IAC-22-B6.IP.20

Comprehensive High-level Avionics Systems for Exploration

Nathaniel Hargrave[a]*

[a] Nexus Aurora, Canada, nathaniel@nexusaurora.org
* Corresponding Author

Abstract

With a growing demand for cheaper access to space, a new type of avionics system is needed that bridges the gap between larger fault-tolerant systems and smaller systems designed for cubesats. The Comprehensive High-level Avionics Systems for Exploration (CHASE) will fulfill this need with an open-source solution that provides high-reliability avionics in an inexpensive package.

CHASE will provide standards for the development of an ecosystem of inexpensive, reliable avionics, and the development of a processing system following those standards. The CHASE hardware will include radiation tolerance through hardware redundancy, a feature lacking in currently available cost-effective cubesat systems while incorporating a number of features to ease the development of associated hardware and full spacecraft.

The CHASE project includes the development of a comprehensive set of flexible software libraries for common systems. These will allow for a faster development process and standardization between spacecraft. CHASE provides a versatile platform for building a variety of spacecraft, and its open-source nature allows it to be easily expanded for more complex missions.

Keywords: avionics, framework, radiation-tolerance, triple modular redundancy, F prime, standard

Acronyms/Abbreviations

BSP	Board Support Package
C&DH	Command and Data Handling
CAN	Controller Area Network
CHASE	Comprehensive High-level Avionics Systems for Exploration
COTS	Commercial off-the-shelf
ECC	Error Correction Code
ECSS	European Cooperation for Space Standardization
EEPROM	Electrically Eraseable Programmable Read-Only Memory
EPS	Electrical Power System
GN&C	Guidance, Navigation, and Control
GPIO	General Purpose Input/Output
MOSFET	Metal-oxide-semiconductor Field-effect Transistor
MRAM	Magnetoresistive Random Access Memory
OCHRES	Operations Control Hardware for Research and Exploration of Space
PCB	Printed Circuit Board
PCIe	Peripheral Component Interconnect Express
PRU	Programmable Real-time Unit
RGMII	Reduced Gigabit Media-independent Interface
RTEMS	Real-Time Executive for Multiprocessor Systems
RTOS	Real-Time Operating System
SEE	Single Event Effect
SerDes	Serializer/Deserializer
SEU	Single Event Upset
SoC	System on a Chip
SWaP-C	Size, Weight, Power, and Cost
TID	Total Ionizing Dose
TMR	Triple Modular Redundancy

Characteristic	Cubesat components	OCHRES	High-reliability commercial
SWaP-C	~$20,000 for simple processing system[1], small, low-power	<$50,000 for processing system[1], compact, moderate power usage	>$200,000 for advanced components[2], bulky, high power use
Component type	COTS	COTS	Radiation-hardened semiconductors built on special processes
Radiation tolerance	Little to none	High reliability through the use of TMR	High resistance to SEE and TID
Documentation	Existing documentation for COTS components	Complete documentation and examples	Unknown
Software support	Existing support for COTS components	Comprehensive supporting libraries	No existing open-source software support

Table 1: A comparison of key differences between existing commercial systems and CHASE.

1. Introduction

The recent growth of the commercial space industry has brought to light a significant gap in the capabilities of commercially available avionics systems, and the need for systems that are inexpensive, powerful, and reliable. The CHASE project proposes a methodology for developing such systems, and this paper includes the development of a processing module using this methodology. This methodology will foster the growth of an ecosystem of such components, leading to a streamlined experience for end users, and reducing cost and development time.

The central idea of this methodology is to develop a set of guidelines that provide a framework to govern the development of space-worthy avionics modules that are powerful and inexpensive while conforming to industry requirements for reliability. This is accomplished by using Commercial off-the-shelf (COTS) components with hardware and software redundancy to ensure reliability. By following these standards, new projects can quickly and easily meet industry requirements, while creating modules compatible with an existing ecosystem.

The CHASE project aims to develop standardized software libraries for common spacecraft systems and supporting interfaces for common hardware components, leading to a seamless user experience. Sharing hardware and software between multiple missions reduces development and certification time, and the open-source nature of the project creates opportunities for future developers to use pre-certified code for mission development.

The rest of this paper is divided into five sections. Section 2 covers the CHASE project, including key goals of the CHASE methodology, comparison to similar systems, and development of the software library. Section 3 discusses the development of the Operations Control Hardware for Research and Exploration of Space (OCHRES) processing module and explains key design points and hardware choices. Section 4 gives example use cases, Section 5 discusses design elements not considered, and Section 6 concludes.

2. The CHASE project

The CHASE project began with identifying gaps in the avionics market, and defining key goals. Table 1 compares existing commercially available systems to CHASE on a variety of key factors. This comparison guided the CHASE project and development of the OCHRES module.

The goal is for CHASE to fill a gap in the current market, with cubesat systems on one side and expensive radiation-hardened systems on the other. To be a competitive middle option, CHASE needs to be both less expensive and more powerful than radiation-hardened options, and more reliable than inexpensive cubesat systems. Lower cost and higher performance can be achieved by using COTS parts, but this comes at the cost of reliability and the danger of radiation-induced faults. To combat this CHASE uses Triple Modular Redundancy (TMR), which pre-

[1]This value is loosely extrapolated from the high-level design and is subject to change.

[2]There is a lack of publicly available data on radiation-hardened systems, but estimates put the RAD5545 6U-220[2, p. 16] unit at greater than $200,000 per unit. The GR740[3] CPU is reported to be atleast $40,000 per unit.

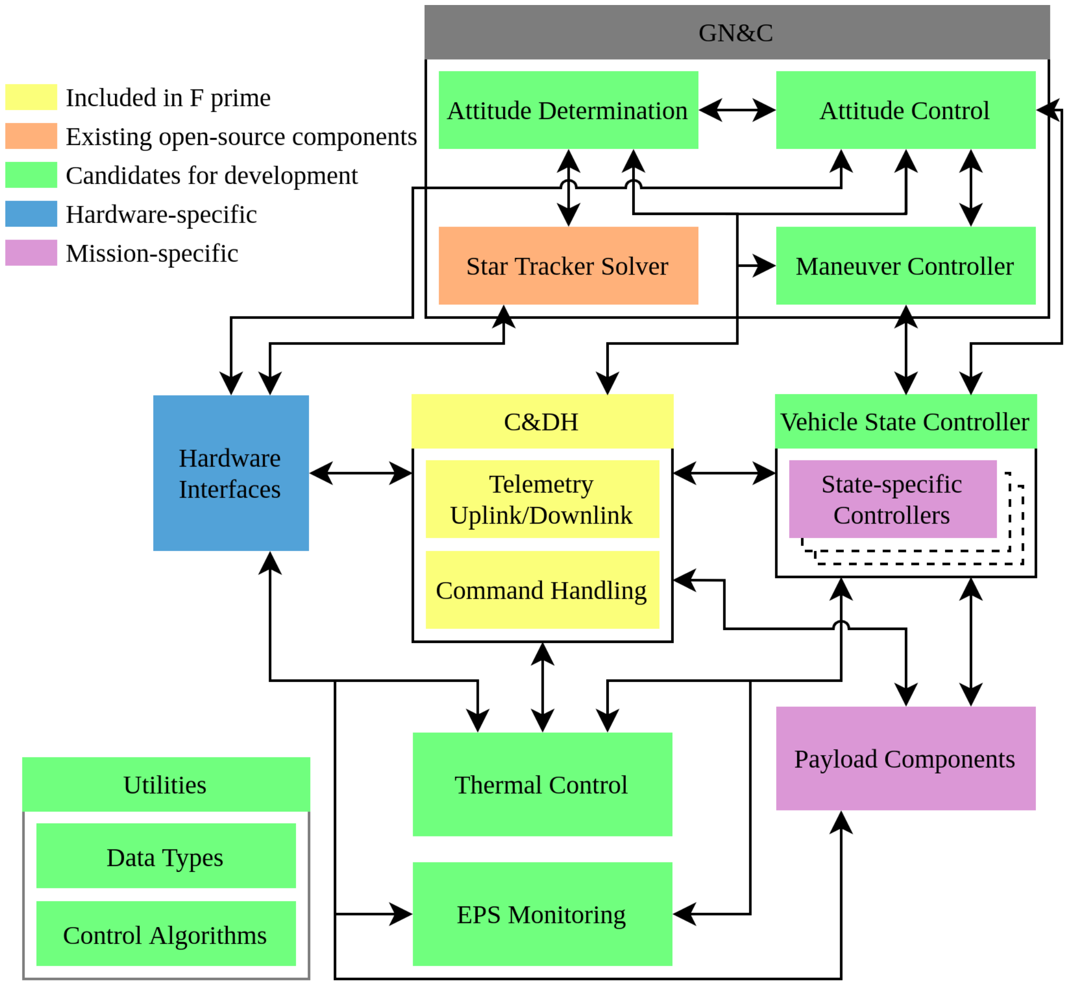

Figure 1: Flight software components

vents faults in one processor from taking down the entire system.

To present a compelling use case, the CHASE project also includes the development of open-source flight software to lower the development barrier for avionics projects. The following sections cover each of these topics in more detail.

2.1 Reliability

The main concern when considering reliability in spacecraft is the effects of ionizing radiation. Ionizing radiation mainly affects semiconductors in two ways, Single Event Effects (SEEs) and Total Ionizing Dose (TID). As the name implies, SEEs are a category of events caused by ionized particles altering the charge in a portion of a semiconductor. The most common SEE is the Single Event Upset (SEU), whereby charged particles affect the digital state of memory devices, thus corrupting memory contents. TID is the long-term effect of radiation altering the structure of the transistors within the semiconductor, which can lead to undefined behavior in digital systems.

TMR increases reliability by using three copies of a system and then comparing the outputs in real-time to detect faults. If not all copies are the same, the system will continue on with the matching two. TMR is a proven concept with a long history of use in spaceflight, having been used in vehicles such as the Saturn V[4, p. 1–6], Space Shuttle[5, Chapter 4], and Dragon[6].

2.2 Size, Weight, Power, and Cost (SWaP-C)

To optimize for SWaP-C, CHASE makes extensive use of COTS components. COTS components offer many advantages over traditional aerospace-grade radiation-hardened semiconductors. COTS components are built on newer process nodes

(GR740 - 65nm[7, §1.1] vs. AM6548 - 28nm[8, p. 3]) which increase performance and decrease power usage. COTS components take advantage of economies of scale to bring costs down and have extensive documentation and existing open-source support, making software development easier. While having three copies of the System on a Chip (SoC) and memory does increase cost and complexity, the increased reliability justifies the tradeoff. Three copies are also significantly cheaper than the closest radiation-hardened option, which doesn't come close in terms of performance[3], [9].

2.3 Software

While there has been an increase in open-source flight software recently, there is still a lack of many key components for a streamlined development experience. Multiple options currently exist for the operating system and framework portions; however, there is a lack of development in other categories. CHASE aims to develop a large ecosystem of mission agnostic modules for the F prime framework. F prime is an open-source flight software framework developed by NASA for small satellite missions[10].

Figure 1 shows a typical mission's software components by subsystem and their connections to each other. Each item is color-coded to indicate what type of component it is. Included in F prime are generic components that come included in the F prime framework. Existing open-source components are items with readily available open-source implementations. Except for hardware- and mission-specific components, everything else was selected as candidates for development. The CHASE project includes developing these components, which provides a library for avionics designers to easily use in future missions.

Releasing the software as open-source encourages contributions from outside parties, increasing quality, quantity, and support for everyone utilizing it.

3. Design of the OCHRES processing module

After defining the methodology and goals for the CHASE project, the next step is the development of the OCHRES processing module. This hardware is being developed in parallel to the methodology and serves as a methodology test case, as well as a usable product. A core processing module was chosen to be the first hardware item because such systems are one of the components with the highest importance and reliability requirements.

The following sections explain the implementation of TMR (§3.1), the hardware subsystems and component choices (§3.2), and the module-specific software components (§3.3).

3.1 Implementation of Triple Modular Redundancy (TMR)

The system's TMR design takes advantage of the Programmable Real-time Units (PRUs) within each System on a Chip (SoC). This offloads the work of message verification and synchronization from the main processors. A PRU from each SoC is dedicated to this task and is connected directly to the two other SoCs using its ethernet interfaces. The TMR design has two major functions, message verification (§3.1.1) and system synchronization (§3.1.2).

3.1.1 Message Verification

The verification software ensures that all outgoing messages contain accurate information by holding transmission until two or more processors have verified the message. Table 2 shows the currently defined modes used for handling messages on different interface types, characterized by bus type, transmit action, and receive action. The design allows for the development of new modes, should the need arise.

To implement message verification, the system retains a table in memory consisting of a list of outgoing messages. An entry contains the following information about each message:

- ID
- Length
- Pointer
- Hash
- Interface Type
- Creation Timestamp
- Verification Count
- Status Flag

Figure 2 is a flowchart demonstrating the behavior of the verification software.

3.1.2 Synchronization

The synchronization component of TMR guarantees that critical data is shared and kept consistent across processors. Periodically processors take a hash of each software module's essential variables and compare the data. If a module's data is incorrect, the correct data is sent over, resynchronizing the system. This procedure is also used to resynchronize in the case of a single-SoC reboot.

3.2 Hardware

The following subsections explain component and design choices for each hardware subsystem. An overview of the design can be seen in Figure 3, showing the triple redundant SoC and memory, dual redundant power supply, and mission-specific interface board.

Bus Type	Example	Transmit Action	Receive Action	Downstream Action
Multi-master	CAN, I^2C	Message-specific master verifies	None (received by all)	None
Point-to-point	PCIe	Verify each	Relay to each	None
Point-to-point (dual-redundant)	Ethernet	Each master verifies	Relay to each	Verify both are the same
Point-to-point (triple-redundant)	Ethernet	None	None	Choose from three

Table 2: Attributes of interface modes

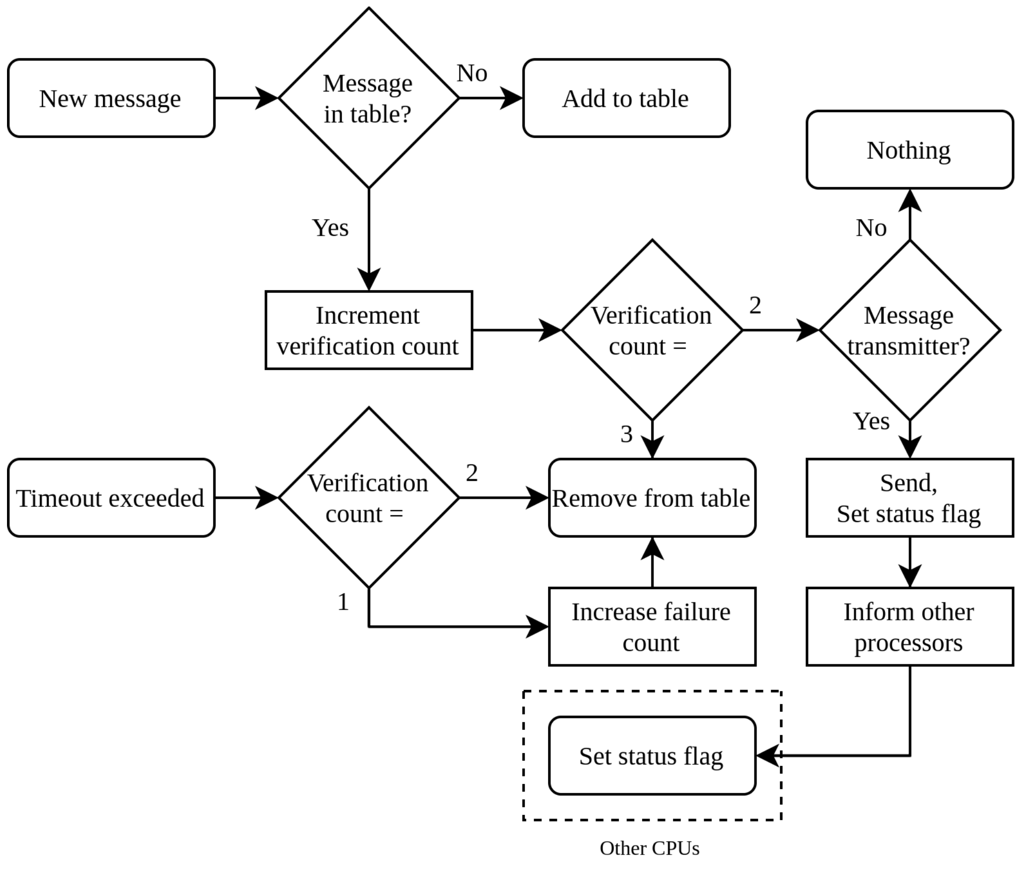

Figure 2: PRU verification flow

3.2.1 System on a Chip (SoC)

The SoC selected is the Texas Instruments AM6548[9], a processor designed for industrial control applications. The SoC requirements included both real-time and application cores, Error Correction Code (ECC) support, and suitable IO. By including both real-time and application cores, tasks with strict real-time concerns (e.g. GN&C) can be put on the real-time cores, while less time-sensitive but more computationally intensive tasks (e.g. scientific data processing) can take full advantage of the processing power of the application cores. The AM6548 contains two ARM Cortex-R5F real-time cores running in lockstep for increased redundancy, four ARM Cortex-A53 cores, and three PRUs. Each PRU contains two processing cores, two real-time cores, two gigabit Ethernet interfaces, and a variety of IO options. The PRUs are of special interest for OCHRES, because one can be dedicated to controlling and synchronizing systems between the three processors, communicating with each through a dedicated ethernet link, and offloading that work from the main processors.

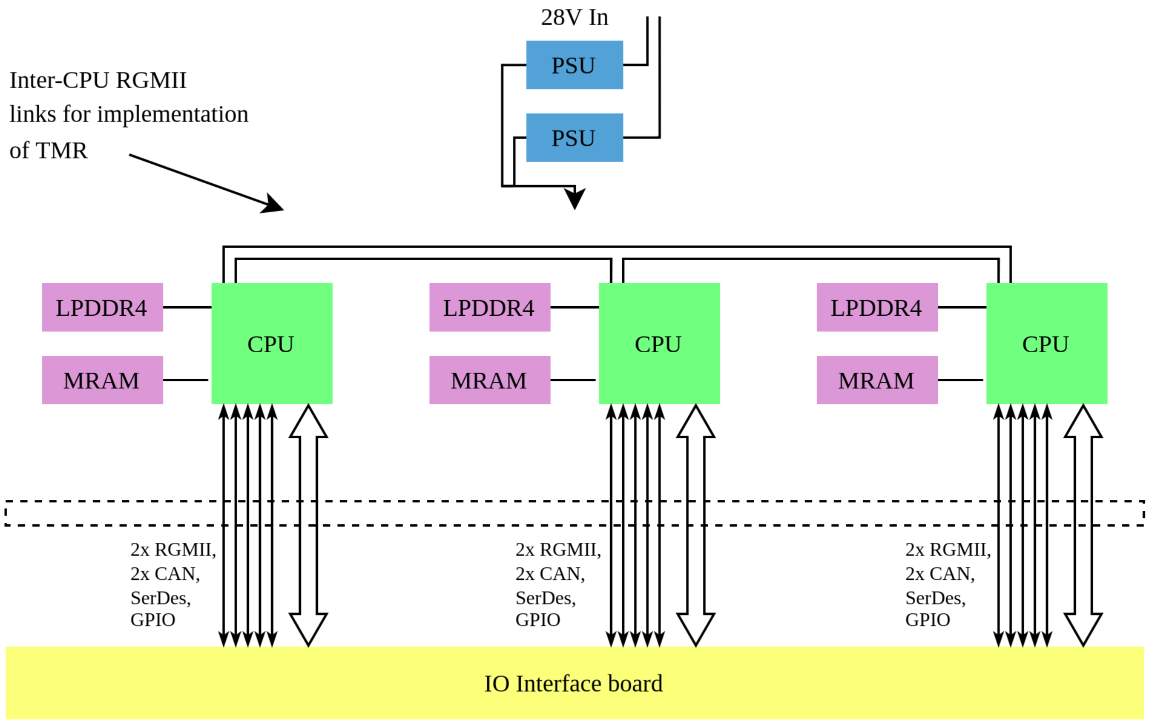
28V In
PSU
PSU
Inter-CPU RGMII links for implementation of TMR
LPDDR4
MRAM
CPU
LPDDR4
MRAM
CPU
LPDDR4
MRAM
CPU
2x RGMII, 2x CAN, SerDes, GPIO
2x RGMII, 2x CAN, SerDes, GPIO
2x RGMII, 2x CAN, SerDes, GPIO
IO Interface board

Figure 3: Processing module diagram

3.2.2 Memory

OCHRES contains one type of volatile memory and two types of non-volatile memory.

For volatile memory, the SoC uses LPDDR4, the low-power version of the common DDR4 standard. The SoC's memory controller includes support for ECC, which is used to negate the risk of SEEs corrupting the memory's contents.

The processing system contains two types of non-volatile memory; one is used for program and parameter storage, and the other is used for mass storage of mission or scientific data.

For program storage, Magnetoresistive Random Access Memory (MRAM) was chosen because of its flight heritage, high reliability, and resistance to radiation[11]. MRAM stores data in magnetic tunnel junctions, instead of the floating-gate MOSFETs used in flash or EEPROM memory. This technique makes MRAM less susceptible to stray charges imparted by ionizing radiation.

For mass storage, OCHRES uses commercial high-reliability flash storage, connected to the exposed PCIe bus of one processor. Flash storage is more sensitive to radiation, but alternatives such as MRAM are simply not available in the required densities. By using a redundant filesystem such as RAID or ZFS, errors from radiation can be corrected, similar to the use of ECC on volatile memories.

3.2.3 Interfaces

The processing PCB is designed to connect to a mission-specific interface board. This board routes the processor IO to the rest of the spacecraft, allowing for more flexibility in design while still allowing full utilization of all available IO. For each processor the processing board passes through the following interfaces:

- RGMII Gigabit Ethernet (2)
- CAN (2)
- high-speed SerDes (2)
- GPIO

These interface types provide a multitude of options for the mission designer to utilize.

3.2.4 Power

The processing board is provided with a nominally 28V supply, which is regulated down to the levels required by the SoC and its subsystems. The regulator subsystem is completely dual-redundant and can be powered by either of two independent buses. This subsystem also provides regulated power to devices on the IO interface board.

3.3 Software

OCHRES requires a multitude of software, which is being developed in conjunction with the hardware. The software stack is shown in Figure 4.

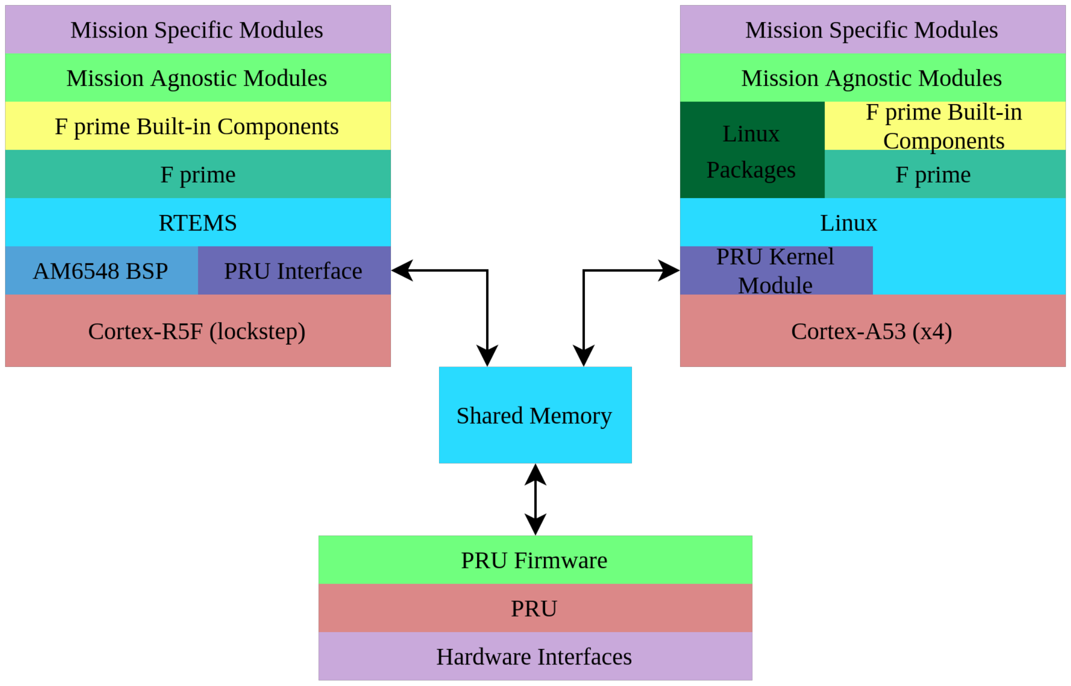

Figure 4: Processing module software stack

The real-time cores run Real-Time Executive for Multiprocessor Systems (RTEMS), an open-source Real-Time Operating System (RTOS) that has been certified by the European Space Agency for European Cooperation for Space Standardization (ECSS) criticality C[12]. RTEMS was chosen because of this certification, and its long history of use in spaceflight. The use of RTEMS on the AM6548 requires the development of a Board Support Package (BSP), the software which connects the SoC hardware to the RTEMS OS.

The Cortex-A53 application cores run Linux. The existing support, knowledge, and heritage make Linux the best choice for general-purpose processing, where real-time constraints do not apply to the same degree as in the real-time cores.

Kernel libraries will be developed for both RTEMS and Linux, allowing user-mode applications to access the hardware interfaces via the PRU.

3.4 Usability

A primary goal of CHASE is accessibility. Documentation and support are major factors in creating accessible products, and current options lack extensive documentation. CHASE aims to remedy this issue. The open-source nature of CHASE allows anyone to look at the exact design, and the system includes extensive examples and reference designs for common applications. Because CHASE uses up-to-date COTS components, many current operating systems, applications, and libraries will support CHASE with minimal modifications. This reduces the need to create application- or hardware-specific software.

4. Design elements not considered

CHASE was never meant to be the be-all-end-all of inexpensive avionics systems, merely a starting point and an example of what is possible. As such, there is plenty of room to expand and continue this and similar work. The CHASE project and OCHRES processing module have some significant shortcomings, which will briefly be discussed here.

A major drawback of the CHASE standard is that it does not define a way of selecting components with the highest radiation tolerance for a given type of COTS part. Ideas such as "Careful COTS"[13] have been proposed, but there is an unfortunate lack of publicly-available radiation test data. Most data is either outdated and/or not easily accessible by the public. This is a problem the author hopes to remedy in the future, but the discussion of that is outside the scope of this document.

A deficiency of the OCHRES design is that not much thought was given to the thermal design. Thermal design for spacecraft is complicated compared to traditional electronics, and powerful processors such as the AM6548 emit a lot more heat than existing aerospace processors. This is something that could be remediated with more research and design but was simply not considered in this iteration.

While the CHASE methodology is designed to be versatile and applicable to many areas of avionics, it currently lacks the breadth to be as useful outside of the area for which it was originally designed (development of OCHRES). The methodology can easily be expanded to encompass more guidelines, offering a better framework for a wider variety of projects.

5. Conclusion

This paper has laid out a set of guidelines for the development of inexpensive, reliable, high-performance avionics. It included the development of OCHRES, a processing module utilizing the standard, and discussed the development plan for a suite of open-source software modules for aerospace applications.

The next steps for the CHASE project are the definition of a detailed guide for the selection of COTS components, the development of the software modules, and the use of the CHASE methodology in the development of more hardware modules. OCHRES can fill a gap in the market, but there are many similar gaps in other areas of the market that CHASE is well-suited to fill.

OCHRES is still in the very early development stage, and in the coming months can progress towards prototype hardware and eventually test flights. The next steps are the development of prototype hardware, software design, and integrated testing.

Acknowledgments

The author would like to thank the Nexus Aurora community for their extensive support throughout the project; William Forsberg, for joining the project early and continually supporting it; and his family, for their generous support, editing help, and for making this project possible.

References

[1] CubeSatShop. “EXA ICEPS Spacecraft System Core.” (2022), [Online]. Available: https://www.cubesatshop.com/product/iceps-spacecraft-system-core/ (visited on 08/25/2022).

[2] BAE Systems. “Radiation-hardened electronics product guide.” (2017), [Online]. Available: https://www.baesystems.com/en-media/uploadFile/20211206201500/1434554723601.pdf (visited on 08/25/2022).

[3] Cobham Gaisler AB. “GR740 product brief.” (Oct. 10, 2021), [Online]. Available: https://gaisler.com/doc/gr740/Product_Brief_A4_GR740_20220616.pdf (visited on 08/15/2022).

[4] I. B. M. Corporation, *Saturn V Launch Vehicle Digital Computer*. Sep. 30, 1964. [Online]. Available: http://klabs.org/DEI/Processor/apollo/19730063841_1973063841.pdf (visited on 07/26/2022).

[5] J. E. Tomayko, “Computers in spaceflight: The NASA experience,” *Encyclopedia of Computer Science and Technology*, vol. 18, Supplement 3 Mar. 1, 1988.

[6] A. Svitak, “Dragon’s "radiation-tolerant" design,” *Aviation Week*, Nov. 18, 2012, Archived from the original on 02/08/2021. [Online]. Available: https://web.archive.org/web/20210208083644/https://aviationweek.com/dragons-radiation-tolerant-design (visited on 07/26/2022).

[7] Cobham Gaisler AB. “GR740 radiation summary.” (Sep. 28, 2020), [Online]. Available: https://gaisler.com/doc/gr740/GR740-RADS-1-1-3_GR740_Radiation_Summary.pdf (visited on 08/22/2022).

[8] Texas Instruments. “AM654x, AM652x Sitara™ Processors, Silicon Revision 2.1 datasheet (Rev. B).” (Mar. 31, 2021), [Online]. Available: https://www.ti.com/lit/ds/symlink/am6548.pdf (visited on 08/22/2022).

[9] Texas Instruments. “AM65x/DRA80xM Processors Technical Reference Manual (Rev. E).” (Dec. 18, 2019), [Online]. Available: https://www.ti.com/lit/ug/spruid7e/spruid7e.pdf (visited on 08/17/2022).

[10] R. L. Bocchino Jr., T. K. Canham, G. J. Watney, L. J. Reder, and J. W. Levison, “F prime: An open-source framework for small-scale flight software systems,” SSC-18-XII-04, 32nd Annual AIAA/USU Conference on Small Satellites, Aug. 2018.

[11] Everspin Technologies Inc. “Aerospace | Everspin.” (2022), [Online]. Available: https://www.everspin.com/aerospace (visited on 08/06/2022).

[12] European Space Agency. “RTEMS SMP Qualification Data Pack.” (2022), [Online]. Available: https://rtems-qual.io.esa.int/ (visited on 08/18/2022).

[13] D. Sinclair and J. Dyer, "Radiation effects and COTS parts in smallsats," SSC13-IV-3, 27th Annual AIAA/USU Conference on Small Satellites, Aug. 2013.