Self-Timing, Self-Checking and Self-Recovery
(Invited talk in Special Session on Test Technologies for High Performance Embedded Systems: Extended Abstract)

Victor Varshavsky[(a)], Alex Yakovlev[(b)], Vyacheslav Marakhovsky[(c)] and Ilya Levin[(d)]
(a) Neural Networks Technologies Ltd., Israel; (b) Newcastle University, UK;
(c) The University of Aizu, Japan; (d) Tel Aviv University, Israel
victor@nnt-group.com; alex.yakovlev@ncl.ac.uk; marak@u-aizu.ac.jp; ilia@post.tau.ac.il

Abstract

The talk shows how the use of the self-timed system design principle [1] *allows constructing dependable embedded multi-processor systems based on self-repair and self-recovery. Systems with modular delay-insensitivity enjoy the property of self-checking with respect to stuck-at faults. Use of special multi-bit codes also improves robustness to intermittent faults. The discussion is based on the design of a multi-token communication channel with self-recovery.*

As embedded control systems become more complex requirements to their design suffer from an important contradiction. On one hand, there is a need for higher performance, which can only be met by an increase in the size of the embedded hardware. On the other hand, greater complexity has a negative effect on reliability, all other things being equal. However, in some applications, such as missile control, maintaining reliability together with appropriate performance levels is of prime concern.

In the last two decades, the concept of dependability has replaced that of reliability when talking about safety-critical systems. Dependability implies graceful degradation of a system's functioning under the effect of faults, allowing the system to retain its ability to carry out its main tasks instead of experiencing a catastrophic failure [2].

Resource replication, e.g., use of triple-modular redundancy, is not the most effective way to improve dependability. Indeed, while increasing the probability of failure-free action on an initial time interval, it actually reduces the mean-time between failures (MTBF) due to a sharp increase in the overall amount of hardware. MTBF is, however, a critical parameter for embedded systems.

A more effective way to improve dependability of systems with long life, such as, for example, satellite probes, is the use of multi-processor or massively parallel systems with automatic reallocation of tasks between the remaining correct processors as some of them become faulty.

Let n be the minimum number of processors required for the system to perform its functions, k the number of redundant processors, T_n the MTBF of a system with zero redundancy and λ the failure rate of a single processor (we assume here exponential distribution), then $T_n = \frac{1}{n\lambda}$ and

$$(k+1)T_n > T_{n+k} > (k+1)T_n \frac{1}{1+k/n}.$$

Today, dependability and survivability of systems becomes increasingly dependant on the reliability of system interconnects and inter-processor communication channels. The latter must be several orders of magnitude higher than the reliability of a single processor. Indeed, if we consider the situation when the system bus is acquired by a faulty processor this may result in a catastrophic failure of the entire system. One of the ways to principally improve the dependability of the communication system is through the use of *self-repair* and *self-recovery*.

The procedures of self-repair and self-recovery firstly require the localization of faulty parts for their subsequent replacement by spare units or for their recovery. The efficiency of self-repair and self-recovery significantly depends on the size of the units being replaced or subject to recovery. For communication systems it is quite reasonable to subdivide communication channels into sections (or nodes) with bit-slice organization [3]. We will therefore consider here, without loss of generality, communication systems based on a multi-token torus, ring (cf. the one whose design was described in [4]), or arbitrary graph. As a unit of fault localization, self-repair or self-recovery we will accept a one-bit line between two adjacent nodes, including its drivers and receivers [5].

The key issue in our presentation is the use of the principles of *self-timing* [1] for building a communication system with self-repair and self-recovery. There are three main motivating factors behind this idea:

- self-timing helps synchronize and communicate events in the system without using a global clock;
- self-timing itself improves the dependability of the system through the automatic adaptation of its temporal behavior to the variations in the values of

operational parameters, such as temperature, supply voltage, aging etc.;

– self-timing provides the system with the property of *self-checking*.

Let us consider the latter issue in more detail.

In a system with modular organization, self-timing is a way to realize the dynamic behavior of the modules by organizing their interaction using causal relations. The behaviour of each module is co-ordinated through its handshake interface with the environment, which includes all other modules of the system and the outside world. The handshake can be implemented in the form of one-bit Request (Req) and Acknowledgement (Ack) signals, as well as using special multi-bit codes. It is crucial that delays in handshake circuitry are assumed to be variable and unpredictable and that the system's behaviour is insensitive to their possible variations. We call this property *modular delay-insensitivity*.

A stuck-at fault in a delay-insensitive module is equivalent to the insertion of an infinite delay in the handshake circuitry and stoppage of the system. Consequently, the absence of response in a handshake after a critical time interval can be interpreted as the presence of a stuck-at fault inside the module. This simple observation manifests the property of self-checking and thus facilitates the localization (up to a module) of stuck-at faults. The transition from one-bit handshake signalling to multi-bit one, allows one to apply the same principles for the detection of some intermittent faults and failures.

As an illustration of the above ideas we consider the implementation of a multi-token system channel with self-recovery.

References

[1] V. Varshavsky (Ed.), *Self-Timed Control of Concurrent Processes*, Kluwer AP, Dordrecht, 1990.

[2] A. Avizienis, H. Kopetz and J.C. Laprie (Eds.) *Dependability*, Springer, N.Y. , 1991.

[3] V. Varshavsky, V. Marakhovsky and A. Yakovlev, "Towards self-checking and self-recovery in embedded self-timed systems", *Proc. IEEE Int. Workshop on Embedded Fault-Tolerant Systems, Dallas, Texas, Sept. 1996*, .

[4] A. Yakovlev, V. Varshavsky, V. Marakhovsky and A. Semenov, "Designing and asynchronous pipeline token ring interface", *Proc. 2nd Working Conf. on Asynch. Design Methodologies, London, May 1995*, IEEE CS Press, N.Y., pp. 32-41, 1995.

[5] V.I. Varshavsky, V.B. Marakhovsky, L.Ya. Rosenblum, Yu.S. Tatarinov, V.Ya. Volodarskii, A.V. Yakovlev, Algorithmic and structural organization of test and recovery facilities in a self-synchronous ring, *Control and Computer Science*,(Translated from Russian, Alerton Press), vol. 23, No.1,pp. 53-58, (1989).