DEVELOPING PLUG-AND-PLAY SPACECRAFT SYSTEMS: NASA GODDARD SPACE FLIGHT CENTER'S (GSFC) ESSENTIAL SERVICES NODE (ESN)

Robert Caffrey
NASA/GSFC
Code 970.2
Greenbelt, MD 20771
(301) 286-0846
robert.caffrey@gsfc.nasa.gov

Harry Shaw
NASA/GSFC
Code 310
Greenbelt, MD 20771
(301) 286-3178
harry.shaw@gsfc.nasa.gov

Leon Wagner
FORTH Inc.
111 N. Sepulveda Blvd., Suite 300
Manhattan Beach, CA 90266-6847
800-55-FORTH
lwagner@forth.com

ABSTRACT

The phrase 'plug-and-play' refers to hardware and software devices that, after being installed ("plugged in"), can immediately be used ("played with"). The system can be used for a specific application without requiring additional software programming or hardware design.

The Essential Services Node (ESN) takes a big step in moving space systems to the 'plug and play' environment of commercial systems. Systems that use the ESN to interface sub-systems to the spacecraft bus (either MIL-STD-1553B or MIL-STD-1773) provide a standard interface to both the sub-systems and the spacecraft bus.

The ESN is a multi-chip module (MCM) that contains an Application Specific Integrated Circuit (ASIC), memory devices, and analog circuitry. The ASIC contains a microprocessor, a MIL-STD-1553B / -1773 bus interface, and many industry standard devices. The memory includes program-, data- and shared-RAM. The analog logic includes a 16-channel analog-to-digital (A/D) converter, an 8-channel 1.0 mA current source, and an analog power strobing circuit.

The ESN MCM is a low-power (<1W), low-cost (<$25k), lightweight (2.6" x 2.6" x .3"), rad-hard (>100K), industry standard, spacecraft control and data system. The ESN eliminates custom interfaces by choosing a set of industry standard interfaces. It reduces system development cost and time by providing system designers with a 'tool box' of common devices, and supports a standard bus architecture in the MIL-STD-1553B / -1773 bus. It eliminates the need for 'orphan' telemetry by incorporating their functions, and it provides the capability for autonomous operations with its powerful microprocessor.

The ESN connects instruments and subsystems to a spacecraft's MIL-STD-553B/-1773 bus. The ESN reads commands or data from instruments or subsystems, processes the information, and sends it to a spacecraft's command and data handling (C&DH) system. The ESN also receives commands from the C&DH system, processes the information, and sends it to an instrument or subsystem.

This paper describes how the ESN provides a standard interface to the spacecraft's bus and its sub-systems. This paper also describes the plug-and-play standard, Open Firmware, and our plans to port it to the spacecraft environment.

1. INTRODUCTION

Plug-and-play is a combination of hardware and software support that enables a system to recognize and adapt to hardware configuration changes with little or no user intervention. In a plug-and-play system, a user can add or remove devices without making hardware, software, or configuration changes and without having a detailed knowledge of the system. A plug-and-play system allows a user to change the system configuration with the assurance that all devices will work together and that the machine will boot correctly after the changes are made.

With the advent of commercial plug-and-play computer systems, both devices and systems have become less expensive, more robust and more 'user friendly'. Plug-and-play systems have standardized hardware and software interfaces and industry has become more competitive and more innovative.

A plug-and-play satellite system is a realistic goal and NASA Goddard, with the development of the ESN, has taken the first steps at achieving that goal. A plug-and-play system requires both hardware and software interface standards. This paper describes the ESN and how it has helped standardize a spacecraft's hardware interfaces. This paper will also briefly describe the standard software interface being developed, which is based on the industry standard Open Firmware standard.

The ESN is a spacecraft control and data system in a single MCM package. The ESN, shown in Illustration 1.0, is embedded into the subsystem of a spacecraft. It provides a standard means of performing the "essential" tasks for many of the spacecraft's subsystems. It also combines many of the common functions found in these subsystems into a single multi-chip module that provides a standard interface node to the spacecraft's communications bus.

For more detailed design information on the ESN, refer to the ESN Hardware Specification [1] and the GSFC ESN Hardware Users Guide [2]. For more general information, application data, and the development history of the ESN, refer to "A Standard Spacecraft Data System on a Chip: NASA Goddard Space Flight Center's Essential Services Node (ESN)", (Caffrey et al) [3].

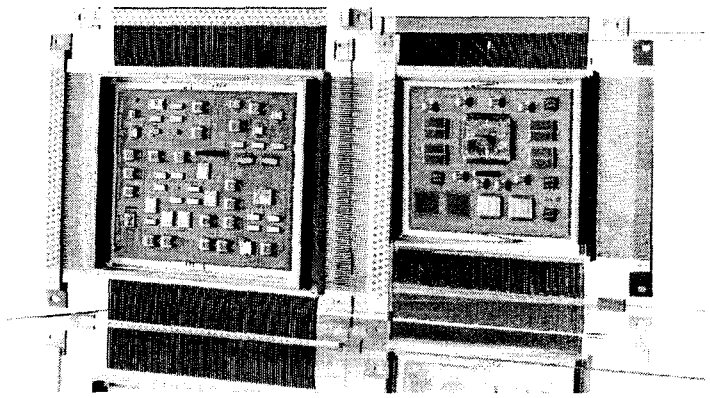

Illustration 1 - The ESN, mounted in front of a mirror with its lids removed, is packaged in a 392 pin 2.6"x2.6"x0.3" double-sided MCM weighing less then 90 grams.

2. SPACECRAFT OVERVIEW

The block diagram in Figure 1 shows where the ESN fits into a typical spacecraft architecture. The MIL-STD-1553B or -1773 bus lends itself to the plug-and-play environment and such an environment has many potential advantages. This environment lowers development costs, decreases system integration times, reduces system documentation, increase competitiveness among vendors and overall spacecraft reliability is improved.

The following are two examples of using the ESN in a plug-and-play spacecraft environment. If during spacecraft integration the GPS from vendor A has problems, a GPS from vendor B could be added to the spacecraft with little impact to the spacecraft's development schedule. In another example, a new spacecraft could be assembled with spacecraft systems from previous satellites and the majority of the new spacecraft development would be complete.

The ESN, as the hardware standard of a plug-and-play system, is used to interface a spacecraft's subsystem to the MIL-STD-1553B / -1773 bus. In this bus architecture, the bus controller (BC) node orchestrates the communication between all other nodes on the bus. The other nodes are designated as remote terminals (RT). The BC node is typically contained in the C&DH system. As a BC, the spacecraft's C&DH ESN can process commands and distribute them to RT ESNs or requests that RT ESNs send status and telemetry to the C&DH ESN for processing and storage.

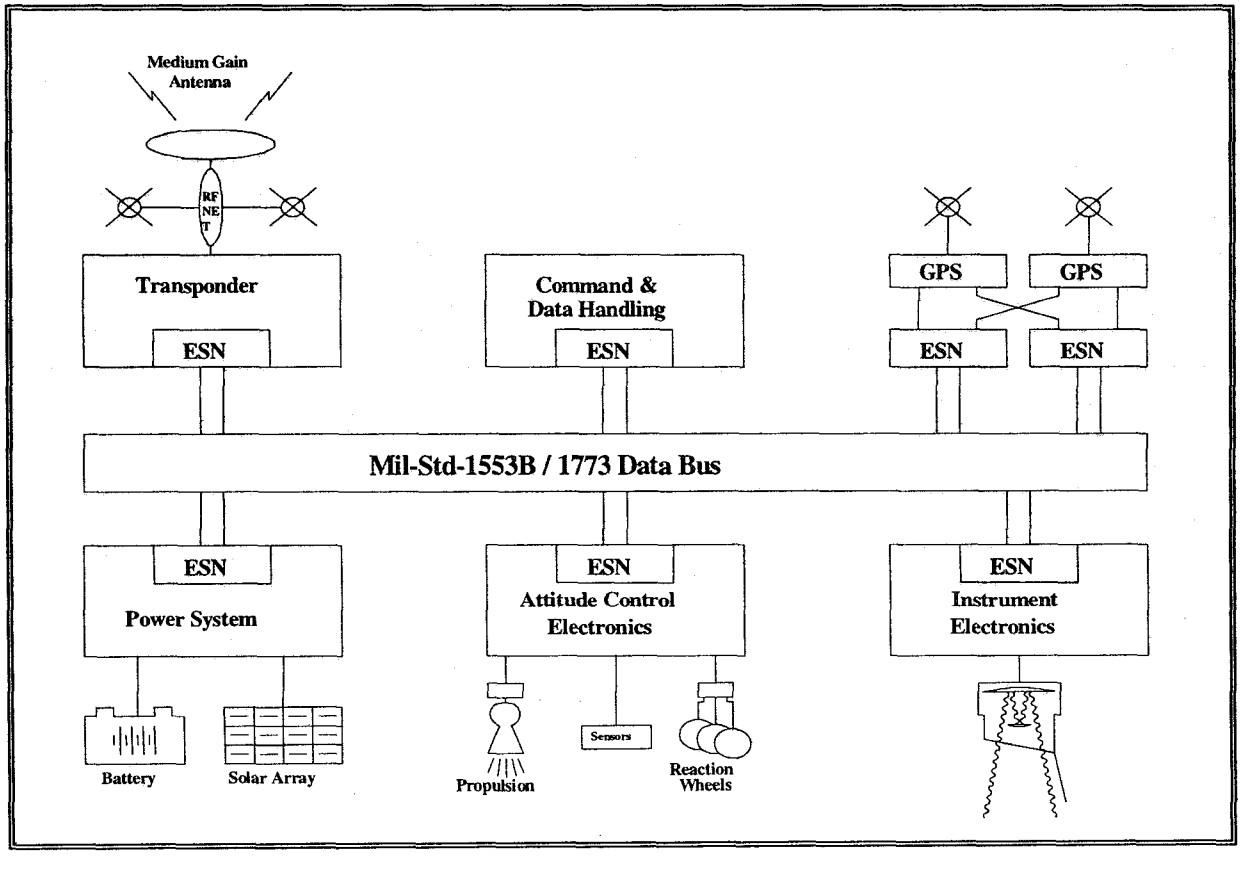

Figure 1- A Generic Satellite Block Diagram

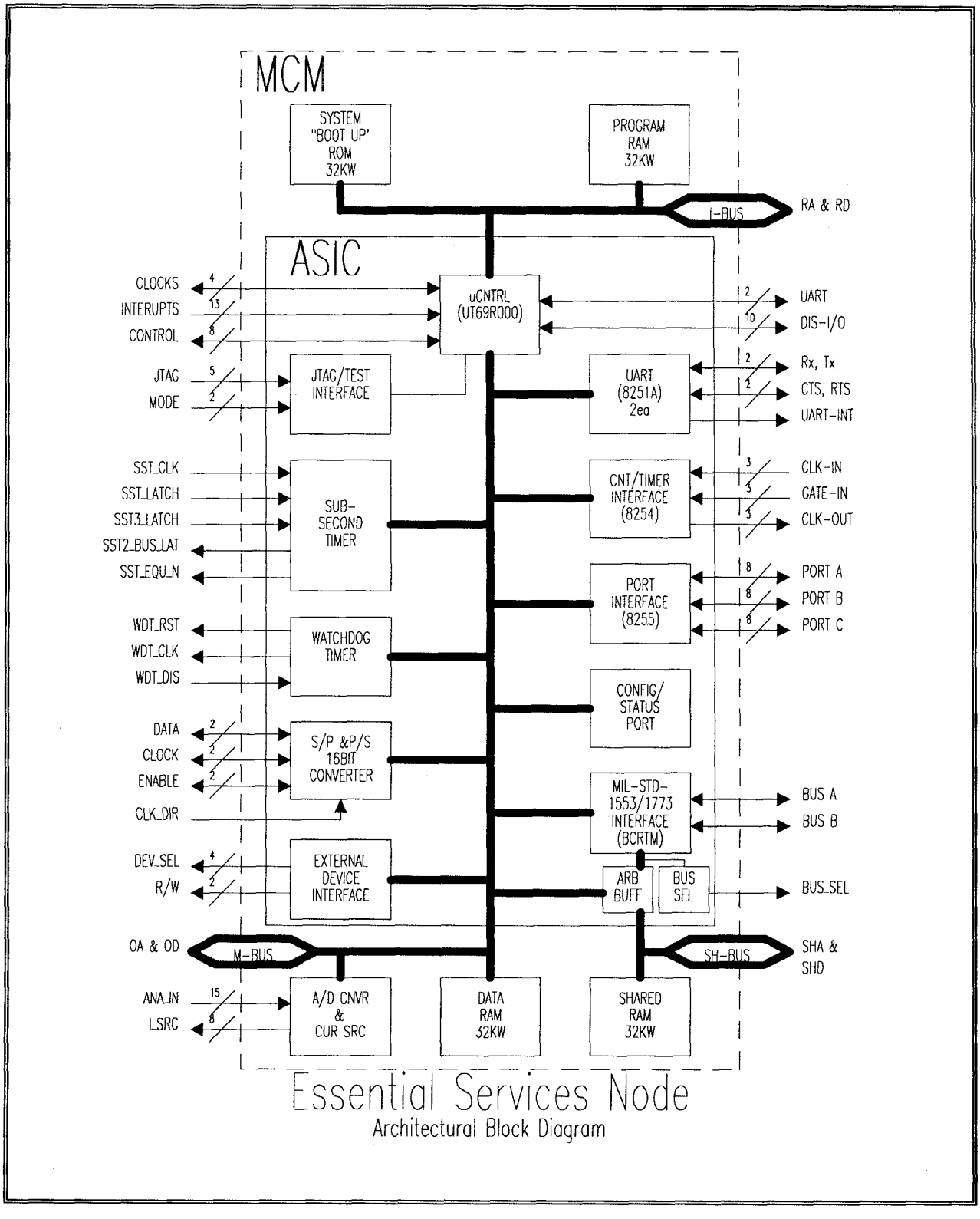

Figure 2 – The ESN's MCM and ASIC Block Diagram

3. ESN SYSTEM REVIEW

The ESN, shown in Figure 2, is a spacecraft data and control system contained in a single MCM device. The ESN contains all the digital and analog circuitry required for controlling a subsystem and connecting it to the spacecraft's MIL-STD-1553B / -1773 bus.

The following paragraphs describe the ESN's MCM, ASIC, memory, and analog circuitry. They also describe the ESN's digital and analog devices in detail. For more information and details on the ESN, refer to Table 1.0, the ESN's Contact List or Section 7, the ESN's Reference list.

3.1 ESN MCM Description

The MCM contains an ASIC, analog circuitry, and memory devices. The MCM package is a hermetically sealed dual cavity ceramic quad flat-pack with 392 leads. For more details on the ESN's MCM, refer to the Honeywell Source Control Drawing (SCD) [4].

3.2 ESN ASIC Description

The rad-hard ESN ASIC is in a 50,000 gate package and is available in die form, a 280-pin flat-pack, and a 304-pin grid array (PGA) package. The primary devices in the ASIC are described in the following paragraphs.

3.2.1 UTMC UT69R000 Microprocessor—The ESN's microprocessor is the 16-bit UTMC UT69R000 RISC microcontroller. This processor is a 16-bit Harvard architecture machine with 1MW of instruction space, 64KW of data space, and 64KW of I/O space. The processor's operating rate is two clocks per instruction and at 12MHz, achieves a performance of 6 MIPS. The processor also has 15 levels of interrupts, two 16-bit timers, two discrete input signals, eight discrete output signals, DMA support, and a built-in 9600-baud UART.

3.2.2 UART Device (8251A)—The ASIC contains two UARTs that are based on the industry standard 8251A UART.

3.2.3 Counter/Timer Device (8254)—The ASIC contains two counter/timer devices, based on the industry standard 8254 device. Timer #1 generates external clock and control signals while timer #2 generates internal clocks and signals for other ASIC devices.

3.2.4 Parallel Port Device (8255)—The parallel port device is based on the industry standard 8255 device. It contains three 8-bit I/O ports, which are independently software programmable for input, output, or bi-directional communication.

3.2.5 UTMC UT1553 BCRTM Interface—The MIL-STD-1553B/ -1773 bus interface is the standard UTMC BCRTM chip. The BCRTM operates at 12MHz (either the processor's clock or its own) and can be configured as a bus controller (BC), a remote terminal (RT), or a bus monitor.

3.2.6 Sub-Second Timer (SST)—The SST is a 24-bit incrementing counter clocked by an external clock. There are four latches associated with the SST. SST#1 is internally latched, SST#2 is either externally or internally latched, SST#3 is externally latched, and SST#4 is a comparison latch.

3.2.7 Watchdog Timer—The Watchdog Timer (WDT) is a 24-bit up counter clocked by the CPU's system clock. When the WDT rolls over (every 1.4s), the ASIC verifies that the software has cleared the WDT. If the WDT is cleared, everything is operating correctly. If the WDT is not cleared, an error condition has occurred and the ASIC generates a WDT reset.

3.2.8 16-Bit S/P and P/S Data Converters—The ASIC contains a synchronous S/P and P/S converter. The S/P converter device reads a serial data stream into a 16-bit shift register. Conversely, the P/S device clocks out a serial bit stream from a 16-bit shift register. Each device operates independently from one another and is controllable in either a master or slave configuration.

3.2.9 Sleep Mode Device—The sleep mode device is a software activated circuit which gates the master clock to the processor allowing the processor to temporally suspend its own activity. The UT69R000 processor is a static machine. This permits the processor clock to be stopped and the ESN's power consumption to be reduced.

3.2.10 Wait-state Generator—The ASIC's Wait-state Generator device expands on the wait state function of the UT69R000 processor.

3.2.11 Analog Control Interface—The Analog Control Interface is a set of digital signals, which controls the MCM's analog operations. The Analog Control Register controls the analog power strobing bit, the three current channel bits, and the four analog channel bits. When the processor writes or reads to the 12-bit or 8-bit A/D register, the ASIC generates the required signals to start or read an A/D conversion.

3.3 ESN Analog Description

The ESN's analog circuitry is divided into three parts—the A/D converter circuit, the current source circuit, and the power strobing circuit. Jumper pins on the MCM configure most of the ESN's analog circuitry

3.3.1 A/D Converter—An A/D converter is provided for digital conversion of analog values on any of the 15 analog channels or 8 current source channels.

3.3.2 Current Source (1.0 mA)—The combination of a voltage reference, transistors, resistors, capacitors, and an OP AMP generate a 1.0 mA current source. The current source is then connected to an 8-channel multiplexer for the purpose of measuring passive devices such as thermistors.

3.3.3 Analog Power Strobing—The combination of resistors, capacitors, and transistors form the ESN's power strobing circuit. This circuit enables the ESN's software to turn the analog power off and on. This feature greatly reduces the ESN's average power consumption.

3.4 ESN Memory Description

The ESN has three 16-bit data buses: the instruction bus, the operand bus, and the shared data bus. Two 32Kx8 SRAMs are connected to each of the ESN's three busses.

3.4.1 Program Memory—The ESN's instruction bus connects up to 1MW of memory to the processor instruction port. The first 32KWs are defined for boot PROMs, the second 32KWs are for internal program RAM, and the remaining address space is user defined.

3.4.2 Data Memory—The UT69R000's operand bus has two 64KW pages; one page for memory and one page for I/O. The top 256 words of the I/O page are reserved for ASIC I/O operations, but the remaining address space is user defined.

3.4.3 Shared Bus Interface—The shared memory is accessible from either the BCRTM or the UT69R000. The ESN uses this space to buffer data and commands both to and from the MIL-STD-1553B / -1773 bus interface.

4. THE SOFTWARE STANDARD

A plug-and-play system requires both standard hardware and software interfaces. The MIL-STD-1553B/-1773 bus and the ESN provide the hardware interface standard, but the software interface standard is more elusive. The software standard must be CPU independent; bus independent; operating system (OS) independent; and it must be auto configurable. The computer industry has faced similar problems and the standard many of them (SUN, Apple, Motorola, and more) have adopted, is the Open Firmware standard. Therefore, the software interface standard, under development, is based on the industry standard Open Firmware standard. Open Firmware also provides many additional benefits to spacecraft systems. For more information on Open Firmware, refer to "Open Firmware" (Bradley) [5].

4.1 Open Firmware Summary

Firmware is ROM-based software that controls a device from the time that it is turned on until the operating system takes control of the device.

The main function of boot firmware is to initialize the hardware and then to "boot" the operating system. Secondary functions include testing the hardware, managing hardware configuration information, and providing tools for debugging in case of faulty hardware or software.

4.1.1 Portability—Open Firmware is not tied to any particular processor family or to any particular bus. Open Firmware is in use on over a million machines, and is supported by several system vendors.

4.1.2 Plug-and-play drivers—One key Open Firmware feature is support for self-identifying devices. The device driver is stored in ROM on the device being "plugged-in" to the system. Open Firmware uses the "plug-in driver" technique with drivers encoded in a machine-independent language called "FCode". FCode is a byte-coded "intermediate language" for the Forth programming language. Forth, an industry-standard interactive programming language, is based on a stack-oriented "virtual machine" that may be easily and efficiently implemented on any computer. The device drivers are read by the operating system and compiled into executable code. The operating system then uses this code to interface to the devices.

4.1.3 Configuration Information—Open firmware also reports the characteristics of the device to the operating system software. System software may use this information to automatically configure itself for correct operation with the particular device. This specification is open and extensible, and may be updated to meet requirements, protecting against obsolescence of the interface.

4.1.4 Open Firmware Debug Feature—Open Firmware can debug hardware,

software, plug-in drivers, and even the firmware itself. Open Firmware provides interactive tools for debugging systems

4.2 Open Firmware on a Spacecraft

Open Firmware is an industry standard with many powerful features and many obvious benefits. Its framework and structure is also a good match with the spacecraft bus. The first step in developing the spacecraft version of Open Firmware is to develop the necessary software tools for the ESN. Open Firmware and its spacecraft implementation, however, will not be tied to the ESN. The standard is in fact, machine independent. Therefore, when a better or cheaper spacecraft bus interface is developed, the Open Firmware standard will continue to be a viable spacecraft plug-and-play standard.

5. CONCLUSION

The ESN has taken a big step in standardizing flight systems and reducing a spacecraft's cost, weight, power, and development time. A hardware solution, however, is never the final solution. Hardware will always get cheaper, faster, smaller, and have improved performance. Plug-and-play spacecraft systems and the spacecraft version of Open Firmware may be the key to better, faster and cheaper spacecraft systems. Plug-and-play spacecraft systems will reduce a spacecraft's cost, documentation, development time, and integration time. It will also increase a spacecraft's reliability and performance.

6. REFERENCES

[1] GSFC ESN Hardware Specification (No. ICD-735-2827 REV 2.0)

[2] GSFC ESN Hardware Users Guide

[3] Caffrey et al, "A Standard Spacecraft Data System on a Chip: NASA Goddard Space Flight Center's Essential Services Node (ESN)", 1997 IEEE Aerospace Conference Proceedings, p. 505-521.

[4] Honeywell ESN MCM Source Control Drawing (SCD) (No. 1902918).

[5] Bradley, "Open Firmware", 1992 Usenet Conference.

Table 1.0 ESN Contact List

Topic	*Contact*
– ESN Development NASA/GSFC	Robert Caffrey (301) 286-0846
– ESN MCM Honeywell CTO	Ed Roberts (602) 436-2658
– ESN ASIC FirstPass Inc.	Bob Schneiderwind (303) 688-6866
– Forth Software FORTH, Inc.	Steve Agarwal (800) 55-FORTH
– ESN Applications Jackson & Tull Inc.	Bill Dentel (301) 805-6090
– 69R000 & BCRTM UTMC Inc.	Bob Bauer (800) 722-1575

Acknowledgments: M. Bradley FirmWorks, M. Cuviello J&T, Inc., P. Hestnes Litton-Amecom, E. Rather FORTH, Inc.