1988

The MICON system for computer design

William P. Birmingham
Carnegie Mellon University

Anurag P. Gupta

Daniel P. Siewiorek

Carnegie Mellon University.Engineering Design Research Center.

The MICON System for Computer Design

by

William P. Birmingham, Anurag P. Gupta,
Daniel P. Siewiorek

EDRC 18-10-89

The MICON System for Computer Design[1]

William P. Birmingham
University of Michigan
Electrical Engineering and Computer Science Department
Ann Arbor, Michigan 48109
313-665-8598
wpb@crim.eecs.umich.edu

Anurag P. Gupta
Daniel P. Siewiorek
Carnegie Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, PA 15213

November 4, 1988

[1]This **work was funded by in part by National Science Foundation grant DMC-8405136 to** the Demctcr Project, and the Engineering Design **Research Center, Carnegie Mellon University, an NSF engineering research center supported by grant** CDR-8522616.

Abstract

The MICON system is an integrated collection of programs which automatically synthesizes small computer **systems from high level specifications. The system address multiple** levels of design, from logical through physical, **providing a rapid prototyping capability. Two programs form MICON**'s nucleus: **a** knowledge-based synthesis **tool called MI; and, an automated knowledge acquisition tool named CGEN** which **is used to teach MI how to design. Other tools in the MICON system are an integrated database and associated data management tools. The system is fully functional, having been used to generate working designs. This paper describes the architecture and operation of the MICON system.**

1 Introduction

Semiconductor technology is providing tremendous opportunities for high processing power, reliable, and low-cost computing in the office or lab. Many of these computers, spanning the range from personal computers to super-mini's, are based on commercially available micro-processor family components. The high-end microprocessors available today commonly support virtual memory and main memory cache. Complementing these devices are a full array of high-performance dedicated processors (e.g. graphics controllers and numerical co-processors) and communication components. A very sophisticated machine can be constructed almost entirely from *off the shelve* components. It appears that semiconductor designers have just begun to tap the well of possibilities, promising more performance and function in future chips.

However, there is a price for all this. Sophisticated components require sophisticated hardware designers; designers must be experts in high-speed logic design[2] and computer architecture. The days of simple designs consisting of a processor and a serial input/output (SIO) interface are long gone. The smallest of today's computer systems contain at a minimum, disks, buses, and graphics terminal interfaces. Workstations must utilize a minimum of three levels of memory hierarchy to achieve maximal performance. In addition, such traditional *extras* as networking and graphics are standard equipment All of this places a strain on hardware designers trying to keep abreast of a rapidly evolving technology.

In the marketplace where product lifecycles are, if anything, shrinking, the competition is growing stronger. So, the hardware designer is being squeezed by three forces: a reduced design time, the ceaseless demand for increased performance at a lower price, and a constantly evolving technology. It appears unlikely that these forces will abate of themselves; therefore, some assistance must be provided in the form of computer-aided design (CAD) tools. The MICON system is intended to provide support for computer hardware designers.

The objective of the MICON system is to reduce the time required to construct a hardware system. The approach to achieving this objective is based on the following two points: capture and disseminate design expertise in a format that *actively* assists designers; provide a tool environment which supports all aspects of computer design.

By making design expertise commonly available, a number of benefits accrue. The learning curve facing designers when a new component is introduced is reduced and the number of new components which a designer can consider is expanded. The number of errors in the design, and the amount of time spent in *trial and error* learning can be reduced. This allows the designer more time to create alternative designs, or to just create a working design more quickly.

The creation of computer hardware is not limited to logic design and physical design; other tasks, such as reliability analysis, must be performed as well. By providing a design environment which supports multiple experts, the designer can single-handedly address the entire spectrum of design activities. Through such an environment, data consistency is handled more efficiently and fewer design errors will result.

The MICON system utilizes a set of technologies. Artificial intelligence (AI) is used for design synthesis and the acquisition of design expertise. Databases are used to provide consistent views of data to all tools. Networking allows the system to efficiently share common resources, such as the database, across many users of the system. The following paper describes the architecture of the MICON system, provides examples of its use, and describes a set of experiments to test its viability. The paper begins with a comparison of MICON to other design systems.

---

[2]**Processors today commonly run in the 16-25 MHz range.**

2 Contrasts with Related Systems

Synthesis is the process of producing an artifact that meets some set of specifications. In general, the specifications are abstract relative to the details needed to build the artifact The synthesizer's task is to provide the missing information. The mechanisms used to implement the synthesis system are as widely varying as the types of synthesis tasks. The next several paragraphs discuss related synthesis work in digital domains, but work is also being done in other domains, such as: analog circuit design [20] and mechanical engineering[16].

Digital system synthesis has been an area of particular interest in the CAD field[29]. Interest has grown from graphics-oriented schematic capture systems to fully automatic systems capable of producing sophisticated, fully operational designs.

Recently, much of the work in digital system synthesis has been oriented around the development of application-specific integrated circuits (ASIC)[3]. In a broad sense, two approaches exist, depending upon the type and abstraction level of the input and the resulting design. The first approach proceeds from a high-level behavioral description into an IC. The types of systems produced typically include micro-processors and mini-computer CPUs. The SAW project [3231], evolved from the CMU-DA project [14], synthesizes digital systems from a high-level behavior language, ISP[3]. SAW's objective is to produce a very-large scale (VLSI) IC that implements the machine described by the ISP input program. The synthesis process consists of transforming the behavioral description into a technology-independent data flow representation called a VT graph. The data flow graph is then subjected to a set of complex operations, resulting in a data-path and controller design represented as a set of register transfer level (RTL) components. The RTL representation is bound to modules in an implementation technology, such as standard cell libraries (see [15] for examples). Examples of other well-known systems which follow the same basic design paradigm can be found in [28][25][27].

The second broad class of ASIC synthesis systems take as input a lower-level description of a system to implement. The descriptions vary, but are typically based on any of the following: a hardware description language, a set of boolean equations, state tables, *et cetera.* These tools produce sophisticated, but smaller scale artifacts than the systems outlined above. Examples of these artifacts, which are used as building blocks of larger digital machines, include: programmable logic arrays (PLA), state-machines, and blocks of combinational logic. Logic synthesizers, typical of this class of tools produce ASICs using a set of pre-defined cells (e.g. standard cells or gate arrays). International Business Machines (IBM) has a long standing and successful effort in this area with the development of a production quality system called LSS[13]. There are a number of other systems which employ a similar paradigm to produce combinational logic and/or state machines[17,4].

Artificial intelligence is being applied to a wide range of synthesis problems. The appeal of AI for synthesis is the ability to use non-mathematically based representations (An example set of representations for digital systems is found is [23]) and to use domain knowledge to reduce the search inherent in many synthesis problems. The success of the XCON/R1 project[24] illustrated the potential of AI technology for synthesis (constructive) problems. Examples of knowledge-based approaches for large digital systems synthesis (the first class of tools outlined above) can be found in [12][33], The DAA system[22], part of CMU-DA, used domain knowledge to aid the generation of an IBM 370 design. The VEXED system[26] is an interactive knowledge-based editor which assists a user in traversing a hierarchy of functional blocks with the aim of creating an NMOS implementation of a design.

In surveying the literature, it is interesting to notice that the design of pieces of computers (e.g. CPU's), not complete computer systems, has driven most of the current synthesis research. However, as these tools become more proficient system integration issues will become prominent The MICON project has been interested in the synthesis of systems through the configuration of existing components. The emphasis of this synthesis approach is on matching the interfaces between components (and in this way is similar to Critter[21], although the intentions and implementations of the systems are very different).

The MICON project originated with MO (a recently dubbed name)[5,8], a knowledge-based system designed to assess the feasibility of automated single board computer synthesis based on commercially available micro-processors. MO embodied, in a more primitive form, many of the concepts used in the design of the Ml system. Several micro-processor families were programmed into the system. A number of designs were produced by MO, and a Z80-based design was built. MO initiated the idea of templates for representing structural information for synthesis. MO, however, had a number of drawbacks which limited its general applicability (see [18] for more

[3] The term application-specific is used *very* loosely, connoting systems ranging from a simple controller to a full micro-processor.

details). These drawbacks lead to the development of M0.5[2,9]. While M0.5 was not fully implemented, the ideas garnished from it were essential to the development of M1. In particular, M0.5 introduced component abstraction which enables M1 to have a much larger degree of freedom in choosing components thereby creating a larger number of interesting designs.

3 The MICON System

The MICON system was originally developed to design small computer systems[4]. The computers designed by MICON consist of the following subsystems:

Processor: designs utilize a micro-processor as the CPU. Presently, the MICON system is capable of designing with the following processors: Motorola 6809, 68008, 68010 and the Intel 80386.

Memory: designs contain read-only memory (ROM), and either static or dynamic memory (SRAM or DRAM). In addition, MICON also supports the use of cache memories where support chips exist in the microprocessor family, such as the Intel 80386 and 80385 combination.

Peripheral: all input/output (IO) is performed by dedicated parts, such as serial IO (SIO) and parallel IO (PIO) devices. IO devices may utilize standard interfaces, such as RS-232-C.

Bus Interface: a selection of standard bus interfaces (e.g. AT bus) are supported by the system.

Support Circuitry: a set of circuitry provides support functions. Examples of support circuitry include address decoders, wait state generators, oscillators and bus drivers. Included in support circuitry are specialized structures for improving system reliability, such as Hamming encoders/decoders.

The complexity of designs produced is roughly equivalent to a small workstation.

MICON is intended to provide a rapid prototyping facility, therefore the overriding concern for the synthesis system (M1) is to quickly generate working designs. Design cost— as measured by a variety of metrics including part cost, board area, and power dissipation— provide a framework for evaluating tradeoffs. Since designs will not be produced in volume, a user is typically willing to lessen the cost constraints in order to get a design quickly, therefore, cost optimization is not necessary. In general, designs produced by the MICON system compare favorably to human generated designs, Section 4 provides more details.

The MICON system is shown in Figure 1. Note that a distinction is made between a *user* and a *domain expert*. The domain expert teaches M1 through CGEN how to design, and uses the data entry tool to add new part models to the data base. The user's goal is creating designs with M1, exploiting the work of the domain expert. If the domain expert imparts sufficient knowledge to M1, the user can be a novice hardware designer. Each of the tools is described below.

M1[18,19] is the synthesis module of the MICON system. Input to M1 is a set of high-level specifications that describe the functionality required of the computer. For example, a user may specify the type of micro-processor, amount and type of memory, and the amount and type of input/output devices required. An additional set of parameters describing *multiple objective criterion*, such as design constraints (*e.g.* board size, cost), and system reliability requirements, are input to the program. M1 uses its knowledge of components and micro-processor system structures to develop a design that satisfies the requirements given to the system. During the design process, M1 interacts with the user to resolve tradeoffs in the evolving design.

M1 is implemented as a rule-based system written in OPS/83. The rule-base consists of knowledge about the components M1 can design with, micro-processor system design techniques, and design for reliability techniques. In order for the MICON system to produce competitive designs the rule-base must be updated with new components and design techniques as they are developed.

ASSURE[11] (Automated SynthesiS for REliability) is a sub-module within M1 that analyzes and modifies designs for improved reliability. ASSURE's input consists of reliability criteria, such as mean time to failure or mission time. ASSURE uses external reliability analysis tools to analyze the quality of the evolving design. Designs failing to meet the user specified criteria are modified by a variety of techniques, ranging from simple

[4]MICON is being applied to design problems in other domains, such as design environments (frameworks) and mechanical engineering[7,10].

System data	
Total part models	382
Total rules	1050

Table 1: Parameters indicating the size of MICON after the experimental period.

integrated circuit package changes to the addition of entire structures (e.g. error correcting codes on memory). While ASSURE is closely tied to Ml, it has a separate, distinct problem-solving architecture.

CGEN[6,7] (JCode GENerator) is the knowledge acquisition tool for the Ml module. Typically, a rule-base can only be updated by whose intimately familiar with the program, making the acquisition of new knowledge difficult. CGEN allows a hardware designer, not familiar with Mi's implementation details, to add his expertise to the rule-base. Inputs to CGEN consist of schematic drawings and simple equations.

A suite of tools are used to transform the output of Ml into a complete computer, performing the physical design function. There are three steps in physical design. The first step is placement and routing of the design for a board. A set of commercial physical design tools, TANGO-PCB[1], *is* used for this task. The second step is to fabricate the board. Two board fabrication technologies are available: wire-wrapping which is done in-house; and, printed circuit board (PCB) which is done at a local PCB manufacturer. The final step is to populate the board with components and test it

The database (DB) is the central repository for the part models used by all modules in the system. The data base is built on a commercially available product, Informix[30]. To ensure consistent information in the MICON system, all modules get data from a common central database - local databases do not exist.

The system is tightly integrated, wherein all tools communicate via UNIX interprocessor communication (IPC) links to the database. The database uses a server to establish communication links with each tool, or client. This scheme allows the database to run *dp* a separate process. In addition to providing a great deal of flexibility, such as allowing a large number of clients to be served simultaneously, it allows the dedication of a single workstation on the network as the database node. This centralizes the database management functions and eliminates many inconsistency problems. Figure 2 depicts the database server communicating with a set of clients running on separate workstations across the network.

4 Experiments and Other Experience

MICON, a working system, was subjected to a set experiments to test its design abilities. The experiments involved the following steps:

1. Teaching the system, using CGEN, to design with the set of micro-procesors mentioned previously (sec Section 3).
2. Exercising Ml to generate a set of designs.
3. Pass an Ml generated design through the physical design and manufacturing processes.

The experiments were conducted using a set of designers who were not implementers of MICON[5]. The designers captured data and knowledge about a large number of components, Table 1 provides an overview of the data (see [7] for complete details). The database contained every type of part necessary to build the computers, ranging from microprocessors to NAND gates, capacitors to RS-232-C ports (DB-25s). Mi's knowledge-base contained over 1050 rules, all of which were created by CGEN. These rules described how to design with each of the parts in the database.

With its knowledge-base and database, Ml was able to create an interesting variety of designs. Since knowledge is acquired concerning parts individually, unique combinations of parts can be configured into a design. The ultimate limitation on the designs comes from:

[5]Except for Birmingham.

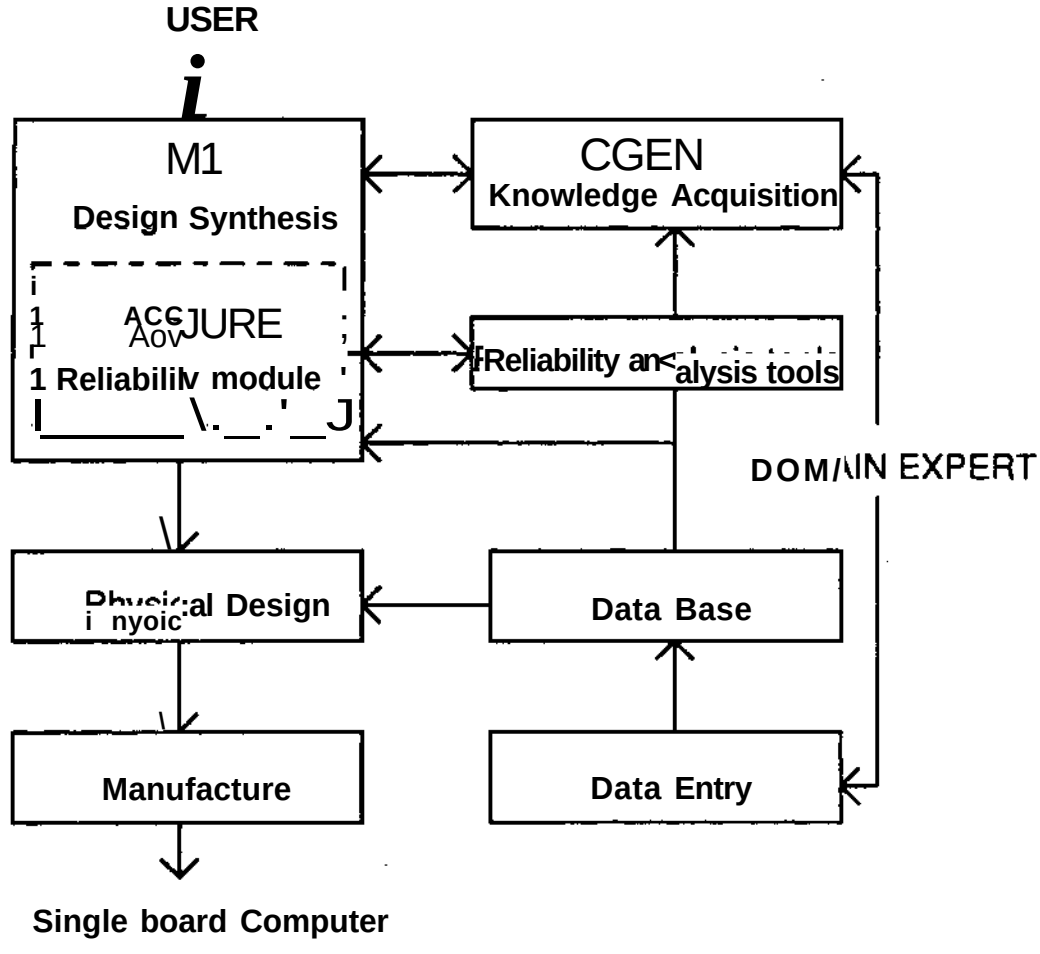

Figure 1: The MICON system.

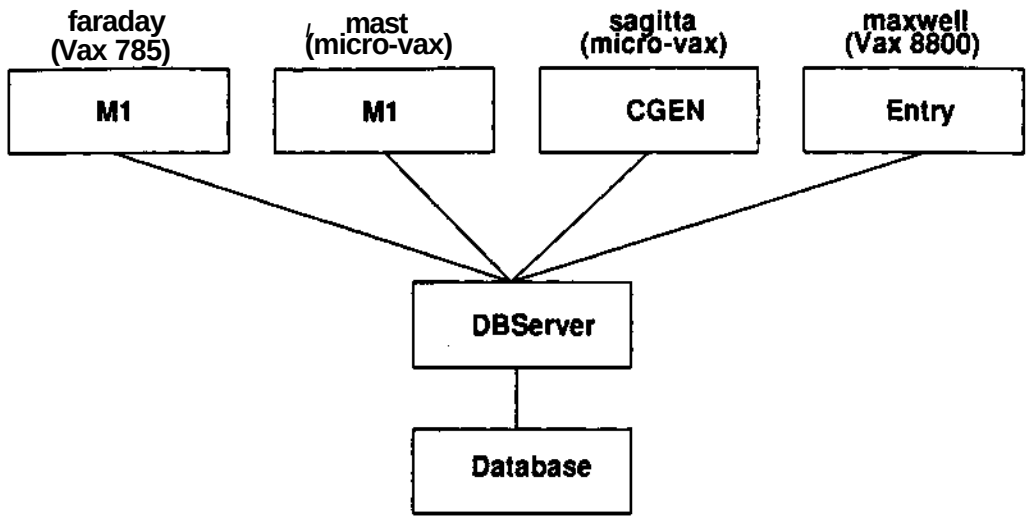

Figure 2: Multiple clients being supported simultaneously by the database server.

Design Specifications				
Set	Amount SRAM	Amount ROM	Amount PIO	Amount Timer
1	4 KBytes	1 KByte	1	0
2	4 KBytes	1 KByte	0	1
3	58 KBytes	4 KByte	1	1
4	58 KBytes	4 KByte	2	2

Table 2: Partial specifications of the designs created by Ml.

- the size of the memory space
- the size of the IO space (if applicable)
- physical constraints (power, area, cost)

The specifications used for each design are shown in Table 2. Each of these designs represents a point in the large space of designs that can actually be generated.

The design Ml produced for the 68008 using specification Set 3 from Table 2 was fabricated and is running. Analysis of the designs shows:

- The designs work at the expected clock rates.
- The number of parts used is no greater than those generated by a human designer.

The favorable comparison with hand design in terms of part count is to be expected since the designs are built almost entirely from VLSI components.

Other Ml generated designs were verifed by manually converting the wire-list to a schematic. While creating the schematic the correctness of the design was checked. Several errors were found, which fell into one of two categories:

missing part: several support parts (e.g. pull-up resistors and by-pass capacitors) were mistakenly absent from some templates.

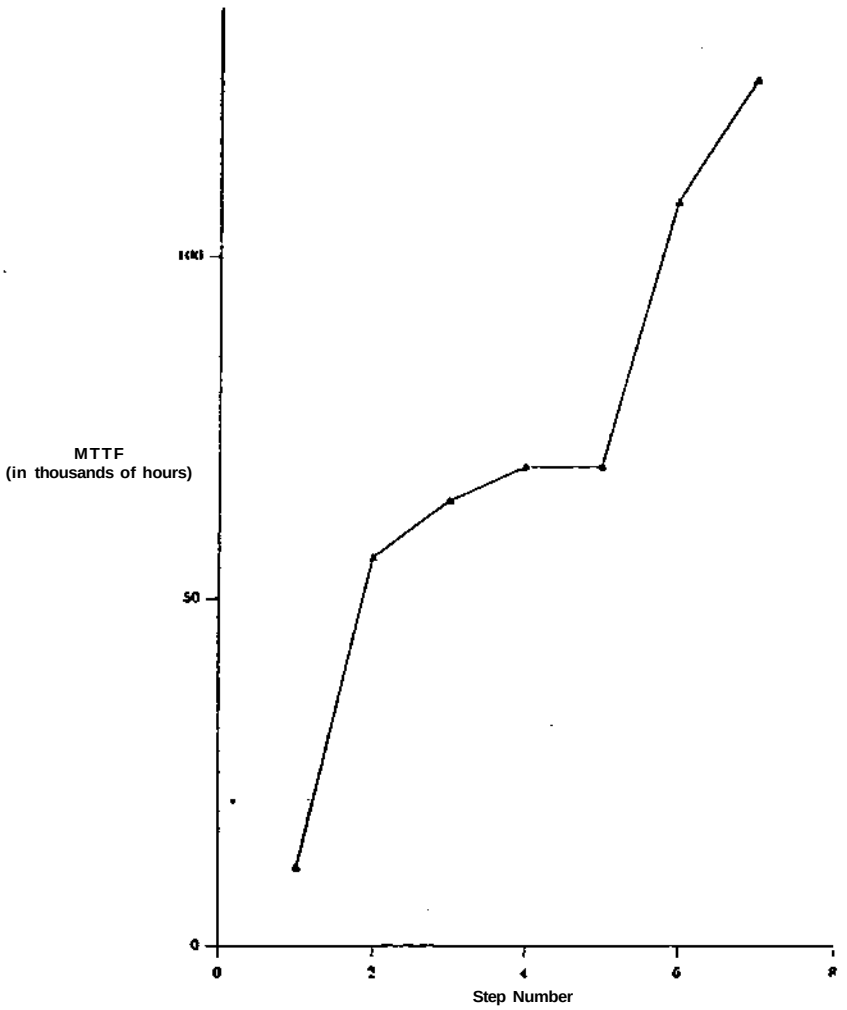

Figure 3: ASSURED MTTF improvement plot for 6809 design

missing connections: some necessary connections between components were not made.

The errors all stem from mistakes in the schematics input to CGEN, which then propagated to Mi's knowledge-base. The schematics containing the errors were corrected and new rules were generated. All designs were then bug free.

A 6809 based design produced by Ml was analyzed and modified by ASSURE for improved system reliability. Figure 3 graphs the steps taken by ASSURE when the reliability goal was maximizing mean-time-to-failure (MTTF). ASSSURE successively identifies the least reliable sub-system or part and applies a reliability enhancement technique for it. ASSURE starts by applying simpler fault-intolerent techniques *{e.g.* changing a package from plastic to ceramic, upgrading the quality of the package) until these result in marginal improvements (corresponding to the flattenning-out of the curve after five steps in Figure 3). ASSURE next applies fault tolerent techniques for the most failing sub-system. Here, Single Error Correcting/Double Error Detecting (SEC/DED) structures were added to the SRAM and ROM array. ASSURE also attempted triplicating the processor and the io peripherals but found that the fault-intolerance did not make up for the failure rate of additional support circuitry it had to add and hence rejected those changes.

Since the experimental period the MICON system has been used very extensively. The user group has grown to over a dozen, including industry. Two workshops with industrial partners were held. The MICON system

functioned without flaw in this more realistic environment[6]. MICON is now being evaluated at several industrial sites.

Recently, the system has been used to generate more sophisticated designs than those described above. In particular, an 80386 design with cache, co-processors, and AT-bus interface was designed by the system. Work is beginning on a new multi-processor computer. Also, the capability to form P solid-model of the design to evaluate its mechanical properites *(e.g.* enclosure interference), and thermal properties using an air-flow analysis package, is under development.

MICON is also being used for a micro-processor interfacing course. As part of the laboratory requirement, students build knowledge-bases of 8086 family components. Designs are then generated and verified. MICON is a very powerful aid for teaching students good design methodology; a topic which is difficult to cover in the short period of one semester. Student in this class gain exposure to design issues not generally experienced until they begin industrial jobs.

5 Summary

MICON is a system which supports all aspects of computer design from logic design through physical design to manufacture. One of the major strengths of the system is its ability to readily incorporate new technologies, allowing it to track the state-of-the-art. MICON has proven its design capabilities through extensive use both inside and outside of the laboratory.

Research is continuing on MICON. Several aggressive designs are being created with the system. In addition, MICON is being ported to domains outside of computer design.

6 Acknowledgments

The MICON system is the result of the efforts of many people. The authors thank the following people for their efforts: Joe Najjar, Raj Merchia, Andrea Dusseau, Nino Vidovic, Ajai Kapoor, Audrey Brennan, and Patrick Edmond.

References

[1] Accel Technologies, Inc. *TangoPCB User's Manual.* 1986.

[2] N. Balram, W.R Birmingham, S. Brady, D.P. Siewiorek, and R. Tremain. The micon system for single board computer design. In *1st Conference on Application of Artificial Intelligence to Engineering Problems,* Computational Mechanics, April 1986.

[3] M.R. Barbacci. Instruction set processor specification (isps): the notation and its applications. *IEEE Transactions on Computer-Aided Design,* CAD-5(4), 1981.

[4] W. Birmingham and J. Kim. Das/logic: a rule-based logic design assistant. In *The Second Conference on Artificial Intelligence Applications,* IEEE Computer Society, December 1985.

[5] W. P. Birmingham. *MICON: A Knowledge-based Single Board Computer Designer.* Master's thesis, Carnegie-Mellon University, Department of Electrical and Computer Engineering, November 1983.

[6] William P. Birmingham. *Automated Knowledge Acquisition for a Hierarchical Synthesis System.* Technical Report CMUCAD-88-25, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1988.

[7] William P. Birmingham and Daniel P. Siewiorek. Capturing designer expertise - the cgen system. In *Submitted to The 26th Design Automation Conference,* IEEE and ACM-SIGDA, IEEE Computer Society, 1989.

[6]The industrial participants, while not manelovent, were interested in testing the capabilities and flaws of the system.

[8] William P. Birmingham and Daniel P. Siewiorck. Micon: a knowledge-based single board computer designer. In *Proceedings of the 21st Design Automation Conference,* IEEE and ACM-SIGDA, IEEE Computer Society, 1984.

[9] William P. Birmingham and Daniel P. Siewiorek. *Single Board Computer Syntfiesis.* Technical Report EDRC-18-02-87, Engineering Design Center, Carnegie Mellon University, 1987.

[10] W.P. Birmingham, A. Kapoor, D.P. Siewiorck, and N. Vidovic. The design of an integrated environment for the automated synthesis of small computer systems. In *To appear in the Hawaii International Conference on System Sciences - 22,* IEEE Computer Society, January 1989.

[11] Audrey A. Brennen. *Automatic Synthesis for Reliability.* Master's thesis, Carnegie-Mellon University, Department of Electrical and Computer Engineering, January 1988.

[12] ED. Brewer and D.D. Gajski. An expert-system paradigm for design. In *23rd Design Automation Conference Proceedings,* IEEE Computer Society, 1986.

[13] J.A. Darringer, D. Brand, J.V. Gerbi, W.H. Joyner Jr., and L. Trevillyan. Lss: a system for production logic synthesis. *IBM Journal of Research and Development,* 28(5):537-545, September 1984.

[14] S.W. Director, A.C. Parker, D.P. Siewiorek, and D.E. Thomas, Jr. A design methodology and computer aids for digital vlsi systems. *IEEE Transactions on Circuits and Systems,* CAS-28(7):634-645, 1981.

[15] E. Dirkes. *A Module Binder for the CMU-DA System.* Technical Report CMUCAD-85-43, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1985.

[16] J.R. Dixon. Artificial intelligence and design: a mechanical engineering view. In *Proceedings of AAAI-86,* Morgan Kaufmann Publishers, 1986.

[17] D. Gregory, K. Barlett, A. de Geus, and G. Hachtel. Socrates: a system for automatically synthesizing and optimizing combinational logic. In *Proceedings of the 23rd Design Automation Conference,* IEEE and ACM-SIGDA, IEEE Computer Society, 1986.

[18] Anurag P. Gupta. *A Hierarchical Problem Solving Architecture for Design Synthesis of Single Board Computers.* Master's thesis, Carnegie-Mellon University, Department of Electrical and Computer Engineering, February 1988.

[19] A.P. Gupta and D.P. Siewiorek. Hierarchical design synthesis - the ml system. In *Submitted to The 26th Design Automation Conference,* IEEE Computer Society, 1989.

[20] R. Harjani, R. Rutenbar, and L.R. Carley. A prototype for knowledge-based analog circuit synthesis. In *24th Design Automation Conference,* IEEE Computer Society, 1987.

[21] V.E. Kelly. The critter system: automated critiquing of digital circuit designs. In *Proceedings of the 21st Design Automation Conference,* IEEE and ACM-SIGDA, IEEE Computer Society, 1984.

[22] TJ. Kowalski. *An Artificial Intelligence Approach to VLSI Design.* Kluwcr Academic Press, 1985.

[23] D. Bobrow M. Stefik, A. Bell, H. Brown, L. Conway, and C. Tong. *The Partitioning of Concerns in Digital System Design.* Technical Report VLSI-81-3, Xerox Palo Alto Research Centers, 1981.

[24] J. McDermotL Rl: a rule-based configurer of computer systems. *Artificial Intelligence,* 19(2), 1982.

[25] M.C. McFarland. Using bottom-up design techniques in the synthesis of digital hardware from abstract behavioral descriptions. In *23rd Design Automation Conference Proceedings,* IEEE Computer Society, 1986.

[26] T.M. Mitchell, L.I. Steinberg, and J.S. Shulman. A knowledge-based approach to design. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* PAMI-7(5), September 1985.

[27] B.M. Pangrle and D.D. Gajski. Design tools for intelligent silicon compilation. *IEEE Transactions on Computer-Aided Design,* CAD-6(6), 1987.

[28] N. Park and A.C. Parker. Sehwa: a software package for synthesis of pipelines from behavioral specifications. *IEEE Transactions on Computer-Aided Design,* CAD-7(3), 1988.

[29] T.C. Raymond. Lsi/vlsi design automation. *Computer,* 14(7)$_f$ July 1981.

[30] Incorporated Relational Database Systems. *Informix-SQL Relational Database Management System and Reference Manual.* 1986.

[31] D.E. Thomas, E.M. Dirkes, R.A. Walker, J.V. Rajan, J.A. Nestor, and R.L. Blackburn. The system architect's workbench. In *25th Design Automation Conference,* IEEE Computer Society, 1988.

[32] R.A. Walker and D.E. Thomas. Design representation and transformation in the system architect's workbench. In *IEEE International Conference on Computer-Aided Design,* IEEE Computer Society, 1987.

[33] W.H. Wolf, TJ. Kowalski, and M.C. McFarland. Knowledge engineering issues in vlsi synthesis. In *Proceedings ofAAAI-86,* Morgan Kaufmann Publishers, 1986.