Abstraction Based Analysis and Arbiter Synthesis: Radar Memory Interface Card Case Study Revisited

Juhan Ernits

Institute of Cybernetics
Akadeemia tee 21, 12618 Tallinn, Estonia

Abstract

The work focuses on the analysis of an example of synchronous systems containing FIFO buffers, registers and memory interconnected by several private and shared busses. The example used in this work is based on a Terma radar system memory interface case study from the IST AMETIST project [1].

More general application domain of this work is targeted at abstraction and analysis methods that enable to extract scheduling problems from an attributed component models. An overview of the component model of the memory interface board is given in Fig. 1. The purpose of the board is to perform computations on the streams A and B using previous values in the streams and previous values of computations on the streams. The requirement for correct operation of the system is that none of the buffers nor registers should neither be starved nor overflowed.

The solution described in this paper differs from the solution described in [2] in the following aspects:

- The solution presented in [2] involves using SMV [3]; the current solution uses Uppaal [4] as the model checker;
- In [2] the schedule for the example is reached using a parameterized model as model checking the full system was infeasible due to state space explosion. In the current approach, the Uppaal model of the component system contains abstractions that enable the schedule to be synthesized for the whole system.

The current solution builds on Uppaal as an explicit state model checker and exploits bit-state hashing as the model checker provided memory consumption reduction technique for finding schedules. Clocks are used for bounding the depth of the search. The main characteristics of the system modelled in terms of Uppaal automata are the following:

- The state of the buffers and registers is represented by integers.
- To model synchronicity, all variables representing buffers and registers are updated on one transition discarding a great deal of interleavings that are irrelevant in this context. There is a separate variable representing the behaviour of the arbiter that controls which register can access memory at a time.
- The granularity of time ticks is aligned with the duration of a transfer on the shared bus. The buffers and registers are updated by relevant multiples of bytes at the beginning of each such abstract tick.

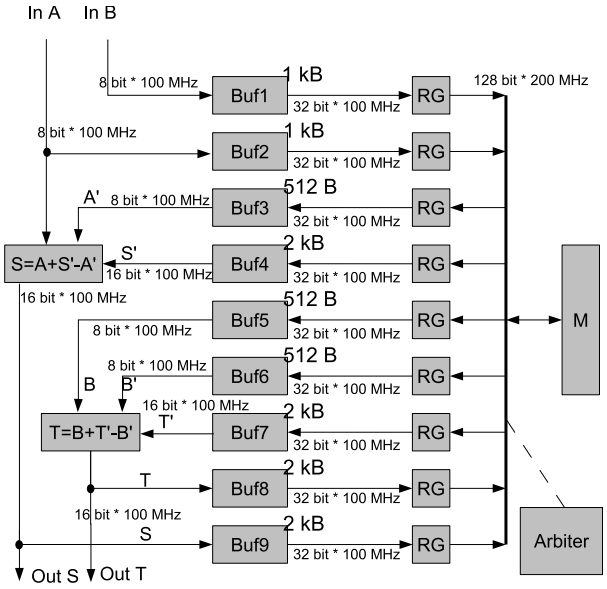

Fig. 1. Radar memory interface board

– The schedule is found using a $E\Diamond$ query and relevant parameters to the Uppaal backend *verifyta*.

The existence of a solution to this problem under further restrictions on resources yields optimisation as a by-product of formal analysis. The contribution of the work is a way to partition the problem into independent subproblems that enable to abstract away details that are irrelevant with respect to the concurrency issue while still guaranteeing correct behaviour in the concrete system.

References

1. Behrmann, G., Bernicot, S., Hune, T., Larsen, K.G., Lecamp, S., Skou, A.: Case study 2: Memory interface for radar system. Deliverable to the IST AMETIST project (2002)
2. Weiss, G.: Optimal Scheduler for a Memory Card. Research report, Weizmann (2002)
3. McMillan, K.L.: The SMV language (1999) Cadence Berkeley Labs.
4. Pettersson, P., Larsen., K.G.: UPPAAL2k. Bulletin of the European Association for Theoretical Computer Science **70** (2000) 40–44