Abstract—This panel proposes a discussion on the role and challenges of embedded software testing. Invited speakers represent different industrial and academic points of view on this topic. Specifically, the panelists were invited to discuss what exactly defines the test of an embedded software, what is the role of the platform designer and of the software designer with respect to test, how embedded software testing really differs from traditional software testing, and whether current solutions and tools are sufficient for this task.

Embedded software testing; embedded test; software test;

I. INTRODUCTION

Software testing has grown as an important technique to evaluate and help to improve software quality. Numerous techniques and tools have appeared in the last decade, ranging from static analysis to automatic test generation and application.

One can argue that software is the dominant part of an embedded system, either as a final product (executable code) or during its development lifecycle (system modeling in specific languages and computation models). In both cases, software must be thoroughly verified to ensure product quality and reliability. One can observe a growing number of academic and industrial works on the topic of embedded SW testing in the last four or five years, and this seems to be a good time for reflection: how exactly is embedded software testing different from traditional software testing? Is it an engineering or computer science problem? Does it need extra support from platform developers? What is the role of the SW engineer and of the designer in developing a high-quality software-based embedded application?

Many authors suggest that, on top of the ordinary software testing challenges, software usage in an embedded application brings additional issues that must be dealt with: the variety of possible target platforms, the different computational models involved during the design, faster time-to-market and even more instable and complex specifications, platform-dependent constraints (power, memory, and resources availability), etc. On the other hand, current platforms are more and more powerful, and the specificities of the embedded application can help to reduce the search space during test generation: application domains, strong code reuse paradigm, use of less advanced programming language resources, and common availability of system models subject to or already verified with respect to specific properties, for instance. Furthermore, a major part of the so-called embedded software does not depend directly on hardware, and one can argue that only a small percentage really needs to be tested together with the target platform, and this test is part of the platform design, not the system design.

The panelists will present their view on the challenges for the test of embedded software. Specifically, the panelists are invited to answer/discuss the following questions:

1. How exactly embedded SW testing differs from traditional SW testing? Which are the target points for testing in an embedded SW? Are current software testing techniques and tools sufficient?

2. Can the SW testing help the system test? Can the HW help the SW testing? How can HW and SW integration be tested?

II. INVITED SPEAKERS

- Achin Nohl – CoWare, DE
- Bruce Douglass – IBM, US
- Florian Schäfer – Research & Consulting, DE
- Harmke de Groot – IMEC, The Netherlands
- Franco Fummi – University of Verona, IT

III. PANEL MODERATOR

- Chris Rowen, Tensilica, US