3rd HCM BELSIGN Workshop, April 11 and 12, 1996, Corsica.

Some Complexity Results in Memory Mapping

Erwin G. Woutersen* and Sabih H. Gerez
University of Twente, Department of Electrical Engineering (EL-BSC-NT)
P.O. Box 217, 7500 AE Enschede, The Netherlands.
E-mail: wouter@natlab.research.philips.com and
s.h.gerez@el.utwente.nl

Abstract

Four combinatorial optimization problems occurring in the memory-mapping subtask of high-level synthesis turn out to be intractable as is proven here. The first problem is *cyclic register assignment* in which repetitive read-write patterns of storage values have to be stored in a minimal number of locations situated in a register file with random access (RAM). The second problem is to partition a set of storage values in as few RAMs as possible. The remaining problems concern the counterparts of the RAM problems for *sequential read-write memories.*

1 Introduction

Many combinatorial optimization problems that are encountered in the field of VLSI design automation are intractable [Gar79]. In this paper a number of problems related to the memory mapping task in high-level synthesis are presented and proven to be either NP-complete or NP-hard. *High-level synthesis* [Gaj92, DM94] is the task of transforming an algorithmic description of hardware into an interconnection of functional units, such as adders and multipliers, and memory elements. The problems to be solved include *scheduling*, the mapping of operations from the algorithmic descriptions to time instants, and *assignment*, the mapping of operations to functional units and the mapping of *storage values* [Sto92], the intermediate values that have to be stored for some time before they are needed again, to memory elements. Obviously, the creation and consumption times of the storage values are known if the scheduling task has been completed.

2 NP-completeness

Proving NP-completeness requires two steps [Gar79, Pap94]:

1. one should prove that the decision problem at hand is in the complexity class NP;
2. one should show that a problem already known to be NP-complete can be reduced in polynomial time to the problem at hand.

This last step is often dealt with in a sloppy way by reversing the requirement: the problem at hand is reformulated as e.g. a problem from graph theory that is known to be NP-complete. This reformulation does not imply that the problem at hand is NP-complete: the graph theoretic problem instances resulting from the reformulation may have a specific structure not present in the general graph theoretic problem that enables to solve the problem instances in an efficient way.

NP-completeness always deals with the *decision version* of a combinatorial optimization problem and not with the *optimization version*. In the following text, even when not always explicitly stated, the decision version is meant when referred to NP-completeness. The decision version has only two possible answers: *yes* or *no*.

3 Cyclic Register Assignment

The first problem considered is the cyclic assignment of storage values to registers. The context for all problems to be discussed in this paper is some repetitive calculation with an iteration period of T_0 as e.g. encountered in digital signal processing algorithms [HdG92]. After scheduling, all intermediate values that need to be stored, are known. Assuming that all values have a lifetime of at most T_0, each value will be characterized by a *creation* or *write time* and a *death* or *last-read time*. The *lifetime interval* defined in this way can be represented as the arc of a circle with total circumference T_0. The goal is to use as few memory locations as possible by mapping values with nonoverlapping lifetime intervals to the same memory location (register). This problem is equivalent to the *circular-arc graph coloring* problem known to be NP-complete [Gar80]. A circular-arc graph is a graph induced by a collection of arcs around a circle, where each arc defines a vertex and overlapping arcs define the edges be-

*Author's current address: Philips Research Laboratories, Building WAY 5.67, Prof. Holstlaan 4, 5656 AA Eindhoven, The Netherlands.

tween their corresponding vertices. For a correct proof of NP-completeness of cyclic register assignment, one needs to prove that for each instance of circular-arc graph coloring, an instance of cyclic register assignment can be defined, which is a trivial thing to do. A circular-arc graph coloring problem is characterized by n arcs $a_i = [b_i, e_i]$ $(1 \leq i \leq n)$, where b_i and e_i indicate the begin and end positions of the arc along the circumference of a circle. The goal is to assign a color to each arc, such that the number of colors used is minimal and overlapping arcs have distinct colors. In reducing this problem to cyclic register assignment, choose T_0 equal to the circumference of the circle and reserve one storage value s_i for each arc a_i such that the creation time of the value is equal to b_i and the death time is equal to e_i. Obviously, an instance of the circular-arc graph coloring problem can be colored with k colors if and only if the storage values in the corresponding instance of the cyclic register assignment problem can be assigned to k memory locations.

If all read and write times of the values are distinct, the registers can be grouped into a single-port register file, a *random-access memory* (RAM). The problem of mapping values to a RAM with the goal of minimizing the number of memory locations remains NP-complete (circular-arc graph coloring with distinct interval endpoints is NP-complete and it can be reduced to the RAM assignment problem in a similar way as described above).

4 Assignment to a Single Sequential Read-Write Memory

Recently, the *sequential read-write memory* (SRWM) has been introduced as an alternative for the random-access register file [Ger95, Wou96]. It originates from a low-power design style as applied by Heubi et al. [Heu94]. Instead of the address decoding found in a RAM, a shift register is used to point to the location to be written or read at a specific time instant. In each cycle step this pointer can be moved to the next location or reset to the first one. This is illustrated in Figure 1. The *SRWM realizability* problem is the problem of mapping a given set of storage values to a single SRWM. The mapping should make possible that each value can be written or read at the right time and that the number of locations in a the SRWM is minimized. This problem is NP-complete as well. The proof uses a reduction from cyclic register assignment. If all read and write times in an instance of cyclic register assignment as well as T_0 are multiplied by the maximum number of values present, an instance of SRWM realizability is created that does not "suffer" from the limitations of sequential access (there is always sufficient time to put the pointer at the required position). The decision version of the newly created instance has a *yes* answer if and only if the corresponding instance of cyclic register assignment has a *yes* answer.

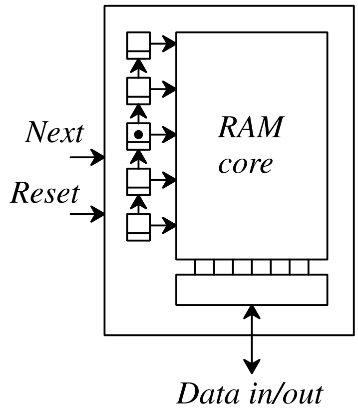

Figure 1: *The sequential read-write memory.*

5 RAM Value Grouping

The *RAM value grouping* problem [Sto90, Sto92, Sto94] occurs when the target architecture for high-level synthesis allows the use of multiple random-access register files [Har89]. It is assumed that single-port register files are used which means that only storage values that have distinct read and write times can be grouped in one register file. The objective of value grouping is to use as few register files as possible for all the set of values. Both in the work of Stok [Sto90, Sto92, Sto94] as in the one of Haroun and Elmasry [Har89], value grouping is approached by building a bipartite graph in the case that there are distinct clock phases for register reading and writing. This clocking scheme will be called the *multiple-phase clocking* model as opposed to *single-phase clocking*, where a memory location is either read or written in one control step (but not both). The two sets of vertices in the bipartite graph are formed by the write times and read times respectively. A value is represented by an edge between its write time and read time. A minimal edge coloring of this bipartite graph gives a minimal number of register files. A disadvantage of this model is that values that are read more than once, cannot be modeled properly. One has either to assume that they are rewritten after being read or that multiple copies of the same value have to be stored.

A new result is that value grouping with the proper modeling of multiple reads (which means that a value is only once written and read as often as required using a single memory location) is NP-complete. The proof is quite simple and based on a reduction from *graph vertex coloring* known to be NP-complete [Gar79]. Consider an undirected graph $G(V, E)$ with n vertices in the set $V = \{v_0, v_1, \ldots, v_{n-1}\}$ and m edges in the set $E = \{e_0, e_1, \ldots, e_{m-1}\}$. An edge $e_k \in E$ that connects two vertices $v_i, v_j \in V$ is denoted by $e_k = (v_i, v_j)$. This problem is reduced to minimal value grouping as follows:

- Create a storage value for each vertex in the graph, re-

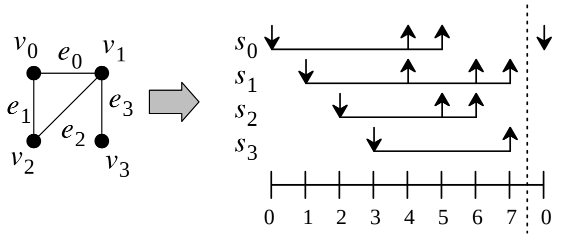

Figure 2: *The reduction from graph vertex coloring to RAM value grouping.*

sulting in the set $S = \{s_0, s_1, \ldots, s_{n-1}\}$.

- The write time for a storage value s_i is set to i ($0 \leq i \leq n$).
- For any edge $e_k = (v_i, v_j)$ add a read time $n + k$ for storage value s_i and s_j.
- Choose T_0 equal to $n + m$.

Figure 2 shows an example graph and the RAM value grouping problem derived from it. In the figure, an arrow pointing down indicates a storage value's write time and an arrow pointing up its read time. Clearly, the storage values of the problem constructed in this way can be mapped on K register files if and only if the original graph can be colored with K colors. It is interesting to note that the NP-completeness of value grouping holds both for a multiple-phase clock and a single-phase clock as a result of the reduction chosen.

6 SRWM Value Grouping

The last problem considered is *SRWM value grouping*. In this problem, the goal is to partition a set of storage values such that each subset can be mapped on a single-port SRWM while minimizing the number of SRWMs. This problem is NP-hard. It is not difficult to see that RAM value grouping can be polynomially reduced to SRWM value grouping using a similar procedure as in the NP-completeness proof of SRWM realizability. However, the first requirement of an NP-completeness proof (the membership of the class NP) cannot be fulfilled. This is due to the fact that it is not possible to verify in polynomial time that a given partition of storage values is a solution, because checking that each subset fits in a single SRWM requires solving SRWM realizability, which is itself NP-complete.

7 Conclusions

To summarize, four combinatorial optimization problems were presented here: cyclic register assignment, SRWM realizability, RAM value grouping, and SRWM value grouping. It turns out that the first three problems are NP-complete and that the last one is NP-hard.

References

[DM94] G. De Micheli. *Synthesis and Optimization of Digital Circuits*. McGraw-Hill, New York, 1994.

[Gaj92] D.D. Gajski, N.D. Dutt, A.C.H. Wu, and S.Y.L. Lin. *High-Level Synthesis, Introduction to Chip and System Design*. Kluwer Academic Publishers, Boston, 1992.

[Gar79] M.R. Garey and D.S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, San Francisco, 1979.

[Gar80] M.R. Garey, D.S. Johnson, G.L. Miller, and C.H. Papadimitriou. The Complexity of Coloring Circular Arcs and Chords. *SIAM Journal on Algebraic and Discrete Methods*, 1(2):216–227, June 1980.

[Ger95] S.H. Gerez and E.G. Woutersen. A High-Level Synthesis Tool for the Assignment of Storage Values to Sequential Read-Write Memories. In *IFIP TC 10 WG 10.5 International Workshop on Logic and Architecture Synthesis*, pages 220–230, Grenoble, December 1995.

[Har89] B.S. Haroun and M.I. Elmasry. Architechtural Synthesis for DSP Silicon Compilers. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 8(4):431–447, April 1989.

[HdG92] S.M. Heemstra de Groot, S.H. Gerez, and O.E. Herrmann. Range-Chart-Guided Iterative Data-Flow-Graph Scheduling. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 39:351–364, May 1992.

[Heu94] A. Heubi, S. Grassi, M. Ansorge, and F. Pellandini. A Low Power VLSI Architecture with an Application to Adaptive Algorithms for Digital Hearing Aids. In M.J.J. Holt, C.F.N. Cowan, P.M. Grant, and W.A. Sandham, editors, *Signal Processing VII: Theories and Applications (Proceedings of the EUSIPCO-94 Seventh European Signal Processing Conference)*, pages 1875–1878, 1994.

[Pap94] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, Reading, Massachusetts, 1994.

[Sto90] L. Stok. Interconnect Optimisation During Data Path Allocation. In *European Design Automation Conference, EDAC*, pages 141–145, 1990.

[Sto92] L. Stok and J.A.G. Jess. Foreground Memory Management in Data Path Synthesis. *International Journal of Circuit Theory and Applications*, 20:235–255, 1992.

[Sto94] L. Stok. Data Path Synthesis. *Integration, The VLSI Journal*, 18:1–71, 1994.

[Wou96] E.G. Woutersen and S.H. Gerez. The Application of Sequential Read-Write Memories in High-Level Synthesis. In *GRONICS '96: Groningen Information Technology Conference for Students*, pages 93–100, Groningen, The Netherlands, February 1996.