Update on using multicore processors with a commercial ARINC 653 implementation

Paul J. Parkinson
Principal Systems Architect
Wind River
Swindon, United Kingdom
Paul.Parkinson@windriver.com

***Abstract*— With the wide availability of multiple core (multi-core) processors, their reduced size, weight and power (SWaP) properties make them extremely attractive for use in Avionics systems. In order to implement a solution on a multi-core platform, the developer will be confronted with numerous implementation and certification obstacles that are not present in uni-core or discrete multiple processor implementations. Achieving safety certification of a multi-core system requires close collaboration between the avionics developers, semiconductor vendors and regulatory agencies. Evolving certification policies and guidance will include both hardware and software aspects of certification. This paper will provide an update of work by Wind River on implementing a COTS ARINC 653 solution for multi-core and provide guidance to the developer on the issues that must be addressed from both a hardware and software perspective in order to understand the potential benefits and certification limitations of multi-core solutions.**

Keywords—avionics; certification; multicore;

I. The Challenge of Multi-core Certification

The introduction of MCP architectures has provided performance gains for enterprise general purpose applications; it has also presented some unique challenges for their use in safety-critical avionics systems. This is because avionics applications have specific requirements, including (but not limited to) application isolation and determinism, and these are not the primary considerations of semiconductor manufacturers when designing MCPs for the commercial market.

The avionics industry, academia and certification authorities have undertaken research projects into the use of MCP architectures in avionics applications. A number of researchers have found that there is variation between MCP designs in terms of their suitability for use in avionics applications, due to the impact of architectural design features on application isolation and determinism [1][2]. These relate to factors arising from shared resources on the device, which include use of a single memory controller or shared bus is used by multiple cores (providing a risk of resource contention), and similarly use of separate or shared Level 2 caches per core, as shown in Figure 1.

This uncertainty about the selection of multi-core processors for avionics programmes presents a challenge for avionics programmes, and was discussed at length in a previous paper [3].

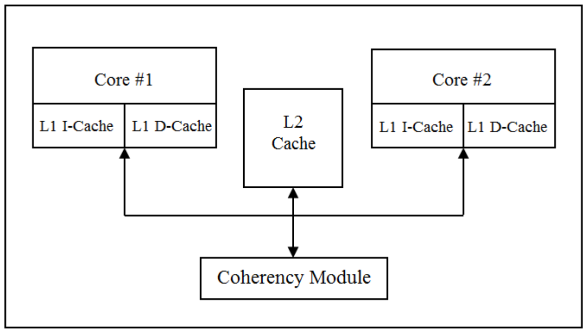

Fig 1. Notional multi-core cache architecture with shared L2 cache

EASA and the FAA have not yet published formal policy / guidance on multi-core certification, which presents a challenge to avionics programmes. However, EASA has published the MULCORS research report [4] and has published the CAST-32 position paper [5], and more recently CAST-32A paper [6], which should be taken into consideration when planning a safety-critical multi-core avionics project in order to reduce certification risk.

Programmes may wish to consider the use of a multi-core processor in their next hardware platform even if their current processing requirements do not exceed that provided by a single core, in order to provide adequate processing capacity to meet future processing requirements. The selection of a multi-core processor may also become a necessity due to the lack of availability of single core processors as mentioned earlier. Similarly, some programmes may wish to use multi-core processors which have more than two cores, as 4-core and 8-core devices are now relatively common. The initial CAST-32 paper did not consider multi-core processors with more than two active cores, although the more recent CAST-32A extends the scope to more than two cores. However, certifying multi-core processors will require substantial research and certification leadership to extend the guidance in the MULCORS and CAST-32 papers.

A. *Core Deactivation*

In both of the above scenarios, programmes will need to be able to utilise certain processor cores and deactivate the unused cores, as discussed in a previouspaper [3]. The ability of safety-critical avionics programmes to be able to deactivate individual cores and develop a safety-case which includes robust arguments for the deterministic operation of the processor may depend on the ability to obtain detailed technical information on the design and operation of the processor from the semiconductor manufacturer. Some semiconductor companies may make this information publicly available, while others may only provide certain levels of information under non-disclosure agreement. For programmes undertaking DO-254 [7] certification of airborne electronic hardware (AEH) this is an important requirement to achieve certification, and they will need to ensure that the selected semiconductor manufacturer will provide access to the required information, even if they do not formally support DO-254 certification.

B. *Multi-core Interference*

Programmes intending to use MCPs in safety-critical avionics applications will need to manage contention between cores for shared resources. In particular, consideration should be given as to whether potential interference paths will result in actual interference channels [5] [8].

Wind River has undertaken research to measure inter-core perturbation on the QorIQ P4080 processor due to shared cache(s) and share memory controller(s) [9]. This included the development of a benchmark suite which continuously passed through a code loop of configurable size, to access consecutive data in a data set of configurable size. The intent was to generate continuous cache misses on both the instruction and data caches.

The benchmark was run on a Wind River SBC P4080 reference board which contained a QorIQ P4080 processor with 64KB L1 cache per core (32KB for instructions, 32KB for data); 128K L2 cache per core (64KB for instructions, 64KB for data); a 1024KB L3 cache per memory controller, and two memory controllers. When the benchmark was run in a partition in virtualized environment (known as a virtual board or VB, or virtual machine), on a single core of the P4080, the performance degraded in predictable manner as the data size increased (as shown in Figure 2).

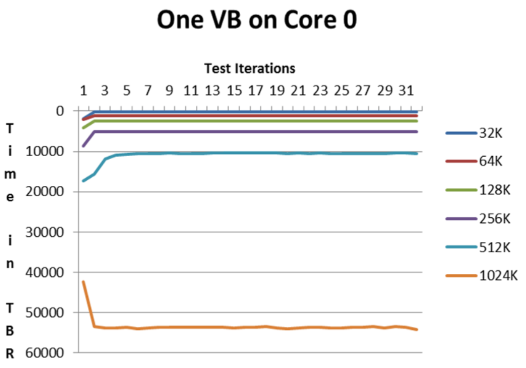

Fig. 2. Benchmarks results for P4080 single core operation.

The vertical axis indicates time in ticks measured by the PowerPC 64-bit Time Base Register (TBR), and the horizontal axis indicates the number of iterations of the benchmark that were performed. The lines of the graph are shown in order of increasing size of benchmark, and show that when the benchmark is small, the code and data are contained within the on-processor L1 instruction and data cache. As the benchmark size is increased, the benchmark overflows into shared L2 cache and eventually into the L3 platform cache with predictable degradation in performance.

However, when the benchmark was run simultaneously on two cores on the P4080 using the same memory controller, the results became unpredictable once the data size had overflowed into the L3 cache, shown by the wavy lines for 512KB and 1024KB benchmark sizes in figure 3.

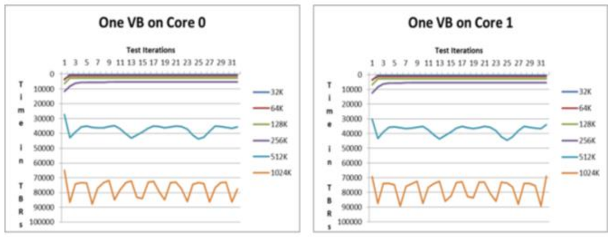

Fig. 3. Benchmarks results for P4080 dual core operation with single memory controller

When the test was re-run simultaneously on two cores on the P4080 but this time using different memory controllers for each core, this resulted in predictable results once more, shown by the 512KB lines in figure 4.

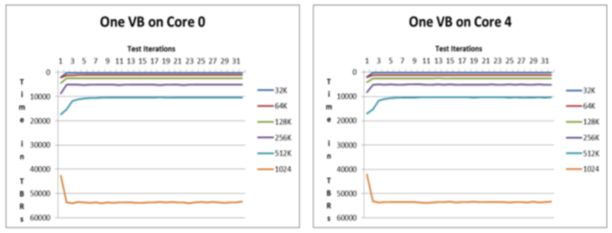

Fig. 4. Benchmarks results for P4080 dual core operation with dual memory controllers

These benchmark results illustrate the potential interference paths for a specific processor architecture, but this does not necessarily indicate the actual interference channels which will occur for an avionics system, as this is dependent on the characteristics of the applications. Therefore multicore interference analysis cannot be performed on the underlying operating system in isolation, but needs to be undertaken at the system-level including the application.

II. CERTIFICATION OF AN ARINC 653 RTOS ON MULTI-CORE PROCESSOR ARCHITECTURE

We will now consider the challenges of avionics software safety certification and multicore in the context of an ARINC 653 RTOS.

A. *ARINC 653 update for multicore*

ARINC 653 is the leading industry open standard for avionics software architecture in an integrated modular avionics environment, with ARINC 653-based systems having being widely deployed in civil and military aircraft. ARINC 653 Part 1 Supplement 3, Required Services (ARINC653P1-3) [10] which was published in 2010 did not address the use of ARINC 653 in multicore processor avionics systems. However, as there was strong market demand for support for multicore, the AEEC APEX Subcommittee undertook the updating of ARINC 653P1-3 to support the use of MCPs. This industry effort involved Tier-1 suppliers, system integrators, and COTS software suppliers, including proactive participation and contribution by Wind River.

The evolution of the standard resulted in the publication of ARINC 653 Part 1 Supplement 4 (ARINC653P1-4) [11] in 2015 to support the use of MCPs, and contains an important provision, stating (page 5) that an application developed under ARINC 653P1-3 to run on a single-core processor, should also run on a single core on a multi-core platform under ARINC 653P1-4 with the same behaviour. This preserves the investment of previously-developed ARINC 653 applications when migrating to multi-core platforms.

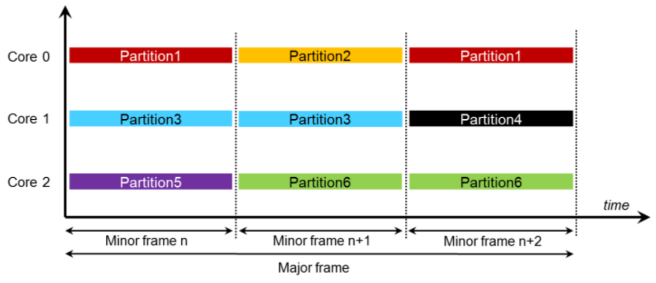

Fig. 5. ARINC 653P1-4 Multicore Scheduling Example

ARINC 653P1-4 also provides ability to support multiple partitions on each processor core using a timeslot scheduling approach where an individual partition will execute on a specific core within a timeslot of defined duration (known as a *minor frame*). An ARINC 653 schedule can be composed of multiple minor frames of similar or differing durations, and the time taken to execute all of the minor frames within the ARINC 653 schedule is known as the *major frame* period. After the execution of the last minor frame has been completed, the ARINC 653 scheduler will then schedule the first minor frame in the schedule again, in a cyclic manner. The ARINC 653P1-4 architecture provides the potential for many potential scheduling configurations (as illustrated by figure 5). However, as discussed earlier, the System Integrator will need to ensure that the configuration of specific applications on a particular IMA platform will provide deterministic behaviour, and that potential interference paths are reduced to the minimum number of interference channels.

ARINC 653P1-4 does not include the ability to run an instance of a partition across multiple cores (known as *a multicore partition*), but states that this capability may be added in a future update of the standard.

B. *VxWorks 653 RTOS Multi-core Requirements*

For the earlier releases of the VxWorks 653 RTOS targeting single-core operation, the requirements were defined in the Software Requirements Specification (SRS) contained in the VxWorks 653 DO-178B [12] Level A certification evidence package for the respective processor architecture. The software architecture of VxWorks 653 2.x RTOS is discussed in [13]. For *VxWorks 653 Multi-core Edition*, the following high-level goals were defined for the product on multi-core architectures:

1. Certifiable to DO-178C DAL A [14].
2. Support multiple DALs on multiple cores
3. Perform fault isolation and containment (health monitors)
4. Perform static configuration and enforcement in accordance with ARINC 653
5. Enable role-based development as per DO-297

These goals were addressed through the product requirements, design implementation and certification strategy using an agile development process in conjunction with DO-178C processes. This enabled the product definition to evolve and to track enhancements to the ARINC 653P1-4 standard.

C. *VxWorks 653 Multi-core Edition RTOS Design Considerations*

In order to achieve the high-level goals of support the safety certification of multiple applications at different DALs up to and including DAL A on multiple cores, the design of VxWorks 653 Multi-core Edition RTOS therefore needed to support isolation of applications running individual partitions through spatial partitioning, temporal partitioning, resource partitioning and multi-core partitioning. The RTOS design also needed to minimise the potential for multicore interference paths where possible.

1) *Hypervisor & Hardware Virtualisation Support*

Microprocessors have typically provided two privilege levels: *User mode*, for user application context, and *Supervisor Mode*, which is usually reserved for use by the operating system kernel. However, an increasing number of modern multicore processors (such as the QorIQ T2080, Intel 64-bit processors and some ARM processors) implement a third privilege level, known as *Hypervisor Mode* [15]. This enables an operating system to be run at hypervisor privilege level, utilising the processor's full hardware virtualisation support to run multiple virtual machines containing unmodified guest operating systems (GOS) and applications.

Wind River took the decision to utilise this capability within the VxWorks 653 Multi-core Edition by implementing the Module Operating System (MOS) at hypervisor level running across the processor's multiple cores, with an unmodified guest OS in virtual machines, as shown in figure 6. A MOS board support package (BSP, not shown in figure 6) is used to map the board-independent functionality of the MOS to the hardware architectures of a specific PowerPC

QorIQ board. This approach enables the MOS to run on different boards by changing the underlying BSP.

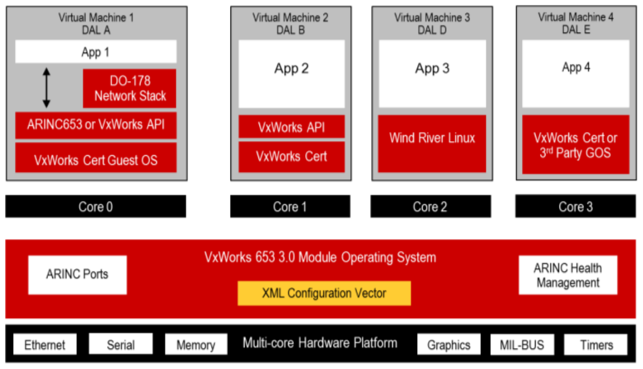

Fig. 6. VxWorks 653 3.0 RTOS Architecture

At the partition-level, Wind River decided to continue to use a partition operating system (POS) approach which had previously been used in VxWorks 653 2.x on single-core processor architectures, with runtime libraries supporting the deployment of applications which use ARINC 653 processors, POSIX APIs, or VxWorks native applications, any of which could be written in C or Ada programming languages. This architecture also enables a GOS running in a partition at *Supervisor level* to have memory protection from the application running at *User Level* within the virtual machine. Full hardware virtualisation enables a physical device to be allocated to specific virtual machines on a processor core, enabling a device driver within the Guest OS in the virtual machine (VM) to efficiently access the device in a controlled manner without significant performance penalty overheard.

VxWorks Cert 6.6.x was used for the POS to enable applications at different Development Assurance Levels (DAL) up to and including DO-178C DAL A [14]. This also facilitates the migration of federated applications developed to run on VxWorks Cert 6.6.x to run in a VxWorks 653 3.x system. The ability to support unmodified GOS also enables consolidation of federated applications develop to run under Linux and third-party / legacy OS on an IMA platform running VxWorks 653.

2) *4.3.2 Two-Level OS Scheduler*

The two-level OS scheduler was also extended to support multi-core architectures with a scheduling model that uses the Partition OS to perform scheduling of contexts (ARINC 653 processes, POSIX threads or VxWorks 653 tasks) without having to perform a system call into the MOS and the associated overhead of processor context switch. This is efficient and very scalable because as the number of partitions increases, the scheduling overhead on the MOS remains constant. This is because the MOS is only responsible for scheduling of partitions at fixed timeslot durations as defined by the ARINC 653 schedule (as shown by the example in figure 6), rather than having to manage an increasing number of processes, threads and tasks across multiple partitions. This approach minimises the potential for performance degradation and partition jitter in a system with a large number of ARINC 653 partitions, and helps to reduce worse-case execution times (WCET). Figure 7 also illustrates how schedules may be defined on each core, and VxWorks 653's ability to support multiple ARINC 653 schedules.

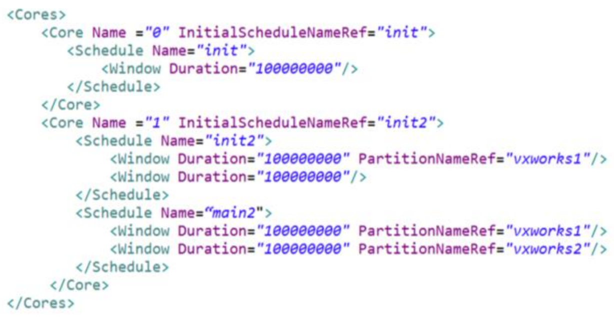
```
<Cores>
    <Core Name ="0" InitialScheduleNameRef="init">
       <Schedule Name="init">
          <Window Duration="100000000"/>
       </Schedule>
    </Core>
    <Core Name ="1" InitialScheduleNameRef="init2">
        <Schedule Name="init2">
           <Window Duration="100000000" PartitionNameRef="vxworks1"/>
           <Window Duration="100000000"/>
        </Schedule>
        <Schedule Name="main2">
           <Window Duration="100000000" PartitionNameRef="vxworks1"/>
           <Window Duration="100000000" PartitionNameRef="vxworks2"/>
        </Schedule>
     </Core>
</Cores>
```

Fig. 7. VxWorks 653 3.0 MOS Schedule XML configuration example

3) *Multi-core Communication*

ARINC 653P1-4 defines inter-partition communications in terms of APplication/EXecutive (APEX) services via *ports*, which provide access at both ends of a communications channel between partitions. Applications access the ports via a pre-defined port name, and ports at both ends of the channel can either be both configured as sampling ports or queuing ports.

In single-core processor systems, the underlying transport mechanism can be implemented by performing memory copy of message buffers between POS and MOS address spaces, and then from MOS to POS for the destination partition. However, extending APEX ports to an MCP system presents some challenges. An application using APEX ports should not be aware of whether the application at the other end of the communications channel is also executing on the same processor core or on a different core, which is known as *location transparency*. This would enable the System Integrator to reconfigure a system, and migrate an application partition from one core to another core, but without impacting the APEX port configuration at the application level.

Wind River achieved this in the implementation of VxWorks 653 Multi-core Edition by maintaining location transparency of ARINC 653 ports at the partition-level, and mapping an APEX port to an underlying *Safe IPC* transport mechanism within the MOS. The mapping of APEX ports to *Safe IPC* channels is defined in XML, enabling the System Integrator to reconfigure the system without requiring individual applications to be modified and recompiled.

4) *Fault Isolation and Containment*

In VxWorks 653 2.x, the MOS ran in the processor's Supervisor Mode to provide isolation from POS applications running in User Mode, and also uses the processor's memory management unit (MMU) to prevent an application from making a programmed I/O access outside its allocated address space (e.g. through de-referencing an invalid pointer).

For the design of VxWorks 653 Multi-core Edition, Wind River decided extend the isolation capabilities to utilise the full hardware virtualisation capabilities available on the QorIQ T2080 processor to verify that DMA transfers to/from a partition are using only valid source and destination address ranges for that partition. This prevents illegal DMA transfers from occurring.

VxWorks 653 Multi-core Edition continues to provide support for ARINC 653 Health Monitoring (HM), by providing the ability to perform cold and warm starts of partitions, and cold restart of the entire module. This provides System Integrators with the ability to configure an ARINC 653 system via an HM framework, enabling a system to provide resilient operation.

5) System Configuration and DO-297 Role-based Development

In VxWorks 653 2.x, system configuration was defined using XML configuration and a process known as independent build link and load (IBLL) to configure and initialise an IMA platform using a single configuration vector (CV). This approach enables platform providers, application developers and system integrators to collaborate according to role separation [16] and also reduce the cost of change, as system and partition configurations can be changed without rebuilding the entire application or platform, which significant reduces the impact-analyses burden when upgrading and modifying an existing system (this is discussed further in [13]). This software architecture continues to be supported in VxWorks 653 Multi-core Edition due to the significant benefits which it has provided.

6) DO-178C DAL A Certification Strategy for VxWorks 653 on QorIQ T2080

Since 2000, Wind River has developed and released DO-178 Certification Pack-ages as certifiable COTS components which organisations could use in their certification programmes. When COTS MCPs started to become widely available, customers started asking Wind River to provide DO-178C Certification Packages on MCP processors, utilising multiple cores. However, as EASA and FAA had not published formal policy on multi-core certification, Wind River regarded undertaking development of a safety-critical multi-core RTOS platform as presenting a significant technical risk with no guarantee of success.

For these reasons, Wind River decided not to start certification until after the publication of the EASA MULCORS research report [4], and FAA CAST-32 position paper [5]. Although these documents do not constitute formal policy at this time, they provide insights into multicore considerations and best practices for multicore certification. Therefore, Wind River developed its Plan for Software Aspects of Certification (PSAC) for VxWorks 653 Multi-core Edition on QorIQ T2080 at DO-178C DAL A, with reference to the FAA CAST-32 paper's objectives in relation to MCP Determinism, MCP Software and MCP Error Handling. Although the CAST-32 paper only addresses two cores, it is largely applicable to more than two cores (as borne out by the changes in CAST-32A), with the caveat that the slow down due to MCP interference, known as the *interference penalty* can grow exponentially as the number of cores increases [8]. This means that system integrators will need to enforce restrictions, known as *interference mitigations*, to reduce interference.

Wind River is working with a lead customer and the FAA on an avionics programme in order to gain early feedback from DO-178C certification audits on the design and certification approach and guidance on application of CAST-32 from the certification authority through the four *Stage of Involvement* (SOI) audits. This approach was regarded as presenting lower technical risk, increasing the probability of successful completion of certification, and in shorter overall time-scales.

D. Future Challenges

Although ARINC 653P1-4 does not currently support multi-core partitions, it indicates that this may be supported in a future update of the standard. This would enable more computationally-intensive applications to be hosted on ARINC653 systems, enabling further consolidation of avionics LRUs onto IMA common computing platforms. ARINC 653 support for multi-core partitions may also increase the potential of using manycore1 processors such as the MPAA® [17] and Tilera (now part of Mellanox) in IMA applications.

At the time of writing, Wind River is in the process of porting the VxWorks 653 Multi-core Edition RTOS to Intel 64 bit multi-core processor architectures (Core, Xeon D). The release dates for the Early Access Release (EAR) and Generally Available (GA) release are published in the current official published Wind River Product Roadmap.

The DO-178C certification of an ARINC 653 RTOS, on other MCP architectures could present different requirements, as other architectures have different initialisation sequences. For example, Intel processors use a BIOS or Intel Firm-ware Support Package [18] [19], which might require optimisation in order to meet the AC2511-B [20] start-up time requirement for an avionics flight display) and undergo DO-178C certification.

Finally, as ARM-based system on chip (SoC) devices increase in processing performance, these may become an attractive option for an IMA platform, especially if DO-254 certification artefacts are provided by the semiconductor manufacturer.

III. Conclusions

The avionics market is currently undergoing a significant transition from single-core to MCP architectures, driven by demands for greater system functionality and the semiconductor product lifecycles which primarily target the much larger commercial market segments. The advances made by semiconductor manufacturers now present a much broader range of viable processor choices for avionics applications than was available in the past.

Although there currently is some uncertainty about the best choice of processor for individual aerospace application use cases, it is likely that positive experiences gained by early adopters on multi-core programmes will result in a virtuous circle of support, further adoption and success, in a similar way to single-core avionics programmes of previous decades generated a rich supplier ecosystem of COTS avionics certification solutions.

The evolution of the ARINC 653 standard to support MCP architectures, combined with the provision of ARINC 653 multi-core software support from COTS RTOS suppliers will enable previously-developed ARINC 653 applications to be re-hosted on MCP IMA platforms, facilitating software reuse and preserving investment. Experience gained from DO-178C certification on multi-core programmes should enable further adoption and proliferation to other MCP architectures.

ACKNOWLEDGMENT

The author wishes to thank the following Wind River colleagues for their input into this paper: A. Wilson, C. Downing, and S. Olsen.

REFERENCES

[1] Kinnan, Larry M., “Use of multicore processors in avionics systems and its potential impact on implementation and certification”, Digital Avionics Systems Conference, October 2009.

[2] “Microprocessor Evaluations for Safety-Critical, Real-Time Applications: Authority for Expenditure No. 43 Phase 5 Report”, DOT/FAA/AR-11/5, US Federal Aviation Administration, 2011. https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/media/11-5.pdf.

[3] Parkinson, Paul J. “The challenge of developing embedded real-time aerospace applications on next-generation multicore procesors”, https://www.researchgate.net/publication/301626435_The_challenges_of_developing_embedded_real-time_aerospace_applications_on_next_generation_multi-core_processors.

[4] Jean, Xavier; Gatti, Mark; Berthon, Guy; Fumey, Marc, “MULCORS - Use of MULticore proCessORS in airborne systems”, Research Project EASA.2011/6, EASA, 8th November 2012. http://easa.europa.eu/system/files/dfu/CCC_12_006898-REV07%20-%20MULCORS%20Final%20Report.pdf

[5] “Multi-core Processors”, Position Paper, Certification Authorities Software Team, CAST-32, FAA, May 2014. https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/cast/cast_papers/media/cast-32.pdf

[6] “Multi-core Processors”, Position Paper, Certification Authorities Software Team, CAST-32A, FAA, Novermber 2016. https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/cast/cast_papers/media/cast-32A.pdf

[7] DO-254 “Design Assurance Guidance for Airborne Electronic Hardware”, RTCA Inc. and ED-80 (EUROCAE), 2002. http://www.rtca.org/store_product.asp?prodid=752.

[8] “White Paper on Issues Associated with Interference Applied to Multicore processors”, US Federal Aviation Administration, 29 January 2016. https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/media/SDS_DO005_White_Paper.pdf.

[9] Kinnan, Larry M., “Multicore in Avionics – Current Practices, Trends and Outlook for Certification”, 32nd Digital Aviation Systems Conference, Syracuse, New York, 2013.

[10] ARINC653P1-3 “Avionics Application Software Standard Interface, Part 1, Required Services”, ARINC Specification 653 Part 1 Supplement 3, , ARINC, 2010.

[11] ARINC653P1-4 “Avionics Application Software Standard Interface, Part 1, Required Services”, ARINC Specification 653 Part 1 Supplement 4 (653P1-4), ARINC, 2015. http://www.aviation-ia.com/cf/store/catalog_detail.cfm?item_id=2510.

[12] DO-178B “Software Considerations in Airborne Systems and Equipment Certification”. RTCA Inc. and ED-12B (EUROCAE), 1992.

[13] Parkinson, Paul J., Kinnan, Larry M., “Safety Critical Software Development for Integrated Modular Avionics”, technical white paper, Wind River, 2015. http://www.windriver.com/whitepapers/safety-critical-software-development-for-integrated-modular-avionics/.

[14] DO-178C “Software Considerations in Airborne Systems and Equipment Certification”. RTCA Inc. and ED-12C (EUROCAE), 2011.

[15] QorIQ T2080 Family Reference Manual, T2080RM Rev 1, NXP, May 2015. https://www.nxp.com/webapp/Download?colCode=T2080RM.

[16] DO-297 ” Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations”, RTCA, 2005. http://www.rtca.org/store_product.asp?prodid=617.

[17] Kalray MPPA® MANYCORE processor, product data sheet, Kalray, 2014. http://www.kalrayinc.com/IMG/pdf/FLYER_MPPA_MANYCORE.pdf.

[18] Wong SH, Sun J, Mahesh D, "Intel® Firmware Support Package for Intel Architecture", white paper, Intel, 2014. http://www.intel.my/content/dam/www/public/us/en/documents/white-papers/fsp-iot-royalty-free-firmware-solution-paper.pdf.

[19] Yao J, Zimmer VJ. Rangarajan R, Ma M, Estrada D, Mudusuru G, "A Tour Beyond BIOS Using the Intel® Firmware Support Package Version 1.1 with the EFI Developer Kit II", Intel, April 2015. https://firmware.intel.com/sites/default/files/resources A_Tour_Beyond_BIOS_Creating_the_Intel_Firmware_Support_Packag e_Version_1_1_with_the_EFI_Developer_Kit_II.pdf.

[20] "Electronic Flight Displays", Advisory Circular AC25-11B, US Federal Aviation Administration, 2014. http://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_25-11B.pdf.