EUROPEAN WORKSHOP ON ON-BOARD DATA PROCESSING (OBDP2021), 14-17 JUNE 2021

ON-THE-GROUND TESTBED FOR AI/ML-ASSISTED ON-BOARD-PROCESSING ON NANOSATELLITE PLATFORM: BRAIN IN SPACE INITIATIVE

Wagner Samantha[(1)], Cappaert Jeroen[(2)]

[(1)]Spire Global, Inc., 33 Rue Sainte-Zithe, 2763 Luxembourg (Luxembourg), Email: samantha.wagner@spire.com
[(2)]Spire Global, Inc., 33 Rue Sainte-Zithe, 2763 Luxembourg (Luxembourg), Email: jeroen@spire.com

ABSTRACT

With support from the ESA Earth Observation Science for Society Programme and in cooperation with Φ-lab, Spire has created '*Brain in Space*' - an on-ground testbed, accessible via a web-based interface, replicating Spire's LEMUR 3U platform, the flagship of Spire's global nanosatellite constellation of over 110 satellites. *Brain in Space* consists of the same hardware that can be found on a flight version of our LEMUR spacecraft including communications and power systems, processing payloads, onboard computer and other utilities. In addition to the standard LEMUR systems and computing payloads (Zynq Ultrascale+, Jetson TX2i), the testbed includes a Google Coral, a Jetson Nano and an UP Myriad X module.

The AI/ML modules allow for scheduling, uploading and testing of AI-powered applications to rapidly process space sensor data of various types and from different sources : Automated Identification System (AIS), Automatic Dependent Surveillance - Broadcast (ADS-B), GNSS-R (reflectometry), GNSS-RO (radio occultation), or Space Environment Monitoring.

The use of an easily accessible on-the-ground testing environment like *Brain in Space* enables acceleration of new services development, innovations within smart data processing and edge computing directly on-board of small satellites and, stress-testing new solutions ahead of the launch to space. It also creates an opportunity to pilot new, AI-empowered ways of how nanosatellite constellations are operated and managed.

This paper provides an overview of the development of the *Brain in Space* testbed, details on the available components and the infrastructure for external parties.

The Brain in Space project was developed under a programme of, and funded by, the European Space Agency. The views expressed in this paper are in no way to be taken to reflect the official opinion of the European Space Agency.

1. INTRODUCTION

Over the last few years, the space industry has seen an exponential growth in the launch of small satellites, as seen in Fig 1. Mechanically, this explosive growth in launches of private and public satellites has induced a breakneck speed of growth for space-generated data. This has contributed to the across-the-board rise in users for space-generated data as both volume and reliability have tremendously improved, while data latency requirement has often been reduced to almost real-time for certain data types. This process has created a considerable challenge in the data flow management and, in parallel, satellites themselves have grown increasingly capable at generating increasing datasets.

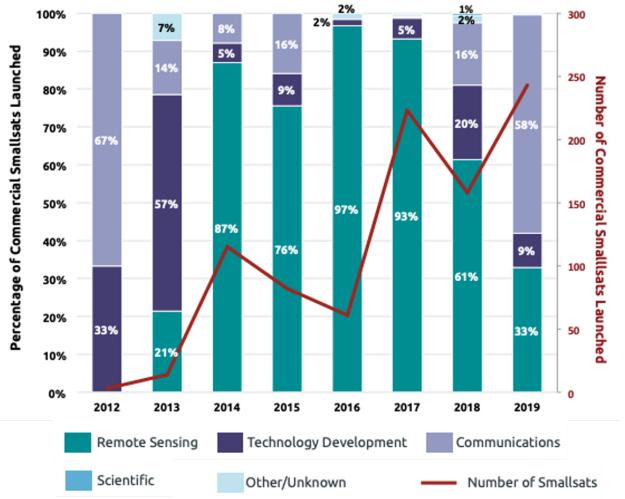

Figure 1. Commercial smallsats by use [1].

This massive rise in data production from space has shifted the traditional issue of satellite operators and their customers from classic data-gathering ("*Can I get this data?*") to a more complex data-triage problem ("*Can I get the right data to the right people at the right time with minimal use of additional resources?*"). In other words, limitations in satellite-data applications are moving from the technology of acquiring the data to the techniques required to optimally exploit the information within the remotely sensed data.

The latest technological developments in chipmaking have made possible the processing of an increasing part of the data analytics directly at the satellite level, in space, thus progressively reducing the amount of data to be downloaded back to ground stations. This factor improves constellation efficiency by reducing demand on the infrastructure on the ground, and the bandwidth required for the transmissions. Moreover, it allows to raise the speed at which critical information is transmitted to users by allowing the satellite to prioritize independently which data will be downloaded first, thus decreasing latency for truly relevant information, as well as to allow the satellite to autonomously direct sensors and make time-critical decisions.

In parallel, many new AI-empowered chips have ignited a lot of interest by allowing small platforms to run complex pattern determination and identification programs. These developments are potentially a game-changer for many current space-based products and are likely to create profoundly disruptive applications that will largely exceed the capabilities of current space-based platforms in generating large sets of "smart" - directly actionable - data.

1.1. Objectives

The *Brain in Space* project aims to offer a nanosatellite testbed with embedded AI/ML chips for users to schedule, upload and test their own AI applications and frameworks. The purpose of this testbed is to create a simulated operating environment on the ground that allows to tests the ability of a chip to enable the running of AI algorithms, to perform :

1. time-critical missions
2. reduction of download bandwidth requirements
3. autonomous decision-making

These three dimensions are today major bottlenecks in the operation and management of small satellites constellations as the number of satellites in orbit has grown geometrically since the early 2010's, with each spacecraft able to generate and transmit an exploding amount of data back to Earth, creating a strain on the constellation's infrastructure and creating major road bumps in the transmission of the data to the final user.

1.2. Development Process

This project has been organized sequentially over one year, with three phases allowing for the setup of the testbed (chip selection, hardware and software implementation), and one longer phase dedicated to the operations of the testbed and actual testing of the retained chip's capabilities by end users.

First, a comprehensive mapping of all the AI/ML chips commercially available that can meet the project's technical requirements has been carried out. A trade-off analysis of the pros and cons of each available solution has been made to select the most relevant chips only for the testbed.

Second, following the chip selection, the hardware components required for the development of the ad hoc chip carrier board have been acquired and assembled to provide the physical infrastructure required as part of this testbed project. This has been realized by creating the schematic design of the board, designing the PCB layout, and manufacturing the hardware. A schematic view of the assembled testbed is shown in Fig. 2 hereafter.

The last phase was the programming of the software architecture required to run the testbed efficiently for Spire teams and other interested parties. The required software interface between the chip, Spire sensors and the testbed itself (including the dummy ground stations and data feeds) were created, while a dedicated VPN for access to third parties outside Spire has been put in place for end users.

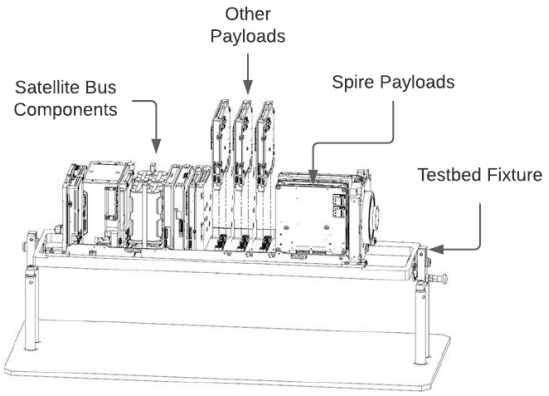

Figure 2. A schematic view of the testbed.

Now that those three phases to develop the *Brain in Space* testbed are completed, the operation phase of the testbed has been initiated, which entails the maintenance, monitoring and operation of the testbed systems. External parties are welcome to participate in running experiments using the retained chips and our available sensors.

1.3. Scope of this Paper

This paper presents the work done on the *Brain in Space* project, including the final testbed assembly, and provides a reference entry about the available infrastructure for external parties.

2. CHIP SELECTION

For the chip selection, a set of requirements have been established. In particular, the chip must be adapted to the size, weight and power (SWaP) constraints of the

platform (here, the Spire LEMUR2 3U platform and its PCB module as reference baselines). It must also be able to survive the space environment for the duration of a mission (which could mean several years of operations).

The chip tradeoff analysis was then based on high-level tabulated specifications (such as power consumption and processing capacity), rather than performance benchmarks. This is because the performance of different chips is difficult to assess without benchmarking each chip using identical parameters, as it is highly dependent on the benchmark setup, training framework (e.g. TensorFlow, Caffe or Keras), type of data and even chip architecture.

2.1. Chip Architectures

Several different hardware architectures can be employed for running AI algorithms, and these all have their own advantages and disadvantages. We have considered the following architectures when selecting chip candidates, as they are commonly used for edge AI applications: Graphical Processing Units (GPUs), Tensor Processing Units (TPUs), Vision Processing Units (VPUs), and Field-Programmable Gate Arrays (FPGAs).

2.2. Framework Compatibility

In addition to hardware, AI chips rely on frameworks [2], which are libraries and sets of functions that allow programmers to quickly design AI algorithms and implement machine learning. There are many frameworks on the market, some of which are fully open source while others are associated with a hardware license. Hardware framework compatibility is important, as it opens up more options for programmers when designing AI algorithms for data processing. Each framework has both advantages and disadvantages, and the choice of a framework is driven by the type of application, project timescale and programmer knowledge. For each candidate chip we have taken into account the frameworks compatibility, such as TensorFlow/TensorFlow Lite, Microsoft CNTK, Caffe, Theano, Amazon machine learning, Torch/PyTorch, Accord.Net, Apache Mahout, Keras, CUDA, and Spark MLlib.

2.3. Selected Chips Available on the Testbed

The data and power interfaces, as well as the size and power consumption, of each payload candidate has been evaluated. We have used the interface requirements of each chip for an initial down-selection, as some interfaces may require a significant amount of engineering effort to integrate with the testbed. Other potential candidates required a higher voltage supply than what Spire platform offers. In addition, the Zynq boards are very similar to each other in architecture and performance, and from a ML testing perspective it may not make a significant difference to results. We have therefore chosen one of these chips for the testbed, the Zynq Ultrascale+, carried on Spire satellites. The selected chips are detailed hereafter :

Xilinx Zynq Ultrascale+: this is a FPGA used by Spire on-board many modules. Xilinx has created engines for running AI algorithms on the chip [3] [4].

Nvidia Jetson TX2i: a GPU flown by Spire as part of a computing module to support on-board data processing. While not necessarily intended for AI, Nvidia has released toolkits for it, and the Jetson series are popular in the AI community due to the ease of using these toolkits as well as the high processing power of the GPUs.

Google Coral: this TPU is currently the main offering by Google, and is sold as a development kit. It runs TensorFlow Lite natively, and is marketed as an easy to use, high processing to power to ratio low-cost AI hardware platform. Google Coral is also particularly suited for cloud applications, and an AI cloud service based on TPUs is offered by Google.

Nvidia Jetson Nano: the Nano is designed to be the most portable and lowest cost GPU in the Jetson series, and has lower processing power but also lower power consumption than other GPUs. It runs the same libraries and frameworks as the other Jetson chips. It is currently planned to be flown on a USC mission that will use it to demonstrate on-board data processing using AI [5]

Intel Myriad X: originally designed by Movidius (later acquired by Intel), this VPU is optimised for deep neural networks and rapid prototyping. The Myriad is compatible with both the TensorFlow and Caffe frameworks, but it is otherwise fairly closed in as it is only currently available as a USB compute stick for evaluation and benchmarking. It is unclear whether the chip would be easily available if deployed at a larger scale, or in an in-orbit demonstration mission. We were also unable to find evidence of flight heritage, but Myriad-2 has been flight-tested on-board the Phisat mission as well as radiation-tested at CERN. While the Myriad X is available as a stick from Intel, we have chosen a SoM from UP as the candidate for the testbed. The UP Vision Plus X carries three Myriad X chips and is connected to a SoM (UP Core Plus). As two of the Myriad chips are accessible only via the PCI-E interface, which is not compatible with the Spire bus, only one Myriad chip (accessible via USB interface) is usable on the testbench.

Additionally, but not directly available to end users, the *Brain in Space* testbed integrates a **Xilinx Zynq 7000 Series**: similar to the Ultrascale+, the Zynq 7000 Series is a FPGA that is part of the Spire communication module. While not used for AI applications at the moment, it shows potential for it and as with the Ultrascale+, specific AI engines have been created by Xilinx. The main difference between these chips is the lower power consumption and processing power of the Zynq 7000 Series.

3. TESTBED ASSEMBLY

Spire *Brain in Space* testbed consists of much of the same systems that can be found on a flight version of a LEMUR2 satellite platform, with the exception of some components that serve no purpose on a testbed, such as solar arrays.

The hardware setup consists of a satellite sled, which is a ground model of the Spire LEMUR2 3U satellite. It includes a satellite bus with all standard Spire flight components. The following core subsystems are present on the testbed:

- Power systems: Electrical power system, Battery dummy, lab power supply
- Communication systems: UHF radios, S-band radio
- Utilities: Low-power on-board computer (used for scheduling, telemetry collection and automation)
- ADCS modules

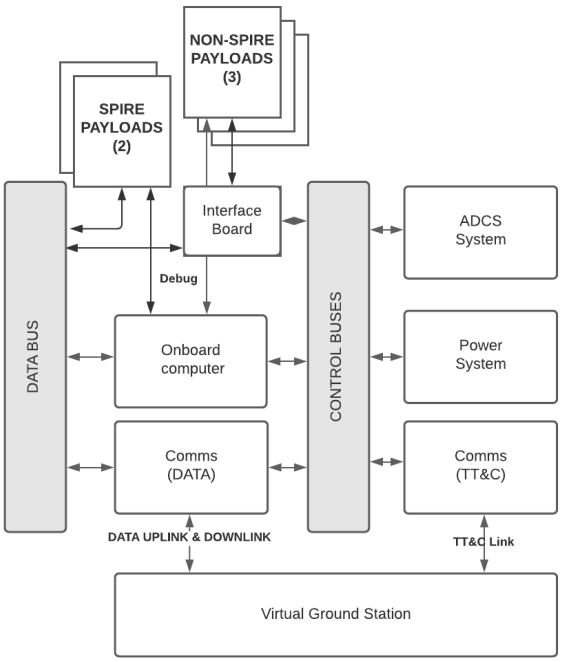

Figure 3: Testbed configuration block diagram

The testbed is divided into a bus section, and a payload section. The payload section contains an interface board used to connect the chips to the bus. A block diagram of the LEMUR2 satellite is shown in Fig. 4 below. The SPIRE PAYLOADS block contains any ML chips carried by Spire modules, while the NON-SPIRE PAYLOADS block in the figure contains the other AI/ML chips.

An interface board provides power and data connections between the Non-Spire ML chips and the on-board computer(OBC), via the satellite backplane. The OBC controls the payloads and links them to the user API (Application Programming Interface). There are several interfaces from the bus to control the Spire payloads, debug, one board at a time, and for data transfers.

Fig 3. shows the assembled testbed, including the satellite bus, Spire payloads, the interface board connecting non-Spire payloads to the bus, as well as the virtual ground stations.

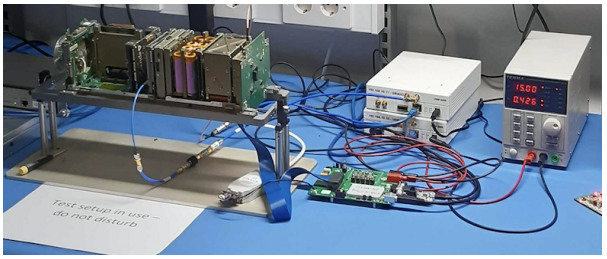

Figure 4. The assembled testbed.

4. SOFTWARE DEVELOPMENT

The *Brain in Space* testbed incorporates 5 AI/ML boards for testing end-user code in an environment that closely mimics on-orbit capabilities.

4.1. Architecture

There are three major components of the Spire system: Tasking API, Data Pipeline, and Satellite Bus.

The Tasking API is a cloud-based REST API that allows end-users to schedule operations, uplink data, and monitor status of their operations. The Tasking API is the end-user's interface to the Data Pipeline. By scheduling payload windows via the Tasking API, the end-user is creating operational window requests that will be communicated to the Satellite Bus at the next contact. These operational windows have a start and end time. The Satellite Bus is responsible for ensuring that these tasks are executed at the appointed time, and any output is collected. The Tasking API also allows the user to create uplink requests by submitting a file and specifying the particular payload and file system path that the file is destined for. Files submitted for uplink are placed in a data bucket where the Data Pipeline picks them up. Window and uplink requests may be monitored via appropriate endpoints on the Tasking API.

The Data Pipeline is responsible for making contact with the satellite, uplinking any window requests and packages that users have submitted, and collecting any output that is present on the Satellite Bus. Output is ultimately delivered to a data bucket location as defined on a per-user basis. The software components of the Data Pipeline consist of a database, a contact windows scheduler, mission stack, and delay-tolerant network. The database contains the authoritative schedule of all contacts and payload operations that have been requested, as well as the associated configuration details, and record of any in-progress data uplinks. The mission stack reads from this schedule in order to determine when to make contact with each satellite, and what requested payload operations and data packages to uplink.

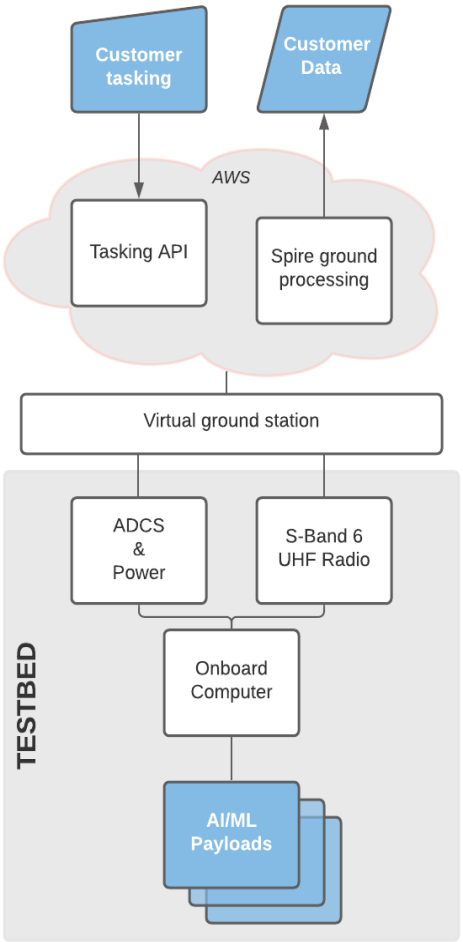

Figure 5: Brain in Space architecture overview

The software components of the satellite bus include a coordinating OBC process, and a payload process that is responsible for invoking any end-user supplied code. Window requests and uplink packages submitted by end-users are placed on the OBC file system by the Data Pipeline where they are received by the OBC process. It parses the window requests and constructs a schedule of operations to coordinate power channel activation. It is also responsible for monitoring the health of the satellite and taking any necessary actions to counteract an undesirable situation. When a scheduled window operation approaches, the OBC process performs any necessary setup including power-on and transfer of any uplink packages resident on the OBC that are destined for the payload. At the start time of the window operation, it invokes the payload process on the payload system, which is responsible for invoking any commands specified by the end-user and submitting any user output to the delay-tolerant network. After the window has concluded, the OBC process performs any necessary shutdown procedures on the payload.

4.2. Command and Control (C&C) Pipeline

The data pipeline can be split into three pieces:
1) command, control and configuration
2) window scheduling
3) customer data.

Window Scheduling via the C&C Pipeline
Payload windows generated via the Tasking API are executed on the satellite after the window has been synced. The customer requests operations via the Tasking API. The scheduler creates the contact windows in the database, so the satellite can execute the mission at the appropriate times. During payload windows, the OBC will power on payloads and trigger operations such as configuration and data collection.

Customer Software Upload via the C&C Pipeline
The customer uses the upload endpoint of the Tasking API to upload a new software package to a particular payload. During the next available bidirectional contact, the Production Mission detects that a new package is available, and uploads the package onto the satellite.

Downloading Generated Payload Data
The data is generated by the customer software on the payload during an operational window. When this data is generated, an agent transfers data files from the payload's storage to the OBC. The OBC encodes the data files into data packets, which are sent to the radio for transfer on the ground station (in the testbed, a dedicated Host PC acts as the ground station). Once received, the ground station relays the data packets to the customer cloud storage, where the incoming data is decoded, and shared with the customer.

4.3. Limitations

There are a few limitations to the *Brain in Space* software pipeline that should be called out, some of

which are limitations to the satellite constellation as a whole.

1. Low-bandwidth links are used for uploads (simulated). Very large uploads can be uploaded manually by the Spire team.
2. Users will not have direct access to debug the testbed (using eg: SSH).
3. When a window is created through the Tasking API the window is in a “pending” state. On the next scheduled contact the window will be synchronized with the satellite, after which the window moves to a synced state. Windows can only be deleted when they are in a pending state.

4.4. Testbed Architecture Fidelity

The test satellite, which resides on the ground, is executed and communicated with tools identical to those used in production. to the end user, the experience and interface is intended to be nearly identical to that of an operational satellite in space. Therefore, the architecture aims to reuse and share as many components as possible that are used in production. Moreover, to the end user, the difference between operating on a testbed and an operational satellite relates mostly to schedule and availability. Tab. 1 details these differences.

Table 1. Differences between testbed and production satellite operations

Testbed	***Operational Satellite***
A single "ground station" (a dedicated host PC) is always available (except during sled hardware maintenance, which is performed on an as-needed basis).	A satellite communicates to various groundstations. An available transit between a satellite and a ground station depends on the ground station location and satellite orbit.
The contact schedule is generated by the periodic scheduler, and operates on a regular period. This schedule can be customized to contact frequency and time.	Contact schedule is a subset of available transits. The production scheduler manages time sharing between the fleet of production satellites and ground stations. Due to this, increasing contact frequency and time for a satellite is a more involved process.
Radio link between ground and satellite is reliable and consistent.	Radio link quality depends on ground station -to-satellite transit (elevation, etc.) and satellite pointing.
Uploading of available customer provided packages is prioritized over downloading of data; therefore, uploads complete sooner.	Uploading of customer provided packages is time-shared with downloading of data during the mission, and therefore may take longer to complete.
Debugging problems is naturally easier. Physical connectivity to the satellite can be used (by Spire engineers only) to drill into and identify bugs, making testbeds ideal for working out issues before placing onto an operational satellite.	Compared to testbeds, access to the satellite is more limited, and the retrieval of debug data will entail a longer latency.

5. CONCLUSION

With support from the ESA’s Earth Observation Science for Society Programme and in cooperation with Φ-lab, Spire has created ‘*Brain in Space*’.

Brain in Space consists of much of the same systems that can be found on a flight version of our LEMUR spacecraft including communications and power systems, processing payloads, onboard computer and other utilities. In addition to the standard LEMUR systems and computing payloads (Zynq Ultrascale+, Jetson TX2i), the testbed includes a Google Coral, a Jetson Nano and an UP Myriad X module. The *Brain in Space* testbed offers an environment with embedded AI/ML chips for users to schedule, upload and test their own AI applications and frameworks. It allows to test the ability of a chip to enable the running of AI algorithms, to perform, for example, time-critical missions, reduction of download bandwidth requirements and autonomous decision-making.

This testbed is the first step towards the completion of an AI chip-equipped satellite able to carry out a long tail of commercial applications.

REFERENCES

[1] Bryce Space and Technology, 2020, Smallsats by the Numbers, *https://brycetech.com/reports/report-documents/Bryce_Smallsats_2020.pdf*

[2] Sofiya Merenych, 2019, Modern artificial intelligence frameworks that make AI available to everyone. *https://clockwise.software/blog/artificial-intelligence-framework/*

[3] Xilinx Vitis AI, 2021, *https://www.xilinx.com/products/design-tools/vitis/vitis-ai.html*

[4] Xilinx, 2019, DPU for Convolutional Neural Network v1.0, DPU IP Product Guide *https://www.xilinx.com/support/documentation/ip_documentation/dpu/v1_0/pg338-dpu.pdf*

[5] Nefi Alarcon, 2020, Lockheed Martin and USC to Launch Jetson-Based Nanosatellite for Scientific Research Into Orbit *https://developer.nvidia.com/blog/lockheed-martin-usc-jetson-nanosatellite/*