

# Designing Reliable Embedded Systems Based on 32 Bit Microprocessors

C. Bolchini, F. Salice, D. Sciuto  
Politecnico di Milano – Dip. Elettronica e Informazione  
P.zza L. da Vinci, 32 – 20133 Milano – Italy  
bolchinilsalicesciuto@elet.polimi.it

## Abstract

Mission critical applications need reliable systems in order to prevent data integrity or at least the detection of failure in the system for discarding erroneous data to be used. We present a design approach for the realization of an embedded processor able to self-detect faults during its operational life. Such a system can also be considered as a starting point for achieving fault-tolerant properties.

## 1. Introduction

Target of the approach presented in this paper is the microprocessor core of an embedded system. The aim of the design methodology is the runtime detection of faults occurring in the CPU, assuming that assessed techniques can be adopted for the remaining parts of the complete system. It is worth noting that the proposed design methodology is mainly conceived for off-the-shelf components in order to meet the set of design requirements imposed, by the consumer electronics market (i.e. short time to market, high flexibility...).

The proposed fault detection technique is based on the straightforward duplication and comparison scheme, applied to the CPU. The attention is devoted to the design of an interface able to concurrently detect the failure of the CPU (the nominal or the duplicated, *checking* one) or its own failure, thus achieving complete fault coverage with respect to a single module fault model. Our goal is to define a methodology suited for achieving reliable 32 bit microprocessors based on the FRC scheme [1], by designing a self-checking component for complete fault coverage. The self-checking component is the *Interface for Functional Redundancy Checking*, IFRC.

The realization of an FRC system refers to the general scheme presented in Figure 1. The logical scheme is constituted by the nominal system opportunely interfaced with the checking microprocessor subsystem.

The functionality of the *interface* performing the

*functional checking* (IFRC) is dynamically modified in relation to the activity of the system, which is univocally determined by the control bus. The role of the interface is twofold: it compares bus signals revealing possible differences due to temporary or permanent faults inside and outside the microprocessor subsystem (signal anomalies on the bus), and it allows signals directed toward the nominal processor to reach the checking processor.

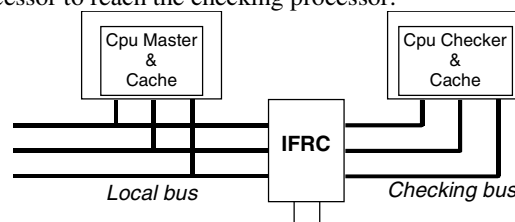


Figure 1. The architectural description of the IFRC.

There are two significant aspects in the design of a self-checking interface: a) when dealing with bi-directional signals (as data signals are) it is necessary to monitor them only when they are generated from the CPU and are valid; b) it is necessary to deal with the fact that the CPU is not always the bus master, also behaving as a slave during DMA controlled data transfers.

The concurrent error detection property of the global embedded system is achieved by allowing both CPUs, nominal and checking, to operate independently on the same data. We consider data to be monitored the data directed toward the CPU master; the checking CPU and the interface being transparent to the rest of the system. If both CPUs are error free the signals they generate have to be equal. Error detection is thus implemented by comparing signals (control, data and address) produced by the two CPUs. The IFRC consists of a set of two-rail checkers (TRC) and a set of controllable two-rail checkers (CTRC), to inhibit the checking functionality when the interface receives signals not to be compared.

[1] Intel, Embedded Ultra Low Power Intel 486T GX Processor Datasheet, <http://developer.intel.com/design/intarch/DATASHTS/272755.HTM>