference,* June, 1984, pp 708-711

S. P. McCormik (1984) EXCL: A Circuit Extractor for IC Designs, *Proc. 21th Design Automation Conference,* June, 1984, pp 616-623

M. Takashima et.al. (1982) Programs for Verifying Circuit Connectivity of MOS/LSI mask artwork, *Proc. 19th Design Automation Conference,* June, 1982, pp 544-550

D. L. Waltz (1972) Generating Semantic Descriptions from Drawings of Scenes with Shadows, AI-TR-271, M.I.T. A.I. Lab, Cambridge, Mass.

A. S. Wojcik, J. Kljaich & N. Srinivas (1984) A Formal Design Verification system Based on an Automated Reasoning System, *Proc. 21th Design Automation Conference,* June, 1984, pp 641-645

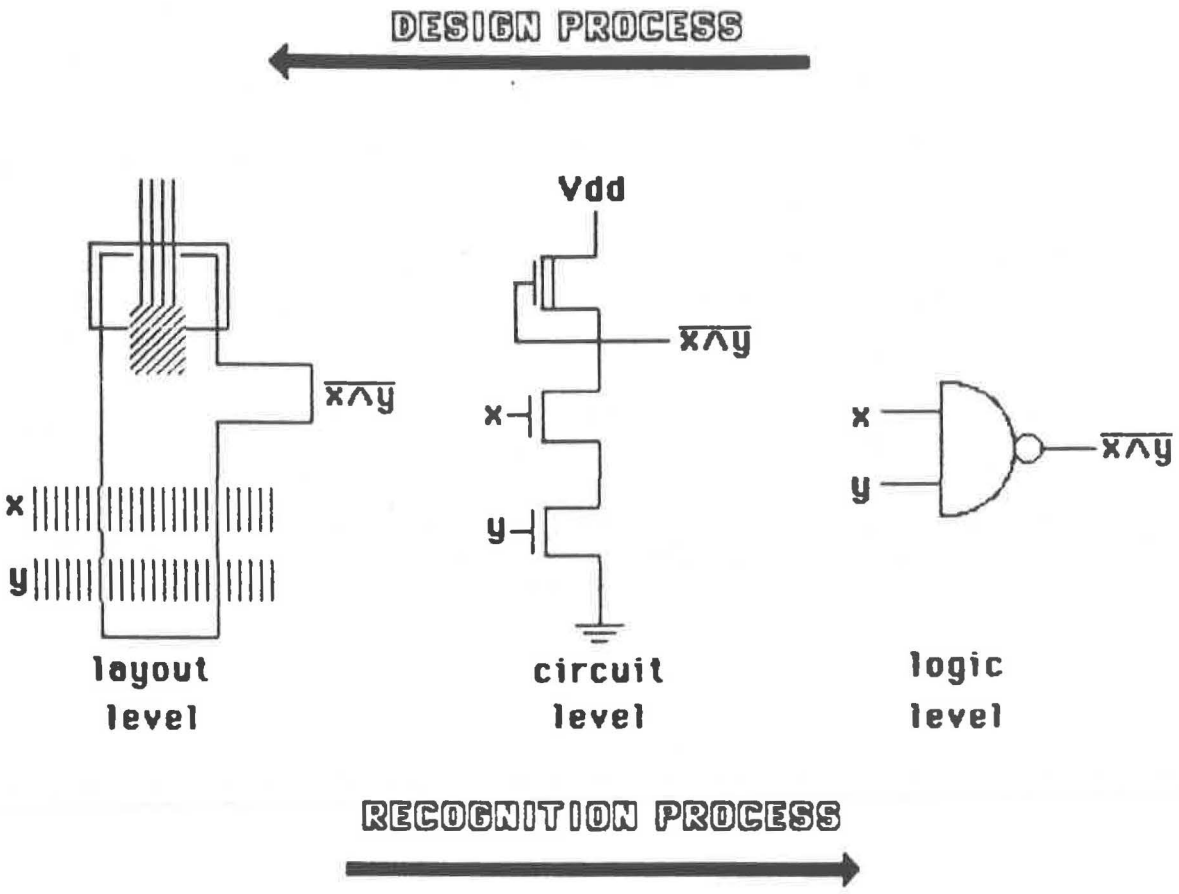

Figure 1 :The Design and Recognition Processes

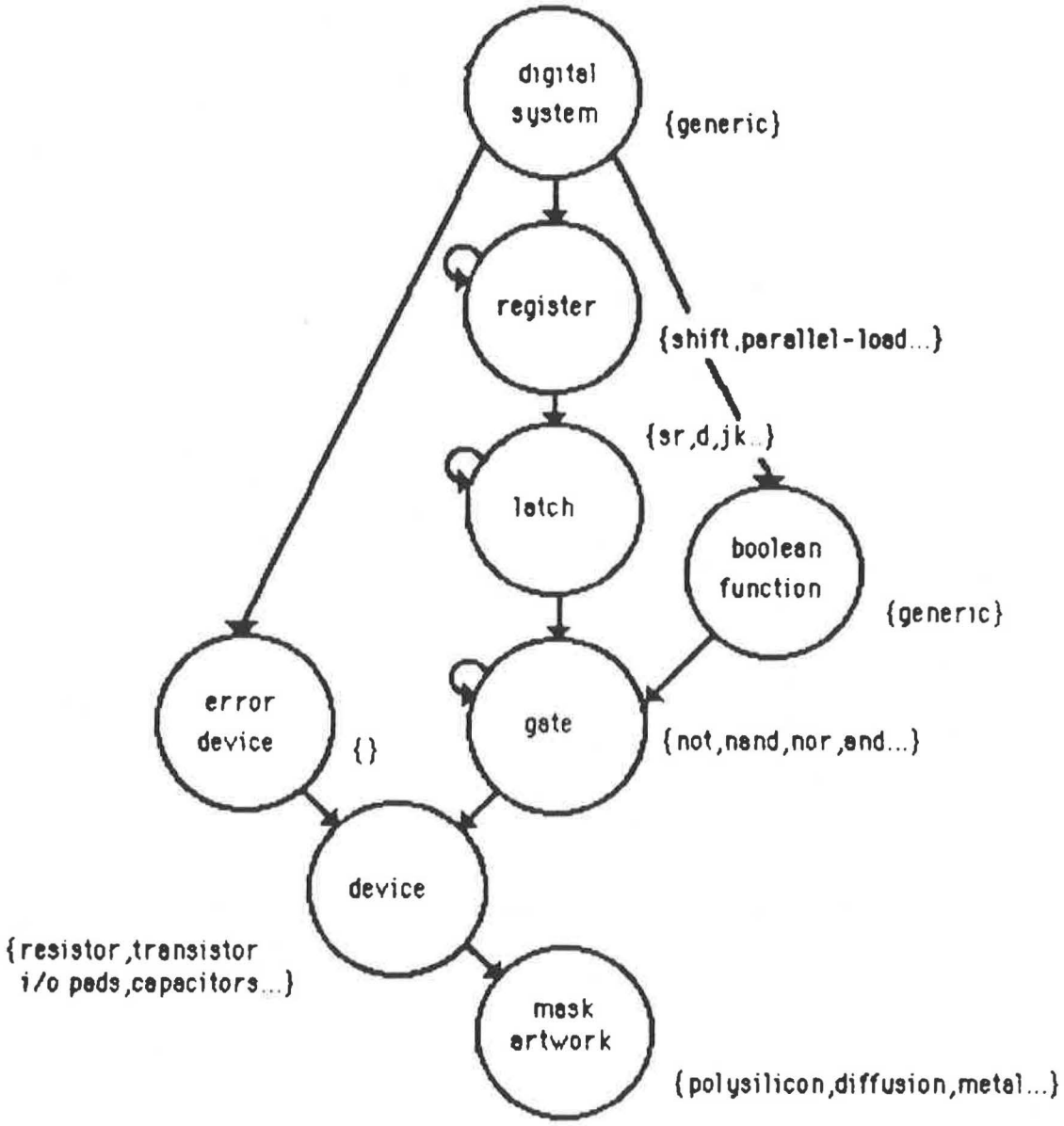

Figure 2: Composition Hierarchy

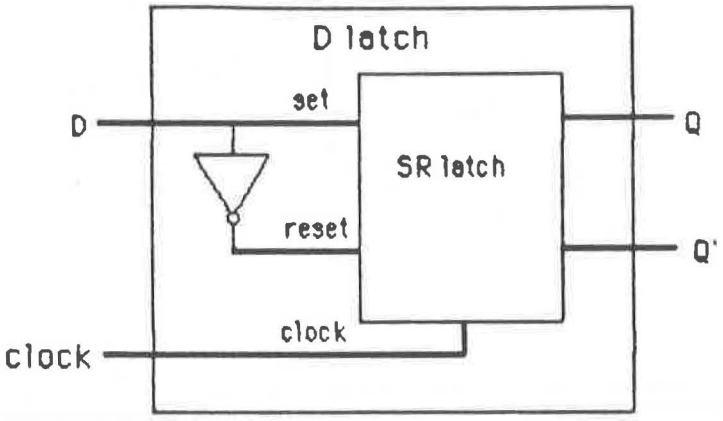

Figure 3:Composition for a "D-Type" Latch

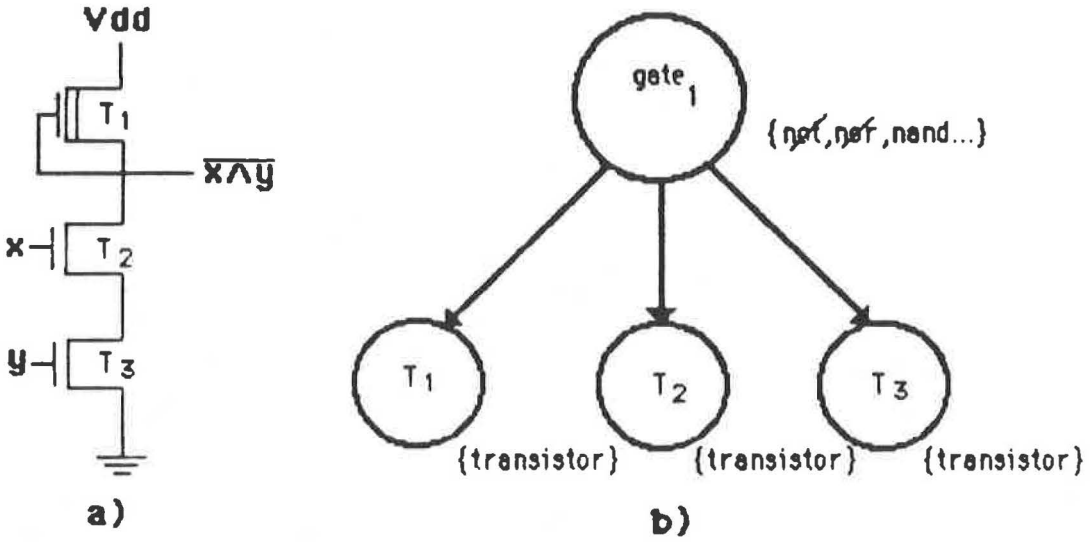

Figure 4: Network Description for NAND gate

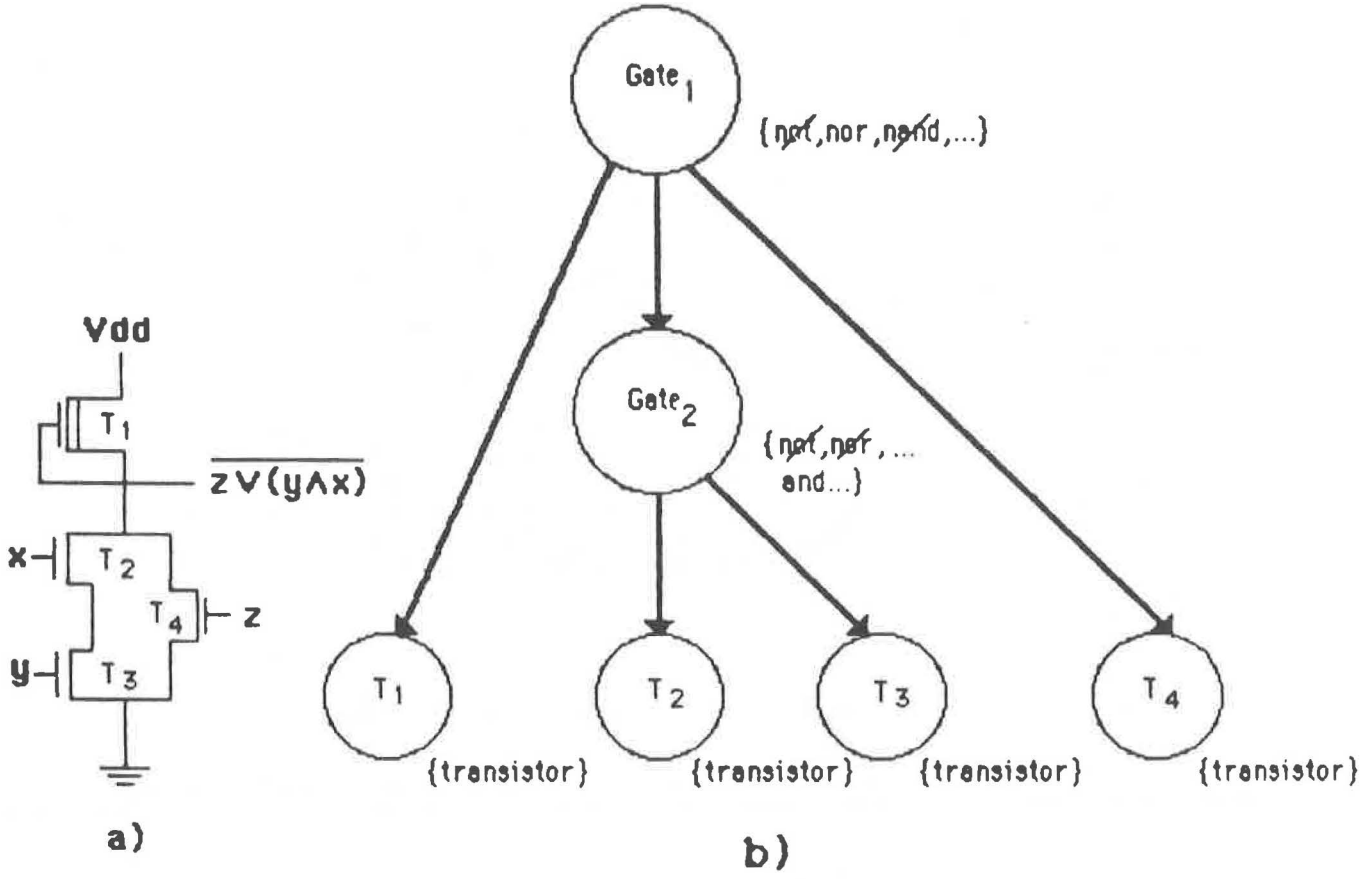

Figure 5: Network Description for Random Logic

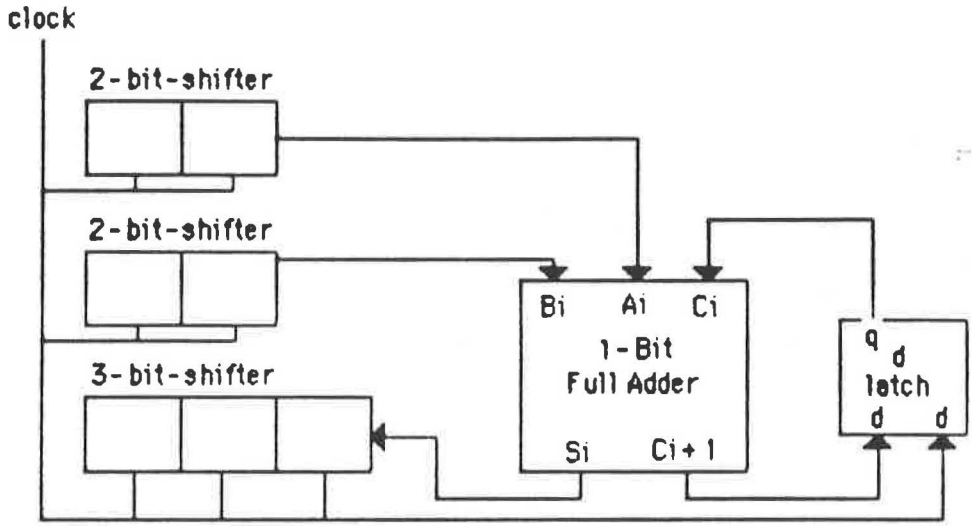

Figure 6: Serial 2-Bit Adder

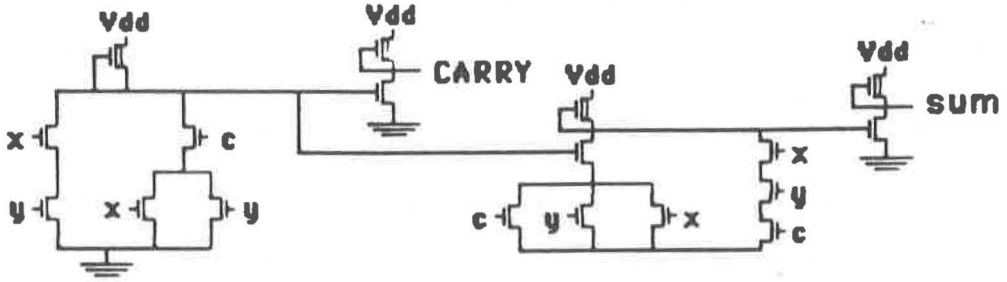

Figure 7: 1-Bit Full Adder

function extracted	# transistor	cpu time
$\overline{(x \vee y)}$	3	6 sec
$\overline{(x \vee y \vee z)}$	4	9 sec
$\overline{((x \wedge y) \vee (z \wedge w))}$	5	18 sec
$\overline{((x \wedge y) \vee (z \wedge w) \vee (z \wedge v))}$	6	28 sec
$\overline{x}$ $\overline{(x \wedge y)}$ 2 error transistors	7	18 sec
carry function sum function	18	109 sec
3–bit–shifter	42	164 sec

Table 1: more examples