Certification Authorities Software Team (CAST)

Position Paper
CAST-32A

Multi-core Processors

COMPLETED November 2016 (Rev 0)

NOTE: **This position paper has been coordinated among representatives from certification authorities in North and South America, Europe, and Asia. However, it does not constitute official policy or guidance from any of the authorities. This document is provided for educational and informational purposes only and should be discussed with the appropriate certification authority when considering for actual projects.**

Table of Contents

1. PURPOSE

The purpose of this CAST paper is to identify topics that could impact the safety, performance and integrity of a software airborne system executing on Multi-Core Processors (MCP). For each topic, the paper provides a rationale that explains why these topics are of concern and proposes objectives to address the concern.

Note: This CAST paper is a means to communicate approaches accepted by certification authorities in projects using MCP technology. Although this paper is formulated around a set of objectives, this paper does not constitute or replace certification material applicable to a project.

2. BACKGROUND:

MCPs were designed to provide a substantial increase in performance over traditional single-core processors (SCPs). MCPs also contain other embedded functions such as network interfaces, embedded security, memory management, etc., which could reduce the chip count for a system implementation. Having several cores integrated onto one device could allow more functions to be integrated together on one processor and in one piece of equipment. Aerospace equipment suppliers are therefore interested in using MCPs in their systems. An additional reason for using MCPs is that many SCPs are likely to become obsolete.

MCPs can execute several software applications at the same time because they have two or more processing cores that can each host and execute software applications. Several applications may therefore attempt to access the same shared resources of the MCP (such as memory, cache and external interfaces) at the same time, causing contention for those resources.

Most MCPs have internal mechanisms such as “interconnects” to handle and arbitrate the demands for MCP resources, but the contention for shared resources between applications usually causes delays in access to the resources. These delays are a form of time interference between applications, which can cause applications to take much longer to execute than when executing on their own. There could also be functional interference between applications via MCP mechanisms. Interference could also occur due to software components installed on the MCP, such as operating systems or software hypervisors.
Interference between software applications 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.

If an MCP hosts software applications from more than one system, interference could also cause a safety-critical software application from one system to be adversely affected by an application from another system.

3. REFERENCES AND RELATED DOCUMENTS / GUIDANCE AND STANDARDS

a. SAE ARP 4754A / EUROCAE ED-79A, Guidelines for Development of Civil Aircraft and Systems

b. AC 20-174, Development of Civil Aircraft and Systems

c. SAE ARP 4761 / EUROCAE ED-135, Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment

d. RTCA DO-178B / EUROCAE ED-12B, Software Considerations in Airborne Systems and Equipment Certification

e. RTCA DO-178C / EUROCAE ED-12C, Software Considerations in Airborne Systems and Equipment Certification

f. AC 20-115C – Airborne Software Assurance

g. AMC 20-115C - Software Considerations for Certification of Airborne Systems and Equipment

h. RTCA DO-254 / EUROCAE ED-80, Design Assurance Guidance for AEH

i. Advisory Circular (AC) 20-152 / RTCA DO-254, Design Assurance Guidance for AEH

j. EASA Certification Memorandum CM - SWCEH – 001, Development Assurance of Airborne Electronic Hardware

4. DEFINITIONS

NOTE: **This position paper has been coordinated among representatives from certification authorities in North and South America, Europe, and Asia. However, it does not constitute official policy or guidance from any of the authorities. This document is provided for educational and informational purposes only and should be discussed with the appropriate certification authority when considering for actual projects.**

Applicable AEH Guidance: ED-80 / DO-254 plus any certification authority AEH guidance applicable to the project

Applicable Software Guidance: the version of ED-12 / DO-178 that applies to the project, plus any certification authority software guidance applicable to the project.

Asymmetric Multi-processing (AMP): an MCP software architecture in which each individual functional process is permanently allocated to a separate 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 extends the SMP architecture by allowing the developer to bind any process and all of its associated threads to a specific core while using a common operating system across all cores.

Critical Configuration Settings: those configuration settings that the applicant has determined to be necessary for the deterministic execution of the software or any settings that, if inadvertently altered, could change the behavior of the processor so as to cause the hosted software to no longer comply with its requirements. (See objectives MCP_Resource_Usage_1 and MCP_Resource_Usage_2.).

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. (From DO-297/ED-124).

MCP Platform With Robust Partitioning: an MCP platform that complies with the objectives of this document and provides Robust Resource and Time Partitioning as defined in this document, not only between software applications hosted on the same core, but also between applications hosted on different cores of an MCP or between applications that have threads hosted on several cores.

Multi-core processor (MCP): a device that contains two or more processing cores. A core in the MCP is defined as a device that executes software. This includes virtual cores (e.g. Intel's Hyperthreading microarchitecture).

Robust Partitioning: both Robust Resource Partitioning and Robust Time Partitioning.

Robust Resource Partitioning (adapted from DO-248C / ED-94C and DO-297 / ED-124) 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 allocations of shared resources.
- 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 DAL as the highest DAL of the software that it partitions.

Robust Time Partitioning (on an MCP) 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 all of the other active cores.

Safety Net: A safety net is defined as the employment of mitigations and protections at the appropriate level of aircraft and system design to help ensure continuous safe flight and landing. The safety net methodology focuses on the assumption that a microprocessor will misbehave.

The safety net approach is a means to mitigate the risks associated with COTS microprocessors via both passive and active methods designed into aircraft systems. This approach requires the safety net to be designed as a function within the aircraft system. The safety net can include passive monitoring functions, active fault avoidance functions, and control functions for recovery of system operations. System architecture and control and recovery functions should be designed to facilitate effective system recovery from anomalous events.
(From DOT/FAA/AR-11/5, Microprocessor Evaluations for Safety-Critical, Real-Time Applications, May 2011.)

Symmetric Multi-processing (SMP): an MCP software architecture in which a single operating system controls the execution of the processes on all the cores and may dynamically allocate sections of processes to run in parallel on separate cores.

5. POSITION

5.1 Applicability to Processors.

This position paper applies to multi-core processors (MCPs) when

- two or more cores are activated

- the IDAL of the hosted software or of the MCP hardware device is A, B or C, and
- the application is not one of the excluded types below (see 5.2.3 i and 5.2.3 ii).

Section 7 of this document describes the objectives that apply according to the assigned IDAL (A, B or C) of the hosted software and the MCP hardware device. (For the definition of IDAL, see ARP4754A / ED-79A).

The Paper does not apply when the MCP software and hardware IDALs are all level D or E.

Applicants whose equipment uses only one activated core of an MCP should comply with the Applicable Software and AEH Guidance (in particular, regarding assurance that the deactivated core(s) do(es) not interfere with the activated core or the software hosted on it).

If an applicant or its supplier using an MCP with only one activated core later decides to install software on a core that was previously deactivated, then the applicant should provide a new set of software and AEH documentation for the equipment in which any additional core is activated, and that equipment will have to comply with this paper if it fulfills the applicability criteria of this document.

5.2 Aspects Not Covered by This Paper

5.2.1 Dynamic Allocation of Software Applications

This paper does not cover MCP platforms on which software applications or threads can be dynamically re-allocated to a different core (or different cores) by the operating system, a software hypervisor or by other means. Allocations of applications to cores may be carried out on start-up of the MCP software, but not during the subsequent operation of the software. Applicants wishing to use dynamic allocation of software applications or threads should contact the certification authority (CA).

5.2.2 Hyperthreading

Applications that enable Hyperthreading on processors with Hyperthreading features are not covered by this paper. Applicants should inform the CA if they intend to use Hyperthreading.

5.3 Processors Exempted from this Paper

Some types of MCP with two or three activated cores are considered to be similar enough in their architectures to existing combinations of single core processors that they do not need to be covered by the additional guidance material of this document. Two types of processors that do not need to be covered by this guidance material are as follows:

i) Lock-step processors with two or more activated cores in which the cores host the same software and execute that same software in lock-step so that their outputs, based on identical input data, can be compared for use in a safety-critical application. (An additional core is sometimes provided for input/output.) These lock-step processors are designed for safety-critical applications and to provide the determinism required, rather than the fast calculations and fast data transfers needed in servers or mobile devices, for which most MCPs are designed. The architectures of lock-step devices do not, therefore, contain features such as shared memory and shared cache that could cause interference. If interference did occur and caused one of the cores to produce a different result from the other(s) or to be delayed in its computations by time interference, these processors are designed to detect differences between the results produced by the cores, so any interference would be detected. The system could then be made safe, or could continue to be available if three cores are used with a voting mechanism. For these reasons, the CAs do not consider that this Paper needs to apply to lock-step processors that operate in the manner described above.

ii) Processors in which two or more identical or different activated cores are incorporated onto the same device, but the activated cores are only linked by conventional databuses, and not by shared memory, shared cache, a ‘coherency fabric / module / interconnect’ or a software hypervisor or any of the other features of MCPs that are mentioned in this document. This category of MCPs includes any MCPs in which a core acting as a co-processor or a graphics processor is under the control of another core that executes software, provided that the cores are not connected by any of the MCP mechanisms mentioned in this document. If the MCP contains any cores in addition to the activated cores, it must be ensured that those other cores are deactivated and that they cannot cause any interference with the activated cores or with the software that executes upon the activated cores.

6. <u>SPECIFIC MCP TOPICS TO ADDRESS</u>

The text in each of the following sub-paragraphs is organized into two sections.

Section a. of each sub-paragraph provides a rationale, which explains the reasons why it is necessary to address the particular topic covered by that sub-paragraph.

Section b. of each sub-paragraph states the objectives that applicants using MCPs should meet in order to address the concerns stated in the Rationale section.

The applicant should meet the objectives of this Paper, with the exception of any objective or part of an objective that the applicant justifies as not being applicable to their MCP, (e.g. if the MCP mechanism addressed does not exist on the selected MCP).

If an objective requests the applicant to provide data on some aspect of their MCP or its hosted software that does not apply in their case, then the applicant should state in the appropriate deliverable document that the particular aspect does not apply and should briefly explain why it does 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.

6.1 Software Planning

a. Rationale

DO-178B / ED-12B / and DO-178C / ED-12C were written before MCPs were used in civil aircraft, so those documents only address the planning, development and verification of software hosted on single-core processors that execute software sequentially and not multi-core processors with multiple cores that execute software concurrently. They do not request applicants to provide any planning data specifically related to software hosted on MCPs and neither does any existing software guidance document. The additional planning objective below clarifies the information needed in the applicable plans to achieve planning data standardization for projects with an MCP.

If an applicant intends their MCP to be used as part of an IMA platform to host applications from more than one system, then it is important for the applicant to clearly state this in their plans or other deliverable documents.

Some operating systems (such as SMP OSs) are capable of dynamically re-allocating software applications to different cores during the operation of the system. It is important for applicants to state at the planning stage whether any such dynamic features of hosted software will be activated. Applicants should note that this Paper does not cover MCP platforms that enable the dynamic allocation of software applications during operation.

b. Objective

MCP_Planning_1: The applicant's software plans or other deliverable documents:

1) Identify the specific MCP processor, 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 device will be used in an 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 document.
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 document and comply with 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 AMP, 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 applications to cores and any other software dynamic features that can affect the execution of the software while it is executing.

6.2 The Planning and Setting of MCP Resources

a. Rationale.

MCPs are highly-complex systems-on-chip that provide many resources, including processing cores, memory devices, peripheral devices, interconnects, and others. Users can usually tailor the resources of their MCP to the requirements of their equipment by selecting aspects such as

- which cores are activated,

- the execution frequencies of the cores,

- which of the peripheral devices of the MCP are activated,

- whether shared memory or shared cache are used and how they are 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 a core in order to save energy. (This might not be desirable for a core hosting a safety-critical application.)

These options are known as the “configuration settings” of the MCP and may be set by connecting certain pins of the MCP to particular voltages or making the hosted software write values to particular registers at the start-up of the software.

Some of these settings, if inadvertently altered during operation of the MCP by Single Event Effects or other conditions, could cause the MCP and its software to behave differently and to no longer comply with their requirements if means to mitigate any inadvertent changes to these “Critical Configuration Settings” are not incorporated.

Using shared memory may cause increased memory access times, and using shared cache may cause repeated cache misses if one application prevents the others from accessing the shared cache. These effects can cause increases in the Worst-Case Execution Times (WCETs) of the hosted applications. The CAs are concerned that such increases in WCET could prevent some applications from having sufficient time to complete the execution of their safety-critical functionality.

Sharing memory may also result in the software hosted on one of the cores being locked out from accessing the data it requires, causing memory access to be delayed and perhaps even causing applications to halt.

If the limited resources of an MCP are not carefully allocated, the demands for access to those resources could be greater than the MCP can provide, so some requested transactions would fail to occur and some applications would be denied the data or the peripheral access that they require.

At the planning stage, it is important to identify which shared resources the applicant intends to use and how the applicant intends to allocate those resources and verify the use of the allocations so as to either avoid contention for these resources or mitigate the resulting problems such as those described above.

b. Objectives.

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.

MCP_Resource_Usage_2: The applicant has planned, developed, documented, and verified a means that ensures that in the event of any of the Critical Configuration Settings of the MCP being inadvertently altered, an appropriate means of mitigation is specified.

MCP_Planning_2: The applicant's software / AEH 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 (*) 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 any Hardware dynamic features of the MCP device that will be active and how they will be used.

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) or shared memory (taking into account the time interference and the software execution problems it may cause such as lockouts, race conditions, data starvation, deadlocks or live-locks, excessive data latency).

b) Dynamic features of the MCP device include any features that can alter the behavior of the MCP or the hosted software during execution, for example, energy-saving features (clock enable/ gating, frequency adaptations, deactivating one or more cores, dynamic control of peripheral access).

6.3 Interference Channels and Resource Usage

a. Rationale

As stated above, applications that execute on different cores of a multi-core processor share MCP resources, so even if there is no explicit data or control flow between these applications, coupling exists on the platform level, which can cause interference between the applications. A platform property that may cause interference between independent applications is called an interference channel.

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 paper. Non-deterministic behavior of the hosted software may occur due to such interference.

It is therefore important to identify the interference channels that could cause interference between the software applications hosted by their MCP platform, to mitigate the effects of each of those interference channels and to verify the selected means of mitigation.

When the MCP user has defined the set of resources available in the MCP by implementing the configuration settings described in section 6.2 above and has installed all the hosted software, the MCP will then be in its “intended final configuration”. It is important that in that intended final configuration, the demands for MCP resources do not exceed the amount of resources available and that their allocations of resources are not exceeded.

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 channeled and the demands are arbitrated. This arbitration can cause interference problems between applications such as jitter on data arrival times or it can change the order in which transactions requested by applications are executed. It is important to identify the effects that the interconnect of their MCP could cause for the applications hosted on the MCP and mitigate them if necessary.

If the demands for interconnect transactions are very high, e.g. in MCPs with a very high level of external databus traffic, the interconnect can become overloaded, which can affect transactions on some MCPs. It is important to ensure that such an overload of the interconnect does not occur or that is mitigated if it does occur.

b. Objectives.

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 I/O, 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 IDAL 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 need to use Worst Case scenarios is implicit in this objective.

6.4 Software Verification

a. 1. Rationale

DO-178B / ED-12B and DO-178C / ED-12C were written before MCPs were used in civil aircraft, so those documents only address the planning, development and verification of software hosted on single-core processors that execute software sequentially and not MCPs with multiple cores that execute software in parallel. Those documents do not provide any specific guidance about the verification of software on a processor with more than one core. They do not require applicants to show that all the software hosted on several cores of an MCP will function correctly and will have sufficient time to execute when all the software is executing at the same time.

As stated earlier, the Worst Case Execution Time (WCET) of an application can increase greatly when other applications are executing in parallel on the other cores of an MCP. This could cause some applications to have insufficient time to complete the execution of their safety-critical functionality. (With single-core processors, there always has to be a margin between the WCET of an application and the available processing time.)

The software verification processes in the existing software guidance therefore need to be adapted for use on an MCP so as to demonstrate that the hosted 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.

Separate verification of an application (i.e. without any other applications on other cores executing at the same time) may be valid under certain conditions if the applicant can demonstrate that the interference between that application and all the other applications hosted on the MCP is mitigated.

Separate determination of the WCET of an application without any other applications executing is only valid if the applicant can demonstrate that they have an MCP Platform with Robust Partitioning or that time interference from other applications is avoided or mitigated for that application.

Data from the interference analysis conducted in order to meet objective MCP_Resource_Usage_3 of this Paper may show that for some software components or set of requirements within a software component, interference is mitigated. Objective MCP_Software_1 below states that separate verification may be conducted in those situations.

Since interference between applications occurs via the proprietary internal mechanisms of an MCP, any simulation of those mechanisms is less likely to be representative in terms of functionality or execution time than testing conducted on the target MCP.

As the Applicable Software Guidance was written to address single core processors, the data and control coupling objectives in that guidance only apply between the software components hosted on a single core processor. With an MCP, there may be data and control flows between software components hosted on different cores of the MCP, so the data and control coupling analysis should be extended to cover those cross-core data and control flows.

a. 2. Categories of MCP Platforms.

The "MCP platform" onto which software applications are loaded consists of the MCP device itself and the platform software, such as an operating system and/or software hypervisor, which provides the interface between the software applications and the MCP device.

In order to adapt the software verification guidance for different types of MCP platform, the two following categories of MCP platforms are considered:

- MCP Platforms With Robust Partitioning,
- All Other MCP Platforms.

b. Objectives

MCP_Software_1: The applicant has verified that all the software components hosted by the MCP comply with 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 is executing in the intended final configuration.

The way in which the applicant should demonstrate compliance with 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 and Time Partitioning (as defined in this document) may verify 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 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 standard DO-178C verification activities.

c) If the highest IDAL of the MCP hardware and of all the software applications hosted on the MCP is IDAL 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 software executing at the same time on other cores of the MCP.

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 applications via shared memory and any mechanisms to control the access to shared memory, and that the data and control coupling is correct.

NOTE : 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 components hosted on different cores.

6.5 Error Detection and Handling, and Safety Nets

a. Rationale

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 in parallel in real-time.

The features of an MCP may therefore contain unintended functionality that may cause errors and produce unexpected behavior. Additional mechanisms would therefore need to be developed and verified to detect and handle the specific errors associated with these features.

Applicants may therefore wish to consider the use of a "safety net" external to 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.

b. Objective.

MCP_Error_Handling_1: The applicant has identified the effects of failures that may occur within the MCP and has planned, designed, implemented and verified means (which may include a 'safety net' external to the MCP) 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.

6.6 Reporting of Compliance with the Objectives of this Document

a. Rationale

The existing software and AEH guidance documents request applicants to provide data on specific topics in their Software and Hardware Accomplishment Summaries, but since those guidance documents were written before MCPs were used in civil aircraft and before this Paper was written, the existing guidance does not request applicants to provide any data related to compliance with the objectives of this document.

b. Objective.

MCP_Accomplishment_Summary_1: In addition to providing in their SAS and HAS the information requested by the existing guidance, the applicant has summarized in their SAS, HAS or other deliverable documentation how they have met each of the objectives of this document.

7. APPLICABILITY OF MCP PAPER OBJECTIVES ACCORDING TO DAL

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

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

MCP PAPER OBJECTIVES	**DAL A or B**	**DAL C**
MCP_Planning_1: The applicant's software plans or other deliverable documents: 1) Identify the specific MCP processor, 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	**Yes**	**Yes**

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 device will be used in an IMA platform to host software applications from more than one system, 6) Identify whether or not the MCP Platform will provide Robust Resource and / or Time Partitioning as defined in this document, 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 document and comply with 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 AMP, 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 MCP used 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 applications to cores and any other software dynamic features that can affect the execution of the software while it is executing.*		
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.	**Yes**	**Yes**
MCP_Resource_Usage_2: The applicant has planned, developed, documented, and verified a means that ensures that in the event of any of the critical configuration settings of the MCP being inadvertently altered, an appropriate means of mitigation is specified.	**Yes**	**No**
MCP_Planning_2: The applicant's software / AEH 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	**Yes**	**Yes**

resources (*) 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 any Hardware dynamic features of the MCP device that will be active and how they will be used. *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) or shared memory (taking into account the time interference and the software execution problems it may cause such as lockouts, race conditions, data starvation, deadlocks or live-locks, excessive data latency).* *b) Dynamic features of the MCP device include any features that can alter the behavior of the MCP or the hosted software during execution, for example, energy-saving features (clock enable/ gating, frequency adaptations, deactivating one or more cores, dynamic control of peripheral access).*		
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 I/O, 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 IDAL 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*	**Yes**	**No** **See note d)**

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 need to use Worst Case scenarios is implicit in this objective.*	**Yes**	**No**
MCP_Software_1: The applicant has verified that all the software components hosted by the MCP comply with 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 is executing in the intended final configuration. The way in which the applicant should demonstrate compliance with 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 and Time Partitioning (as defined in this document) may verify 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 software components executing in the intended final configuration. *NOTES:*	**Yes**	**Yes**

***NOTE:* This position paper has been coordinated among representatives from certification authorities in North and South America, Europe, and Asia. However, <u>it does not constitute official policy or guidance from any of the authorities</u>. This document is provided for educational and informational purposes only and should be discussed with the appropriate certification authority when considering for actual projects.**

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 standard DO-178C verification activities.* c) *If the highest IDAL of the MCP hardware and of all the software applications hosted on the MCP is IDAL 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 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 software executing at the same time on other cores of the MCP.*		
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 applications via shared memory and any mechanisms to control the access to shared memory, and that the data and control coupling is correct. *NOTE: 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 components hosted on different cores.*	**Yes**	**Yes**
MCP_Error_Handling_1: The applicant has identified the effects of failures that may occur within the MCP and has planned, designed, implemented and verified means (which may include a 'safety net' external to the MCP) 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.	**Yes**	**No**
MCP_Accomplishment_Summary_1: In addition to providing in their SAS and HAS the information requested by the existing guidance, the applicant has summarized in their SAS, HAS or other deliverable documentation how they have met each of the objectives of this document.	**Yes**	**Yes**

8. APPENDIX

Note to the reader - Backward Compatibility Table.

Given the significant change between the previous revision and the current revision of the document, the table below has been added in order to provide the relationship between the objectives of this document and those of the previous version.

Current Objectives	***MAPPING TO FORMER CRI / IP / CAST 32 OBJECTIVES***
MCP_Planning_1	***MCP_Software_1, MCP_Software_2, MCP_Determinism_1, MCP_Determinism_4, MCP_Determinism_5.***
MCP_Planning_2	***MCP_Determinism_8, MCP_Determinism_10, MCP_Determinism_12.***
MCP_Resource_Usage_1	***MCP_Determinism_1***
MCP_Resource_Usage_2	***MCP_Determinism_2***
MCP_Resource_Usage_3	***MCP_Determinism_7, MCP_Determinism_9, MCP_Determinism_11, MCP_Determinism_15, MCP_Determinism_16***
MCP_Resource_Usage_4	***MCP_Determinism_13, MCP_Determinism_14***
MCP_Software_1	***MCP_Software_3, MCP_Software_4, MCP_Software_6, MCP_Determinism_5***
MCP_Software_2	***MCP_Software_5***
MCP_Error_Handling_1	***MCP_Error_Handling_1***
MCP_Accomplishment_Summary_1	***No trace***
Covered by System Safety process	***MCP_Error_Handling_2***
Covered by applicable EASA AEH Guidance	***MCP_Determinism_3***
Deleted	***MCP_Determinism_6***

NOTE: **This position paper has been coordinated among representatives from certification authorities in North and South America, Europe, and Asia. However, <u>it does not constitute official policy or guidance from any of the authorities</u>. This document is provided for educational and informational purposes only and should be discussed with the appropriate certification authority when considering for actual projects.**