AMC 20-193 Use of multi-core processors

1. Purpose

1.1 This AMC describes an acceptable means, but not the only means, for showing compliance with the applicable airworthiness specifications for aspects related to multi-core processors (MCPs) contained in airborne systems and equipment used in product certification or ETSO authorisation. Compliance with this AMC is not mandatory, and an applicant may elect to use an alternative means of compliance. However, the alternative means of compliance must meet the relevant requirements, ensure an equivalent level of safety, and be approved by the Agency on a product or ETSO article basis.

1.2 This AMC provides objectives for the demonstration of compliance with the applicable airworthiness specifications for airborne systems and equipment that contain MCPs, according to the applicability in Section 2 of this AMC.

2. Applicability

2.1. This AMC may be used by applicants, design approval holders, and developers of airborne systems and equipment, which contain MCPs, to be installed on type-certified aircraft, engines, and propellers. This also includes developers of ETSO articles.

This AMC applies to systems and equipment that contain MCPs with two or more activated cores for which the item development assurance level (IDAL) of at least one of the software applications hosted by the MCP or of the hardware item that contains the MCP is A, B, or C. The deactivation of cores is handled through the applicable airborne electronic hardware (AEH) guidance.

This AMC does not apply when the IDALs are all Level D or E.

If an applicant modifies the use of the MCP (such as by activating one or more additional cores or adding software of IDAL A, B, or C), then the applicant should reassess the applicability of this AMC.

Section 5.7 of this AMC describes the objectives that apply according to the assigned IDAL (A, B, or C) of the hosted software or of the hardware item that contains the MCP.

2.2. Aspects not covered by this AMC

The following aspects are not covered by this AMC. This does not constitute an exemption, i.e. the objectives of this AMC are still applicable if an applicant uses these features.

Any applicant who uses these features should describe how they are used so that the behaviour of the MCP is not altered, and determinism is still guaranteed.

In their planning activities, the applicant should present the methods employed to cover these aspects, and satisfy the objectives of this AMC or show compliance with the applicable airworthiness specifications if they propose an alternative to this AMC or part of it.

2.2.1. Dynamic allocation of software applications

An assumption in this AMC is that software applications are statically allocated to cores during the start-up of the MCP software, but not during the subsequent operation of the software.

This AMC does not cover MCP platforms on which software applications or tasks can be dynamically reallocated to a different core (or different cores) by the operating system, a software hypervisor, or by other means.

However, justification for using dynamic allocation features within the scope of this AMC may rely on robust and proven limitations that lead to deterministic behaviour, such as:

— restricted usage permitting the applicant to claim equivalence to the conditions expressed in this AMC (for example, multi-static allocation, i.e. selection of a prequalified configuration, instead of pure dynamic allocation).

2.2.2. Simultaneous multithreading support

This AMC does not cover simultaneous multithreading, as this issue is not specific to MCPs.

2.3. Exceptions

An MCP may contain multiple cores of different types, which may interact in different ways and some of the interactions do not produce interference. Therefore, the objectives of this AMC do not apply to the interactions between two or more activated cores of an MCP in the following cases:

— The activated cores are set up in lockstep mode (in lockstep processors with two or more activated cores, the cores host the same software and execute that same software in parallel so that their outputs, based on identical input data, can be compared for use in a safety-critical application); or

— The activated cores are only linked by the conventional databuses typically used in avionics systems, and not by any of the following: shared memory, shared cache, or a 'coherency fabric' / 'module interconnect'. This category includes the case where the cores only act as co-processors or graphics processors, each under the control of another core that executes software.

The objectives of this AMC apply to the interactions between all the other activated cores of an MCP.

3. Background

MCPs can execute several software applications at the same time by hosting them on different cores; therefore, several software applications and/or hardware functions may attempt to access the same shared resources of the MCP (such as memory, cache, 'coherency fabric' / 'module interconnect', or external interfaces) at the same time, causing contention for those resources.

Most MCPs have internal features to handle and arbitrate the concurrent demands for MCP resources, which may cause delays in access to the resources. These delays are a form of time interference between the software applications or tasks, which can cause the software applications to take much longer to execute than when executing on their own.

The execution of software applications may be different on MCPs than it is on single-core processors (due to parallelism and other MCP mechanisms, or software components such as operating systems or hypervisors). This may result in new or different data or control coupling paths, and functional interference between the software applications or tasks.

Interference between the software applications or tasks executing on an MCP could cause safety-critical software applications to behave in a non-deterministic or unsafe manner, or could prevent them from having sufficient time to complete the execution of their safety-critical functionality.

4. Definitions

Applicable airborne electronic hardware (AEH) guidance: AMC 20-152( ) and any project-specific guidance.

Applicable software guidance: AMC 20-115( ) and any project-specific guidance.

Asymmetric multi-processing (AMP): an MCP software architecture in which each individual functional task is permanently allocated to a specific core and each core has its own operating system (however, the operating systems may be multiple copies of the same operating system or be different from core to core).

Bound multi-processing (BMP): an MCP software architecture that restricts symmetric multi-processing (SMP) (see definition below) architecture by constraining tasks to be bound to specific cores while using a common operating system across all cores.

Determinism/deterministic: the ability to produce a predictable outcome generally based on the preceding operations and data. The outcome occurs in a specific period of time with repeatability (definition taken from ED-124/DO-297).

Integrated modular avionics (IMA) platform: in this AMC, this term refers to an integrated modular avionics MCP platform that provides both robust resource partitioning and robust time partitioning (as defined in this AMC).

Intended final configuration: the configuration of the software and hardware in which the set of the MCP resources has been defined by implementing the configuration settings and all software components have been installed on the target MCP.

Interference channel: a platform property that may cause interference between software applications or tasks.

Item: a hardware or software element that has bounded and well-defined interfaces (definition taken from ED-79A / ARP 4754A).

Item development assurance level (IDAL): the level of rigour of development assurance tasks performed on item(s), e.g. IDAL is the appropriate software level in ED-12C / DO-178C and design assurance level in ED-80 / DO-254 objectives that need to be satisfied for an item (definition taken from ED-79A / ARP 4754A).

MCP platform: it consists of the MCP itself and, in many cases, the platform software, such as an operating system and/or software hypervisor, which provides the interface between the software applications and the MCP.

MCP platform with robust partitioning: an MCP platform that complies with the objectives of this AMC and provides robust resource partitioning and robust time partitioning as defined in this AMC, not only between software applications hosted on the same core, but also between software applications hosted on different cores of an MCP or between software applications that have tasks hosted on several cores.

Multi-core processor (MCP): an AEH device that contains two or more processing cores. A core in an MCP is defined as a device that executes software. This includes virtual cores (e.g. in a simultaneous multithreading microarchitecture, although as stated in Section 2.2.2, the use of simultaneous multithreading is not covered by this guidance). An MCP is typically implemented in a device that may also include resources such as memory or peripheral controllers, internal memory, peripherals, and internal interconnects.

Robust partitioning: both robust resource partitioning and robust time partitioning.

Robust resource partitioning (adapted from ED-94C / DO-248C and ED-124 / DO-297): robust resource partitioning is achieved when:

— software partitions cannot contaminate the storage areas for the code, I/O, or data of other partitions;

— software partitions cannot consume more than their allocation of shared resources; and

— failures of hardware unique to a software partition cannot cause adverse effects on other software partitions.

Note: Software that provides partitioning should have at least the same IDAL as the highest IDAL of the software that it partitions.

Robust time partitioning (on an MCP): this is achieved when, as a result of mitigating the time interference between partitions hosted on different cores, no software partition consumes more than its allocation of execution time on the core(s) on which it executes, irrespective of whether partitions are executing on none of the other active cores or on one, more than one, or all of the other active cores.

Safety net: a safety net is defined as the employment of mitigations and/or protections at the appropriate level of aircraft and system design as a means to satisfy the safety objectives. The safety net methodology may be applied when it is assumed that part of a system will misbehave. The safety net is by nature independent of the source of misbehaviour. The safety net can include passive monitoring functions, active fault avoidance functions, and control functions for effective recovery of system operations from anomalous events.

Simultaneous multithreading: this is when virtual cores are used to execute more than one execution thread on a single physical core.

Software application: generally, it designates the software part of a function installed on an MCP.

Software component: any part of the software which may access MCP shared resources; it may designate either a software application or an operating system or a hypervisor.

Hardware component: any part of the hardware which may independently access MCP shared resources.

Symmetric multi-processing (SMP): an MCP software architecture in which a single operating system controls the execution of the software on multiple cores and may dynamically allocate tasks to cores at run-time.

Task: the smallest unit of software execution that can be managed independently by a scheduler. For the purposes of this AMC, this term encompasses ‘threads’ or ‘processes’ (in the sense of

ARINC 653). For simplification in this AMC, when addressing interference, a task also represents any part of an application or any part of a software component that executes on one core.

5. Multi-core processor (MCP) guidance

This section takes stages of a typical life cycle of a project involving an MCP in turn, explains the important issues involved in each stage, and provides objectives for applicants to meet for each of those stages.

The applicant should meet the objectives of this AMC, with the exception of any objective or part of an objective that the applicant justifies as not being applicable to the MCP in their system or equipment (e.g. if the MCP mechanism addressed does not exist on the selected MCP). The applicant should state in the appropriate deliverable document which particular aspects do not apply and explain why they do not apply.

Some of the objectives have notes provided after them. These notes should be considered to be part of the objectives, as they provide additional information that is relevant to the objectives. Objectives and their included notes are formatted in italics to differentiate them from the rest of the text.

5.1. Planning

The additional planning objectives below clarify the information to be included in the applicable plans to achieve planning data [standardisation]/[standardization] for projects with MCPs.

Objective MCP_Planning_1

The applicant's plans or other deliverable documents:

1. *Identify the specific MCP, including the unique identifier from the manufacturer.*
2. *Identify the number of active cores.*
3. *Identify the MCP software architecture to be used and all the software components that will be hosted on the MCP.*
4. *Identify any dynamic features provided in software hosted on the MCP that will be activated, and provide a high-level description of how they will be used.*
5. *Identify whether or not the MCP will be used in an integrated modular avionics (IMA) platform to host software applications from more than one system.*
6. *Identify whether or not the MCP platform will provide robust resource partitioning and/or robust time partitioning as defined in this AMC.*
7. *Identify the methods and tools to be used to develop and verify all the individual software components hosted on the MCP so as to meet the objectives of this AMC and the applicable software guidance, including any methods or tools needed due to the use of an MCP or the selected MCP architecture.*

Notes:

a) The MCP software architecture includes asymmetric multi-processing (AMP), symmetric multi-processing (SMP), or any other architecture used by the applicant.

b) The software components identified should include any operating systems, hypervisors, software applications, and all functions that are provided in software. In the case of an MCP used in an IMA platform, the software components that are identified do not have to include the hosted software applications.

c) The dynamic features provided in software should include such aspects as the dynamic allocation of software applications or tasks to cores and any other software dynamic features that can affect the execution of the software while it is executing.

d) Where the applicable software and AEH guidance calls for independence in meeting the objectives, the plans should identify how verification independence will be applied.

Multiple software applications and/or hardware functions may use resources of the MCP and may cause contention for resources and interference between software applications or tasks. Even if there is no explicit data or control flow between software applications or tasks running concurrently on different cores, MCP resources (e.g. cache or interconnects) may be shared. Therefore, coupling may exist on the platform level which can cause interference between the software applications or tasks and cause increases in the worst-case execution times (WCETs) of the software applications. In addition, there could be interaction between software and hardware functions that would need to be considered (e.g. cases where there are multiple masters).

Objective MCP_Planning_2:

The applicant's plans or other deliverable documents:

1. Provide a high-level description of how MCP shared resources will be used and how the applicant intends to allocate and verify the use of shared resources (see Note a)) so as to avoid or mitigate the effects of contention for MCP resources and to prevent the resource capabilities of the MCP from being exceeded by the demands from the software applications and/or the hardware components of the MCP.

2. Identify the MCP hardware resources to be used to support the objectives in this AMC.

3. Identify any hardware dynamic features of the MCP that will be active, and provide a high-level description of how they will be used.

4. Identify the aspects of the use of the MCP that may require a safety net or other mechanisms to detect and handle failures in the MCP.

Notes:

a) The description of the use of shared resources should include any use of shared cache (taking into account the time interference it may cause due to cache misses or other effects) or shared memory (taking into account the time interference and the data and control flow effects it may cause, such as lockouts, race conditions, data starvation, deadlocks, live-locks, or excessive data latency). The description of shared resources should also include any use of shared interconnect and take into account the time interference due to arbitration for access to the shared interconnect.

b) Hardware dynamic features of the MCP include any features that can alter the behaviour of the MCP or the hosted software during execution — for example, energy-saving features (clock

enable / gating, frequency adaptations, deactivating one or more cores, or dynamic control of peripheral access).

5.2. The setting of MCP resources

In the context of MCPs, some of the configuration settings are especially relevant to the MCP hardware and software architectures, such as:

— which cores are activated;

— the execution frequencies of the cores;

— the priorities and allocation of shared interconnect;

— which of the peripheral devices of the MCP are activated;

— whether shared memory or shared cache is used, and how each is allocated; and

— whether dynamic features that are built into some MCPs are allowed to alter the frequency of execution of the cores or to deactivate one or more cores in order to save energy (this might not be desirable for cores that host safety-critical software applications).

Objective MCP_Resource_Usage_1:

The applicant has determined and documented the MCP configuration settings that will enable the hardware and the software hosted on the MCP to satisfy the functional, performance, and timing requirements of the system.

Objective MCP_Resource_Usage_2:

RESERVED.

Covered by AMC 20-152A, Objective COTS-8.

5.3. Interference channels and resource usage

As stated above, the software applications or tasks that execute on different cores of an MCP share MCP resources; so, even if there is no explicit data or control flow between these software applications or tasks, coupling exists on the platform level, which can cause interference between them.

There may be software or hardware channels through which the MCP cores or the software hosted on those cores could interfere with each other, in addition to those channels specifically mentioned in this AMC. For instance, many MCPs include an 'interconnect' / 'coherency fabric', through which the demands for MCP resources (e.g. from the software applications hosted on the MCP) are channelled and the demands are arbitrated. This arbitration can cause interference effects such as jitter on data arrival times, data consistency issues, or it can change the order in which transactions requested by the software applications are executed.

Non-deterministic behaviour of the hosted software applications may occur due to such interference.

Moreover, the complexity of the MCP, executing tasks in parallel and the interference could lead to the demands for resources exceeding the available resources. For instance, if the demands for interconnect transactions are very high in MCPs with a very high level of external databus traffic, the interconnect can become overloaded, which can affect transactions on some MCPs.

MCP_Resource_Usage_3:

The applicant has identified the interference channels that could permit interference to affect the software applications hosted on the MCP cores, and has verified the applicant's chosen means of mitigation of the interference.

Notes:

a) This objective includes the identification of any interference caused by the use of shared memory, shared cache, an interconnect, or the use of any other shared resources, including shared peripherals, and the verification of the means of mitigation chosen by the applicant.

b) If the applicant identifies interference channels that cannot affect the software applications in the intended final configuration, then those interference channels do not need to be mitigated and no verification of mitigation is needed.

c) The applicant should handle any interference channel discovered at any time during the project in the same manner as in this objective and these explanatory notes.

d) If the highest IDAL of the MCP hardware and of all the software applications hosted on the MCP is C, and the hosted software applications are not required by the safety analysis to be robustly partitioned, then the applicant has the option to not conduct an interference analysis and, therefore, to not meet this objective. However, applicants should note that opting to not meet this objective affects the manner in which they are permitted to conduct their software verification (see Objective MCP_Software_1 *and* Note c) *of that objective.)*

MCP_Resource_Usage_4:

The applicant has identified the available resources of the MCP and of its interconnect in the intended final configuration, has allocated the resources of the MCP to the software applications hosted on the MCP, and has verified that the demands for the resources of the MCP and of the interconnect do not exceed the available resources when all the hosted software is executing on the target processor.

Note: The use of worst-case scenarios is implicit in this objective.

5.4. Software verification

The software verification processes in the applicable software guidance need to be adapted for use on an MCP to demonstrate that the hosted software applications function correctly and have sufficient time to execute in the presence of the interference that occurs when all the hosted software is executing on an MCP.

With an MCP, there may be data and control flows between software components or tasks hosted on different cores of the MCP. Therefore, the data and control coupling analysis performed on the software hosted on each separate core (as required by the applicable software guidance) may not reveal the improper software behaviour associated with features such as hardware runtime optimisations and memory models on MCPs.

The WCET of a software component or task may increase significantly when other software components or tasks are executing in parallel on the other cores of an MCP. This could cause some software applications to have insufficient time to complete the execution of their safety-critical functionality.

Interference and interactions between software applications or tasks occur via the proprietary internal mechanisms of an MCP. Any simulation of those mechanisms is, therefore, less likely to be representative in terms of functionality or execution time than testing conducted on the target MCP in the intended final configuration, and thus is less likely to detect errors.

To adapt the software verification guidance for different types of MCP platforms, the two following categories of MCP platforms are considered:

— MCP platforms with robust partitioning, and

— all other MCP platforms.

MCP_Software_1:

The applicant has verified that all the software components hosted by the MCP meet the objectives of the applicable software guidance. In particular, the applicant has verified that all the hosted software components function correctly and have sufficient time to complete their execution when all the hosted software and hardware of the MCP is executing in the intended final configuration.

The way in which the applicant should satisfy this objective depends on the type of the MCP platform:

— ***MCP platforms with robust partitioning:***

Applicants who have verified that their MCP platform provides both robust resource partitioning and robust time partitioning (as defined in this AMC) may verify software applications separately on the MCP and determine their WCETs separately.

— ***All other MCP platforms:***

Applicants may verify separately on the MCP any software component or set of requirements for which the interference identified in the interference analysis is mitigated or is precluded by design. Software components or sets of software requirements for which interference is not avoided or mitigated should be tested on the target MCP with all software components executing in the intended final configuration, including robustness testing of the interfaces of the MCP.

The WCET of a software component may be determined separately on the MCP if the applicant shows that time interference is mitigated for that software component; otherwise, the WCET should be determined by analysis and confirmed by test on the target MCP with all the software components executing in the intended final configuration.

Notes:

a) All the interfaces between the hosted software and the hardware of the MCP should be included in this testing.

b) The robustness testing mentioned above is intended to cover the specific aspects of an MCP that are not specifically covered by the standard verification activities described in the applicable software guidance.

c) If the highest IDAL of the MCP hardware and of all the software applications hosted on the MCP is C, and the hosted software applications are not required by the safety analysis to be robustly partitioned, then the applicant has the option to not conduct an interference analysis and therefore to not meet Objective MCP_Resource_Usage_3. In such a case where no interference

analysis has been performed, the hosted software components should be verified according to this objective as components for which interference is not avoided or mitigated and for which separate verification is, therefore, not permitted.

d) To 'verify separately' and 'determine the WCET separately' mean to conduct these activities without all the software executing at the same time on other cores of the MCP.

e) Interference may occur between tasks of a single component when the tasks execute on different cores.

MCP_Software_2:

The applicant has verified that the data and control coupling between all the individual software components hosted on the same core or on different cores of the MCP has been exercised during software requirement-based testing, including exercising any interfaces between the software components via shared memory and any mechanisms to control the access to shared memory, and that the data and control coupling is correct.

Notes:

a) When this objective cannot be completely met during the software verification, applicants may propose to use system-level testing to exercise the data and control coupling between software components hosted on different cores.

b) Interference may occur between tasks of a single component when the tasks execute on different cores.

5.5. Error detection and handling, and safety nets

As well as the types of errors and failures normally detected and handled in a system that incorporates a single-core processor, additional types of errors and failures may need to be detected and handled in an MCP environment due to problems caused by the features of MCPs and due to the additional complexity of executing several software applications or tasks in parallel in real time.

Features of an MCP may, therefore, contain unintended functionality that may cause errors and produce unexpected behaviour. Applicants may, therefore, wish to consider the use of a 'safety net' independent from the MCP to detect and handle failures within the MCP and to contain any such failures within the equipment in which the MCP is installed.

MCP_Error_Handling_1:

The applicant has identified the effects of failures that may occur within the MCP and has designed, implemented, and verified means commensurate with the safety objectives, by which to detect and handle those failures in a fail-safe manner that contains the effects of any failures within the equipment in which the MCP is installed. These means may include a 'safety net' independent from the MCP.

5.6. Data to complement the accomplishment summaries

The applicant is expected to describe how the objectives of this AMC were satisfied.

MCP_Accomplishment_Summary_1:

In addition to providing the information requested by the applicable software and AEH guidance, the applicant has provided documentation that summarises how they have met each of the objectives of this AMC.

5.7. Applicability of the MCP objectives according to their IDALs

The column 'IDAL A or B' shows the objectives applicable when the highest IDAL of any of the software applications hosted by the MCP or of the MCP hardware device is A or B.

The column 'IDAL C' shows the objectives applicable when the highest IDAL of any of the software applications hosted by the MCP or of the MCP Hardware device is C.

MCP OBJECTIVES	IDAL A or B	IDAL C
MCP_Planning_1	**Yes**	**Yes**
MCP_Planning_2	**Yes**	**Yes**
MCP_Resource_Usage_1	**Yes**	**Yes**
MCP_Resource_Usage_2	Covered by AMC 20-152A Objective COTS-8	**n/a**
MCP_Resource_Usage_3	**Yes**	**Refer to Note d**
MCP_Resource_Usage_4	**Yes**	**No**
MCP_Software_1	**Yes**	**Yes**
MCP_Software_2	**Yes**	**Yes**
MCP_Error_Handling_1	**Yes**	**No**
MCP_Accomplishment_Summary_1	**Yes**	**Yes**

6. RELATED REGULATORY, ADVISORY, AND INDUSTRY MATERIAL

(a) Related EASA Certification Specifications (CSs)

(1) CS-23 *Certification Specifications and Acceptable Means of Compliance for Normal, Utility, Aerobatic, and Commuter Category Aeroplanes*

(2) CS-25 *Certification Specifications and Acceptable Means of Compliance for Large Aeroplanes*

(3) CS-27 *Certification Specifications and Acceptable Means of Compliance for Small Rotorcraft*

(4) CS-29 *Certification Specifications and Acceptable Means of Compliance for Large Rotorcraft*

(5) CS-E *Certification Specifications and Acceptable Means of Compliance for Engines* and AMC 20-3B *Certification of Engines Equipped with Electronic Engine Control Systems*

(6) CS-P *Certification Specifications for Propellers* and AMC 20-1A *Certification of Aircraft Propulsion Systems Equipped with Electronic Control Systems*

(7) CS-ETSO *Certification Specifications for European Technical Standard Orders*

(8) CS-APU *Certification Specifications for Auxiliary Power Units* and AMC 20-2B *Certification of Essential APU Equipped with Electronic Controls*

(b) EASA Acceptable Means of Compliance (AMC)

(1) AMC 20-115( ) *Airborne Software Development Assurance Using EUROCAE ED-12 and RTCA DO-178*

(2) AMC 20-152( ) *Development Assurance for Airborne Electronic Hardware (AEH)*

(c) FAA Advisory Circulars (ACs)

(1) AC 20-115 *Airborne Software Development Assurance Using EUROCAE ED-12( ) and RTCA DO-178( )*

(2) AC 20-152 *Development Assurance for Airborne Electronic Hardware (AEH)*

(3) AC 00-72 *Best Practices for Airborne Electronic Hardware Design Assurance Using EUROCAE ED-80( ) and RTCA DO-254( )*

(4) AC 20-170 *Integrated Modular Avionics Development, Verification, Integration and Approval using RTCA DO-297 and Technical Standard Order C-153*

(5) AC 27-1309 *Equipment, Systems, and Installations (included in AC 27-1, Certification of Normal Category Rotorcraft)*

(6) AC 29-1309 *Equipment, Systems, and Installations (included in AC 29-2, Certification of Transport Category Rotorcraft*)

(d) Industry Documents

(1) EUROCAE ED-12 *Software Considerations in Airborne Systems and Equipment Certification*, dated May 1982 (no longer in print)

(2) EUROCAE ED-12A *Software Considerations in Airborne Systems and Equipment Certification*, dated October 1985 (no longer in print)

(3) EUROCAE ED-12B *Software Considerations in Airborne Systems and Equipment Certification*, dated December 1992

(4) EUROCAE ED-12C *Software Considerations in Airborne Systems and Equipment Certification*, dated January 2012

(5) EUROCAE ED-79A *Guidelines for Development of Civil Aircraft and Systems*, dated December 2010

(6) EUROCAE ED-80 *Design Assurance Guidance for Airborne Electronic Hardware*, dated April 2000

(7) EUROCAE ED-94C *Supporting Information for ED-12C and ED-109A*, dated January 2012

(8) EUROCAE ED-124 *Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations,* dated June 2007

(9) EUROCAE ED-215 *Software Tool Qualification Considerations*, dated January 2012

(10) EUROCAE ED-216 *Formal Methods Supplement to ED-12C and ED-109A,* dated January 2012

(11) EUROCAE ED-217 *Object-Oriented Technology and Related Techniques Supplement to ED-12C and ED-109A*, dated January 2012

(12) EUROCAE ED-218 *Model-Based Development and Verification Supplement to ED-12C and ED-109A*, dated January 2012

(13) SAE International Aerospace Recommended Practice (ARP) 4754A *Guidelines for Development of Civil Aircraft and Systems*, dated 21 December 2010

(14) RTCA DO-178 *Software Considerations in Airborne Systems and Equipment Certification,* dated January 1982 (no longer in print)

(15) RTCA DO-178A *Software Considerations in Airborne Systems and Equipment Certification*, dated March 1985 (no longer in print)

(16) RTCA DO-178B *Software Considerations in Airborne Systems and Equipment Certification*, dated 1 December 1992

(17) RTCA DO-178C *Software Considerations in Airborne Systems and Equipment Certification*, dated 13 December 2011

(18) RTCA DO-248C *Supporting Information for DO-178C and DO-278A*, dated 13 December 2011

(19) RTCA DO-254 *Design Assurance Guidance for Airborne Electronic Hardware*, dated 19 April 2000

(20) RTCA DO-297 *Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations*, dated 8 November 2005

(21) RTCA DO-330 *Software Tool Qualification Considerations*, dated 13 December 2011

(22) RTCA DO-331 *Model-Based Development and Verification Supplement to DO-178C and DO-278A*, dated 13 December 2011

(23) RTCA DO-332 *Object-Oriented Technology and Related Techniques Supplement to DO-178C and DO-278A*, dated 13 December 2011

(24) RTCA DO-333 *Formal Methods Supplement to DO-178C and DO-278A*, dated 13 December 2011