A Scalable Methodology for Cost Estimation in a Transformational High-Level Design Space Exploration Environment

Joachim Gerlach Wolfgang Rosenstiel

University of Tübingen
Department of Computer Engineering
Sand 13 • D-72 076 Tübingen • Germany

Abstract

Objective of the methodology presented in this paper is to perform design space exploration on a high level of abstraction by applying high-level transformations. To realize a design loop which is close and settled on upper design levels, a high-level estimation step is integrated. In this paper, several estimation methodologies fixed on different states of the high-level synthesis process are examined with respect to their aptitude on controlling the transformational design space exploration process. Estimation heuristics for several design characteristics are derived and experimentally validated.

1 Introduction

Since a few years the increasing complexity of digital circuits represents the main problem in digital circuit design, existing methodologies which allow a specification of the design on higher levels of abstraction and therefore support a homogeneous design process are getting more an more important. One reason for this is, that higher design levels hold a large optimization potential. Cause of this, design space exploration should start on upper levels of abstraction. In our methodology, design space exploration is done on behavioral level by applying high-level transformations. If, like most of the conventional high-level transformation tools do, an evaluation of the design is performed by explicitly executing many (or even all) design steps of lower levels, the resulting design loop covers a large number of design steps and therefore a single cycle of the design loop is quite expensive to perform. Our approach to face this problem is to cut the design loop on a high level of abstraction by integrating a high-level cost estimation step. This causes a trade-off between estimation quality and computation effort of the estimation heuristic. In this paper, estimation heuristics are presented which can be evaluated almost within behavioral synthesis and which can be fixed on different execution stages of the high-level synthesis process (and for this called scalable with respect to the state of synthesis). The techniques are examined with respect to their aptitude on controlling the task of transformational design space exploration, and estimation heuristics are derived, which allow to decide the quality/ effort trade-off in an (user specific) optimal way.

The paper is organized as follows: After a look at the state of the art in chapter 2, an overview on our design space exploration methodology is given in chapter 3. Chapter 4 presents our scalable estimation methodology in detail, chapter 5 describes the experimental derivation and validation of efficient estimation heuristics and summarizes the results.

2 State of the Art

Most of the estimation heuristics include approximative execution of particular synthesis steps (and for this, results are tightly coupled with the applied synthesis algorithms). In [11], a heuristic for area estimation is presented, which bases on performing module- and register-binding, succeeded by an approximative floorplanning. [8] describes an approach for computation of upper bounds on area based on explicitly performing ASAP and ALAP scheduling. The PEPPER [6] analysis tool performs an estimation of delay by explicitly executing tasks of placement and routing. In the CADDY system [12], area/delay estimation bases on corresponding models of the module library, which is completed before the real synthesis process starts. Power estimation heuristics are often based on simulation [10] or iterative structural propagation of switching activities [9], but up to now, behavioral level power estimation techniques are restricted to special application domains (DSPs) [7].

3 High-Level Design Space Exploration

Figure 1 shows an overview on our high-level transformation design space exploration environment. On the highest level of abstraction, three activities can be identified: the tasks of high-level transformation application (transformations), evaluation of transformation quality, including the step of cost estimation (transformation analysis), and control of the transformational exploration process, including identification of transformation candidates and initiation of design cycles (transformation control).

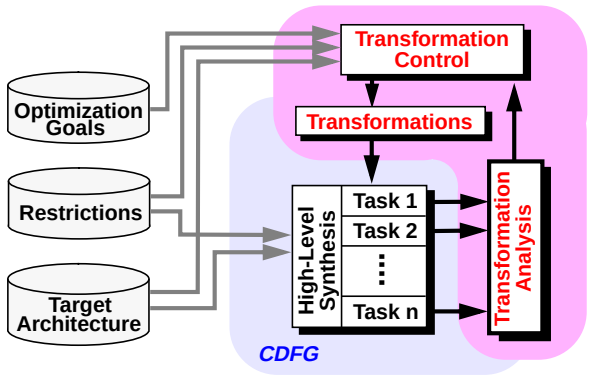

Figure 1. Transformation environment.

The term *high-level transformations* covers optimization techniques known from compiler theory [1]. However, the preconditions of their applicability (e.g., target architecture, optimization goals, restrictions) differ significantly from software domain. In the current state of implementation, the transformation task comprises a set of fifteen basic transformation types, including pure datapath optimizations, e.g., constant-propagation, common subexpression elimination and expansion, strength reduction, algebraic optimizations, optimization of array-accesses and array-scalarization, as well as controlflow optimizations, e.g., function inlining, loop unrolling and tiling, elimination of dead or empty paths. Most of the basic transformations can be scaled in terms of functionality and locality of their application. Regarding this, a set of about fifty individual transformation steps results. In the current state of implementation, selection and activation of transformations is done by the designer.

Our approach for realizing a design loop which is close and settled on high level of abstraction bases on a tight coupling of high-level transformation and high-level synthesis tasks. Cost estimation is done by trial execution of particular steps of high-level synthesis. Therefore, transformations immediately act on the internal high-level synthesis design representation, a control-/dataflow graph (CDFG). Proof of transformation applicability, transformation application itself, and analysis of the transformation result (including the step of cost estimation) is completely performed on the CDFG structure. For this, a complete cycle of the design loop can be executed efficiently without leaving the high-level design representation [5].

4 High-Level Cost Estimation

4.1 Demands on a cost estimation heuristic

The application of a cost estimation heuristic within our high-level transformation design space exploration environment leads to a set of specific demands:

- To allow cost evaluation *within* high-level synthesis, the *execution stage of the synthesis process* has to become a parameter of the cost estimation heuristic. The cost estimation heuristic should be applicable in different execution states of the high-level synthesis process (and for this, is called *scalable*) and should be able to identify and utilize all design information which is available in this state.
- In spite of the high level of abstraction, on which cost estimation acts, the heuristic has to be able to estimate design costs in a fine-grain fashion, because designs resulting from high-level transformations possibly differ in a very slight way only (e.g., because a transformation only affects very local parts of the design).
- In regard of applying the cost estimation heuristic within transformational design space exploration, the ability to quantify *relative* dependencies of design characteristics (and for this, to *compare* design alternatives) is much more important than the ability to capture *absolute* values. To quantify this characteristic, we apply the *fidelity* measure [3] proposed by Gajski:

$$\text{Fidelity} = 100 \bullet \frac{2}{n(n-1)} \bullet \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mu_{ij}$$

$$\mu_{ij} = \begin{cases} 1 & \text{if} \begin{cases} E(i) < E(j) \ \wedge R(i) < R(j) \\ E(i) > E(j) \wedge R(i) > R(j) \\ E(i) = E(j) \wedge R(i) = R(j) \end{cases} \\ 0 & \text{otherwise} \end{cases}$$

The fidelity measure supplies for a given set of n reference values $R(1),\ldots, R(n)$ and n estimation values $E(1),\ldots,E(n)$ a measure describing the quality of the estimation with respect to its ability to quantify relative dependencies of pairs of values.

4.2 Cost Estimation Methodology

Our cost estimation methodology bases on common application of two contrary principles, *specialization* and

abstraction. In our methodology, two types of estimation heuristics have to be distinguished:

- Techniques to estimate costs of the entire design, referred to as *cost heuristics*.
- Techniques to estimate costs of isolated register-transfer components, referred to as *cost functions*.

Figure 2 shows an overview on our cost estimation methodology. In a cost heuristic, as much as possible of register-transfer information (depending on the actual execution state of the synthesis process) is identified (components as well as interconnection structure). This corresponds to the specialization step. The identified register-transfer components are passed to a cost function, which performs an estimation of the particular component costs. Back in the cost heuristic, the component estimation values are used to generate an estimation value for the entire design, corresponding to the abstraction step. Objective of our methodology is to utilize all design information for estimation, which is available in the current state of the synthesis process. In the following, cost heuristics are evaluated in 5 execution stages of high-level synthesis (see table 1).

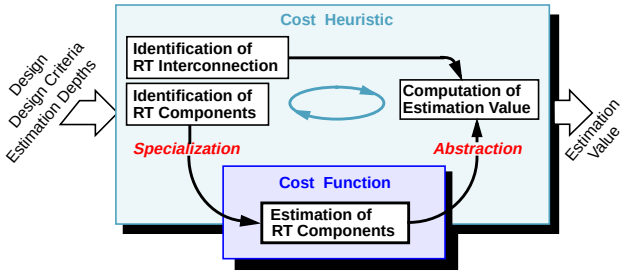

Figure 2. Cost estimation methodology.

Estimation Depth	State of High-Level Synthesis
Estimation Depth 1	After Module Scheduling and Allocation
Estimation Depth 2	After Module Binding
Estimation Depth 3	After Register Allocation and Binding
Estimation Depth 4	After Interconnection Binding
Estimation Depth 5	After Netlist Generation

Table 1: Evaluation depths of cost heuristics.

In estimation depth 5, high-level synthesis is carried out completely, and for this depth 5 is equivalent to register-transfer level. By dividing the estimation step in cost heuristic and cost function, the estimation methodology becomes „high-level“: generation of cost functions has to be done only *once* for a given target architecture, design style and high-level synthesis system (and for this, can be regarded as a precomputation step). A single estimation request only requires the evaluation of the cost heuristic. In our investigations, several cost functions and cost heuristics were realized. Combinations of cost functions, cost heuristics, and estimation depths were extracted which lead to estimation heuristics which can be efficiently applied in our transformational design space exploration methodology.

4.3 Cost Functions

Our cost functions base on *explicitly* performing costs of elementary register-transfer cells of the generic component-library of the applied high-level synthesis system. Typical examples of such elementary cells are full adders, logic-gates or 1-bit 2:1-multiplexors. Those elementary cells are *explicitly* synthesized and design characteristics are extracted. So, characteristics of the applied synthesis tools and target architecture are taken into consideration.

In a first cost function, *GenCost*, determination of costs of register-transfer components of higher complexity is done via cost formulas. Those cost formulas are derived by analyzing the generic generators of the applied high-level synthesis system. Arguments of the cost formulas are given by parameters of the generic generators, e.g., word length, number of inputs or sign. Figure 3 shows an example: for deriving area and delay cost formulas of a ripple-carry adder, $area_{FA}$ and $delay_{FA}$ of an elementary full adder cell are explicitly determined (applying the *concrete* synthesis process). Regarding the generic generator of an arbitrary n-bit adder, it can be found that *area* can be approximated by $n \bullet area_{FA}$. Same formula holds for *delay*, taking into account that the critical path touches all full adder cells.

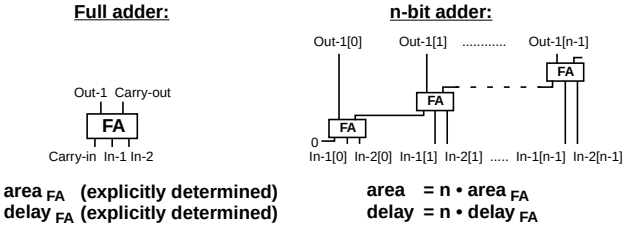

Figure 3. Cost function *GenCost*.

In a second cost function, *FuncCost*, costs of particular instances of a register-transfer component are explicitly determined, too, but in contrast to *GenCost*, not only elementary cells of the parameter space are regarded. The evaluated cost values are used to generate a piecewise linear approximating function, which is applied for a cost function of the corresponding register-transfer component type.

4.4 Cost Heuristics

Cost heuristics identify as much as possible of register-transfer information in the given design (which corresponds to an intermediate result of the high-level synthesis process). Then, cost functions are applied to perform an estimation of the register-transfer components and the results are used to compute an estimation value for the entire design.

A first cost heuristic, *DPApprox*, analyzes the current design representation and identifies connected datapath segments of maximum size. By this strategy, all information which is available in the current state of high-level synthesis is utilized to perform an estimation value. Figure 4 shows an example: In (a), only information concerning functional units and registers is available. In (b), estimation is performed in a more advanced stage of the synthesis process, which results in more accurate predictions.A second cost heuristic, *ProbDPApprox*, acts in very

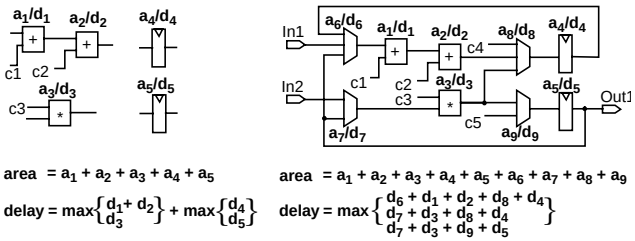

Figure 4. Cost heuristic *DPApprox*.

similar fashion, but in addition, a weighting of datapath segments (or sub-segments) corresponding to the maximum interconnection structure which can be identified in the current state of the design process is performed. Thereby, a uniform distribution is assumed in branching nodes, which leads, as the following chapter will show, to satisfactory results. A third cost heuristic, *CPApprox*, considers costs produced in the controller part of the design. This is done by regarding several static characteristics of the controller (e.g., number of states, transitions, input/outputs). Applying a set of standard high-level synthesis benchmarks, we were able to show a good correlation (fidelity > 80%) of an estimation based on such static characteristics. For a common consideration of datapath and controller, correlation factors describing the ratio of datapath costs (heuristics *DPApprox* and *DPApprox*) to controller costs (heuristic *CPApprox*) are derived in a statistical way. For more details see [5].

5 Experimental Results

Objective of our investigations is to derive combinations of cost functions, cost heuristics and estimation depths which result in estimation heuristics of maximum quality/effort trade-off with respect to their application in transformational design space exploration. In the following, the design criteria area, delay, and power are considered in detail. Other design criteria (e.g., throughput, FSM size) were regarded, too, but the methodologies differ significantly from those presented above, and for this, are not subject of this paper.

To evaluate the quality of our estimation methodology, heuristics resulting from several combinations of cost functions, cost heuristics and estimation depths were applied to multiple applications, and the results were compared with the synthesis results. In our experiments, high-level synthesis is performed by the PMOSS system [4], logic synthesis is done by the SIS system [14]. Mapping the design on a concrete target architecture bases on the MCNC library [14].

Our methodology is demonstrated in terms of the following three types of applications:

- **Fidelity**: Our methodology is demonstrated in terms of the algorithm for computing the fidelity measure (see section 4.1). For this, the evaluation algorithm itself becomes an application.
- **High-level transformation benchmark suite**: The heuristics derived from the fidelity example are validated via a set of high-level transformation benchmark designs.
- **GSM**: For an application of high complexity and industrial relevance, a submodule of the GSM fullrate speech transcoder [2], integrated in mobile telecommunication systems for real-time compaction of human speech, is regarded.

5.1 Fidelity

The fidelity algorithm was transformed into a set of nine alternative design versions by applying high-level transformations (for the transformations sequence, see figure 5).

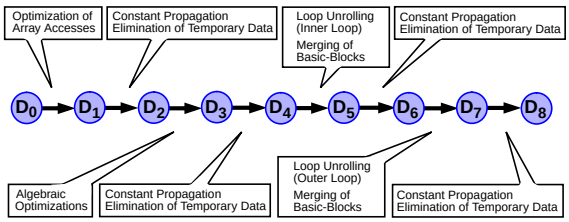

Figure 5. Transformation sequence for the fidelity application.

In figure 6, combinations of cost functions and cost heuristics, which lead to estimation heuristics with maximum

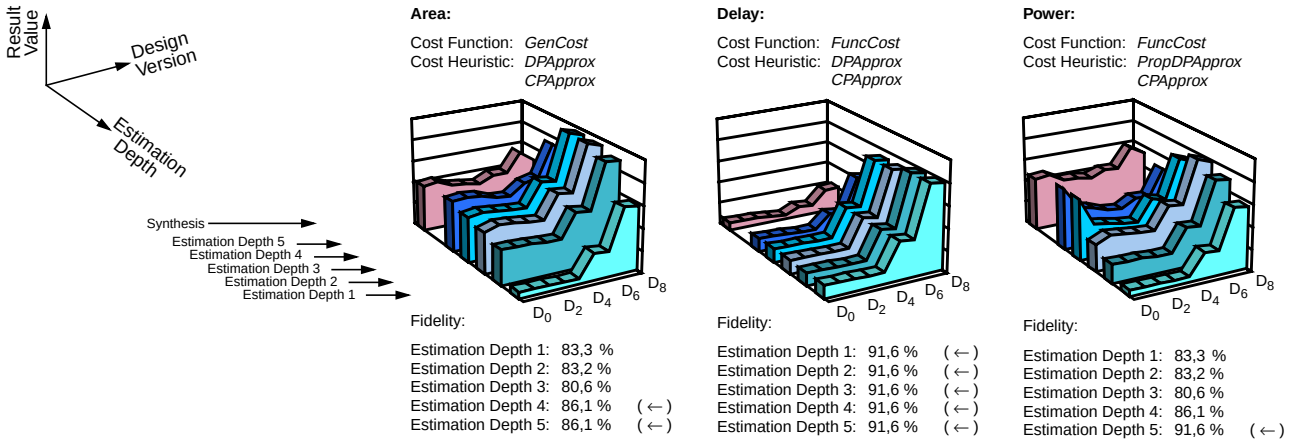

Figure 6. Comparison of estimation and synthesis results.

fidelity (←) are shown. Like mentioned before, the *fidelity* of estimation heuristics is of primary interest, and for this, the *shape* of the curves, not the absolute values has to be considered (the absolute values generated by the heuristics don‘t correspond to any physical unit, only the *relative* relations are of interest). As figure 6 indicates, e.g. combination of the generic cost function *GenCost* and cost heuristics *DPApprox* (non-probabilistic datapath approximation) and *CPApprox* (controller approximation) results beginning with estimation depth 4 in area estimation heuristics with maximum fidelity of 86%. Regarding the speedup applying our high-level estimation methodology with respect to generating corresponding values by explicitly performing lower level synthesis steps, speedup factors from $0.5 \cdot 10^2$ up to $1.1 \cdot 10^6$ can be obtained (depending on design criteria and estimation depth). The results are summarized in table 2. For details (e.g., curves for other combinations of cost functions and cost heuristics which lead to lower estimation fidelity) see [5]

5.2 High-Level Transformation Benchmark Suite

For validation of the estimation heuristics derived in 5.1, alternative design versions of 10 benchmark designs (of different applications, e.g. filters, sorting algorithms, mathematical computations, DSP) were generated via high-level transformations. Then, the design criteria area, delay, and power were estimated applying the heuristics extracted in 5.1. The resulting fidelity values are summarized in table 2.

5.3 GSM Fullrate Speech Transcoder

In scope of a project under grant of *Deutsche Forschungsgemeinschaft*, the GSM fullrate speech transcoder was treated by hardware/software codesign, wherein a real-time critical module was identified and efficiently realized in hardware [13]. This module was optimized applying our transformational design space exploration methodology. Figure 7 shows the corresponding transformation tree. Here, the estimation heuristics derived in 5.1 also decide the quality/effort trade-off in an optimal way. The resulting fidelity values are summarized in table 2, the speedup disposes of similar dimensions as those in 5.1. By rerunning the design space exploration process without applying our estimation step, the identical design was identified for hardware implementation. Thus we were able to show that in the GSM application our estimation methodology leads to an acceleration of the design process by orders of magnitudes without any loss of design quality.

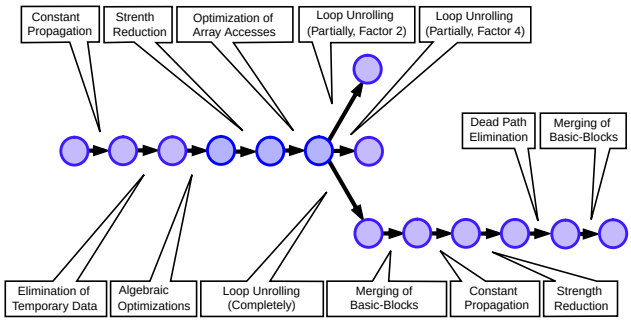

Figure 7. Transformation tree for the GSM application.

5.4 Results

Table 2 summarizes the results of sections 5.1 to 5.3 and for this, suggests estimation heuristics for several design characteristics. In all applications 5.1 to 5.3, the proposed combinations of cost functions, cost heuristics and estimation depths lead to rapid estimation heuristics with high fidelity (≥ 80 %), which points to a high stability of our methodology. For *all* applications and design characteris-

Design Criteria	Estimation Heuristic						Fidelity Result											
	Cost Function		Cost Heuristic			Estimation Depth	Fidelity (in %)	Trans-Bench-1 (in %)	Trans-Bench-2 (in %)	Trans-Bench-3 (in %)	Trans-Bench-4 (in %)	Trans-Bench-5 (in %)	Trans-Bench-6 (in %)	Trans-Bench-7 (in %)	Trans-Bench-8 (in %)	Trans-Bench-9 (in %)	Trans-Bench-10 (in %)	GSM (in %)
	GenCost	*FuncCost*	*DPApprox*	*DPProbApprox*	*CPApprox*													
Area	X		X		X	4	86.1	97.2	80.0	100	82.2	86.7	89.3	91.7	93.9	93.3	97.2	89.1
Delay		X	X		X	1	91.7	86.1	90.0	100	86.1	88.9	96.4	91.7	86.4	100	91.7	81.8
Power		X		X	X	5	91.6	83.3	80.0	89.3	80.1	88.9	96.4	94.4	89.4	93.3	91.7	synthesis timeout

Table 2: Results of estimation heuristic evaluation.

tics, our estimation heuristics were able to reliably forecast the design version with minimum cost. All speedup factors have similar magnitudes as the ones presented in 5.1, and for this lead to a significant acceleration of the design space exploration process.

6 Conclusion

In this paper, our scalable methodology for cost estimation in high-level transformation design space exploration was presented and efficient estimation heuristics for several design characteristics were derived. We were able to show experimentally that our estimation heuristics leads to a significant acceleration of the design process with high quality of result.

Further investigations concentrate on integration of the presented estimation methodologies in algorithms for automated control of the design space exploration process.

Acknowledgments

This work was partly funded by the Deutsche Forschungsgemeinschaft, under grant „Automatisierter Systementwurf“, SFB - 358 - B3.

References

[1] A.V.Aho, R.Sethi, J.D.Ullman. Compilers: Principles, Techniques and Tools, Addison Wesley, 1986.

[2] European Digital CellularTelecommunications Systems, Phase 2: Full Rate Speech Transcoding. DIN Norm ETS 300580-2, DIN Deutsches Institut für Normung e.V., Beuth Verlag GmbH, 10772 Berlin., 1994.

[3] D.D. Gajski, F. Vahid, S. Narayan, J. Gong. Specification and Design of Embedded Systems. Prentice Hall, Englewood Cliffs, 1994.

[4] J. Gerlach, H.-J. Eikerling, W. Hardt, W. Rosenstiel. Von C nach Hardware: ein integratives Entwurfskonzept. In GI/ITG/GMM-Workshop Allgemeine Methodik von Entwurfsprozessen, Paderborn, March 1996 (in german).

[5] J. Gerlach, W. Rosenstiel. Ein skalierbarer Ansatz zur Kostenabschätzung für die Steuerung von High-Level Transformationen. Technical Report WSI-97-5 at University of Tübingen, September 1997 (in german).

[6] D. LaPotin, Y. Chen. Early Matching of System Requirements and Package Capabilities. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, November 1989, pp. 394-397.

[7] R. Mehra, J. Rabaey. Behavioral Level Power Estimation and Exploration. In Proceedings of the International Workshop on Low-Power Design, April 1994, pp. 197-202.

[8] S.Y. Ohm, F.J. Kurdahi, N. Dutt. Comprehensive lower Bound Estimation from Behavioral descriptions. In Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, November 1994, pp. 182-187.

[9] S.R. Powell, P.M. Chou. A Model for Estimating Power Dissipation in a Class of DSP VLSI Chips. IEEE Transactions on Circuits and Systems, Vol. 36, No. 6, June 1995, pp. 646-650.

[10] T. Quarles. The SPICE3 Implementation Guide. Tech. Rep. M89-44, Electronics Research Laboratory, University of California, Berkeley, California, 1989.

[11] M. Rim, A. Mujumdar, R. Jain, R. De Leone. Optimal and Heuristic Algorithms for Solving the Binding Problem. IEEE Transactions on VLSI Systems, Vol. 2, June 1994, pp. 211-225.

[12] W. Rosenstiel, R. Camposano. Synthesizing Circuits from Behavioral Level Specifications. In Proceedings of the 7th International Conference on Computer Hardware Description Languages, August 1985.

[13] W. Schwarz, G. Fettweis, A. Mögel. Sonderforschungsbereich 358: Automatisierter Systementwurf - Synthese, Test, Verifikation, dedizierte Anwendungen. In Wissenschaftliche Zeitung der Technischen Universität Dresden 46 (1997), Heft 2, pp. 31-46 (in german).

[14] E.M. Sentovich, K.J. Singh, L. Lavagno, et al. SIS: A System for Sequential Circuit Synthesis. Technical Report Memorandum No. UCB/ERL M92/41, Department of Electrical Engineering and Computer Science, University of California at Berkeley, May 1992.