A Standardized, Distributed Computing Architecture: Results from Three Universities

Michael Swartwout
Aerospace Systems Laboratory, Department of Mechanical & Aerospace Engineering
Washington University in St. Louis
Campus Box 1185, St. Louis MO 63130; (314) 935-6077
mas@me.wustl.edu

Christopher Kitts, Pascal Stang
Robotic Systems Laboratory, Department of Mechanical Engineering
Santa Clara University
500 El Camino Real, Santa Clara, CA 95053
ckitts@me.scu.edu

E. Glenn Lightsey
Department of Aerospace Engineering and Engineering Mechanics
University of Texas at Austin
1 University Station, C0600, Austin, TX 78712-0235
lightsey@csr.utexas.edu

Abstract. At the 16th AIAA/USU Conference on Small Satellites, researchers at Santa Clara University (SCU) proposed a distributed computing architecture for small or multi-spacecraft missions. This architecture extended existing I²C, Dallas 1-wire and RS232 data protocols and was adaptable to a number of microcontrollers. Since then, that architecture has been implemented on six university-class space missions at three different universities.

As "early adopters", these universities had the typical challenges of working with a new, evolving standard and adapting the standard to their hardware and mission needs. Each faced additional, program-specific challenges related to project size, scope and infrastructure as well as the student background/training. Still, because of this architecture, every school saw three improvements: accelerated integration and training of new students; rapid modifications of existing systems; and school-wide collaboration among robotics projects.

This paper reviews SCU’s distributed computing architecture, discusses the details of its implementation at all three universities, and provides lessons learned/lessons applied to six spacecraft programs: Akoya-A/Bandit-A & Akoya-B/Bandit-C at Washington University in St. Louis, EMERALD & ONYX at SCU, and FASTRAC and ARTEMIS at the University of Texas-Austin. The merits of adopting this architecture as a standard for university-class spacecraft are also reviewed.

INTRODUCTION

Distributed computing architectures offer numerous advantages in the development of complex devices and systems. These advantages include well-defined interfaces, flexible composition, streamlined integration, straightforward function-structure mappings, standardized components, incremental testing, and other benefits. For these and other reasons, the Robotic Systems Laboratory (RSL) at Santa Clara University (SCU) has developed a distributed command and data handling (dCDH) system for their on-board computational requirements. These systems are typified by the notion of “smart subsystems” such that each subsystem or functional module has its own microcontroller capable of providing local control and execution of processes specific to that particular functional piece of equipment; subsystem microcontrollers communicate with each other through a linear data bus topology.

The RSL implementation uses the commercial I²C and microLAN 1-wire linear bus specifications for connections between microcontrollers and other components; in addition, several microcontroller motherboards and software code libraries have been developed for specific subsystems and devices using this implementation. Furthermore, custom command and data protocols crucial to the application of the dCDH system across a varying set of applications have been developed. Finally, advanced applications generally applicable to a wide range of systems have been built to include a file system, a command scheduler, and an expert system.
Apart from the technical benefits from the use of such an architecture, RSL discovered significant educational benefits. For example, one overriding advantage of a dCDH architecture over a conventional “main computer” architecture is the ability to focus subsystem design teams on achieving functional capability. This means that each subsystem design group becomes responsible for specifying, developing, and verifying the performance of the entire subsystem to include all of the mechanical, electrical and processing-oriented components of that system. In addition, technical subsystem interfaces are naturally specified and align themselves nicely with the managerial, task-oriented focus of each subsystem team. By contrast, deliverables in a “main computer” architecture often are not functionally driven, and the architecture itself can obscure who is responsible for defining and ensuring functionality at the subsystem level. These influences on the educational experience led RSL to employ dCDH architectures for complex systems with increasing frequency. Much of the original dCDH development took place for RSL’s EMERALD nanosatellite project, although it was quickly adapted for other robotic systems ranging from underwater robots to unmanned aerial vehicles.

In 2003, student satellite projects at Washington University in St. Louis (WU) and the University of Texas at Austin (UT) were selected for the AFRL/NASA/AIAA University Nanosat-3 (NS-3) design competition. Independently, each PI opted to subcontract RSL to provide the dCDH system for their vehicles, and thus, the RSL dCDH system became the de facto standard across three University Nanosat programs. The UT and WU teams earned 1st and 2nd place, respectively, in the NS-3 competition. Subsequently, all three universities were selected to participate in the Nanosat-4 (NS-4) competition.

As “early adopters” of the dCDH standard, these three universities had the typical challenges of working with a new, evolving standard and adapting the standard to their hardware and mission needs. Each faced additional, program-specific challenges related to project size, scope and infrastructure as well as student background/training. Still, because of this architecture, every school saw improvements in three areas: accelerated integration and training of new students; rapid modification to existing systems; and school-wide collaboration among robotics projects.

The next section provides a technical overview of RSL’s dCDH system, followed by case studies of dCDH implementation at WU, SCU and UT, respectively, with an emphasis on university-specific conditions and their effect on adoption and use of the dCDH standard. While the term “standard” is used to describe the RSL dCDH architecture, the authors do not claim that it is at the point of being a true standard. Rather, the RSL dCDH hardware and protocols are a working “standard,” the details of which will be described in this paper. In the conclusion, the RSL dCDH architecture will be evaluated and its potential for use as a broad standard for satellite command & data handling will be addressed.

OVERVIEW OF RSL DCDH STANDARD

The RSL dCDH system was introduced at the 16th AIAA/USU Conference on Small Satellites. It consists of specifications for data/power harness, command/data protocol and microcontroller system. An overview of that system is provided here, emphasizing the hardware, software and architecture updates since that paper was written.

In the context of this paper, distributed computing refers to computational decentralization across a number of processors which may be physically located in different components, subsystems, systems, or facilities. These processors may be general-purpose computers with data/application sharing capabilities (e.g. a typical personal computing network), they may have an architecture that enables collaborative processing focused on a specific task (e.g. parallel computation), and/or each may be optimized to efficiently execute particular tasks or control specific subsystems (e.g. smart peripherals).

The dCDH is an example of a linear bus distributed computing architecture: a standardized, shared, linear data bus to which all subsystems are connected (Figure 1). The use of a linear bus for the computing architecture leads to a simple and relatively small data bus that promotes standardized methodologies for interfacing at a range of levels. At the signal level, standardization of physical interconnections is a natural objective. At higher levels, standardization of data communication protocols for arbitration, error handling and other functions is a straightforward strategy. For the student satellite projects, one of the most important achievements of a linear bus is component-level modularity; components can be easily connected, disconnected, replaced, swapped and/or upgraded in a rapid and transparent manner.

---

**Figure 1. Linear Bus Architecture**
While distributed computing systems and their advantages are common in modern personal computer, consumer product, industrial automation, automotive, and other industries, they have been slowly adopted in the satellite industry. Past initiatives such as the NASA/JPL X2000 development program have recognized the advantages of distributed computing strategies and are beginning to develop such systems for space flight.

System Design

The RSL team sought to develop a linear bus system with a balance of simplicity and cost while also a) providing performance capable of supporting research-quality microsatellite instrumentation, and b) being feasible for use by student design teams.

Protocol. Given the desired balance of simplicity, reduced wiring, and speed, the Inter-Integrated Circuit (I\(^2\)C) protocol was selected. I\(^2\)C is used in audio/visual equipment, on PC motherboards, and in “smart” batteries. This protocol easily scales to networks of arbitrary size and includes built-in support for multiple masters. However, it does not provide existing, suitable high-level communication protocols.

I\(^2\)C is a synchronous serial protocol that uses only 2 wires, one for data (SDA) and one for clock (SCL). It also requires a common ground. It operates at 100 kbps in standard mode. Faster modes (400kbps and 3.4 Mbps) are also specified, but fewer devices support these modes. The protocol is specified to a fairly high level from reading and writing to multi-master support and arbitration.

Communication is always initiated by a master, which also drives the clock. Each I\(^2\)C message consists of an arbitrary number of 9 bit “words.” These words are 8 bits of information plus an acknowledge.

High Level Messaging. I\(^2\)C standardizes many layers of the communication protocol. However, because it does not specify any data integrity checks, the development team has developed a simple error checking message format involving checksums and reply packets.

This approach permits subsystem designers to choose the exact format for commands and data relevant to subsystem tasks. In addition, a variety of standard commands are defined for controlling common tasks for all subsystems on the bus. These include functions for checking subsystem status, synchronizing time, and querying the subsystems for a list of defined commands (help function). This last feature is very attractive because it allows subsystems to change and expand their functionality without requiring extensive system knowledge by operations personnel.

Telemetry Bus. In addition to the higher bandwidth I\(^2\)C command/data bus described above, the current architecture also has a low bandwidth telemetry bus built on the Dallas 1-wire protocol. Dallas microLAN supports an arbitrary number of devices connected with a single bi-directional data line. It is an asynchronous protocol that operates at 14.4kbps. Like many standardized serial protocols, a wide range of off-the-shelf components is available. Of particular interest for satellite telemetry are the temperature sensors (DS18B20) and analog-to-digital converters (DS2450, quad channel 8-bit accurate).

One particularly appealing characteristic of the Dallas technology is that each individual device has a completely unique 64-bit ID number for addressing. This eliminates the need for external address configuration pins, or chip select lines when connecting multiple devices to the bus. Adding another temperature node, or A2D node truly is as easy as just connecting 3 wires: power, ground, and the data line.

Design-level Fault Tolerance. Since the nodes of the I\(^2\)C bus are connected as open-collector, it is possible for an errant subsystem to hang the entire data bus. In other aerospace systems (airplanes, larger satellites, etc.), a common solution is to incorporate the additional complexity of multiple redundant back-up data buses. For the power/mass/volume-constrained university-class vehicles, such redundancy is precluded. Instead, such bus faults are handled by controlling subsystem power over the Dallas microLAN, thereby allowing a component to be reset or completely shut down in the event of a component failure that results in an I\(^2\)C bus lock-up. An analog switch at each I\(^2\)C connection ensures that the subsystem circuitry is isolated from the I\(^2\)C bus. Therefore, if a controller detects that the I\(^2\)C bus is hung, it can selectively turn off power (using Dallas) to successive subsystems until the fault is eliminated.

AVR-SAT Microcontroller

For the distributed architecture to work effectively, it is not necessary for all subsystems to use the same processor; adherence to the data and wiring protocols ensures that processor differences are transparent to all other subsystems on the bus. But to simplify baseline subsystem development and to leverage economies of scale, RSL converged on a very mature design for the standard microcontroller system, called the AVR-SAT. Shown in Figure 2, this 3-module system is based on the Atmel AVR processor family. The 3-module design allows users to mix/match/upgrade capabilities as required by a specific application:
• The system motherboard provides an array of memory and I/O support to the processor. Specific communications supported include wired local connections as well as both short- and long-range wireless connections. The board also provides expanded memory as well as subsystem-level power processing.

• The processor board currently uses the AVR family Atmega128 processor, although a wide range of other processors are compatible with the design. The board also supports expanded RAM, two serial ports and 32 I/O lines.

• For space applications, a latch-up board completes the 3-board set by providing circuitry for resetting system power when radiation latch-ups (indicated by sudden, high current draws) occur.

Table 1 provides a more comprehensive description of the technical features of the AVR-SAT system.

The AVR-SAT system is an extended version of the processor and i/o design of the AVRmini board, which is used as part of the undergraduate MECH 143 / ELEN 123 Mechatronics course at SCU. This allows students to rapidly learn the new features of the board in order to exploit its advanced capabilities for more complex projects. Furthermore, the AVR-SAT system is size and power compatible with the university CubeSat specification, which is being used by more than 60 universities worldwide. SCU’s AVR-SAT is the only academic implementation of a standard dCDH design in this community; there are two commercial implementations.

Table 1. AVR-SAT dCDH System Features

<table>
<thead>
<tr>
<th>System Components</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor Board</td>
<td>Powerful ATmega128 Processor Module&lt;br&gt;-16MIPS processor, 3mW/MIPS&lt;br&gt;-128KB FLASH, 512KB RAM&lt;br&gt;-General I/O (32 pins available to user)&lt;br&gt;-Dual serial ports&lt;br&gt;-I²C bus (Inter-Integrated Circuit Bus)&lt;br&gt;-SPI bus (Serial Peripheral Interface bus)&lt;br&gt;-Advanced power management&lt;br&gt;-Hardware Dallas 1-Wire support</td>
</tr>
<tr>
<td>Motherboard</td>
<td>Structural interconnects for AVR-SAT system&lt;br&gt;Built-in short-range radio link&lt;br&gt;Enables local distributed data networks&lt;br&gt;Direct Support for Long-range Radio Links&lt;br&gt;I/O Backbone Bus for Intra-Subsystem Comm &amp; Control&lt;br&gt;Used for command and control of payloads/circuits within the subsystem&lt;br&gt;On-board Non-Volatile memories (up to 512KBytes)&lt;br&gt;On-board high-efficiency power system (optional)&lt;br&gt;Allows subsystem to operate without a supporting power system&lt;br&gt;Allows direct connection to satellite batteries&lt;br&gt;Emerald Data and Power Buses</td>
</tr>
<tr>
<td>Radiation Protection Board</td>
<td>Dual power control and latchup protection modules&lt;br&gt;Allows subsystem power switching and measurement&lt;br&gt;Automatic overcurrent cutout protects against latchup&lt;br&gt;Optional automatic overcurrent reset for stand-alone operation</td>
</tr>
</tbody>
</table>
**Wiring Harness**

Given the adoption of the AVR-SAT as the standard microcontroller for the linear data bus, a standard wiring harness naturally evolved. As seen in Figure 2, the AVR-SAT motherboard has two DB9 connectors: one for power and one for data. These connections have been defined such that an arbitrary number of AVR-SATs (or other pin-compatible devices) can be safely harnessed in parallel. Not only does this greatly simplify and reduce the fabrication and certification of the wiring harnesses, but it enables rapid integration and safe swap-out of subsystems.

**Predicted Benefits**

In their 2002 paper, the RSL dCDH designers predicted that four benefits would be realized.

- In the design stage, the distributed system will simplify the command and data flow within the satellite by clarifying which specific component is responsible for each task and what information exchange is required to initiate the task; ideally, each functional block in the system’s signal flow diagram will map directly to a physical box on the satellite. Furthermore, this characteristic will allow easy hierarchical scaling of the functional and data flow designs for multi-satellite missions and comprehensive space/ground segment systems.

- During development and integration, the distributed architecture will promote the rigorous and independent test/verification and the controlled, incrementally integration of each subsystem/component. Such an achievement will assist in de-coupling the reliance of one subsystem’s development on the operation of another (e.g. such as the central processing unit in most centralized architectures). Furthermore, with network bus “gateways”, components can be easily integrated remotely using TCP/IP or other protocols to bridge and test subsystems being developed at different locations.

- On-orbit, the distributed architecture will simplify resource sharing (e.g. computational power, memory, etc.) among components, subsystems, and even satellites. It also promotes fault tolerance since computational functions can be supplied by other units (possibly even located in other spacecraft or on the ground) in the event of component outages.

- When exploited in a multi-satellite mission, the distributed architecture will allow components deployed across multiple satellites to interact in much the same manner as those within a single satellite. This has significant implications in the simplification of collaborative processing schemes both at the conceptual and implementation levels.

The experiences of three universities in adopting the dCDH standard will be presented in the three sections that follow, and these predictions will be evaluated in the conclusion.

**IMPLEMENTATION AT WU**

The Aerospace Systems Laboratory (ASL) at Washington University is part of the Department of Mechanical & Aerospace Engineering; it is staffed by two research faculty, less than a half-dozen graduate students and approximately 20 undergraduates drawn from mechanical, aerospace, computer science, electrical engineering and other majors on campus. The faculty and graduate students lead the research activities with the undergraduates performing most of the system design, fabrication, test and operations of laboratory vehicles. Undergraduates participate through capstone design courses, independent study projects, a one-credit seminar, paid summer internships and as volunteers. On average, an undergraduate will participate for two years, with a core team of 6-10 students active for three or more years.

The ASL projects relevant to this paper are the Akoya/Bandit missions for the NS-3 and NS-4 programs. Under NS-3, ASL developed Akoya-A and Bandit-A, a 28-kg mothership and 2-kg daughtership, respectively; the primary mission was to demonstrate automated deployment, inspection and redocking by Bandit-A. The Akoya-A engineering demonstration unit (EDU) is shown in Figure 3, the Bandit-A prototype dock is in Figure 4, and the air-bearing demonstration version of Bandit-A is in Figure 5.

![Figure 3. Akoya EDU During Integration](image-url)
As shown in Figure 6, Akoya-A was designed to have eight AVR-SATs on the data bus; the Bandit-A service vehicle carried a ninth (via a wireless link), and the ground system has a tenth microcontroller (an AVRmini) to seamlessly link the spacecraft data bus to the ground systems. Average communication loads on the data bus are under 4 kbps, with occasional increases to a few dozen kbps for transferring files or images.

As the NS-3 activities ended, ASL was selected for the NS-4 competition. The Akoya-B/Bandit-C mission is a modified version of the Akoya-A/Bandit-A mission, with a stronger emphasis on reliable/autonomous Bandit operations and a simplified Akoya configuration. As of the writing of this paper, the Akoya-B spacecraft carries six AVR-SATs and the Bandit-C service vehicle one.

**Motivations for Adoption of RSL Standard**

At the start of NS-3 activities, the authors observed that mechanical & aerospace engineering students at WU exhibited an almost pathological fear of electronics; this aversion is in stark contrast to students at SCU and other universities. Part of this can be attributed to WU’s lack of a mechatronics program and small electrical & computer engineering population. Thus, there were few mechanisms to equip students to work with their electronics and CDH systems, and even resistance to learning to work with such devices.

ASL student teams had struggled to integrate a COTS PC104 CDH architecture into previous robotic projects. In addition, the ASL chief engineer had been the project manager for the Stanford/USNA Sapphire student spacecraft (NO-45) and as such had experienced the negative aspects of a “main computer”, student-developed CDH architecture: tremendous schedule delays, unusual performance, and difficult integration. For all of these reasons, the ASL team needed an existing CDH system geared towards student projects. The perceived benefits of the RSL dCDH system (modular development, rapid integration and troubleshooting, “entry-level” training) made it an obvious (and perhaps, the only) option.

**Adoption Process**

In May 2003, ASL sent two undergraduates to a dCDH training seminar at SCU; they returned with AVRmini hardware and development software. These students spent the next two years as the primary CDH programmers/troubleshooters for ASL. They organized several AVR-SAT training sessions for ASL undergraduates and, as subsystem development and integration ramped up, these students took on the role of dCDH mentors to the rest of the team.

Subsequent to the training seminars, ASL maintained frequent e-mail and phone contact with RSL developers for additional troubleshooting. There were several other face-to-face meetings at conferences. In October 2004, RSL released the initial version of the AVR-SAT, followed by the final version in December 2004.

**Assessment of Standard**

The most obvious benefit of ASL’s adoption of the RSL dCDH standard is in education: in the nine months from the arrival of the AVR-SAT microcontrollers to the writing of this paper, there has been a significant increase in the number of ASL students capable of programming microcontrollers. In September 2004, there were three capable students and in June 2005, there were ten (with four more to be trained that summer). Indeed, the pathological fear of electronics among ME/AE students seems to be waning.
Furthermore, familiarity with microcontrollers has spilled over into more rapid integration and development of the spacecraft subsystems. The communications, power and attitude control subsystems were developed in parallel and quickly integrated over a matter of weeks by student volunteers.

Students who have worked with the dCDH standard and AVR-class microcontrollers are adopting them for other robotics projects at WU. For example, a graduate student in the CSE department used several AVRminis and the RSL dCDH standard to rapidly integrate several sensing and actuating devices to their SMRT-HumV outdoor mobile robot, including a GPS receiver, laser rangefinder and a student-built motor-driven sensor deployment device (Figure 7). As expected, the adoption of the RSL dCDH standard by other WU projects has increased the pool of students capable of working with the microcontrollers, as well as improved knowledge capture and peer-to-peer mentoring.

In May 2005, ASL faced its largest-ever turnover challenge: all of the core Akoya-A/Bandit-A students would leave campus for the summer, “lost” to internships or full-time employment. Of the eight undergraduate summer interns, four were completely new to the program and only two had any previous microcontroller experience. However, in less than two weeks, this completely new group of students was able to assemble and operate the Akoya-A and Bandit-A EDUs from their component pieces and even begin software upgrades – all with little to no coaching from the previous student teams. In a similar manner, the RSL dCDH system was demonstrated to be easy to learn and implement during Akoya-A integration, when two untrained ASL team members were able to learn the system and contribute functional controller code in a span of two hours.

Figure 6. Akoya System-Level Block Diagram (AVR-SATs are diamonds)

Functionally, the RSL dCDH standard and accompanying AVR-SAT microcontroller has met all of ASL’s computing and data handling needs. Admittedly, neither the Akoya nor the Bandit vehicles tax the performance limits of the system. Still, ASL has functionally demonstrated the value of a distributed architecture in two ways. During initial integration of

Figure 7. SMRT-HumV
the Akoya-A EDU, it was more convenient for developers to work on the scheduling and expert system software apart from the main spacecraft. Instead, both of these spacecraft subsystems were “integrated” as data bus nodes on the ground station side of the system; the physical location of the microcontrollers running those spacecraft functions had no effect on the behavior or integrated performance of those subsystems. Similarly, both the Bandit-A service vehicle and ground controlling station operate as nodes on the Akoya-A data bus, enabling transparent operation of Bandit-A from the ground.

The RSL dCDH standard has also improved the process by which the Akoya-B/Bandit-C vehicles are developed. The Akoya-A EDU is being kept in the ASL lab as a training/development testbed for new students. As Akoya-B subsystems are developed, they can be integrated into the Akoya-A EDU and demonstrated. This ability further improves the incremental and parallel development of Akoya-B subsystems, as the ASL team can test each part in an end-to-end system before the Akoya-B EDU is complete.

The biggest challenge ASL faced regarding the standard was the evolving nature of the hardware; when the original WU-SCU subcontract was awarded, neither the AVR-SATs nor the AVRmins existed; the subcontracted system used a PIC microcontroller. But AVRmini and AVR-SAT upgrades were so much better-suited to the Akoya-A/Bandit-A mission that the ASL team opted to delay integration until their development. However, the Fall 2004 delivery of the AVR-SATs hampered completion of the Akoya-A EDU (which was to be delivered to the January 2005 Final Competition Review).

This problem is certainly not a reflection on the RSL dCDH architecture but on the risks that must be accepted by an “early adopter” of a new technology. While the delayed AVR-SATs contributed to the incomplete integration of the Akoya-A EDU, it is not the sole reason. More importantly, given ASL resources, it is unlikely that an alternate architecture/microcontroller could have been adopted and integrated any faster. The AVR-SATs that ASL will use in NS-4 have an anticipated delivery date of mid-summer 2005, which means that there will be no integration delay for Akoya-B/Bandit-C.

**IMPLEMENTATION AT SCU**

SCU’s Robotic Systems Laboratory is hosted by the school’s Mechanical Engineering Department although the lab runs an interdisciplinary program with students from mechanical/electrical/computer engineering as well as from physics and math/computer science. RSL runs a very active, externally-supported field robotics program in which undergraduates routinely develop robotic systems for performing advanced technology demonstrations and performing world-class scientific investigations¹. Each year, approximately 35-40 seniors participate in this program on any of 5-10 interdisciplinary projects aimed at developing a particular robotic system or device. These systems include spacecraft, unmanned aerial vehicles, terrestrial rovers, surface vessels, and underwater robots. Graduate students are integrated into the program as key technologists who typically use the robotic systems as experimental platforms for experimentally verifying and validating a technical capability that is developed as part of an engineering thesis.

**Original Motivation**

The motivation for developing the dCDH system arose due to the difficulties that several of the authors had experienced while working on educational microsatellite projects with a single main computer. While a single processor can provide a significant amount of computational power, it was found that its use in these projects led to several drawbacks. The majority of these problems were related to the fact that all of the functional subsystems had custom connections to the central processor.

From a technical perspective, this central connectivity caused a number of problems. For example, changes in the processing requirements of one subsystem (such as the need for additional i/o pins, memory, computational power, etc.) could easily affect the resources available to the other subsystems given that such resources are limited.

Educationally, even more significant drawbacks were observed. Students on specific subsystem teams would often develop the mechanical and electrical portions of their subsystem and then hand them to the processor team for integration and testing since a) the high-cost and often custom processors were very limited in number, and b) the complexity of the computer hardware and multi-objective software required a processor-specialist to develop the custom program. As a result, the already overburdened software team became the de facto integration and test engineering team since every functional subsystem required some level of software support. This practice would overwhelm the limited number of software developers who were already taxed with developing and debugging the custom processors. In addition, it left a significant hole in the educational experience for the students developing the mechanical and electrical elements of the system since a) they were largely absolved of any rigorous test and integration tasks (which occasionally led to significant quality control problems), and b) their
focus on delivering only portions of a subsystem made it difficult to provide any sort of comprehensive focus on subsystem functionality.

**Early dCDH Work on the Emerald project**

When the opportunity arose to develop the two-spacecraft Emerald nanosatellite system as part of the first University Nanosat Program (NS-1), the SCU-Stanford-MIT team elected to include the development and use of a dCDH system as a developmental experiment focused on ways to address the aforementioned drawbacks of a central computing architecture. Other mission objectives of the Emerald system included demonstrations of key sensing and control technologies for satellite formation flying as well as verification of advanced autonomous operations techniques.

Early dCDH work on Emerald attempted to balance the dual development of the dCDH standard with the design of a specific hardware and software implementation of that standard. It was during this stage that many of the key trades were developed regarding the choice of the linear bus, the use of the I²C and microLAN implementations, etc. The first hardware implementation of the standard microcontroller board was a PIC16F877-series motherboard; this was the primary board baselined for the Emerald mission. A more capable PIC17C56-series motherboard was later prototyped in order to conduct specific tests regarding the monitoring of the linear bus as a means of assessing dCDH performance. These original boards are shown in Figure 8.

![Figure 8. Initial dCDH Motherboards: Left – the PIC16F877-based board used for Emerald subsystems. Right – the PIC17C56-based board used for bus monitoring.](Image)

Initial use of the dCDH system for Emerald resulted in several key benefits. First, the use of standard protocols and equipment saved significant cost and time that would have otherwise been invested in custom designs (as had been done in previous programs). Second, the modularity promoted by the approach allowed entire subsystems to be completed, tested, and integrated without being impeded by slow progress elsewhere in the development effort. Third, the use of simple microcontrollers allowed a subsystem team to completely develop all mechanical, electrical and software elements relating to the specific subsystem functionality, a feature that significantly improved the educational experience of students. Furthermore, the simple interface allowed these functional subsystems to be easily verified using a standard PC interface, thereby allowing these teams to easily provide the test data necessary to properly support claims of “we’re done!”

This PC interface has also been used by RSL to support remote, internet-based system integration and test through the lab’s NETROL system.

Of special note is the fact that one of the primary delays with Emerald involved the very late delivery of and the grueling learning curve associated with a commercially available flight processor advertised as catering specifically to the needs of small satellites. The Emerald team had originally baselined the use of such a computer in order to expedite functionality of basic command and telemetry functionality, to provide more robust processing if such proved necessary as the program evolved, and to serve as a hedge against the risks associated with the dCDH experiment itself. As it turned out, the Emerald team completely developed and accepted as final several subsystems prior to the ultimate delivery of the commercial flight processor.

Furthermore, this flight processor was later cut from the design due to its complex and proprietary design that made it difficult to learn and implement within the parameters of an academic environment.

**Maturation of the dCDH System**

In early 2003, Stanford and MIT withdrew from the Emerald development effort due to a shifting focus of the involved laboratories, difficulties with the UNP-1 flight verification process, and the explosion of the Space Shuttle (which had been the targeted launch vehicle for the mission). RSL continued with this effort, however, and used the opportunity to iterate both the dCDH standard and the hardware/software implementation. This work included updates to the I²C and microLAN drivers, refinement of the inter-facility data encapsulation (used for satellite-to-satellite transmissions) as well as an explicit definition of a power bus protocol. In addition, this work included the development of a new hardware implementation, AVR-SAT as described previously, and the composition of its supporting software libraries.

During this period of time, the RSL team also developed several significant software extensions for use with the dCDH systems, although they are not explicitly part of the dCDH standard or implementation. The first of these was a file system which provided an extended layer of data management to individual dCDH microcontroller boards. Next, a command scheduling system was developed which allowed a single dCDH microcontroller to cache and process commands scheduled for future execution by sending these
commands out onto the data bus at the appointed time. An extended version of this system was also developed in order to provide full service command processing which consists of releasing the scheduled command and also storing any immediate response by the remote subsystem. In addition, at the application level, a production rule system was developed in order to provide a basic level of on-board automation. It should be noted that these extended software services provide the dCDH system with one form of flexible resource sharing given that command scheduling, command execution, command verification, and data storage for a single functional objective may be shared over any number of dCDH modules. In addition, the fact that these functions may be easily incorporated in parallel and/or across segments of the space system opens the door for a powerful form of fault tolerance.

Also during this time, RSL developed a simpler non-space rated version of the AVR board, known as the AVRmini. This board has been incorporated into SCU’s undergraduate mechatronics course which is typically taken by students during their junior year. This introduction significantly shortens the learning curve when many of these students begin work on satellite projects using the AVR-SAT boards during their senior year.

**dCDH for the ONYX Mission**

The newest dCDH hardware and software services have been selected for use in the newest RSL small satellite initiative, ONYX (ON-board autonomY eXperiment); indeed, the use of the dCDH is considered to be a developmental experiment in and of itself for this particular mission.

Like the WU Akoya-B/Bandit-C mission and the UT ARTEMIS mission, development of the ONYX space vehicle is being funded by AFOSR through the NS-4 program. ONYX is being designed for a three month LEO mission to a) demonstrate advanced model-based anomaly management techniques (in which faults will be deliberately injected via ground command in order to run double-blind experimental verification and validation of this technology), b) characterize the space operation of a novel multi-spectral imaging system that has been developed by the Jet Propulsion Laboratory, and c) demonstrate a variety of distributed command and control operations technologies through the use of the dCDH avionics and RSL’s distributed ground segment, and d) support interactive web-based experiments in K-12 classrooms and as part of university laboratory exercises.

The ONYX dCDH architecture, depicted in Figure 9, consists of an array of AVR-SAT modules linked together through the dCDH linear buses. Dedicated dCDH processor boards exist for the communications, power, and attitude subsystems as well as for each payload subsystem. Although not technically required, the quest for developmental simplicity within the student environment will most likely lead to additional AVR-SAT modules for executing the command scheduling and expert system services.

**ONYX (as well as FASTRAC and Akoya/Bandit) will be controlled through the use of RSL’s Distributed Robotic Control Network (DRCN). This system consists of a centralized mission control center, which is located in the Space Technology Center at NASA Ames Research Center. This control center is connected via the internet to several geographically distributed communication stations to include four OSCAR-class amateur radio groundstations (located throughout the United States) as well as an 18 meter dish that is leased from SRI International. Elements of the RSL DRCN are pictured in Figure 10.**

Although not currently implemented, engineering upgrades to these groundstations will include use of AVR-SAT modules in order to implement a modular ground station control architecture. Use of this ground segment to perform highly autonomous, distributed satellite control is one of the specific missions for ONYX, and the dCDH system plays a vital role in achieving the technical demonstrations being developed. These demonstrations are projected to include distributed anomaly management through the use of model-based reasoning, distributed and autonomous command planning and fault-tolerant execution, etc.

**Figure 9. ONYX dCDH Architecture.**

**Figure 10. RSL’s Mission Control Center and Two Types of Communication Stations.**
In addition to using the dCDH system in spacecraft projects, RSL personnel have also begun to incorporate its use into other robotic systems being developed by the Laboratory as shown in Figure 11.

For example, a network of AVRmini boards (RSL’s non-space rated precursor to the AVR-SAT design) is currently being developed to control several of the Lab’s underwater robots, such as the Triton and Mantaruis tethered vehicles and the Bronco AUV (autonomous underwater vehicle). An initial version of the same system has been incorporated into testflights of the latest RSL UAV (unmanned aerial vehicle) that has been undergoing test flights during the past year. And portions of the standard have been incorporated into one of RSL’s two multi-robot rover testbeds that are used to experimentally demonstrate cluster control techniques for a wide range of applications; incorporation of the dCDH standard into the other system is slated for the 2005-06 academic year.

The use of commercial standards and parts certainly contributes to a low-cost implementation, which is of course critical in an academic environment. It should be noted that the cost of the single industry-grade satellite processor originally selected for the Emerald mission was nearly an order of magnitude more expensive than the handful of dCDH boards that ultimately replaced it. This, combined with the immediate ability to manufacture new boards and an open design that is simple enough for undergraduates to understand, have made the AVR-SAT implementation of the dCDH system the current on-board processing system of choice for RSL projects. And while the computational power is limited, to date, the processing power of a single board has yet to limit the functionality of any subsystem developed as part of a limited student design project; and when such a time comes, RSL’s plan is to simply develop a more powerful implementation that still conforms to the dCDH standard (and which, because of the interface standard, can be seamlessly integrated with AVR-SAT and other implementations within the same system).

Finally, the ability to distribute task-specific functionality across more than one processing platform has opened the door to new control approaches. While the benefits of many of these approaches have not been operationally validated (they are, by definition, a mission objective of the RSL flight systems), the powerful nature of this capability has been demonstrated in the lab and anecdotaly suggests that these capabilities will significantly enhance the flexibility and robustness of operational missions.

**IMPLEMENTATION AT UT**

The University of Texas at Austin created a Satellite Design Laboratory (SDL) in 2002 in response to a long-standing desire within the Aerospace Engineering and Engineering Mechanics (ASE) Department to improve the satellite hardware design program. The SDL currently retains about 40 students per semester, working on various projects such as CanSat and CubeSat payloads, high altitude balloons, and NASA microgravity (“Weightless Wonder”) experiments, in addition to the University Nanosatellite program. The student population in the SDL is a mixture of graduate and undergraduate students, primarily from aerospace engineering backgrounds, but also with students from other disciplines. The modularity of the dCDH architecture is the primary benefit with regard to its use within the RSL educational environment. Casting the student design challenge in terms of implementing a network of intelligent subsystems with individual subsystem teams responsible for developing and verifying all subsystem-level functionality (regardless of the mix of mechanical, electrical, and computer components) is an overwhelmingly successful strategy. It reinforces student involvement throughout multiple lifecycle phases for a given piece of equipment, it mandates a truly interdisciplinary engineering approach, and it simplifies management with respect to the development and execution of the project’s work breakdown structure. The modularity also allows the separate implementations to be seamlessly mixed within a single system as was demonstrated (although unintentionally) during development of the Emerald project.
mechanical, electrical, and chemical engineering, physics, and the information school. Ideally, students are recruited into the lab as sophomores and will stay with the lab for several years while working on different projects. A vertical mentoring process is used where new students are mentored by senior students and students move up in responsibility as they become the next group of mentors.

The University Nanosat-3 entry for UT’s SDL is a technology demonstration mission known as FASTRAC (“Formation Autonomy Spacecraft with Thrust, ReIrnav, Attitude, and Crosslink”). The main feature of the mission is that two nearly identical 15 kg satellites will separate on-orbit and perform formation flying experiments. Among the experiments planned are GPS-enabled relative navigation and attitude determination, radio crosslink between satellites, and the operation of a low thrust plasma propulsion system. These are all technologies that are of interest to future satellite formation flying missions. A picture of the stacked system of twin satellites is shown in Figure 12 as they were delivered at the UN-3 Flight Competition Review in January 2005. An individual satellite is shown in a partially integrated state in the SDL’s clean room tent in Figure 13. FASTRAC was the winner of the UN-3 competition and is scheduled to fly on an Air Force launch opportunity in 2006 or 2007.

Selection of RSL Standard

UT benefits from a talented pool of student engineers. But as a result of its creation within 1 year of the University Nanosatellite-3 proposal deadline, UT’s SDL lacked experience in space hardware and electronics fabrication at the time of the UN-3 proposal. An early decision was made to partner with RSL for a quasi-off the shelf distributed computing and data handling system. UT could then focus its limited resources on the systems design studies and technology experiments.

In a similar manner to what occurred at WU, UT sent a couple of students to SCU’s RSL in 2003 to meet with the dCDH development team. These students received training in the AVR system and became designers of the communications and software systems.

The final data bus block diagram for FASTRAC is shown in Figure 14. There are four AVR boards per satellite, including a sensor AVR (GPS + magnetometer) and an actuator AVR (thruster). Although the core capability and many system utilities were provided by RSL, UT students designed and wrote all top-level software for FASTRAC’s components. It was originally thought that the complexity of the GPS receiver, which contains many different commands and output data types, might pose problems for the AVR system. Since the GPS telemetry interface is a simple string passthrough, however, it actually turned out to be straightforward. This is an example of the elegance of the distributed bus concept. The particular details of each device are handled within that subsystem and do not affect the overall spacecraft bus complexity.

Assessment of Distributed System

As has been mentioned at the other universities, several advantages were obtained by using the distributed bus system. It was easy to train students in the system standards and the UT team was then able to customize the bus for its own uses. The system scaled well and was able to handle relatively complex sensor subsystems such as GPS receivers without problems.
There were a number of additional benefits. Because the system was made out of commercial standard components, the C&DH hardware was very cost effective – the entire system cost only a few thousand dollars. A hidden cost went into the training, integration, and software development that was required. These costs were carried by the students through the educational process. In fact it was the goal of the program to train the students. But these costs are perceived to be modest compared to those of a custom designed satellite computing system.

Another advantage had to do with the ease of developing ground support equipment (GSE) and testing the hardware. The engineering approach was to demonstrate functional capability through testing. During the two year design competition, the team performed microgravity separation experiments, GPS hardware simulations, radio crosslink tests, structural vibration tests (both shock and frequency sweep), and thermal/vacuum chamber tests. It was easier to develop software that could support all of these tests because the communications interface was well defined. Also, because each subsystem could be integrated separately onto the satellites, it was possible to develop and test C&DH software for each subsystem individually. This greatly simplified the software development process.

Finally, because FASTRAC uses two crosslinked satellites, it is possible for these satellites to communicate with each other as components of the same distributed architecture. Therefore, minimal added effort is required to perform information exchange between the satellites. This innovative feature allows the crosslink to be designed simply and efficiently.

Future of the dCDH Standard at UT

Based on the success of the RSL system on the FASTRAC mission, it is currently being considered for UT’s University Nanosatellite-4 experiment, ARTEMIS (“Autonomous Rendezvous & Rapid Turnout Experiment Maneuverable Inspection Spacecraft”). This project builds upon the previous FASTRAC mission concept by adding a three-axis attitude control system and impulsive thruster system to enable on-orbit rendezvous of the two satellites. The satellites also contain a camera imaging system so that one satellite can “inspect” the other. Finally, to demonstrate a rapid turnout capability, the satellites will be integrated and functionally tested within 5 days.

The RSL dCDH standard is being considered as one design candidate for this mission. The primary challenges facing the command and data handling system are:

- The active guidance system requires greater computing resources for rendezvous and control. The Guidance Navigation and Control AVR will need a more advanced microprocessor than the current design.
- The imaging system requires substantially more data storage and throughput than has been previously attempted.
- The 5 day rapid turnout goal will require a bus-level plug and play capability.

Although several different design approaches are being evaluated at this time, the distributed architecture has many advantages that are applicable to this mission, including team training, cost, and retention of previous C&DH expertise.
CONCLUSIONS

The dCDH architecture presented at the 2002 Conference on Small Satellites was implemented on three university-class spacecraft projects by 2005 and is being evaluated or implemented on three more. With the launch of FASTRAC in 2006 or 2007, it will be one of the first distributed data bus architectures to operate on-orbit.

In 2002, the RSL team made four predictions regarding their dCDH architecture. Those predictions now can be evaluated:

Prediction 1: Design Simplification Through Clear Functional/Data Decomposition. As described by all three universities, the RSL dCDH standard has improved the design process; the conceptual functional block diagrams directly map to physical boxes on the spacecraft. Such clear decomposition has allowed student teams to independently develop their designs. This simplification is very important given the naturally high rate of turnover in student projects.

Prediction 2: Rapid Integration & Test Through Decoupled Operation. The University Nanosat-3 projects were on an extremely short (2-year) concept-to-EDU design cycle; one important reason why UT and WU finished so strongly in the NS-3 competition was the rapid integration afforded by the dCDH architecture. Similarly, the NS-4 projects proposed by these universities are all modular upgrades to previous space projects; the flexibility of the linear data bus allows for rapid system-level integration and test of new spacecraft components even before the new spacecraft is fully developed.

Prediction 3: Robust Operational Behavior. While no RSL dCDH-implemented mission has flown to date, aspects of robust operational behavior have already been demonstrated. Fault-tolerance through dual-use or alternate configurations has been demonstrated in the design and testing phase of the NS-3 program.

Prediction 4: Enabled Multi-Agent Missions. All of the UT and WU projects described above are multi-agent missions, involving two vehicles controlled from a single ground station, usually with one vehicle acting as a relay for the other. The RSL dCDH architecture has made the design and operations of these systems truly transparent; integrated operation of the second vehicle has been no different than integrating any other on-board components.

In fact, the RSL dCDH architecture is one path towards enabling extremely rapid assembly, integration and test (A&IT) of space vehicles. Using the data protocol and wiring harness standards, many kinds of components can be integrated literally in minutes, in the same way that terrestrial computers can be connected to the Internet. This is possible because no other device on the linear data bus needs to “know” about the presence of the new device; as long as commands from the ground are formatted properly, the new device will function properly.

Clearly, such rapid integration is only possible for compact, self-contained devices with nominal power and thermal behaviors. But the concept is so promising that the all three teams are considering a demonstration of rapid A&IT on their NS-4 spacecraft using the RSL dCDH system.

The RSL dCDH Architecture as a Standard

The RSL dCDH architecture is a mature system for distributed control of robotic systems which has been adopted for spacecraft projects at three different universities. Thus, it is worth discussing whether this architecture is valid for a broader CDH standard.

As discussed above, the RSL dCDH architecture has numerous benefits, including simplified, modular, decoupled design, ease of integration, and the ability to incrementally upgrade devices in the architecture. For example, the RSL team has already performed several upgrades to the core microcontroller, from the PIC to the AVRmini to the AVR-SAT. These upgrades have been made to existing systems with no change in functionality; some systems have different microcontrollers interacting seamlessly.

On the other hand, this architecture has modest data throughput rates, less than 400 kbps in optimal configurations and less than 100 kbps in nominal configurations. There is also a concern whether the presence of so many microcontrollers may increase system susceptibility to radiation-induced events.

In an academic environment, there is a concern that the flexibility and decoupled operation enabled by the dCDH could encourage careless design or operation; in the absence of rigorous interface management and operational testing, it is possible to add devices that conflict with one another (for example, using the same address) or for students to develop a deep understanding of how the entire system operates.

The final obstacle to broad adoption of the RSL dCDH standard is that RSL is the sole provider of the AVR-SAT microcontroller. Of course, as stated above, any processor that adheres to the wiring and data protocols is compatible with the RSL standard, but one of the primary educational benefits of the program is the rapid development enabled by the AVR-SAT and the mentoring community of schools who use the system.

It is the authors’ suggestion that the RSL dCDH system is an excellent option for “university-class” spacecraft
(i.e. those space missions that emphasize student education through hands-on development). As demonstrated by the SCU, UT and WU programs, the dCDH system and associated hardware enables very rapid, parallel development of spacecraft subsystems and accelerated integration. The well-defined interface also aids good design practice through functional decomposition of mission tasks as well as accelerating the integration of new students into the program.

The AVR-SAT microcontroller is being demonstrated in University Nanosat-class (30 kg) spacecraft, but it has been sized for CubeSat-class applications. Broad adoption of the RSL dCDH system could improve the development of these student missions in the same ways as described, above.

It is the authors’ experience that CDH systems in university-class spacecraft can quickly become the dominant schedule drivers, as such systems invariably take longer to develop than expected – especially when the students are designing the hardware and software themselves. RSL’s experience with the EMERALD mission indicates that even professional CDH systems can be extremely difficult to work with in a academic setting. For most universities, it would be far more productive to implement an established CDH/operating protocol and instead spend student labor on the development of high-level software and extensive operational testing, provided there is easy access to drivers, documentation and design experts. In that regards, the RSL dCDH system provides a unique opportunity for university-class missions.

ACKNOWLEDGMENTS

The authors acknowledge the other RSL team members responsible for this dCDH architecture: Bryan Palmintier, and Rob Watson. They also thank the Nanosat development teams at their respective universities for their work in implementing, refining and demonstrating the value of this architecture.

Development of this distributed computing architecture has been made possible through sponsorship by the U.S. Air Force, NASA Goddard Space Flight Center, the Universities Space Research Association, the California Space Grant Consortium, and Santa Clara University. Partial development of this system and its integration with RSL’s Distributed Satellite Operations Network has been supported in part by the National Science Foundation under Grant No. EIA0079815 and EIA0082041; any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The Nanosat activities at all three universities have been partially sponsored by AFOSR through AFRL/VSSV.

REFERENCES


