Hierarchical Design for VLSI: Problems and Advantages.

by
W.M. vanCleemput
Computer Systems Laboratory
Stanford University
Stanford, California 94305.

ABSTRACT

This paper describes the hierarchical design process for VLSI circuits and discusses the potential benefits and disadvantages.

1. INTRODUCTION

Over the past decade software designers have learned to cope with increasingly complex programs. In order to deal with this complexity a structured programming methodology has been developed. The advent of VLSI technology has brought similar complexity within the reach of hardware designers.

The objective of this paper is to explore the hardware design process and the problems that hierarchical design approaches create as well as their advantages.

Figure 1 shows the major steps and interactions in the hardware design process. The designer starts with a set of initial specifications that are often incomplete and possibly incorrect. The designer makes a number of design decisions both in terms of logic design and physical design. In the mean time he also modifies or refines the specifications until a final design is reached.

Design decisions by the designer can take one of two major forms:

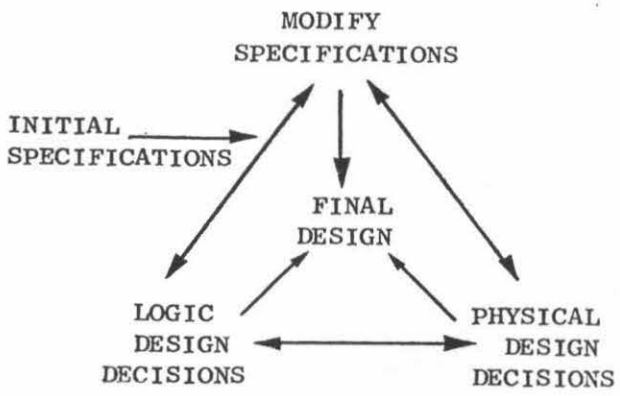

Figure 1: The Hardware Design Process

1. top-down decomposition of a behavior specification into less complex behavior specification modules.

2. bottom-up combination of physical building blocks into larger building blocks.

At some point in time the designer has to map the behavior specifications into some of the building blocks and to assure himself that this mapping will indeed result in a correctly operating design.

From the previous discussion it is clear that the design process is a combination of top-down and bottom-up processes that are happening concurrently in the designer's mind until he reaches a final correct design.

In an idealized model of the design process, there are three concurrent tasks:

1. behavior design, where the designer decomposes the initial specification into subproblems, possibly refining the specification by doing so.

2. structural design, where the designer tries to realize a block or module by the interconnection of more primitive modules.

3. physical design, where the designer tries to realize his design in a given technology.

In most cases the behavior and structural design processes go on concurrently, while the physical design process is done separately. It is, however, quite clear that the physical design process can have a tremendous influence on the behavioral or structural design of a system, necessitating many iterations during a design.

The physical design process itself may also be hierarchical in nature: a system can be partitioned into physical subsystems that in turn can be partitioned into physical subsubsystems and so on.

From the previous discussion on the digital system design process, one can conclude that there exist at least three major design hierarchies: a behavioral hierarchy, a structural hierarchy, and a physical hierarchy. The mapping of a behavior into a structural hierarchy is usually known as thea logic design process, while the mapping of a structure into physical hierarchy is usually known as the physical process.

2. MAJOR DESIGN HIERARCHIES

2.1 BEHAVIORAL HIERARCHY

Figure 2 shows a possible hierarchical decomposition of a general purpose computer. In this particular example the CPU process consists of a fetch cycle and an execute cycle process. The fetch cycle process consists of instruction fetch address calculation and operand fetch subprocesses, while the execute cycle process consists of arithmetic and logical subprocesses. The arithmetic process consists of addition and substraction. Each of those may consist of integer and floating point operations.

This is clearly an example in which the specifications did not pay attention to the possible physical implications of the hierarchical decomposition.

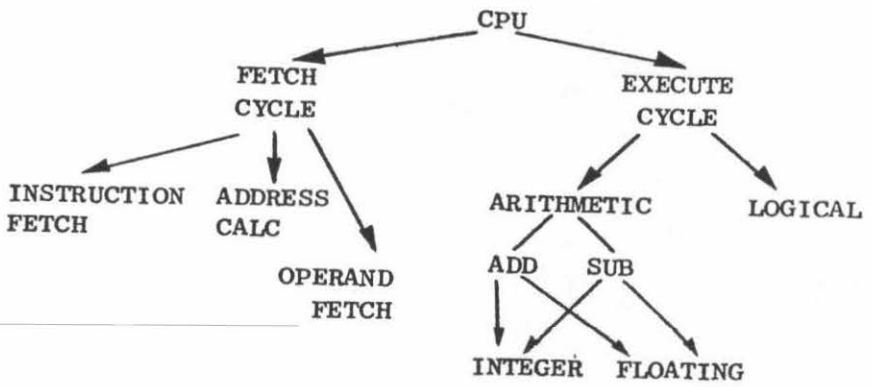

Figure 2: Behavior Hierarchy

A behavioral hierarchical decomposition can be implementation independent. For instance, an ISPS-like description [Ba77] of the IBM/370 architecture would be the same for all IBM/370 implementations.

One system that has adapted this hierarchical decomposition of a behavioral description is the SARA system [Es78] at UCLA. The behavior of a system or subsystem at every level of the hierachy is modeled in terms of GMB graphs. The hierarchy of behavioral descriptions of a hardware design is similar to structured programming techniques.

This hierarchical behavior design process provides for a iterative refinement of the initial design specification.

Design automation tools that could be used during this phase of the hardware design process are: formal verification of the behavior specification at various levels of the hierarchy, as well as simulation of the design at various levels.

2.2 STRUCTURAL HIERARCHY

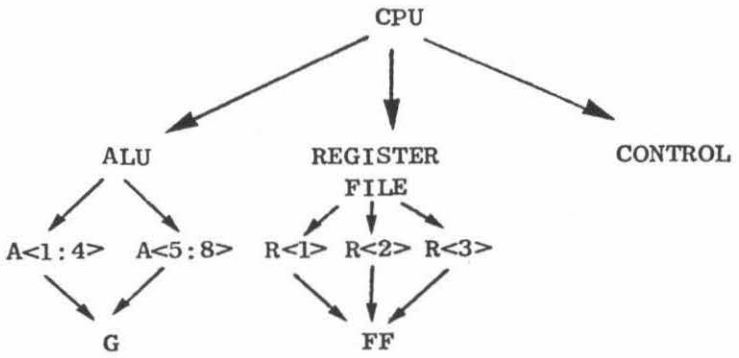

Figure 3: Structural Hierarchy

A possible structural hierarchy for a general purpose computer system is shown in Figure 3, where the CPU consists of an ALU register file and a control section. These three subsystems are interconnected by data paths which are not shown explicitly in this hierarchy. The ALU may consist of two identical sections, one for the first four bits and the second one for the last four bits. Both sections of the ALU are made out of gates of type G. The register file on the other hand consists of three registers R1, R2, R3, which are constructed out of flipflops of type FF.

The SARA system at UCLA [Ga74] also includes a structural modelling tool, called SL1 (a structural description language). This language allows the designer to specify the structural hierarchy of the design.

In the SCALD design system [MW78] a similar methodology is employed. The SCALD system was used for the design of an actual machine, the Stanford-1 processor. This processor

was designed within a few man-years (a fraction of the time normally spent on such a design project). One of the advantages of structural hierarchical design is that one has a structural modularization of the design with well-defined interfaces. This can reduce the design time considerably. The structural hierarchy of a system is developed concurrently with the behavioral hierarchy. At every level of the structural hierarchy one can can associate a behavior description with every module. This behavior description itself can be hierarchical in nature, as was discussed in the previous section. This multitude of behavior hierarchies can be used for formal verification of the design decisions the designer makes.

Another function that one wants to perform during the structural design is to verify the intended behavior of a design against the structure that one is proposing. This so-called dataflow verification was implemented in the LCD system at IBM [OE77]

A final important aspect of a structural hierarchy is the fact that one can do a hardware macroexpansion of a design into lower level primitives. This provides an alternative to a direct hardware compiler from a behavior description into physical hardware. Since such a mapping is totally user-controlled, one can achieve very satisfactory results, as was demonstrated in the SCALD system [MW78].

2.3 PHYSICAL HIERARCHY

In order to package a system the designer has at his disposal a hierarchy of cabinets, racks and printed circuit boards. In integrated circuit design he may have at his disposal an hierarchy of supercells, macrocells, simple cells and transistors (Figure 4).

When designing large scale integrated circuits the human designer has a natural approach to lay out a design first in global terms, and then to successively refine this design down to the transistor level.

In order to cope with the complexity of VLSI circuits, several hierarchical IC design systems have been described

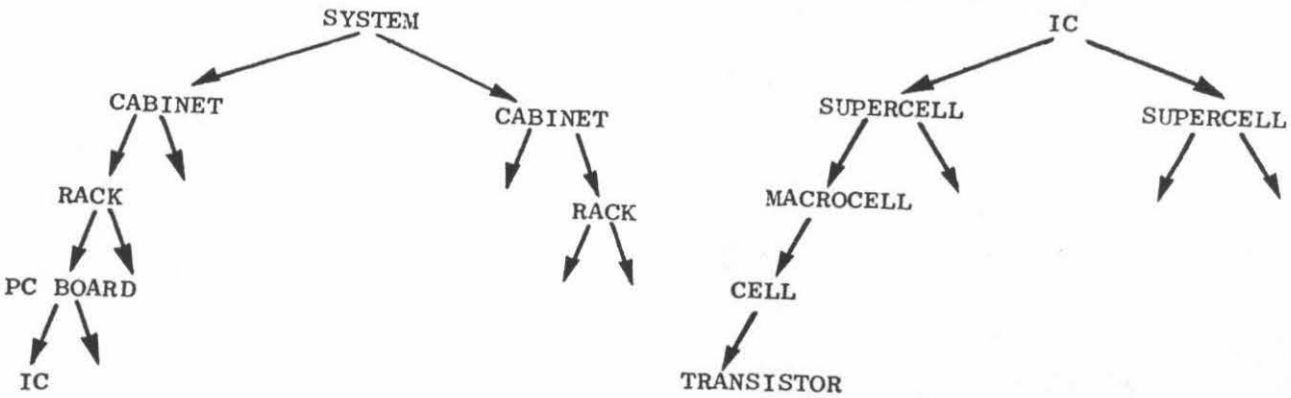

Figure 4: Physical Hierarchy

[PG78, vS77]. The potential advantage of these systems comes from reducing the amount of design time, while maintaining a reasonably efficient area utilization.

During the physical design process extensive use is made of design automation tools. A subtask of mapping the structural design hierarchy into a physical hierarchy is known as the partitioning process. This partitioning is usually done by human designers and very few automatic algorithms are used.

During the physical design process one can obtain sufficiently detailed information to allow a detailed behavior prediction of a system. This predicted behavior can then be verified against the intended behavior as it was specified by the designer during the behavior specification process.

3. ADVANTAGES AND PROBLEMS

3.1 DEMONSTRATED ADVANTAGES

A major advantage of hierarchical design methods is the reduction of design time. This technique is often used informally by IC layout designers: a cell is designed once and then replicated several times; further, IC designers often use a top-down planning phase to determine the global layout of a circuit. However, today's use of hierarchical layout by IC designers is totally informal.

Formal hierarchical methods for IC layout were proposed in [vS77] and [PG78]. Preliminary results show that hierarchical automatic layout yields results that are superior to single-level (classical) automatic IC layout algorithms [Pv79a, Pv79b].

The most convincing evidence that hierarchical design can reduce design time can be found in the use of the SCALD system for the design of the Stanford-1 processor [MW78]. The SCALD system uses a hierarchy of structural (connectivity) diagrams, specified graphically by means of SUDS (the Stanford University Drawing System). The SCALD system further consists of a macroexpander, which produces a wirelist at the physical module level and of a physical design system, which automatically produces the wirewrapped design. The amount of time and the number of engineering changes to the Stanford-1 design were at least an order of magnitude smaller than those of comparable ECL-based machines.

3.2 POTENTIAL ADVANTAGES

If one were to formalize hierarchical design one could require the designer to specify intended behavior for every module in the structural hierarchy.

The first advantage of such an approach would be extensive documentation of the design, now a major cause for misunderstanding and hence design errors and iterations.

If one were given the intended behavior and the structure (realization) of a system at a given level as well as the intended behavior of all subsystems at the next lower level, then formal verification could be used to check the consistency of the designer's decisions.

3.3 HIERARCHICAL DESIGN PROBLEMS

3.3.1 The Difference in Hierarchies

One of the important problems that one has to solve in hierarchical design is the mapping from a behavioral hierarchy into a structural hierarchy, and from a structural hierarchy into a physical hierarchy. In the past this mapping has always been done by human designers with little or no assistance from design automation tools. Due to the increased complexity, the need for formal verification tools as well as synthesis tools becomes more and more apparent.

One possible solution is to avoid the mapping problem by making every behavioral module the same as every structural module and the same as every physical module. This may seem a good solution at first sight, but it has some farreaching implications. For instance, a physical design decision may be made that could require changing the behavior specification or the structure specification of a design. Such a change may invalidate all the verification and simulation results that were obtained on the behavior or structural design. This may be a non-acceptable solution that may result in a large number of iterations during the design process. If the hierarchical structure of the behavior of a design is not the same as the structural or physical hierarchy then automated or computer-aided mapping processes are needed. The main objective of these automated mapping tools would be to speed up the time required to perform an error-free mapping of behavior into structure. Unfortunately, very little work has been done on this problem and it is not clear today how feasible such an approach would be.

3.3.2 Circuit Layout Problems

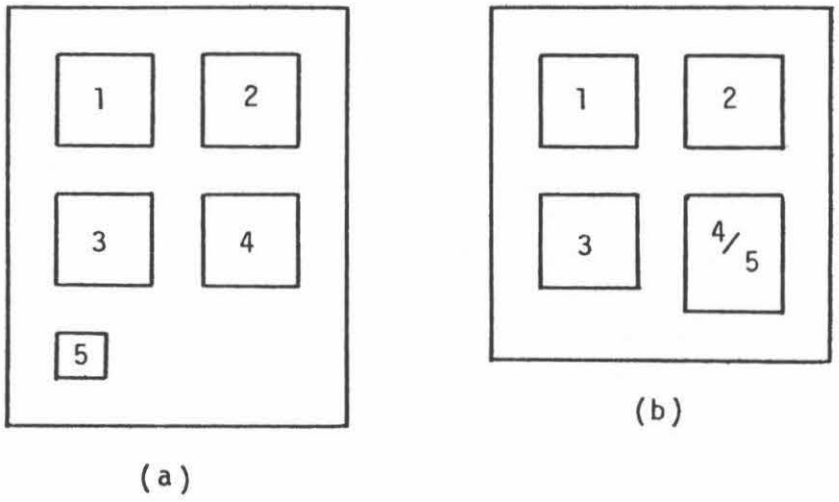

Figure 5: Inefficient Layout due to Hierarchical Process

Figure 5 illustrates the problem of mapping a structural hierarchy into a physical hierarchy. In this example, which represents a simple integrated circuit layout, the design consists of four cells of equal size, numbered one to four, and a fifth cell which is considerably smaller. If one were to consider a one-to-one mapping from the structural hierarchy into a physical hierarchy, the layout of Figure 5a would result with as a consequence a potential inefficient area utilization. However, if in the physical design process one could combine 4 and 5 into a physical module, then the physical design may be much more compact as shown in Figure 5b. If behavior, structure and physical hierarchies were identical then this design decision made during the physical layout of a circuit would have repercussions on both the structural and behavior specifications which may have been verified. Nonetheless the decision made here should have no effect on either the structure or the behavior of the circuit since it is a purely physical design decision.

In a purely hierarchical approach one encounters the problem of optimal shape determination for the blocks in the

layout hierarchy. For example, in the structural hierarchy of Figure 3 all registers are built out of flipflops of type FF, that are physically identical. To associate a single physical design with a structural module can be very inefficient.

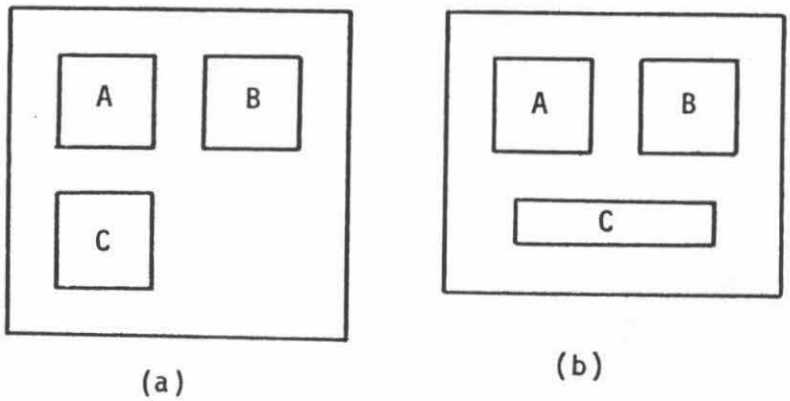

Figure 6: Influence of Different Physical Layout

Consider the example of Figure 6, where the layout consists of three structurally identical modules A, B and C of type FF. The layout could be greatly improved if the physical shape of C were changed as in Figure 6b.

A similar problem of efficient area utilization exists with the assignment of terminals around the periphery of a module. Figure 7a shows two identical modules A and B, with the specified connections. If the order of the terminals could be modified as in Figure 7b, a more efficient layout could be obtained. This however would necessitate the redesign of all modules, thereby greatly reducing the potential benefits.

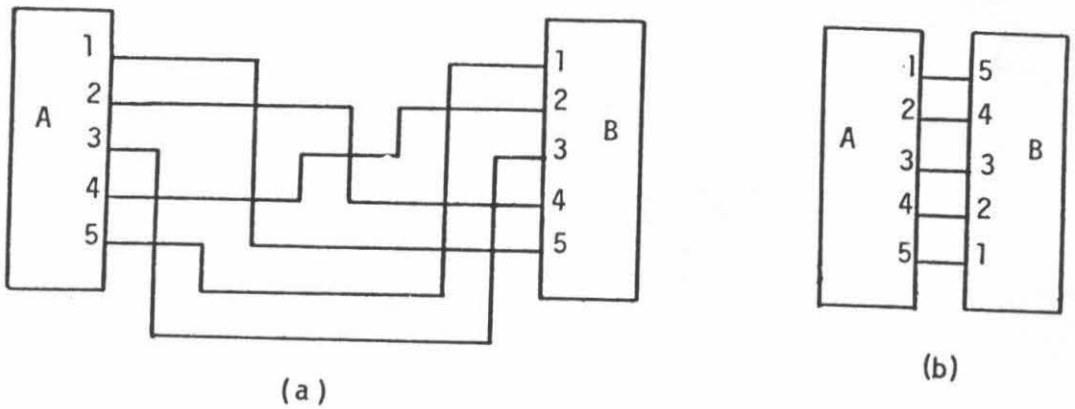

Figure 7: Influence of Terminal Assignment

3.3.3 Multiplicity of Hierarchical Representations

It was already pointed out that a large number of hierarchical behavior descriptiona may exist, complicating the problem of formal design verification.

In a similar fashion, several structural hierarchies may exist, as pointed out in [va77]: for the purpose of simulation using a zero-delay, three-valued simulator, a flipflop may be modelled by a collection of gates and delay elements; for use with another simulator, this same module may be mapped into NAND gates with variable delays and for the purpose of IC layout the mapping would be into a set of CMOS or NMOS transistors. In other words the structural hierarchy of a design depends not only on the designer's decisions relating to price-performance trade-offs, but also on the actual purpose of the description.

In a similar fashion, the physical hierarchy of a design is not unique, but rather a choice of the designer.

The problem of choosing the best hierarchy cannot be solved by the hierarchical design process concept. It is important to realize that hierarchical design can merely produce an acceptable design in an reduced amount of time.

3.3.4 The Need to Exceed the Boundaries of a Level

In the previous section, several examples were given that illustrated the need for look-ahead over one or more levels of the hierarchy.

Most of today's design automation software works on a fixed level of abstraction. This is partly due to the lack of hierarchical concepts in these tools and partly due to the nature of the problem they are trying to solve. Among the latter we mention the layout of a PC board, the calculation of timing delays in a LSI circuit, the calculation of physical wirelength of a design.

For these examples, a simple macroexpansion of the design provides a solution. The basic idea of a macroexpansion is to collapse several levels of a hoerarchy into a single level of abstraction.

However, there are situations in which interactions between the levels of a hierarchy are more complex and hence tend to counteract the advantages. One such case is the problem of geometrical design rule adherence in LSI layout. These design rules are usually expressed as a combination of complex relationships on rectangular or polygonal elements. In a true hierarchical design environment, one has to define design rules between cells that are far more conservative, hence resulting in a less optimal layout.

A similar problem exists with IC layout: a cell is often seen as a closed polygon without allowing use of the inside area for laying out a collection of cells. In reality some of this intra-cell area could be used for optimal layout.

4. CONCLUSIONS

In this paper we have attempted to present a model for the human designer and the design process. We have postulated that a hierarchy of behavioral specification, structural implementation and physical realization is a reasonable model for the human designer.

Design automation tools should capture this hierarchical information from the human designer and use it for making more intelligent design decisions. An open problem in design automation is the problem of mapping from a behavioral specification hierarchy into a physical implementation hierarchy, and from a physical implementation hierarchy into a physical realization hierarchy. This process should be either automated or computer-aided, if we want to deal with complex systems.

Hierarchical design is capable of reducing the amount of resources devoted to a design. As always this yields a trade-off between design optimality and design time. In the VLSI era, a suboptimal inexpensive design may be more important than a more expensive but more optimal design.

In some cases there will be a need to perform a macro-expansion on the design in order to consider the whole design at a single level.

REFERENCES

[Ba77] Barbacci, M.R. "The ISPS Language," Carnegie Mellon Univ., Dept. of Computer Science, Technical Report, 1977.

[Es78] Estrin, G. "A Methodology for the Design of Digital Systems , supported by SARA at the Age of one," Proc. National Computer Conf., Anaheim, Cal., June 1978, pp. 313-324.

[Ga74] Gardner,R. "A Methodology for Digital System Design based on Structural and Functional Modelling," Ph.D. Thesis, UCLA, 1974.

[MW78] McWilliams, T. M., and L. C. Widdoes, Jr., "SCALD: Structured Computer-Aided Logic Design," Proc. of the 15th Design Automation Conference, Las Vegas, Nevada, June 1978, pp. 271-277.

[OE77} Ofek,H.; Evangelisti,C.J. and Goertzel,G. " Designing with LCD: Language for Computer Design," Proc. 14th Design Automation Conf., San Francisco, June 1977, pp. 369-376.

[PG78] Preas,B.T. and Gwyn,C.W. "Methods for Hierarchical Automatic Layout of Custom LSI Circuit Masks," Proc. 15th Design Automation Conf., Las Vegas, Nevada. June 1978, pp. 206-212.

[Pv79a] Preas, B.T. and vanCleemput,W.M. "Placement Algorithms for Arbitrarily Shaped Blocks," Proc. Int. Symposium on Circuits and Systems, June 1979, to be published.

[Pv79b] Preas, B.T. and vanCleemput,W.M. "Routing Algorithms for Hierarchical IC layout," submitted for publication, 1979 Design Automation Conf.

[vS77] vanCleemput,W.M. and Slutz,E. "Initial Design Considerations for a Hierarchical IC Design System," Conf. Record 11th Asilomar Conf. on Circuits, Systems and Computers, November 1977, pp. 334-341.

[va77] vanCleemput,W.M. "An Hierarchical Language for the Structural Description of Digital Systems," Proc. 14th Design Automation Conference, New Orleans, Louisiana, June 1977, p. 377-385.