Published in: *5th International Conference on Software Testing*, Düsseldorf, April 2004.

Automated Testing of Aircraft Controller Modules

Dr. Oliver Meyer, Verified Systems International GmbH, Bremen
Aliki Tsiolakis, Technologiezentrum Informatik (TZI), Universität Bremen
Dr. Sven-Olaf Berkhahn, Airbus Deutschland GmbH, Hamburg
Josef Kruse, Airbus Deutschland GmbH, Hamburg
Dirk Martinen, Airbus Deutschland GmbH, Hamburg

Keywords: Integrated Modular Avionics, Hardware in the Loop Test, ARINC 653, AFDX, Victoria

Most aircraft families currently in use employ a large variety of microcontroller boards based on specialized hardware specifically designed for particular purposes. The Integrated Modular Avionics (IMA) approach aims at unifying the different hardware platforms by using general purpose hardware which allows to implement these different controller applications on a set of instantiated generic IMA modules. These modules will be the main controller hardware for future Airbus aircraft projects and accordingly, they are classified as criticality level A (in conformance with the standard RTCA DO178B).

IMA modules and their operating system are designed to support general aircraft controller requirements, such as

- hard real-time capability, since aircraft controllers are inherently time critical.
- spatial and temporal partitioning which allows to execute multiple applications on a single module without temporal or memory related interferences between these applications. If a single application fails it must not influence any other application. This segregation allows to integrate applications of different criticality levels on the same module.
- Built-In Test Equipment (BITE) facilities for detecting hard- and software defects during operation.
- a large variety of I/O interfaces including CAN, discretes, analogue, ARINC-429 and AFDX (Avionics Full Duplex), which is a fast ethernet technique based real-time communication frame protocol designed to replace ARINC-429. AFDX will be mainly used for inter-controller communication, i.e. communication between IMA modules.

The operating system is based on the ARINC standard 653. It is highly configurable in order to suit the needs of a large variety of controller applications. Apart from the temporal and spatial partitioning, the IMA Module Configuration Definition (ICD) allows to set up I/O message routing and I/O format descriptions and conversions as well as health monitoring reactions and general module characteristics such as caching behaviour or network addresses.

Testing of these controllers is performed in four integration steps:

- Test of the bare module plus operating system, but without application software or application-specific ICD.
- Test of the configured modules (module + OS + application specific ICD).
- Test of the whole network of configured modules, but still without application software.
- Test of the whole network of controllers using a step-wise integration of the application software.

The focus of this presentation will be the first step of this testing hierarchy.

For testing of the bare module, a two-part testing approach has been developed. One part consists of a test application (TA), which implements an interpreter for the API calls and is commanded via reserved AFDX I/O links. After receiving a command and its parameters, the TA executes the desired API call and immediately sends the return values back to the caller, which checks if the results are correct with regard to the current status of the module. Additionally, the TA contains a number of scenarios (which are basically parameterized sequences of API calls and C commands) which allow performing highly time critical sequences of actions without the communication delays caused by external commanding. Furthermore, the TA explicitly permits to call API functions with illegal parameters and thus makes it possible to perform API robustness tests also. In order to be able to check the timing/scheduling and partitioning properties, a configurable number of instances of the TA are executed concurrently within different partitions and processes of the IMA module, each of which can be addressed separately from the outside.

Verified Systems real-time test system RT-Tester builds the external part of this automated hardware-in-the-loop testing approach. In addition to selecting, sending and checking commands for the TA instantiations executed in parallel, it controls the I/O Interfaces of the IMA module. In this, it is also possible to check the interdependabilities between I/O communication, module internal I/O handling and operating system calls on application level. These tests are performed with regard to message integrity, timing properties and independability as well as robustness of I/O devices.

In order to reflect the fact that the IMA module operating system is highly configurable, a test approach based on test specification templates has been chosen. Test specifications in the formal language CSP are taken from a test template library and instantiated for the current ICD. An ICD parser generates CSP include files containing the most important aspects of the current configuration thus allowing the instantiated tests to check exactly the currently configured module properties. Additionally, an ICD generator has been developed to semi-automatically produce a large variety of different configurations, both legal and illegal. This combined parser/generator approach makes it possible to use the same test specification templates for the automated testing of differently configured modules, for normal range tests as well as for robustness tests.

The described concept and the test suite for automated IMA module testing has been developed by TZI and Verified Systems International GmbH in collaboration with Airbus Deutschland GmbH. It is applied in the context of the European research project VICTORIA using the system integration test bench for the cabin domain. VICTORIAs main objective is to design, develop and validate new systems which integrate all on-board functions for commercial airliners using standardized components and open standards.