On Diagnosability in HW-SW Codesign: A Case Study

Gy. Csertán and A. Pataricza *
Department of Measurement and Instrument Engineering
Technical University of Budapest
H–1521 Budapest XI. ker. Műegyetem rkp. 9. R108, Hungary
Phone +36–1–463 2057
E-mail: csertan,pataric@mmt.bme.hu

Introduction

The increasing role of dependability in the assurance of the high quality of information processing services necessitates a proper support for dependable design. It should be integrated into the design flow, that even a practitioner engineer without deep knowledge of the underlying highly abstract theoretical background will be able to dominate it.

Unfortunately currently available hardware-software codesign frameworks lack this inherit support for design for dependability. Therefore the designer is forced to use various other evaluation tools which are based on different modeling techniques than that of the codesign environment. Under such circumstances the isomorphism of the model transformations are not guaranteed. That can be a source of design faults and as such it will decrease product quality, what becomes crucial in case of fault-tolerant embedded systems.

In [1] a novel approach is presented for solving this problem in the first, functional design phase of codesign and a small example is included as a feasibility study. In this paper a larger and much more detailed example is given as a case study. The elaboration of the example will show how the presented method can be used in the practice for:

- test generation
- fault simulation
- diagnostic design
- testability analysis

How these task are integrated into the design flow of HW-SW codesign [3] and the interdependencies among the tasks are shown in Figure 1.

In the presented approach in the first, functional design phase of codesign the system is modeled with dataflow networks (DFN). The basic DFN description of the components is extended with the description of faults, fault propagation and fault effects. As a result not only the normal behavior of the system can be defined but also the faulty one.

At the beginning the abstraction of the model is high; only the flow of data is modeled in the form of token flows without any description of the data transformation performed by the components (uninterpreted modeling). Tokens representing the data are colored either as correct or as faulty. (Note that this qualitative grouping can also be user defined allowing the

*This work was supported by the Hungarian National Scientific Fund OTKA-TO15728 and the German National Scientific Fund DFG SFB-182

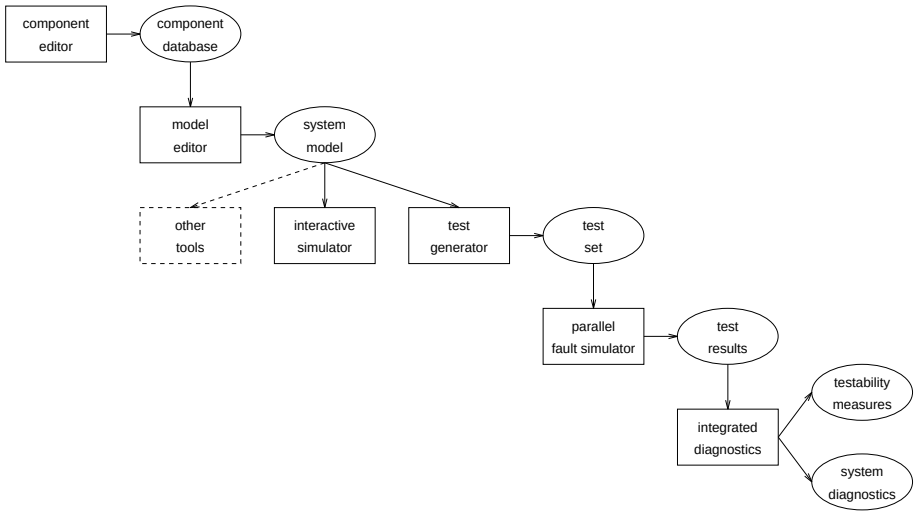

Figure 1: Dependable codesign

user to model different aspects of dependability.) The set of error propagation paths can be estimated by tracing the tokens from the fault site to the outputs of the network.

Due to uncertainties in this high level modeling, diagnostic uncertainty is introduced in form of uncertainly colored tokens in order to express conditional error propagation. This way the test generation and fault simulation algorithms deliver a superset of propagated fault effects, that of course incorporate all potential consequences of the faults of components. For test generation purposes PODEM, the well known gate-level automatic test pattern generation (ATPG) algorithm, has been adopted to the dataflow notation. The relatively low complexity of the high level model allows for using a concurrent fault simulator for fault simulation.

The potential tests, found by the ATPG program, has to be validated in subsequent steps of codesign, but the first rough estimation of diagnostic measures can already be done by diagnosability analysis. Using the fault-dictionary method it is straightforward to extract the test-conclusion relationship from the results of fault simulation. This relationship is the input model of integrated diagnostics [4] that is used for diagnosability evaluation. This way design for dependability is possible from the very beginning of the design as a parallel activity.

After diagnosability analysis the diagnostic design of the system is done, i.e. an effective subset of the test is selected. In the operational phase system diagnostics is based on the results of these tests. For system diagnostics a very efficient new method is presented in [5].

The nondeterminism within the model will be restricted, as more and more design details are incorporated into the model by stepwise refinement (top-down), or as implementation details and validation results from lower levels are back annotated into the model. The usage of model refinement rules will make sure, that the diagnostic measures will monotonically converge to that of the fully designed system. (Note that the method, although it becomes ineffective, can be used even for the fully designed system.)

Example

As a case study the MEMSY [2] system is selected. MEMSY is a modular expandable multiprocessor system. The basic unit is a single node, which is a multiprocessor, shared memory machine. It can consist of up to 4 processors (CPU) with memory management units and cache memories (CMMU). The CPUs and CMMUs can work either stand-alone or can be connected

pairwise in master-checker configuration. Each node has a hard disc with disc controller, a LAN adapter, and communication memory adapters.

Five such nodes can be connected to one pyramid, which is the base of expandability of MEMSY. Additionally to the five nodes a pyramid consists a watchdog processor, that supervises the five nodes. Communication among nodes within and outside of pyramids happens via memory boxes by message passing.

Since it is a general purpose system modeling will be done from the perspective of hardware. The neglect of software will cause further uncertainties in software related components; i.e. a random choice whether a processor executes a read or write command. This kind of uncertainty will not affect the results of the evaluation, it will only make evaluation longer.

The system is modeled at functional block level and dataflow components represent units like: CPU, MMU, cache, disc, disc controller, LAN controller, watchdog, communication memory, DUART, terminal, CIO, reset unit, bus arbiter, master-checker interface, etc.

In the case study system configurations with:

- different number of CPUs
- different number of CMMUs
- CPUs in different master-checker configuration
- CMMUs in different master-checker configuration
- single node vs. one pyramid

will be compared. Tests will be generated possible for all single functional unit faults of the system. Then the test sets will be evaluated and the testability measures of the system computed

Acknowledgment

The authors want to express their gratitude to Prof. M. Dal Cin who hosted them for a period at the Department of Information Science (IMMD 3) of the University Erlangen-Nuremberg.

References

[1] Gy. Csertán, A. Pataricza, and E. Selényi. Dependability Analysis in HW-SW codesign. In *Proceedings of the IEEE International Computer Performance and Dependability Symposium, IPDS'95*, pages 316–325, April Erlangen, Germany, 1995.

[2] M. Dal Cin et al. Architecture and Realization of the Modular Expandable Multiprocessor System MEMSY. In *Proceedings of the 1st Internation Conference on Massively Parallel Computing Sytems MPCS'94, Ischia, Italy*, pages 7–15. IEEE CS Press, May 1994.

[3] J. Rozenblit and K. Buchenrieder, editors. *Codesign.* IEEE Press, 1995.

[4] W. R. Simpson and J. W. Sheppard. *System Test and Diagnosis.* Kluwer Academic Publishers, 1994.

[5] A. Pataric. Algebraic Modeling of Diagnostic Problems in Hw-Sw Codesign. *In this volume.*