SSC19-WKVIII-07

Wireless Bus Interconnects for Flexible and Reliable CubeSat Signal Integrations

Russell Trafford, Nicholas Gorab, Tanner Smith, Adam Fifth, John Schmalzel, Robert Krchnavek, and Sangho Shin
Henry M. Rowan College of Engineering, Rowan University
201 Mullica Hill Road, Glassboro, NJ 08028
+1(609) 972-6160, traffo17@rowan.edu

ABSTRACT

One of the largest, yet easily forgotten, aspects of constructing any complex system is the effort needed to integrate several subsystems. One common way to do this is to standardize the interface between subsystems, whether that is a physical standard, such as USB, or protocol standards, such as Wi-Fi and Bluetooth. In our previous implementations of CubeSat systems and subsystems we have found the PC/104 bus to be volume and mass inefficient while allowing too many potential conflicts to be considered a 'standard'. Our research proposes to implement a wireless, Bluetooth based, communication interface between subsystems to minimize the physical and logistical effort required to build CubeSat systems. By completely removing the need for physical/electrical connections between the subsystems, the barrier to entry for making custom modules for CubeSats can be lowered dramatically. Throughout the course of this research, the applicability of Bluetooth Low-Energy (BLE) utilizing the Generic Attribute (GATT) protocol is investigated.

BACKGROUND

Traditional CubeSat Interconnects

CubeSats consist of multiple interconnected subsystems which operate with regards to their own core area of the satellite system. To truly be interconnected, these subsystems require a means of communication to talk to each other. Commercially, this is popularly accomplished through the implementation of a PC/104 header [1]. This header includes physical channels for both communication and power, leading to its adoption as the de facto standard for Commercial Off-The-Shelf (COTS) subsystems. While PC/104 is a recognized standard interface, most COTS subsystems are only using the form factor and not the signal definitions. This in turn could lead to integration problems with multiple COTS systems, where a single pin might be utilized for more than one signal.

The PC/104 pin header is defined as a 4 x 26 pin header with 2.54mm (0.1 inch) pitch. When looking at the required landing area on a printed circuit board, this header takes up roughly 445mm^2, and this is without considering the height of the connector or any plastic surrounding it. The internal dimensions of a CubeSat following CalPoly's CubeSat standard must be less than 100mm by 100mm (total edge to edge dimension of the completed CubeSat). If a PCB were to be made to the maximum size of the CubeSat, this header would take up 4.45% of the available area on the board. In addition to this, the physical stacking header for each board measures roughly 8.5mm vertically, consuming significant space in the satellite stack [1]. For complex subsystems which require extensive hardware, reserving space for a header may prove to be troublesome and result in excessive design complications. The large size of this header also adds subsequent mass to the CubeSat, which may prove worrisome if the system is reaching the target weight threshold provided by the launch provider.

While the size of these headers in the grand scheme of a CubeSat might seem inconsequential, the inclusion of these headers effectively limits the number of different layers a CubeSat can have and can cause balancing issues with the center of mass. In "larger" CubeSats such as 3U, 6U, and 12U, this is truly not that large of an issue. But as the standard of the CubeSat extends down to 0.5U and the desire to make small, swarm satellites increases, this connector becomes a problem. Routing these headers also becomes an issue in smaller busses, leaving very little room to route critical signals throughout the system. The main motivation of the research is to better implement these types of systems, reducing the mass, volume, and PCB area requirement while maintaining the same functionality.

CubeSat System Architectures

With all the subsystems present on a CubeSat, ensuring proper communication is a paramount task. These subsystems need to coordinate to ensure that data is delivered to the correct subsystem in a timely manner. There are two main approaches used in the complex

systems: Centralized and Distributed architectures, as seen in Figure 1. Back when processing power and memory were limiting factors, a Centralized system architecture was used. This architecture utilizes a central controller, typically called the Command and Data Handling (CDH) system, to manage the communication between subsystems and in some cases control the subsystems. To reflect the dramatic decrease in cost of processing power, in this paper, CDH is still a central "brain" controlling the data flow, but each subsystem can have a brain of its own to better provide more local control. Some researchers have referred to this as a "Semi-Distributed" architecture. Within these architectures, there is no built-in method for subsystems to communicate with one-another outside going through CDH.

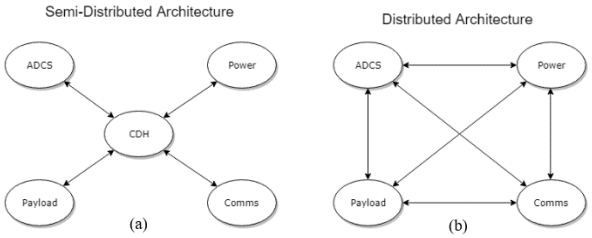

Figure 1: Architecture overview of (a) semi-distributed and (b) fully distributed architectures showing possible communication channels for each.

An example of this is CDH issuing a command to a payload subsystem to run an experiment. The payload processor will take care of all the functions necessary for the experiment, and then return the data to CDH once it is completed. CDH is not responsible for commanding the payload processor through each step of the experiment.

Distributed architectures are usually more complicated when compared to the semi-distributed ones. Instead of having a central brain of the satellite, each subsystem in a distributed architecture can perform tasks without intervention from a master device. Each subsystem possesses the ability to talk to any other subsystem on the satellite, removing the need for a CDH 'brain'. This requires each subsystem to act as both a slave and a master, as it can be receiving information and sending it depending on the state of the system. A major technicality of this system is developing a hierarchy of importance when dealing with communication. For example, a message informing a subsystem that the battery is critically low should be considered more important than experimental data and take precedence when being read.

Requirements for a CubeSat Communications Bus

Through this discussion so far, the PC/104 connector provides the main means for communication and power delivery within the system. For many reasons it is clear why a connector such as this would become the de facto standard within CubeSat COTS systems, but the question is how "one-size-fits-all" is this connector and what downsides could be improved. For a suitable replacement to be determined, a proposed communication and power delivery bus would need to:

- Provide a means for a CDH based central architecture to communicate and control other subsystems;
- Allow for multiple subsystems to communicate with one another in a distributed architecture;
- Provide power to multiple subsystems; and
- Have built-in flexibility to fit system designer's needs.

This would all also have to be done while reducing the amount of mass, volume, and overall hardware integration complexity required by using the PC/104 connector. In a typical COTS power supply system, there are typically only around 10-12 pins out of the 104 available pins. This research seeks to ideally replace as many of the remaining 90 pins with a wireless interface that has a minimum viable communications bus to fall back on.

WIRELESS INTERCONNECT MODEL

To circumvent these prevalent issues with traditional wired CubeSat interconnects, a wireless interconnect model is proposed. This is realized by removing all physical wire-based interconnects from the satellite and implementing a small, wireless communication device on each subsystem. The only remaining physical connections on the board should be for power and structure purposes. A simple redundant hardware communication channel may also be implemented in this design. This wireless model allows for all necessary communication channels to exist but removes the need for the large volume and mass inefficient PC/104 header.

System Considerations

For this wireless model to be applicable to modern CubeSat designs, it needs to recognize certain considerations for the system to be realized as a potential alternative to the physical interconnect model. These include the following concerns:

Physical Volume

As previously mentioned, the PC/104 header takes up considerable volume per subsystem PCB. As this is a significant detraction for the physical header, a wireless interface needs to be realized in a way which consumes less space in both two and three dimensions. This reduction in volume will allow subsystems to be stacked closer together, increasing the density of the satellite and providing more room for hardware [2].

Power Consumption

Introduction of a wireless interconnect system intrinsically consumes more power when compared to a traditional physical interconnect system. To ensure that a wireless interconnect system is possible, it must meet the CubeSat's power budget. Trade-offs with this extra power consumption need to be analyzed, along with preliminary power consumption data for the wireless technology selected.

Antenna Requirements

To communicate, some wireless modules will require an antenna to ensure proper communication. To remain physically smaller than the physical header, the footprint of the antenna needs to be identified. With different wireless technologies utilizing different sizes of antennas, this needs to be considered when designing the wireless interconnect system [2].

Cost

CubeSats are becoming increasingly popular for small institutions and Universities for testing and research. One of the major factors limiting these satellites from being launched is their prohibitive cost. Naturally, wireless communication will be more expensive when compared to traditional interconnects so the trade-off between price and functionality needs to be assessed. Additionally, the price will vary with different wireless communication methods, so this becomes a point of consideration when designing the system.

Redundancy Features

CubeSats are complex systems which require extensive redundant features to provide a long and successful lifespan [2]. Introducing a wireless communication method into this design adds a significant amount of complexity and provides more points of failure which can ultimately lead to a premature end of mission. To combat this, redundancies need to be built into this wireless model to ensure that failures can be accounted for and circumvented. This also includes the implementation of a bare-bones hardware backup communication line which is to be used in case of radio failure.

Benefits

The implementation of a wireless interconnect model possesses many benefits for both commercial and research CubeSats. The removal of the PC/104 header allows each PCB to be more subsystem-oriented, i.e. the elements on the board do not need to take into consideration the location of traces and the consumed volume which is prevalent when using the PC/104. The footprint of these wireless devices is significantly smaller than that of traditional interconnects, so more hardware can be placed on the board while maintaining the same CubeSat form factor.

This removal of the traditional interconnect system also increases the total modularity of the CubeSat system, allowing for a more flexible architecture. Wireless devices may scan through all available connections on-board the satellite to determine which subsystems are present, and then act accordingly by issuing certain commands to ascertain more subsystem specific information. This process promotes a hot swap methodology of constructing a CubeSat where subsystems can be interchanged rapidly during development without any major changes needed in the overall framework of the system. With this methodology, significant operational testing can be performed on the system without the need to physically connect subsystems. If issues arise in the design, this also prevents the potentially damaging process of deconstructing the CubeSat stack to access each individual subsystem. Wireless interconnects also allow the satellite to maintain a much denser design when compared to traditional CubeSats. The physical PC/104 header acts as a potentially wasteful spacer between subsystems, consuming roughly 8.5mm of vertical space per PCB. This causes the header to become the prominent constraint when considering subsystem selection, limiting a 1U CubeSat to roughly 4-6 subsystems. Removal of this header will allow subsystems to be placed much closer together, eliminating empty space on the satellite while at the same time providing a means to increase the capability of the system.

Drawbacks

One of the biggest drawbacks with any wireless system is the possibility of interruptions. There is a certain reliability that is intrinsic to using copper to connect different systems. It typically takes some sort of mechanical event to disconnect a wire in a satellite, although corrosion can be caused from different sources.

Depending on the protocol selected, the bandwidth required, and the quality of the components used, there are many sources of error which could arise. For example, since the transceiver which is located on each PCB plane is made of silicon, there is now an increased sensitivity to radiation environments. Other issues such as thermal sensitivity in the oscillators and other components could lead to radio failure. In some terrestrial systems, resetting a radio or working around this failure could be done. In a space system; however, having just one component in a system aimed to connect the subsystems could easily lead to a total system failure and loss of mission.

What this all means is that in a satellite system, there should still be some copper connecting these subsystems somewhere. But this does not mean there needs to be 104 of them. There needs to be some sort of backup system in place to minimize the chance of total system failure. This would also allow for a subsystem to opt to turn its radio off in the case that it might get stuck in a transmit mode or some unknown state. This does present much more of a software complexity as well since this priority system will need to be put into place. However, this still could be implemented in a way to reduce the integration issues and requirements posed by the 104-pin connector.

IMPLEMENTATION

Bluetooth Low-Energy (BLE)

After careful consideration of multiple wireless communication methods, BLE was chosen for further analysis. This communication method, designed by the Bluetooth Special Interest Group, is based on traditional Bluetooth but is geared more towards power-conscious designs. Since a wireless communication bus will require power to operate from what could already be a tight power budget, the Low Energy version of Bluetooth is more ideal for operation on CubeSats [4].

Generic Attribute Protocol

The Generic Attribute (GATT) protocol can be used extensively in CubeSat applications. This protocol is designed to send small packets of data between a single client and multiple server nodes. The GATT protocol contains a hierarchy of information, starting with a single profile that contains multiple services consisting of characteristics [3]. GATT characteristics store specific information relating to the service using the following attributes: characteristic value, characteristic declaration, client characteristic configuration and characteristic user description. Each attribute for a characteristic also contains the following properties: handle, type and permissions. For clarification, a general architecture for the GATT protocol can be seen in Figure 2.

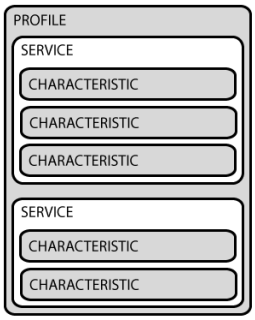

Figure 2: Overview of the GATT Protocol Architecture [4]

A single device will consist of a profile, with each of the areas of interest being considered a service. Each of the services can contain multiple characteristics, which are data being collected through the system. One key low-power element of this architecture is the ability for each characteristic to notify the master device when is it updated, allowing the master to be in a sleeping state until something happens. With all these considerations in mind, BLE and GATT will be applicable for low-power close range communication systems [3]. Hardware wise, a BLE IC has an extremely small footprint for integration onto PCBs, and due to the close-range communication distance, little is needed for antennas. Additionally, there are existing modules which contain all the components needed for a complete BLE package. This often consists of a powerful processor, integrated antenna, and dedicated driver software. All of this allows for simple integration and rapid prototyping of these BLE devices. An example of this Characteristic Architecture can be seen in Figure 3.

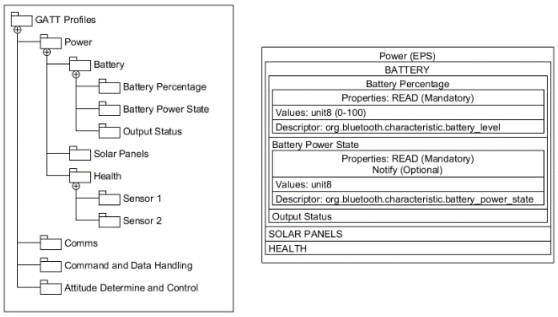

Figure 3 GATT Characteristic Architecture for an example CubeSat.

Overall, the GATT protocol offers system-wide benefits while decreasing power consumption, freeing up more space for other operations and improving the efficiency of the system. This protocol capitalizes on the benefits that come with BLE implementation while providing performance that meets, if not exceeds, traditional hardware interconnects.

ESP32

For the initial testing of this concept, the Espressif ESP32 was chosen as the main development platform. This module contains an 80MHz dual-core ARM processor, and on-board peripherals including a full 2.4GHz and 5GHz Wi-Fi system and BLE capable Bluetooth Radio. While this module is available as a board mounted package, as seen in Figure 4, a development board with included FTDI serial converter and 5V regulator was used. This device can be programmed to be either a BLE Master or Slave device and contains enough CPU power to possibly run subsystems on its own.

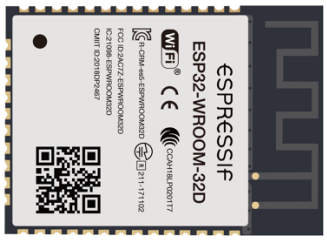

Figure 4: ESP32 WROOM Bluetooth and Wi-Fi Module

The size of this module is 18mm by 25mm (450mm^2), excluding any area required for antenna clearance or placement. This is smaller than the area required by the PC/104 header, but the height of the module is only 3.1mm, making the volume required extremely small. If this module could help reduce the need for the tall headers typically used, this could lead to much denser CubeSat configurations. It should be noted, however, that since this and any other module with an on-board antenna will most likely need a cooper keep-out around the device, there is board and routing space lost on both sides of the PCB.

Minimum Viable Communication Network

To develop a working framework a minimum viable network needs to be created. This system shall consist of the absolute minimum required software and hardware necessary for the system to operate effectively. Using BLE as a basis for wireless communication, this minimum viable communication network must include at least one BLE-enabled radio per subsystem. This basic inclusion ensures that each subsystem will be able to connect to the master device remotely.

For the initial development of the BLE based communication bus a Semi-Distributed architecture was chosen. Three ESP32s were used to represent the CDH, Ground Communications and Payload subsystem. Each subsystem had local processing, but inter-subsystem communication was coordinated through CDH. This can be seen in Figure 5.

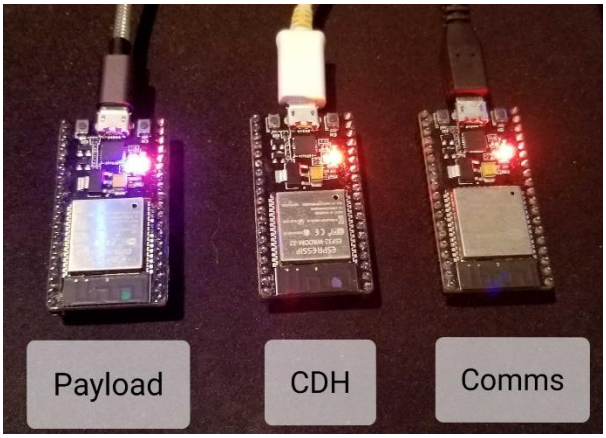

Figure 5: Experimental setup for initial implementation.

EXPERIMENTAL RESULTS

Initial Experiment

As a proof of concept, this initial experiment aimed to demonstrate this system's capability to transmit messages across multiple wireless devices, mimicking the traditional wired interconnect system. To do this, multiple facets of a fully-fledged CubeSat mission needed to be simplified for rapid implementation into the simulated space. To mimic a ground station, a serial connection between a laptop and the designated Comms ESP32 was established to emulate a more powerful transceiver typically found on flight ready CubeSats. Additionally, very basic services were implemented for each subsystem to provide a simple but effective means of testing system functionality. For example, the Comms subsystem had a "new message" service while the payload subsystem had an "LED status" service.

To ensure proper connectivity between all the devices utilized in this experiment, each ESP32 system was initialized with a specific Device ID and name which the CDH subsystem could easily recognize. The experiment

began with CDH powering on and searching for potential server devices to pair with. Subsequently, the Comms and Payload subsystems were initialized to begin broadcasting a signal informing other devices they are ready to be paired with. CDH then proceeded to scan through the available BLE devices until one is recognized. When a known device was found, CDH would initiate pairing and then the two devices would begin transmitting information. This experimental setup can be seen in Figure 6.

Figure 6: Block Diagram of system connectivity for initial experiment

To demonstrate that the system is operating as expected, a characteristic was established on the Payload subsystem which controlled a single on-board LED. The goal of the experiment is to manipulate the state of the LED with access limited to the ground station laptop. During testing, a serial command was sent from the laptop to the Comms subsystem as a mock radio transmission. From there, the Comms system updating a service pertaining to incoming information, populating it with the newly received message. Upon updating this value, the CDH subsystem was notified and was able to access and read the transmitted message. Following the commands issued in the message, the CDH accessed a characteristic possessed by the Payload subsystem controlling the LED. Once CDH updated this characteristic, the Payload subsystem was notified and acted on this change, toggling the state of the LED. The responses of the subsystems can be seen in Figure 7.

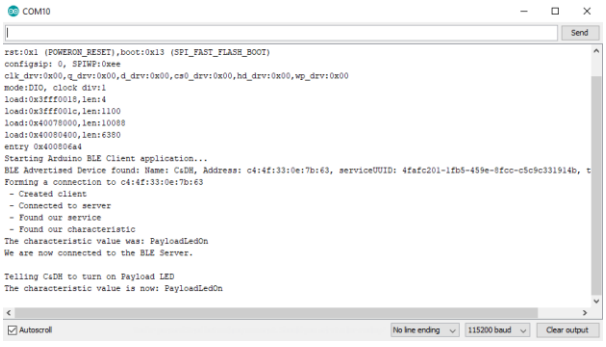

Figure 7: Debug terminal for COMMS module communicating to the CDH and Payload.

Although this demonstration was simple; it effectively demonstrated the basic capabilities of a wireless interface. The flow of communications found in this test are almost identical to that of a fully realized satellite, with the most major difference being that the profiles found on a flight-ready model will encompass much more information about the subsystem.

DISCUSSION AND FUTURE WORK

Expanding and Implementing the Wireless Communication Bus

One of the benefits of using a communication protocol like Bluetooth is the ability for plug-and-play devices. At a high-level, if the GATT protocol is standardized for the subsystems, then there should not be an issue in expanding the network. This does require further development on standardizing the interface, which should be done regardless of the medium used (wired or wireless). While in a simple example presented in this paper may seem simple to establish, once more subsystems are introduced, there will need to be some conformity to allow them to talk to one another.

Additional testing with the BlueNRG 2

Another benefit of using Bluetooth is the abundance of BLE-enabled ICs and SoCs, allowing multiple subsystems to have different radios without sacrificing communicability. For example, the ESP32 provides a powerful and adaptable solution for a Wireless CubeSat bus, but it comes at the cost of PCB area and power. Other modules exist to be integrated into systems which may already have a CPU and are much smaller footprint. An example of the is the BlueNRG 2 by Texas Instruments. This BLE SoC can handle the required GATT protocol while being the size of a QFN 32 or 48 package [5]. This could lead to a further reduction of PCB area compared to the PC/104 header to almost less than 5% of the original. Depending on the module or SoC chosen, there may have to be implemented an antenna or some other elements outside the chip.

Backup Communication Bus

The wireless implementation of the communication bus within the CubeSat did provide the plug-and-play utility desired while supplementing the need for a dense interconnect. There are, however, certain aspects which could not be replaced. Power still must be delivered to the subsystems which still requires some form of common header to be placed or wires to be manually added. To help accommodate for a soft or hard failure in the radio, there needs to be a backup hardwired communication bus which can connect CDH to other critical systems. The implementation and requirements of this bus would change based on the architecture of the satellite.

In a Centralized or Semi-Distributed architecture, CDH would need to be able to communicate to each subsystem and could act as a Master on the bus. This lends itself to a protocol such as SPI, I2C, or CAN. CAN would provide noise resilience due to the differential topology used and is robust against connected devices failing. SPI and I2C would provide an interface more commonly found in processor peripherals and would provide an addressing scheme. However, in SPI the number of pins required would have to vary based on the number of planes attempting to be connected, where I2C would have to very carefully have its Addresses managed. The analysis of each of the communication buses in a space environment is another area of research which could be done.

CONCLUSION

The PC/104 connector, which has become the de facto standard in many CubeSat COTS subsystems, consumes a significant amount of space within a CubeSat system, and has proven to be both mass and volume inefficient. By providing a wireless interface for a considerable portion of connections between subsystems, most pins used in the PC/104 header can be omitted from a design. This paper outlined a proof of concept experiment where an ESP32, acting as a CDH module, paired with and managed COMMS and Payload subsystems. Operating in a simulated launch environment, a mock ground station was able to send Payload commands to Comms and have the wireless subsystems communicate effectively to achieve the optimal outcome. These results show that there is a path forward for the use of a wireless module to replace many of the connector pins and provide an opportunity for more interoperable COTS systems.

Acknowledgments

This work was sponsored in part by the NJSGC grant (no. 5860) and the NSF GAANN Fellowship.

References

1. J. Bouwmeester, M. Langer, and E. Gill, “Survey on the implementation and reliability of CubeSat electrical bus interfaces,” *CEAS Sp. J.*, vol. 9, no. 2, pp. 163–173, Jun. 2017.
2. W. Lan, “The CubeSat Program, Cal Poly SLO CubeSat Design Specification (CDS) REV 13 Document Classification X Public Domain ITAR Controlled Internal Only.”
3. Texas Instruments, “Generic Attribute Profile (GATT) — BLE-Stack User’s Guide for Bluetooth 4.2 3.01.00.05 documentation,” 2016. [Online]. Available: http://dev.ti.com/tirex/content/simplelink_cc2640r2_sdk_1_40_00_45/docs/blestack/ble_user_guide/html/ble-stack-3.x/gatt.html. [Accessed: 11-Jun-2019].
4. K. Townsend, “Introduction to Bluetooth Low Energy.” Adafruit Industries, p. 8, 2019.
5. STMicroelectronics, “BlueNRG-2 Bluetooth low energy stack,” no. STMicroelectronics, p. 175, June 2018.