An Efficient Reuse System for Digital Circuit Design

Annette Reutter
Robert Bosch GmbH,
Dep. K8/DIC, Postfach 1342,
D-72703 Reutlingen, Germany
annette.reutter@rt.bosch.de

Wolfgang Rosenstiel
University of Tübingen
Department of Computer Engineering
Sand 13, D-72076 Tübingen, Germany
rosenstiel@informatik.uni-tuebingen.de

Abstract

In this paper a complete reuse system for digital circuit design is presented. Thereby 'design for reuse' and 'design by reuse' aspects are considered. In particular a repository for IPs with special emphasis on classification and selection, web integration and IP protection is developed. By practising intra-company reuse with our system the efficiency and performance of reuse is demonstrated.

1. Introduction

To keep pace with the strong competition and the increasing complexity of ICs it is absolutely necessary to continuously improve the design process. Mainly a reduction of design time and design costs and an increase in design quality have to be realized. To meet these demands integration of reusability into the industrial design flow is important. VSI Alliance is creating standards for interfacing reusable building blocks [VSIA97], [Bake98], which are often called "Intellectual Property" (IP). Design reuse mainly comprises the tasks "design for reuse" and "design by reuse" [KCGW98]. [KeBr98] and [GKRR98] describe several mechanisms designing modules for reuse. [JDKR97], [ReMR98a] combine high level synthesis and reuse. To establish reuse in the design flow suitable systems managing IPs and the reuse process are necessary. [OAIP98], [BAMS98], [PHSM95] deal with design models and environments considering reuse aspects and enabling "design by reuse". [BeBS98] describes an internet based IP catalog. By our contribution a complete reuse system is presented. In particular the model, the implementation and the integration into the industrial design flow is demonstrated.

[1] This work was supported by the BMBF under contract 01 M 3036 B (MEDEA/SSE project EURIPIDES). The authors are responsible for the contents of this publication.

The proposed system enables the management of informations about IPs as well as the management of IP's object data. All data are stored in the database system and can be uploaded and downloaded by design engineers directly. The system provides different and extensive mechanims for selection. Comprehensive informations about IPs are presented in an user-friendly form. The reuse system is a "living" IP repository: All included IPs are designed based on a uniform design for reuse strategy. The IPs and the informations about IPs match the used design flow and applications. The design engineer is enabled to work actively with the repository. Our system is optimized for intra-company reuse, but nevertheless there is no restriction in the area of external reuse.

By our contribution costs and profits by integrating reuse are not only demonstrated, but also captured by numbers which were extracted during the application of our system.

2. Model

2.1. Global concept

Beside hardware descriptions there is also a necessity to reuse scripts realizing the usage of tools and design flows. Therefore, reuse can be divided into three levels as shown in figure 1. Most popular is reuse of modules as hardware descriptions. Further reuse levels are tools and flows. To reuse tools and flows, scripts have to be prepared for reusability. In particular scripts, but also designs, have to be extended by constructs, which allow parametrization and thereby flexible reusability [RMKR97], [ReMR98a], [ReMR98b]. Flexible reusability is characterized by the possibility of reusing designs in other applications and projects by a simple parameter setting without any manual changes in the source code. Static reusability - reuse of modules without any possibilities of adaption - also allows the reuse of designs, but only with high efforts: The source code has to be well known and adapted manually. So, often it is simpler and faster to develop the design once again.

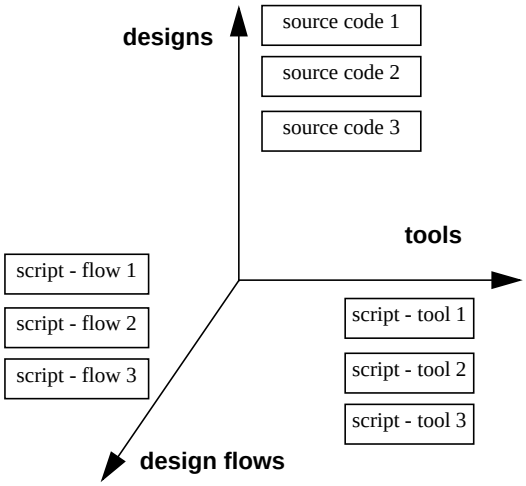

Figure 1. Reuse levels

This concept consisting of three levels of reuse includes dependencies. The practice shows that e.g. not every design description can be handled by every tool: Not every combination is prepared and available in the current database and descriptions often include tool specific extensions. Therefore, a system is required which is able to manage these dependencies.

2.2. Design for reuse strategy

The parts of an IP are defined as shown in figure 2. Because complete verification of parametrizable IPs is impossible by using today's CAD tools, test designs are

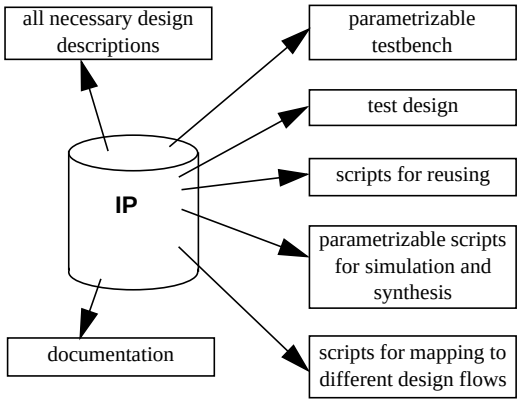

Figure 2. Parts of an IP

essential parts of IPs. A test design is a dummy design environment with an example integration of the IP. It allows to verify the IP with different parameters within a verified and automated design flow, but independently of the final design. Therefore, a testbench which allows the same parametrization as the design and parametrizable scripts for synthesis and simulation are necesssary. In parts these scripts can also be reused for the final designs. Since parametrizable scripts for all possible options of designs, especially for RT synthesis scripts, are too much effort, only realistic parameters are supported. Further parts are scripts or generators for integrating the IP into the test design and the design of the design engineer, parametrizable scripts for mapping to different design flows and a suitable user-friendly documentation explaining the functionality and the reuse process. The proposed design for reuse strategy is explicitly described in [ReMR98a] and [ReMR98b].

2.3. Classification and selection

In the past reuse often failed because the designs could not be retrieved and no documentations were available [LeWM98]. The following mechanisms for selection and classification allow transparent design reuse:

- Keywords
  Keywords are defined for each design in the database. In order to create a useful classification, it is important to keep the set of keywords as small as possible.
- Properties
  In order to search for IPs, which have to meet special demands, classification and selection by properties should be installed. Properties like 'area', 'delay', 'quality degree' and 'silicon available' have to be defined for each IP.
- Reuse levels
  Reuse levels - designs, tools and flows as shown in figure 1 - can be considered as a special kind of properties. After the complying classification, e.g. all IPs can be selected which are designed with a special tool or flow, or for which special scripts are used or which include a certain design.

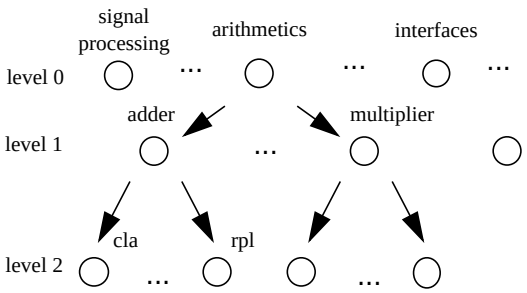

Figure 3. Taxonomy

- Taxonomy
  A well-defined taxonomy allows IP classification according to an hierarchical classification scheme. A part of such a taxonomy is shown by figure 3 and by [BeBS98].
- Dependency and similarity
  Some IPs include further IPs. Such dependencies have to be registered by the reuse system. If a selected IP does not meet the requirements, it may be necessary to select all modules with similar properties. Therefore IPs should also be classified by considering similarities. A practical approach is to register all similar IPs for each IP available in the repository by introducing similarities between the nodes of the taxonomy. Furthermore the degree of similarity can be characterized by weights [FaSe98].

2.4. Version and release control

IPs can exist in different versions. New versions include e.g. extensions, improvements or bug fixes. Old versions should be kept available, because they could be already reused and trace the lifecycle of IPs. To manage different versions and to make IPs invisible, a release control has to be installed. A release includes none, one or several versions of each IP. It is also possible to compose releases for different applications or customers.

2.5. Access history

The number of reuses and in particular the reuse experiences are important informations about IP quality. If problems occur, persons who have already reused the current IP are important to know. Therefore the access to the repository and the reuse success is recorded as shown in figure 4. IP is rejected, because it does not fit. The experiences during the reuse process are marked by '–' and '+'.

IP	Date	Project	Name	Success
IP A	01.01.1997	PRJ_3	Müller	reused, +
IP B	27.03.1997	PRJ_1	Müller	rejected
IP B	26.05.1997	PRJ_2	Maier	reused, -
IP B	01.01.1998	PRJ_4	Klein	reused, +
IP C	02.01.1998	PRJ_3	Jetter	reused, +

Figure 4. Access history

2.6. In future required modules

A reuse system should also contain information about IP required in the close future indicating the responsible divisions within the company. So, redundant development can be avoided and design and reuse of IPs can be planned.

3. Implementation of the reuse system

The reuse system was implemented by developing a relational database application reflecting the described model and by designing a secure web user interface, in which the database is embedded. Databases combine all advantages required for our application. In particular, databases meet the demands of data dependency management and definition of any user-defined properties.

First of all the described model was transformed into an entity-relationship diagram, which is shown in parts by figure 5. The E/R-diagram is used as the basis of database development and interface design.

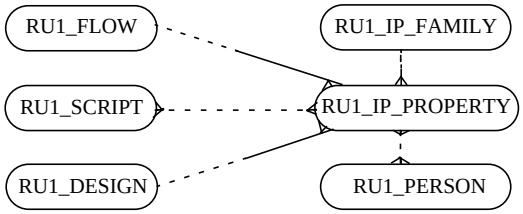

Figure 5. E/R-diagram

To install an easy-to-handle reuse system a standardized and transparent user interface is supposed. To submit reuse between different sites of a company, we decided to implement an intranet web interface. It is universal, easy to handle and it enables reuse without any reference to the data's physical location.

Reuse often fails because the access to IPs is not sufficiently clarified and protected. Designers hesitate to provide IPs in a common repository since there is no protection against unauthorized access [HaKn98]. In our system this protection is provided by the following proceeding: IPs can only be accessed via the web interface. Only the administrator is excluded from this restriction. To control the access several user roles ('administrator', 'user', 'reader') and rights (read, write, change, delete) were defined for database access via web. The 'administrator' is owner of all rights. The 'user' is allowed to read all data and to write to unchecked data. The 'reader' is only allowed to read data. All persons who like to use the reuse system were admitted to user roles and have to authorize with a password.

Several user modules were developed. User modules are graphical interfaces between database and user. An example is given by figure 8. Most important modules are input modules, query modules and maintenance modules. Input modules allow to check in (upload) new IP. Query modules allow to search an IP using the defined selection mechanisms and to check out (download). Maintenance modules allow the administration of the system. For IP upload it is important to know the properties of available IPs. Only by using this knowledge, similarities and dependencies can be defined and the set of keywords can be kept as small as possible. Therefore a special upload mechanism was developed. It is shown in figure 6. As described 'users' and 'administrators' are allowed to read all available data and

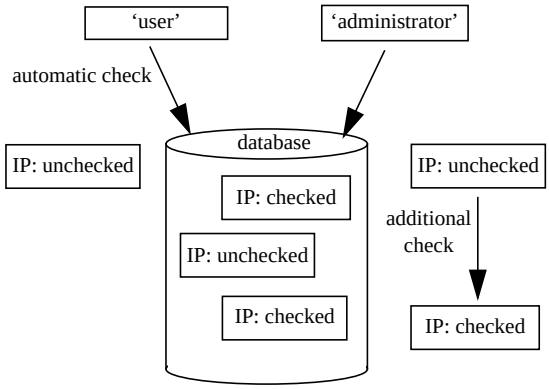

Figure 6. Upload of IPs

write new IPs and the according properties to the database. To keep the database with high quality, all new IPs get a flag 'unchecked'. Only 'checked' data are visible for selection. 'Users' are only allowed to write or to change unchecked data. Only the 'administrator' is allowed to change this flag to 'checked'.

4. Reusing Designs

By an example of IP lifecycle and by results extracted during prototype use of our system in the industrial environment we show the strength of our approach and the complete reuse system.

4.1. Reuse process

The reuse process as shown in figure 7 is starting with design for reuse (number (1) in figure 7). First of all the profits of designing a module for reusability have to be estimated. Thereby the higher efforts must be considered. A specification of design reusability should be created. It

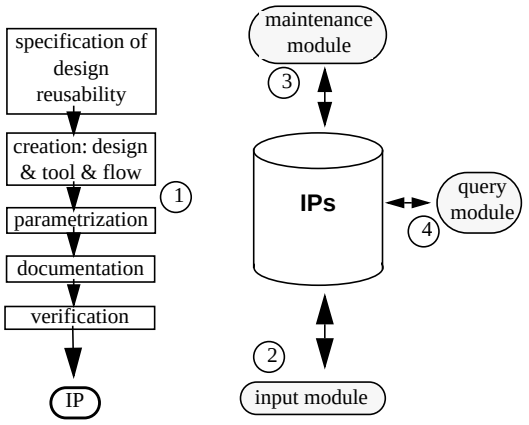

Figure 7. Reuse process

includes all functions and tasks allowing profitable reuse with acceptable efforts. These functions will be realized [ReMR98a], [ReMR98b] and will characterize the design as an IP. Higher efforts are mainly caused by parametrization and documentation work, but also by additional verifications. The resulting IP is uploaded by the designer ('user') using an intranet input module (2). The designer has to characterize the IP by filling in a upload form considering already available IPs. This 'unchecked' data are transfered via intranet into the database. The 'administrator' is creating a visible IP by changing the flag to 'checked' after some additional checks. The maintenance module provides some additional automated possibilities to start checks. E.g. the availability of responsible persons and necessary data or files can be checked (3). Now, any 'user' is able to select and download this IP (4) using a query module as shown in figure 8. 'Readers' are allowed to read all information about IPs, but not to download modules. Using the added testbench and test design designers can perform further verifications before integrating the IP into their designs. Therefore only less efforts are necessary because all scripts are available. The work consists of parameter setting and script starting.

4.2. Results

After integrating the developed reuse system into the industrial design flow results demonstrating efficiency and performance of reuse were extracted.
IPs can be divided into two sections:

- Parametrizable Intellectual Properties (PIP) which are especially designed for reusability
- Static Intellectual Properties (SIP) which are designed for a special project.

PIPs are parametrizable and can be simply adapted to another design environment. SIPs can also be reused, but

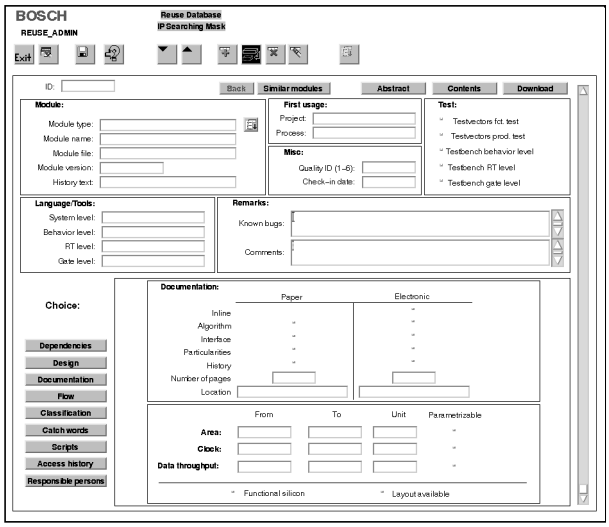

Figure 8. Query module

the designer has to completely know the source code and to adapt manually. For both kinds of IPs design costs, design time and collection of expert's knowledge were measured. Figure 9 to figure 12 summarize the results by representative excerpts.

Design costs. Following costs were measured and outlined in figure 9 for SIPs and in figure 10 for PIPs:

- Costs to develop a specific design module oneself (column A)
- Costs of overhead to create an IP (column C)
- Costs to reuse an IP (column B)

The costs are measured by man weeks. By integrating reuse

module ID	development (A)	reuse (B)	preparation (C)
bus	4 MW	2 MW	1 MW
audio	32 MW	8 MW	4 MW
fft	9 MW	3 MW	2 MW
fir	6 MW	2 MW	1 MW
ioctrl	5 MW	2 MW	1 MW

Figure 9. Design costs - SIPs

into the industrial design flow following results could be summarized:

- Global costs
  The complete costs of creating an IP consist of the costs of specific development (A) and the additional costs of preparation (C).
  $costs_{global} = costs(A) + costs(C)$
  By each reuse the costs are reduced by the costs for a redesign minus the costs for reuse.
  $costs_{saved} = costs(A) - costs(B)$
  It pays off to develop IPs if the global costs are saved by reusing the IP *n* times.
  $costs_{global} = n * costs_{saved}$
  Considering SIPs and PIPs the global costs are normally paid back by two reuses.

moduleID	development (A)	reuse (B)	preparation (C)
MUL	4 MW	0.5 MW	3 MW
I2C_1	3 MW	1 MW	3 MW
I2C_2	8 MW	1 MW	4 MW
SPI	4 MW	1 MW	3 MW

Figure 10. Design costs - PIPs

- Costs of a designer using the reuse system
  The costs of reusing a SIP (B) are for worst case at least half of the costs of developing the SIP once again (A), as shown in figure 9. For best case the costs are only 25 percent of the costs of a redesign.
  Considering PIPs the costs for reuse amount only between 1/8 and 1/4 of the redesign cost as shown in figure 10.
  Thereby a designer can save up to about 85 percent of costs by reusing already available IPs. Because the costs of creating the IP are paid after the second reuse, a cost reduction of 85 percent is realistic.
- Preparation costs
  The preparation costs are higher for PIPs (up to same costs as for development) than for SIPs (up to 1/4 of development costs). But reuse costs are higher for SIPs than for PIPs. Preparation costs occur only once. Reuse costs are to pay often, exactly for each reuse of the IP. Therefore, if it is a module, which will be reused more than two times, it is better to perform higher efforts for preparation. The resulting PIP can be reused with less efforts because it is prepared especially for reusability.

Time-to-market. Following times were measured and outlined in figure 11 for SIPs and in figure 12 for PIPs as a part the time-to-market:

- Time to reuse an IP (column E)
- Time to develop a specific design module oneself again, no reuse (column D)

The times are measured by weeks. Following results could be summarized:

module ID	no reuse (D)	reuse (E)
bus	4 weeks	2 weeks
audio	16 weeks	8 weeks
fft	9 weeks	3 weeks
fir	6 weeks	2 weeks
ioctrl	5 weeks	2 weeks

Figure 11. SIPs - Time-to-market

module ID	no reuse (D)	reuse (E)
MUL	4 weeks	0.5 weeks
I2C_1	3 weeks	1 week
I2C_2	8 weeks	1 week
SPI	4 weeks	1 week

Figure 12. PIPs - Time-to-market

- The time-to-market can be improved essentially by reusing designs. Considering SIPs the time-to-market is improved at least by factor two. For PIPs the improvement factor amounts about three to eight. It has to be emphasized once again, that PIPs achieve better results than SIPs.

Collection of expert's knowledge. Designs collected in the reuse system represent expert's knowledge which will always be available. Without reuse system this knowledge is lost, if the designer leaves the company, because the designs can not be retrieved again with high probability. Therefore the number of designs which were checked in by each designer were counted. On an average each designer supplies considerable knowledge of about five IPs.

5. Conclusion

By this contribution an efficient reuse system including both "design for reuse" mechanisms and tools enabling transparent "design by reuse" is presented. Thereby special requirements like efficient selection of IPs, web user interface and protection against unauthorized access are considered. The integration into the industrial design flow is shown and the efficiency and performance of our reuse system is demonstrated by results obtained by practising reuse.

References

[BAMS98] J. Böttger, K. Agsteiner, D. Monjau, S. Schulze: An Object-Oriented Model for Specification, Prototyping, Implementation and Reuse. Proceedings of DATE in Paris, 1998.

[Bake98] S. Baker: Virtual Socket Interface Alliance™. Designer Track, DATE in Paris, 1998.

[BeBS98] B. Behnam, K. Babba, G. Saucier: IP Taxonomy, IP Searching in a Catalog. Designer Track , DATE in Paris, 1998.

[FaSe98] N. Faulhaber, R. Seepold: An Efficient Similarity Metric for IP Reuse in RMS. Proceedings of 2. Workshop "Reuse Techniques for VLSI Design" in Karlsruhe, 1998.

[GKRR98] D. Garte, T. Kunjan, A. Reutter, S. Riedel, S. Rülke: Survey on a Practicable "Design for Reuse" Strategy Including Design Flow and Testbench Aspects. Proceedings of 2. Workshop "Reuse Techniques for VLSI Design" in Karlsruhe, 1998.

[HaKn98] S. Hauck, S. Knol: Data Security for Web-based CAD. Proceedings of DAC in San Francisco, 1998.

[JDKR97] A. A. Jerraya, H. Ding, P. Kission, M. Rahmouni: Behavioral Synthesis and Component Reuse with VHDL. Kluwer Academic Publishers, Boston/Dordrecht/London, 1997

[KCGW98] M. Koegst, P. Conradi, D. Garte, M. Wahl: A Systematic Analysis of Reuse Strategies for Design of Electronic Circuits. Proceedings of DATE in Paris, 1998.

[KeBr98] M. Keating, P. Bricaud: Reuse Methodology Manual. Kluwer Academic Publishers, Boston/Dordrecht/ London, 1998.

[LeWM98] G. Lehmann, B. Wunder, K.D. Müller-Glaser: A VHDL Reuse Workbench. Proceedings of EURO-DAC in Geneva, 1996

[OAIP98] S. Olcoz, L. Ayuda, I. Izaguirre, O. Penalba: VHDL Teamwork, Organization Units and Workspace Management. Proceedings of DATE in Paris, 1998.

[PHSM95] V. Preis, R. Henftling, M. Schütz, S. März-Rössel: A Reuse Scenario for the VHDL-Based Hardware Design Flow. Proceedings of EURO-DAC in Brighton, 1995.

[RMKR97] A. Reutter, B. Mößner, I. Kreuzer, W. Rosenstiel: Wiederverwendung komplexer Komponenten für Synthese und Simulation unter Verwendung von VHDL. Proceedings of 8. E.I.S.-Workshop in Hamburg, April 1997.

[ReMR98a] A. Reutter, B. Mößner, W. Rosenstiel: Design of generic components for efficient reuse in high level designs. Proceedings of SDA-Workshop in Dresden, 1998.

[ReMR98b] A. Reutter, B. Mößner, W. Rosenstiel: Design of reusable modules for high level designs. Proceedings of International Workshop on IP Based Synthesis and Simulation in Grenoble, 1998.

[VSIA97] "Virtual Socket™ Interface Architecture Document", Virtual Socket Interface Alliance, 1997, http://www.vsi.org/library/vsi-or.pdf