Validated Software Integration Testing Applicability for Multiple RTOS Configurations

Scott Nowell
President, Validated Software Corporation

6 May 2010

1902 Wright Place, Suite 200 • Carlsbad, California 92009 • (760) 531-5290

Validated Software Integration Testing Applicability for Multiple RTOS Configurations

Introduction

The Validated Software µC/OS-II Avionics Validation Suites™ and µC/OS-II Avionics Validation Kits™ contain a set of artifacts intended as part of a DO-178B submission and provide evidence as to the appropriateness of µC/OS-II use in an avionics application. Use of the various artifacts warrants some explanation. Of particular interest are methodologies used in both the verification and validation of µC/OS-II and how they apply to µC/OS-II use in the field. Assuming the applicability of **integration testing** with a single set of configuration options, this paper discusses the tests' validity when µC/OS-II is deployed utilizing a different set of configuration options.

In the context of this discussion, µC/OS-II configuration options are restricted in scope to:

- µC/OS-II components, services or API's that are linked in to the object code
- Compiler and linker settings that are used for testing as well as production code being utilized in the final application

The Validated Software DO-178B integration-testing procedure complies with RTCA DO-178B and does so in a fashion that minimizes expense and maximizes flexibility for the end customer. Please bare in mind that integration testing, in and of itself, is not the only output of compliance with DO-178B; it is one part of a comprehensive approach to reliability.

Integration Testing

Integration testing falls in the middle of the three-tiered testing strategy utilized in aviation: Unit, Integration, and System testing. Unlike unit testing where the focus is on verifying the internal logic of each RTOS function, the purpose of integration testing is to ensure the distinct components of the RTOS work in accordance to user expectations (as defined by the user manual and the software requirements document). Test cases are developed with the express purpose of exercising the interfaces between the components.

The following section covers the assumptions and expectations that VSC has and our rational for the assumption that the current methodology for integration testing is not only valid for our test bench, but also for user applications that go to production using a different configuration.

Overview of µC/OS-II Configuration and Testing

Developed for use in a wide range of applications, µC/OS-II provides easy and flexible configuration. As a result, it has an almost infinite number of permeations in the ways it can be configured. The benefits are many, but one of the greatest is the ability to enable only the features needed by a given system. The various configurations are controlled with the os_cfg.h configuration file. Like many commercial operating systems, µC/OS-II was developed with an eye toward portability. The os_cfg.h file contains settings that select both enabled services and API's and settings that relate to specific hardware platforms.

Some believe that integration testing only yields results and provides assurance for one configuration. That is not the case. The following sections describe in detail how DO-178B Validation Suite/Kit integration testing is applied and why it is valid across all µC/OS-II configurations.

The goal of the µC/OS-II Integration Testing is to exercise and stress all aspects of the RTOS, with particular emphasis on system boundary conditions and the ability of the RTOS to detect and report attempts to exceed the defined boundaries.

µC/OS-II Configuration Assumptions

The os_cfg.h supplied with the Validation Suite/Kit Integration Tests is pre-configured for the values needed by the Integration Tests. It is also configured to enable all services available for use in µC/OS-II. The "global" approach taken by VSC, i.e. configuring all services, provides a single set of tests that cover all supported combinations.

The global configuration for VSC Integration Tests was chosen to provide the maximum flexibility in testing while representing real world implementations. µC/OS-II has been designed such that it allows either a limited or unlimited quantity of "items". For example, in the case of tasks there is a hard limit of 255. By system definition, the lowest possible priority for a task is the value 254; any numerically larger value will cause a compile time error. In the case of events there is no limit, other than available memory, on the number of events that can be configured.

With µC/OS-II task allocation, tasks are defined in a fixed size array, so the size is deterministic. For each defined task there is one task control block (TCB) allocated in the OSTCBTbl.

Unlike tasks, µC/OS-II events are allocated locally and connected via pointers. Each event control block (ECB) points to the next event control block. The final ECB contains a NULL pointer to indicate it is the end of the list.

All dependent counters, e.g., loop counter on task lists, within µC/OS-II have been sized to be of a type larger than the maximum count. For unlimited items such as events, the index counters have been defined as unsigned 16 bit values, which is considered to be far beyond the practical needs of an RTOS of the scale of µC/OS-II. The following table shows the configurable items in µC/OS-II**.**

Item	Min	Max	Variable Size (bits)	Description
TASKS	**2**	**254**	**8**	Number of allowed tasks in the application.
Events	**0**	**0xffff**	**16**	Total number of events (Semaphores , Mailboxes, etc.) allowed in the system..
Flags	**0**	**0xffff**	**16**	Total number of flags allowed in the system
Memory Partitions	**0**	**0xffff**	**16**	Total number of Memory segments allowed in the system
Queues	**0**	**0xffff**	**16**	Total number of Queues allowed in the system
Timers	**0**	**0xffff**	**16**	Total number of Timers allowed in the system

µC/OS-II Integration Test Configuration Assumptions

In the case of event/item quantities, the configuration used with the integration tests has been chosen to represent real world use. Some of the tests require a minimum number of items such as mailboxes. If too few have been configured then the test will not build properly and a compilation error will be reported.

As none of the quantities exceed C type boundary limits, the actual quantity used and tested is irrelevant. The testing always exercises boundary conditions by creating the configured number of items, regardless of the configured quantity, to prove that the configuration is achievable. The tests then attempt to exceed the boundary by allocating one more of the same item than is configured to insure that an error is returned.

Configuration of Critical Items for Production Releases

Certain of the configuration items can be configured with new values, but have critical limits on the maximum and minimum values. Some items should not be enabled as they will introduce untested code in the object code. This section identifies these items and lists the limits.

Item	Min	Max	Description
OS_MAX_TASKS	**2**	**254**	Number of allowed tasks in the application.
OS_ARG_CHK_EN	**1**		If this value is set to zero (0), external provision must be provided to validate all function arguments.
SAFETY_CRITICAL_RELEASE	**1**		This value should be defined for production release code to force testing of safety critical settings. The defined value is irrelevant.

Critical items that must not be configured in Production Releases

Certain of the configuration items should not be enabled as they will introduce untested code in the object code. This section identifies these items and lists the limits.

Item	Value	Description
OS_TICK_STEP_EN	**0**	This value must not be enabled in production/flight code. Enabling this feature activates untested code.
OS_DEBUG_EN	**0**	This value must not be enabled in production/flight code. Enabling this feature activates untested code.

Summary

Integration testing is one of many components involved in the software lifecycle. The goal of integration testing is to test the external software interface between it and the system as a whole.

VSC's approach to integration testing is to test µC/OS-II as a superset of its functionality. Thus ensuring EVERY external interface performs as specified.

When designed and tested:

- With proper design methodologies and processes
- Using configuration values that do not exceed the range of values proven by compile time test macros
- Combined with code coverage, structural analysis, and code inspection
- As a total system

we can conclude that the testing of µC/OS-II using Validated Software's integration test methodology is a valid indication that all other supported configurations shall work as designed.

The final bullet statement, "As a total system", is key to our conclusion that the VSC Integration Tests are comprehensive tests. Regardless of the testing done on the RTOS as a component, it is always tested in context in the finished avionics system. The VSC Integration Tests exercise and test all of the boundary conditions of the RTOS, but even this is only a portion of the overall testing. In addition to boundary and robustness testing, µC/OS-II is subjected to 100% unit test coverage to guarantee that all code and decision paths are thoroughly tested. It is then tested again in the final avionics system using the avionics system configuration.

As stated earlier, integration testing is not sufficient in and of itself to stand alone as a complete test of device software. It is one small component that when taken together with the remaining tests provides assurance that the system as a whole will perform as designed.

Visit ValidatedSoftware.com for the latest product information.

For more information about µC/OS-II, refer to www.Micrium.com. For more information about the Validation Suite, see www.ValidatedSoftware.com. Complete product information, including sample of each DO-178B document in the Validation Suite can be requested from the Validated Software Sales office. Call 251.649.1693 or email Sales@ValidatedSoftware.com.

Validated Software
1902 Wright Place, Suite 200 • Carlsbad, California 92009 • (760) 531-5290
E-mail: info@ValidatedSoftware.com