Plug and Play for Spacecraft Onboard Systems

Chris Plummer, March 2005

Introduction

The aim of this paper is to establish a common understanding of plug-and-play for spacecraft within the SOIS area.

The paper starts by defining what plug-and-play is, and then suggesting a variety of ways in which this could be beneficial in spacecraft onboard systems. It then examines the few key capabilities that are required to implement plug-and-play systems. Finally, it looks at the current state of the art in spacecraft onboard systems and identifies the technological developments that are needed to make spacecraft plug-and-play systems a reality.

What is plug-and-play?

The term plug-and-play refers to a combination of hardware and software techniques that enable a computer system to recognise and adapt to configuration changes with little or no human intervention.

An obvious terrestrial application of plug-and-play is in desktop computing systems, where plug-and-play techniques allow the user to add new devices, such as a printer or an imaging device, without having to manually configure the host computer and install drivers to operate that device. However, plug-and-play techniques are also very widely used in other terrestrial environments, such as factory automation, where reconfiguration of the system through addition or removal of devices is commonplace.

Plug-and-play techniques are also commonly used in networked environments where the network topology is variable and devices can roam and attach at different points in the network. Such characteristics are often found in wireless networks for instance.

Early plug-and-play technologies did not detect configuration changes dynamically. Instead they examined the hardware configuration at start-up or during a reset cycle and adapted accordingly. The latest plug-and-play systems feature dynamic recognition of configuration changes, either using event signalling, or through periodic polling of devices, or through a combination of both.

Use of plug-and-play in spacecraft onboard systems

There are many different scenarios where the use of plug-and-play techniques could offer significant benefits for spacecraft onboard systems. The benefits vary across many different classes of mission, but understanding these benefits is essential to compiling a financial case for the development of spacecraft onboard plug-and-play techniques. In this section of the paper a number of different scenarios are examined.

Automated adaptation to hardware configuration changes offers a number of advantages for virtually all classes of spacecraft mission. Hardware reconfiguration of the spacecraft often occurs when a flight unit fails and a redundant unit assumes its role. Traditionally, this is handled by keeping multiple configurations onboard the spacecraft to handle each combination of prime and redundant hardware units. However, this uses valuable onboard memory resources and, as the number of permutations increase as spacecraft become more complex, becomes a less and less practicable solution. Plug-and-play techniques that could detect one unit dropping out of the configuration and a standby unit replacing it would allow automatic reconfiguration without the need to store multiple configurations onboard. Furthermore, this solution is suitable for any number of possible permutations of hardware configuration.

Manned missions are often characterised by having different payload configurations according to the goals of the mission, resulting a range of different configurations from one mission to another. Moreover, much of the payload equipment may only be powered on during certain parts of the mission so that there are several different configurations even during a single mission. Here again, plug-and-play techniques would allow dynamic reconfiguration of the system to make the best use of the available payload resources, and could even be used to trigger human intervention when the system detects that the configuration is not adequate to support a given experiment.

Planetary exploration missions of the future feature multiple elements that can roam the surface of the planet, moving in and out of range of each other. In some mission scenarios, a landed asset is augmented progressively by successive flights. In both cases, plug-and-play techniques could be beneficially used to manage the configuration of the overall system. In the latter case, plug-and-play techniques could allow completely unforeseen mission elements to be integrated into an established surface facility.

Capabilities required to support plug-and-play

Any plug-and-play system requires three basic capabilities to be provided, namely:

- Detection of configuration changes, e.g. addition or removal of devices,
- A mechanism for identifying devices,
- A mechanism for enumerating devices, i.e. for determining their capabilities at run time.

Detection of configuration changes is the first crucial step in performing plug-and-play operations and is the detection of the addition or removal of a device in the system, whether this involves physically adding or removing a device, or merely powering-on or powering-off a device without changing any physical connections. In the simplest cases, device discovery is carried out once at system start-up, and the system is then automatically configured according to the devices detected. However, for plug-and-play to be really useful more dynamic mechanisms of device discovery are desirable. At least three different classes of discovery mechanism are used, and these are described in the following paragraphs.

In centrally managed systems, changes in configuration can be detected by periodically probing the system. For example, on a centrally managed bus system the bus controller might periodically poll each available address to determine if there is a device at that location. By maintaining a table of connected devices the controller can determine if devices have been added or removed, and can initiate automatic configuration routines accordingly.

In truly distributed systems that use a democratically managed communication bus such as CAN or Ethernet, devices may announce their arrival or impending removal by placing a message on the bus. This is reliable for device arrival, but unexpected removal may go unnoticed if the device being removed does not have time to announce its departure. Therefore, this mechanism is usually combined with other mechanisms to ensure reliable detection of device removal.

The third mechanism is to use electrical signalling to indicate device arrival or removal. This is typically implemented on bus based systems that are designed to support live insertion of devices, such as USB. In these systems an electrical signal is used to generate an interrupt or event indication somewhere in the system that can be used to trigger further plug-and-play actions.

Device identification is the second crucial step in performing plug-and-play operations. Device identification is the process of determining precisely what devices are connected to the system so that the appropriate automatic configuration can be applied. Device identification is achieved by the device providing an identifier value that enables it to be uniquely identified within the system.

Two very distinct strategies exist for device identification in modern plug-and-play systems. The first is based on identification of device type. In this scheme a plug-and-play device provides information that allows the rest of the system to identify the type of device, such as *printer* or *pointing device*, and this information might be qualified with subclasses within the type, or with information such as the vendor and product identifier. However, while the system can determine the device type, it cannot easily distinguish between several instances of the same type connected on the system, for example if several identical file store devices are used. To overcome this, the system must augment the basic device identification process by dynamically assigning system unique instance identifiers to each device.

The second device identification strategy is to assign a globally unique identifier value to every device. This immediately overcomes the problem of identifying instances because however many devices of a particular type are used, they will each have a different ID. However, it has the drawback that a unique identifier must be generated for, and encoded within every device. Also, because device IDs are no longer related to device type, the system must obtain information about device type through other means, e.g. through device enumeration (described below).

Device enumeration is a mechanism used to obtain detailed information about a device so that the system configuration can be optimised. Usually, the information obtained through device enumeration includes details of the device capabilities as well as its current configuration.

Device enumeration can be used to extend the basic plug-and-play facility of dynamic, automatic system configuration in several different directions. For example, the device might hold an electronic copy of its own user manual, or a binary image of the device driver. In systems where asset tracking is important, devices might contain electronic copies of their own service records, generated automatically, that can be used to manage system maintenance activities and identify parts that should be scheduled for replacement or refurbishment.

Mechanisms to implement plug-and-play capabilities

The three basic plug-and-play capabilities – configuration change detection, device identification, and device enumeration can be implemented in a number of different ways.

Configuration change detection may be implemented using electrical signalling to dynamically detect changes in configuration, as is done in USB for instance, or the system may use polling combined with device identification and enumeration to periodically determine the system configuration. In any real terrestrial system, for example in an automated factory, a wide variety of different techniques for configuration change detection might be used, depending on a number of factors including the hardware capabilities, type of device, type of host system, and system topology. We can expect that on a spacecraft it would also be necessary to use a variety of different techniques for configuration change detection.

Both device identification and device enumeration imply digital communication of information between a device and the rest of the system. Any device that connects to the system using a digital communication bus could support device identification and enumeration, but in practice it is essential that communications with the device can be established without manual intervention. This is not inherently true of many commonly used communication buses and it is therefore necessary to apply special communication establishment techniques.

Many modern communication buses, including wireless networks, define standard procedures for establishing communication with devices on-the-fly. For example, wireless networks often specify dedicated hailing channels and signalling rates. Once communication with a device has been established the actual communication channels and signalling rates used may be negotiated during device enumeration. Similarly, wired buses like USB define specific control endpoints that must be used and default communication parameters for initial communications.

Older communication networks, such as Ethernet based LANS, have been augmented with special services like DHCP, which are designed to automate protocol stack configuration and enable communication establishment between nodes in a system. This then enables device identification and enumeration to be implemented on established networks.

Once communication between the device and the rest of the system has been established, device identification can be performed. This is performed by the device providing identification information in the form of digital data. Therefore, in addition to implementing an interface to a digital communication bus, the device must be able to store the identification information in non-volatile memory.

As mentioned above, the identification information provided by a device may relate to the device type, or to the device instance. In the first case, the system may then need to resolve instances if multiple devices of the same type are connected. There are several mechanisms that can be used for this that depend on the capabilities of the connecting medium. In the second case, where device identification relates to the instance of the device, the system must obtain information about the device type during the subsequent enumeration phase. This is usually relatively straightforward, but may require more system resources such as memory in each system node to maintain device information tables!

Device enumeration is the last of the three critical capabilities required to support plug-and-play operations and entails obtaining more detailed information about each device. This invariably involves an exchange of information between the device and the system but, depending on the amount of memory and processing power available in the device, the volume of information that must be transferred, and hence the amount of data that must be stored in the device, can be minimised.

State-of-the-art in spacecraft

Modern spacecraft comprise a few processing units interconnected by a digital communication bus together with many sensor and actuator devices connected using a variety of discrete interfaces. The trend at the moment is to steadily increase the number of processing units in future spacecraft, while the number of sensors and actuators in each new generation of spacecraft is increasing rapidly. However, very little attention is being paid to the interfacing of sensors and actuators, and there remains a large and growing variety of discrete interface types in use.

As mentioned above, almost any device attached to a digital communication bus can be adapted to support the device identification and enumeration capability needed for plug-and-play operations, and several techniques can be used to detect configuration changes dynamically on spacecraft buses. From a SOIS perspective, what needs to be done to support plug-and-play for the units directly connected to the spacecraft bus is to define standard ways of representing device identification and enumeration data, and to define standard identification and enumeration procedures. IEEE-1451, USB, and CAN already have standard formats and procedures that could be used as a reference to this activity. This could be covered under an extension of the TCOAS charter.

Extending plug-and-play capabilities to onboard sensors and actuators represents far more of a problem at the moment because of the variety of interfacing techniques used. We can categorise the sensors and actuators that are not connected directly to the onboard data handling bus into two classes, those with embedded microprocessors or microcontrollers, and those without.

Those devices with processors or microcontrollers typically include reaction wheel and gyro packs, sun and earth sensors, and so on[1]. It would be relatively straightforward to enhance these devices to support plug-and-play operations by

[1] Despite the presence of a processor, these devices still often use discrete interfaces to connect to the rest of the system

including device identification and enumeration support and ensuring that this information can be transferred to the rest of the system via a digital bus. However, this class of device only represents a small fraction of the total numer of devices on the system.

The vast majority of sensors and actuators onboard a modern spacecraft are very simple devices, such as temperature sensors, that do not have processors. There are frequently hundreds of such devices on a small or medium sized spacecraft, and thousands on large spacecraft! Introducing plug-and-play capabilities to these requires a radical rethink to the way that they are interfaced. To support device identification and enumeration these devices must be equipped with a microcontroller to run the necessary protocol engine and to enable them to be interfaced via a digital data bus. At first sight it appears to be counter-intuitive to add this complexity to every simple device on the spacecraft. But plug-and-play capability in these devices could offer very significant benefits during spacecraft integration and test. Moreover, there is already a compelling argument to equip these simple devices with digital bus interfaces in order to reduce the spacecraft harness. Furthermore, it probably is not necessary to equip every device with a microcontroller. Instead, a single microcontroller might be used as a cluster controller capable of interfacing several simple devices.

From a SOIS perspective, the task specifying plug-and-play capabilities for simple sensor and actuator devices seems to fall under the transducer BoF umbrella. This BoF is not yet chartered but is already looking into the use of digital buses for sensor interfacing. Device identification and enumeration capability for simple devices could be handled within the protocol engine that would already be needed for the digital bus interface. Indeed some of the candidate buses, such as 1-Wire, that are already being considered in the transducer BoF already feature device identification capabilities.