Avionics for HyMOVE
A Modular Orbital Transportation System for Small Satellites *

Sergio Montenegro[1], Michael Strohmeier[1], Paola Breda[2], and Michael Vogel[2]

[1]Department of Aerospace Computer Science, Würzburg University
[2]HyImpulse Technologies GmbH

February 14, 2023

Contact: sergio.montenegro@uni-wuerzburg.de
Presentation will be performed by: Sergio Montenegro
Topics: 1.1. Software architectures and frameworks, 4.3. Data controllers, bus and micro-controllers, 4.4. Operating System (OS) front-ends, 5.4. System specific OBC solutions

1 Introduction

Within HyMOVE ("Hybrid Multipurpose Orbital Vehicle"), HyImpulse Technologies GmbH (HIT) and Julius-Maximilians-University Würzburg (JMUW) are developing an orbital vehicle, which can be adapted to a diverse set of mission scopes, thanks to its modular architecture. This modularity stems from the standardization of parts, and thereby to a reduction of cost as general industrial practices like serial production can be adopted into the space industry.

Next to this flexibility, one of the core technologies of the HyMOVE vehicle lies in the utilization of hybrid rocket propulsion, which is defined as the type of rocket propulsion that uses both solid and liquid components. This yields several advantages compared to both liquid and solid propulsion, key among which are safety and simplicity at high Isp-performance [1]. Unlike solid propulsion, hybrid rocket motors are both reignitable and throttleable, allowing for more complex missions to be executed based on this type of engine. Hybrid propulsion has historically been plagued by low thrust densities, an issue which has been overcome due to the advent of liquefying fuels. Among these, paraffin has been the fuel of choice for HIT's propulsion developments, but the baseline choice of oxidizer used in HIT's other projects (liquid oxygen) is unfit for an in-space application due to its cryogenic nature. Instead, alternatives for oxidizers are currently under investigation, specifically "green" oxidizers like hydrogen peroxide (H_2O_2) or nitrous oxide (N_2O), to complement the already non-toxic and soon to be carbon-neutral paraffin fuel.

JMUW develops the corresponding highly modular, scalable, and reliable avionics systems. HyMOVE is intended to have a modular design, so that the technology can be adapted for a wide variety of applications. For HyMOVE, very high dependability is needed and if the system remains in orbit for extended periods, then also very high reliability: Fault tolerant control (software and hardware), radiation resistant hardware and real-time critical tasks/control. In order to achieve high reliability in a very compact and very high-performance components-off-the-shelf (COTS) package, we will evaluate the use of SOI (Silicon on Isolation) Technology. This technology is most promising for this application due to its lower power consumption and superior radiation tolerance compared to CMOS (Complementary metal oxide semiconductor) components. The software includes the real-time kernel, software infrastructure, communications middleware from our building blocks execution platform RODOS, and further key applications for satellite and kick-stage operations. Our fault tolerance mechanism is based on an "ultra-fast recovery" system. The goal is for each computer to take less than a second to reboot and start all applications. This allows the hardware to rapidly rejoin the network after a crash and

*Thanks to the financial support from Bayerischen Staatsministeriums für Wirtschaft

retake control of critical tasks. Beyond this recovery technique, important functions are also replicated in the network several times in different nodes so that a crash does not jeopardize the flight. While novel, we are confident in the successful implementation of our "ultra-fast recovery" concept as described.

1.1 Hardware

All Hardware is implemented as dual-X (X: controller, router, connections, switches, etc.) for fault tolerance, meaning every basic hardware unit is duplicated, as shown in Figure 1. The system consists of two parallel running Time Triggered Ethernet networks. Every controller, e.g. the dual-board computer or dual-payload computer or dual-GNC Computer, has two interfaces to the network, one for each router. Hence, each Dual-X is implemented with a total of four links to the dual-network.

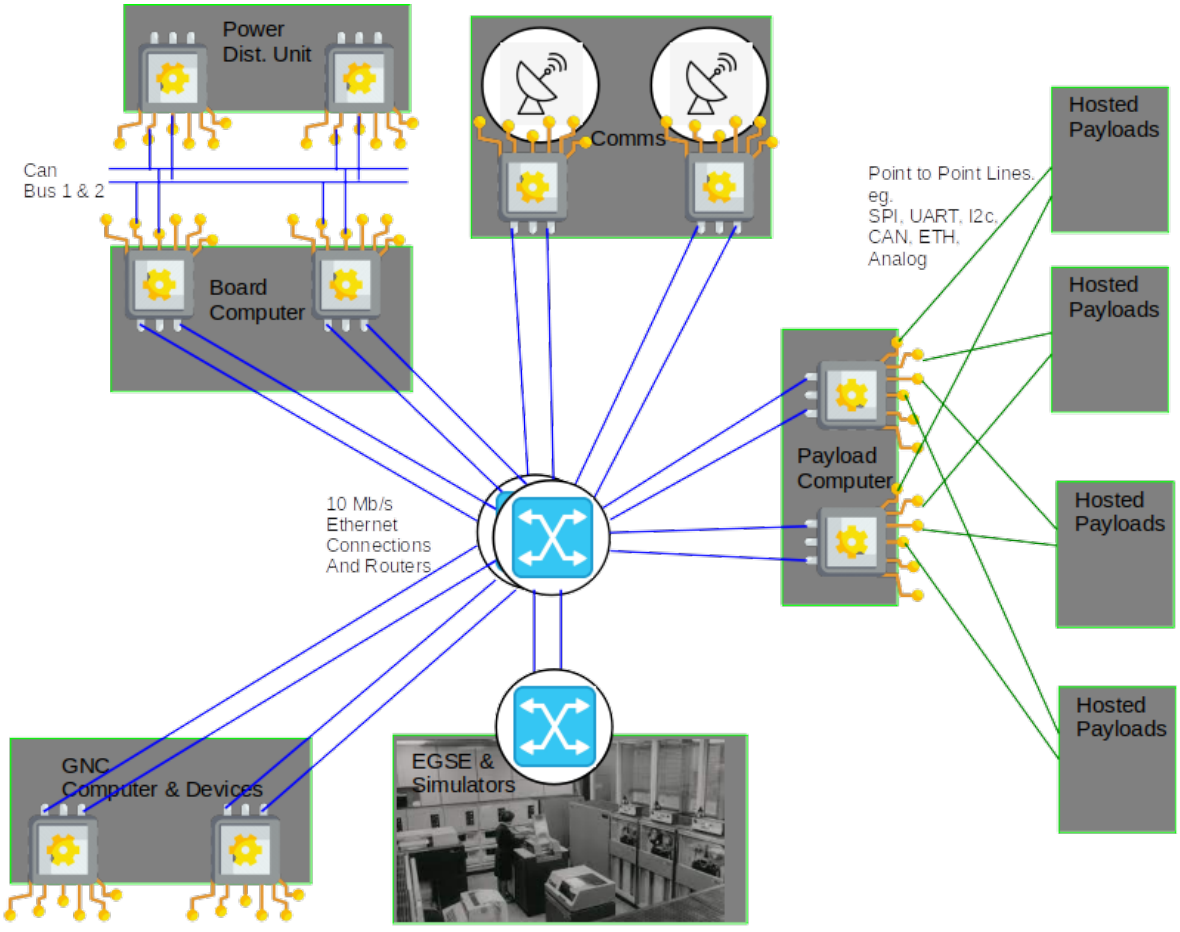

Figure 1: HyMOVE Avionic Hardware

Figure 2 shows the implementation of each dual computer (eg. dual board computer, dual payload computer and dual GNC computer). Described in some further detail below are the following two aspects of hte system:

1. IO Adapter
2. Keep-off

1.1.1 IO Adapter

About the IO Adapter: As there is only one standardized dual computer configuration, an IO adapter module shall be used to achieve the necessary functionality for various payloads and mission types.

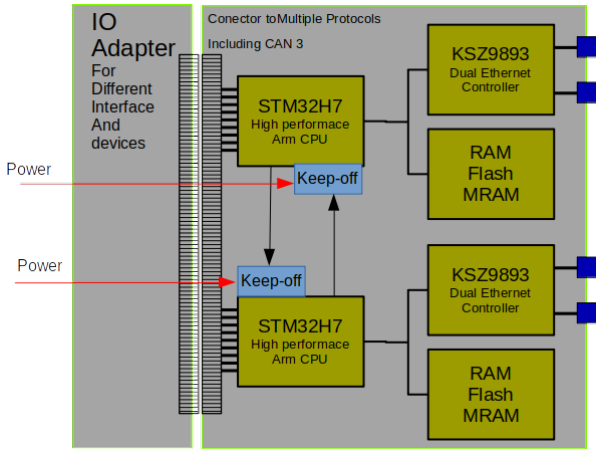

Figure 2: HyMOVE Dual Computer

The computer will interface with the IO adapter through a standardized IO connector to achieve high modularity and interoperability. The IO adapter can therefore readily be re-implemented for unique IO configurations without a substantial redesign of the dual computer.

1.1.2 Keep-Off

About Keep-Off: With the Keep-Off circuit we implement simple yet flexible redundancy management. The redundancy management is completely controlled by software, allowing for straightforward switching between hot and cold redundancy. Each node has a so called Watchdog Application. The watchdog application checks the timings and proper behavior of all other applications and devices on the computer. If everything seems to be working appropriately, it will send a pulse to the keep-off circuit of the other computer. If a computer crashes, its watchdog application will notice the interruption of its processes and stop sending the keep-off pulse, forcing the other computer to be turned on and take over the tasks and processes. As long as one computer (the worker) keeps sending a pulse to the keep-off circuit of the other computer (spare), the spare computer will remain off. If the pulse does not arrive within the set time period, the spare computer will become active and take control (become the worker) after completing its startup self-check. Once operating, the new control computer (previously spare) will send the keep-off signal, forcing the other computer to become the cold spare.

1.2 Software

We implement the control software ”by construction based on building blocks (BB)” within the BB-execution Platform RODOS. RODOS is a dependable distributed real time operating system. RODOS provides a real time micro kernel and a distributed Publisher-Subscriber Middleware. Applications distribute messages by publishing on topics. The middleware will distribute the message to all subscribers of a topic in the system, independently of its position. For the publisher it is discernable if the communication partner is running in the same computer or on another computer attached to the network. It is even possible to build distributed systems, in which one part is running in the spacecraft and another on the ground segment. Even these physical divisions of the system will be discernable in the publisher-subscriber communication. This allows for the dynamic distribution of applications among networked computers, simplifying the fault tolerance and redundancy management. Applications are then the Building Blocks (like chips on hardware), which will be interconnected by the middleware (like on a printed circuit board) as depicted in Figure 3. On a larger scale, complex software can then simply be ”assembled” using the various BBs, without elaborate consideration of the internal structure of each BB.

Figure 3 shows a typical BB (applications) and topics configuration for a spacecraft. Details will be explained in the full paper.

Figure 4 shows a typical BB (applications) and topics configuration for a spacecraft. Details will be explained in the full paper.

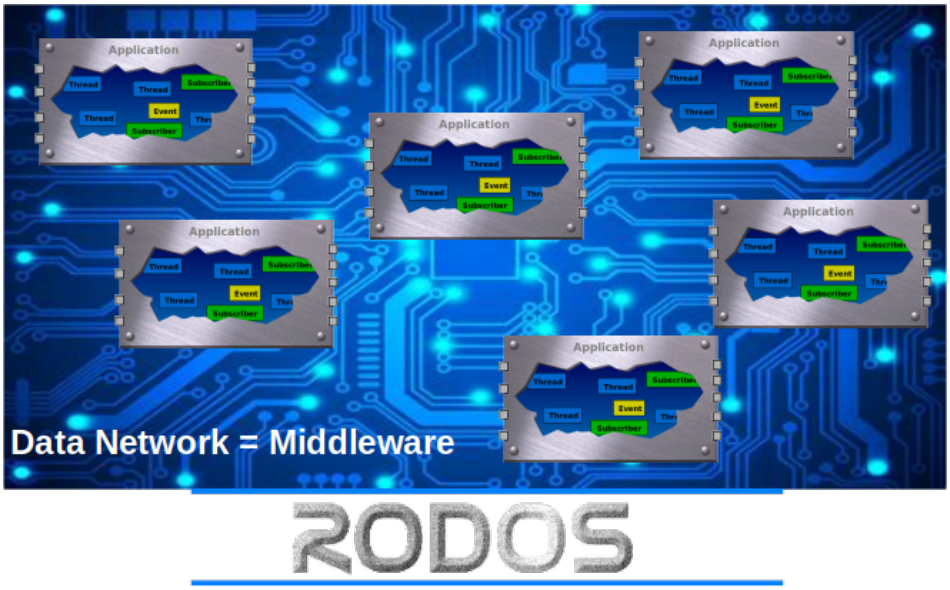

Figure 3: Building Blocks on RODOS BB-Execution Platform

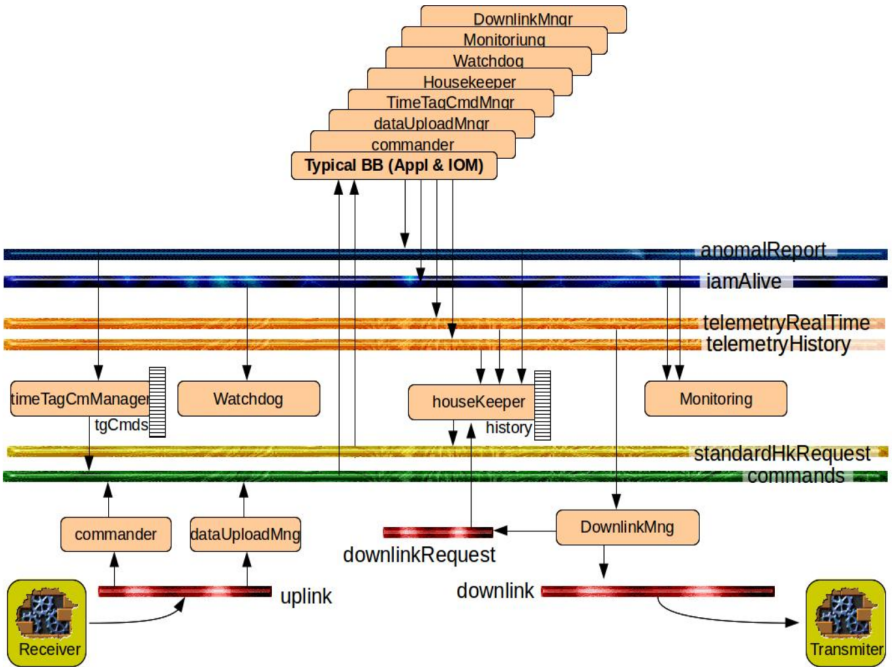

Figure 4: Typical BB and Topics configuration

References

[1] E. Messerschmid and S. Fasoulas. Raumfahrtsysteme, propulsion. In *Raumfahrtsysteme*, 2000.