Experiences in Scheduling of Design Decisions

Reinhard Rauscher
Univ. Hamburg
Vogt-Koelln-Str. 30, D22527 Hamburg
e-mail: rauscher@rz.informatik.uni-hamburg.de

Abstract: *The most important task in design methodology is to determine and to evaluate the set of design decisions to be considered. In many designs the set of design decisions is handled intuitively, and also the set of alternatives for each of the design decisions are highly influenced by experiences from earlier designs. The paper discusses the ordering of such a set of design decisions. A set of practical designs have been analyzed using the system introduced in the paper.*

1 Introduction

In the past, the design of highly integrated systems has been considered a very time consuming process. The design procedure has been done only by specialized designers, which have had detailed knowledge of the underlying technology.
In the meantime this has changed. Nowadays - because of widespread use of technology independant models and corresponding tools - the design can be done quite fast. Typical examples of these models are *standard cell design* [4] and the *Meet-in-the-middle strategy* [5].
But, more and more design decisions occur, which are answered intuitively or have been answered based on experiences from earlier designs ([2]).

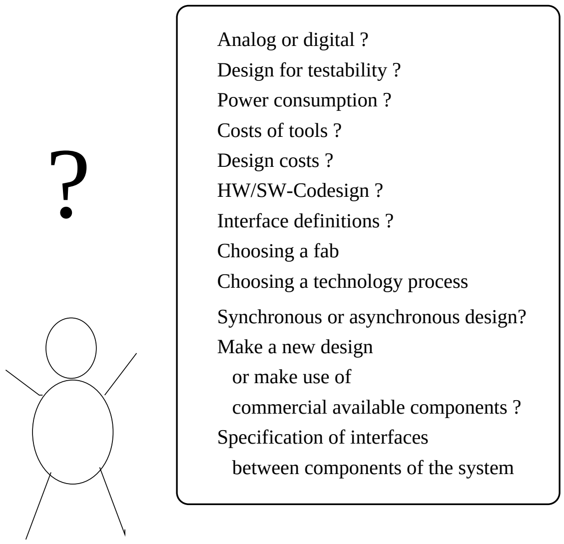

Figure 1: Confusion of the designer

Fig. 1 illustrates the unordered list of design questions to be ordered by the designer. After ordering of the questions he has to choose a solution from a set of alternatives for each of the design decisions. The task of the designer can be roughly sketched by the following quotation ([5]):

"enormous dimensions of the design space have to be explored."

Any a priori sequence of design decisions (valid for all design activities) would be desirable, but it cannot exist ([3]).
The same holds for the amount of design decisions, which differ from a few (10) up to 1000 ([3]).
Summarized, the number of decisions, the alternatives for each of the decisions, and the sequence of the decisions depend on the application problem.
Considering fig. 1 some restrictions - for the actual task - can be defined.

First, some of the decions have to be handled in parallel. Second, some of the decisions have to be done in an a priori defined sequence. An example for decisions one has to deal with in parallel could be *Synchronous or asynchronous design ?* and *Power consumption.*
Another example coming from fig. 1 could be the parallelism of the decisions *Analog or digital design ?* and *Specification of interfaces between components of the system.* Both questions have to be answered in parallel, if the designer has the opportunity to choose between well suited analogue or digital interfaces.
An example for sequential decision making is the following: *Make a new design or make use of commercial available components ?.* This is a decision, which must be handled before the decision *Choosing a fab and/or choosing the production process* can be done.
Obviously, only if the first decision is answered "make a new design" the second decision is relevant for the designers task.
In the following a concept and a corresponding tool is presented, which allows a structured procedure in ordering the design decisions. The basic idea is, that a designer can use his own knowledge concerning the set of design decisoons and his experiences concerning their interdependencies, but also he could use an standard list, which must be extendable. Demands the tool has to meet are:

- The list of design decisions to be considered must be incremental enlargable by the user (designer).
- The list of mutual dependencies between the design decisions (parallel or sequential) must be incremental enlargable by the user (designer).

One goal can be defined as follows: Find a sequence of the unordered list of design decisions, which fulfills all specified mutual dependencies between design decisions.
If not all of the specified dependencies can be fulfilled, the violations of the dependencies must be *weighted.* Based on this, a second goal can be described as follows:
Find a sequence of the unordered list of design decisions, which minimizes the weighted sum of violated dependencies.

2 Implementation

Input to the system is a description of the set of design decisions and a list of dependencies (between design decisions). Due to the demands mentioned above, the input description can be enlarged quite easily. The whole input can be written to a ASCII-file.
Interactively, the user (designer) decides, whether the procedures 'transitive closure' or 'sequentialization' should be executed.
After this, no further parallelisation or sequentialisation can be done. The generated output is the ordered list of the design decisions.

3 Results based on several ASIC-designs

Based on experiences from several ASIC-designs the system's prototype has been analyzed. Starting with a set of 30 general design decisions, first of all the dependencies have been (problem dependent) determined. The results were:

1. For several designs only a subset of the 30 design decisions has been proved to be necessary. Partially, they have been determined by the specification itself or nobody noticed, that a decision has been done intuitively. Furthermore, some of the decisions have not been relevant for the actual design.
2. Partially, the results of the system have been distinct from that sequence of design decisions, which has been choosen while doing the design. The reason for this could be, that most of these designs have been done in a real top-down manner without detailed analysis of dependencies.
3. A comparison between the schedule of design decisions generated using one of the procedures 'transitive closure' or 'sequentialize' with that one without these procedures did result only in quite important changes in the schedule. But, most of the milestones are identical for all these alternatives.

References

[1] Gajski, D.: Silicon Compilation, Addison-Wesley, 1988
[2] Leung, S.; Shanblatt, M.: ASIC System Design with VHDL: A Paradigm, Kluwer Acad. Press., Boston, 1989
[3] Knapp, D.; Parker, A.: The ADAM Design Planning Engine, IEEE Transact. on CAD, Vol. 10, No. 7, July 1991, pp 829-846
[4] Mead, C.; Conway, L.: Introduction to VLSI Systems, Addison-Wesley, Reading, Mass., 1980
[5] Rabaey, J.; De Man, H. et al.: Cathedral-II: A Synthesis System for Multiprocessor DSP Systems, In: Silicon Compilation, D.D. Gajski (ed), Addison-Wesley, 1988, pp 311-360