The interested reader is referred to a survey paper by Breuer\textsuperscript{2} for a comprehensive bibliography of previous work in computer design automation.

REFERENCES

1 D F GORMAN J P ANDERSON
   \textit{A logic design translator}
   1962 Proc FJCC pp 86-96

2 M A BREUER
   \textit{General survey of design automation of digital computer}
   Proc IEEE Vol 54 No 12 December 1966 pp 1708-1821

PANELISTS

Donald F. Gorman
RCA
Cherry Hill, New Jersey
Roy L. Russo
IBM Watson Research Center
Yorktown Heights, New York
Wilm E. Donath
IBM Watson Research Center
Yorktown Heights, New York
Melvin A. Breuer
University of Southern California
Los Angeles, California

Functional design and evaluation

by DONALD F. GORMAN

\textit{Radio Corporation of America}
Cherry Hill, New Jersey

The design of information processing systems consists of the development of (1) the initial marketing specifications, (2) a functional design to meet these specifications including a detailed logic design to achieve these functions, and (3) an implementation of this logic design, in either hardware or stored microprogram, using the latest technology.

Functional design encompasses all activities involved in producing the functional design from the behavioral specifications and the conversion of this design into a logic design.

Evaluation of a design is the process of determining if the functional design, the logic design, and the implementation satisfy the initial behavioral specifications.

What next?

Some areas in which progress must be made in order to bring the present efforts to fruition will now be considered.

Provided that a fixed control philosophy is assumed, design translation techniques can be applied to a functional design to produce a logic design. However, in order to optimize a design, it must be possible to merge several control methods, such as synchronous, asynchronous, a centralized clock, distributed control, etc., into a single design. Techniques must be developed to determine the control method best suited for each portion of the design, and translation techniques are needed to produce a single coordinated design utilizing several different control philosophies.

Design translation techniques will provide, for the first time, sufficient detailed information about alternate designs that tradeoff studies may be effectively undertaken with respect to hardware vs. software, synchronous vs. asynchronous, etc. When criteria to be used in these tradeoffs have been established, the tradeoffs themselves may be incorporated into the translation procedure and designs may then be optimized over a wider range of criteria.

Translation from a behavioral specification to a functional design is the truly creative part of the design activity and is currently considered to be an art rather than a science. Design experience using translation techniques and the ability, via automation, to analyze
numerous designs will eventually lead to a methodology for automating aspects of this phase of design.

Evaluation of information processing systems encounters even more problems than design. One of the fundamental problems today is that there is general disagreement over what constitutes a “good” design. Assuming that satisfactory criteria can be found, techniques for rapidly evaluating systems with respect to these criteria are needed.

Analytic procedures are currently being developed, but are, as yet, far from being general enough for adaptation to automation.

Simulation, which is the only evaluation technique available today, suffers from a severe handicap—inefficiency—in both the writing and running of simulation programs. Efficiency will be achieved and simulation will become a practical design aid through the following steps: (1) the use of simulation program generators that will facilitate the creation of customized simulation programs, (2) the generation of simulation models that allow a detailed view of the area of immediate interest to be combined with a very general view of other portions of the system, and (3) the generation of integrated simulation programs based on such models.

Summary

Design automation, which was first used in the manufacturing and physical engineering areas, has made considerable progress in logical design and is now making inroads into functional design, previously considered to be an entirely creative process. The use of simulation and design translation is the first step in transforming the functional design of both the hardware and the software of information processing systems from an art to a science.

The logic-to-hardware interface area of design automation

by ROY L. RUSSO

IBM Watson Research Center
Yorktown Heights, New York

The logic-to-hardware interface area is concerned with the problems of translating a logic description into a high level physical description. The logic description is an interconnection of the primitive circuits, such as And's, Or's, Nor's, Flip-Flops, etc., that are to be used in the physical packages. A high level physical description takes into account, for example, partitioning of the logic into physical packages, but does not include relative placement of the primitive circuits within a package, or pin assignment, wire routing, etc.

This interface area is extremely important at the present time for two basic reasons. First, the present integrated circuit technology imposes critical and interrelated constraints which must be satisfied during this phase of the design process if the technology is to be used effectively. These constraints include minimizing the number of integrated circuit chips or chip types to reduce cost and specifying chips that can be tested adequately and economically. Second, decisions made in specifying the high level physical description determine to a great extent, whether or not severe machine design goals, such as fast cycle times, will be met.

The problems that design automation programs must help solve for the logic-to-hardware interface area are best categorized by the class of machine being designed: either a small low-performance one or a large, high-performance machine.

In the small machine category, chip type minimization and machine cycle time constraints can be set aside to make the problem more amenable to attack and the cost of the computer programs more reasonable. Since manufacturing costs will depend almost directly upon the number of unique chips in the machine, the major problem in this case will be to partition so as to reduce that number. The relatively small number of terminals on a chip compared to the large number of circuits on the chip makes this partitioning extremely difficult. It also increases the number of tests which must be applied to the chip and this raises the cost of testing. Hence, the effect of the implementation on testing must also be considered at this level. If the packaging hierarchy consists of logic on chips, chips on carriers, etc., then the problems become more difficult and the techniques must be extended.

The problems increase enormously for large, high-performance machines because the number of constraints and their complexity increase sharply. In particular, chip type minimization and proposed machine cycle time constraints must be considered. (The low volume of machines implies that the same chip type will have to be used repeatedly in the machine to reduce costs.) Whereas in the small machine case, the specific placement of chips in the packaging hierarchy was not important, it must now be considered because of its effect on the cycle time. Hence, for this class of machines, there are problems of partitioning logic to chips in such away as to minimize chip types, and of placing the individual chips in the packaging hierarchy in such a way as to satisfy a proposed short cycle time with its implied constraints.