1394 PWG

Plug and Play Task Group Findings

DRAFT

July 2, 1999

Revision 0.1

Lee Farrell, Canon
Mike Fenelon, Microsoft
Laurie Lasslo, Hewlett Packard
Greg LeClair, Epson

Introduction

Requirements for 1394 Plug and Play Device Identification Information.

This document reports on the findings of Plug and Play Task Group within the 1394 PWG. The information contained in this document may be applicable to other device classes and physical interfaces.

Related Standards

- ISO/IEC 13213:1994 / IEEE Std. 1212 CSR and Configuration ROM
- IEEE Std. 1394 High Speed Serial Bus
- ANSI SBP-2 Serial Bus Protocol (Draft – Proposed for profile)
- IEEE P1212r – CSR Revisions, clarifications, FDS keys
- IEEE Std. 1284-1994 – IEEE Standard Signaling Method for a Bidirectional Parallel Peripheral Interface for Personal Computers.

Current Situation

The 1394 PWG has proceeded with the definition of a device profile which incorporates a set of Configuration and Status Registers, a Configuration ROM, the 'SBP-2' protocol and a 1394 PWG defined Transport Command Set to meet the unique requirements for Imaging Devices.

Node Discovery

Current operating systems understand the implementation issues with regards to basic node discovery based on the IEEE Std. 1394-1995. Once a node is discovered on a 1394 Serial Bus, the next level of enumeration to occur is identifying the target device protocol.

Protocol Driver

The SBP-2 protocol implementation can be identified by correct implementation of a Configuration ROM with the appropriate Specifier ID and Software Version in the Unit Directory.

Because SBP-2 encapsulates a specific command set, another level of enumeration must occur. The command set can be identified by correct implementation of a Configuration ROM with the appropriate Command Set Specifier ID and Command Set values in the Unit Directory.

Goals

- Identify scope of the problem
- Identify methods currently used to identify local devices
- Identify methods that would be used in future systems
- Identify the minimum required implementation.
- Identify useful extensions to the minimal implementation.
- Identify method of encoding the information.
- Identify method of accessing the information.
- Note the issues identified as being beyond the scope of this group.

Scope

Device Identification

Device which implement the 1394 PWG Profile may contain one or more functional units. Each of these units may utilize different application layer protocols, command sets and data streams to realize the function.

If the user attempts to connect a device which is not previously known to the client system, it is possible they may be required to install additional software specific to the capabilities of the device.

It is useful to uniquely identify the device, both for the system installation software and also to the user.

Methods Used to Identify Devices

Current Specification

Clause 7.6 of IEEE Std. 1284-1994 requires that a compliant device will supply the MANUFACTURER, COMMAND SET and MODEL keys. These are abbreviated MFG, CMD, and MDL respectively.

Current Systems

Enumerators for the Parallel Port, USB, and IEEE-1394 interfaces will use the ASCII Device ID string originally defined in Clause 7.6 of IEEE Std. 1284-1994. The critical keys and values required are the MANUFACTURER and MODEL entries.

Future Systems

Future enumerators may look for additional information via newer mechanisms under consideration by diferent vendors and standards working groups, however, the existing mechanism (ASCII Device ID string) will continue to be used.

Implementation

Minimal Requirements

To maintain compliance with standards, properly identify the software required and also provide the ability to identify the target device, MANUFACTURER, COMMAND SET and MODEL information is required.

Useful Extensions

CLASS

SYNTAX: CLASS: Class1, Class2, Class3;

The current key of 'CLASS' does not provide adequate definition of usage for devices which may support more than one functional unit (i.e. Scanner / Printer / Fax or Telephone / Answering Machine / Fax)

Keys may have multiple values. In the case of device class, ordering of values has no significance. Multiple values signify the presence of more than one functional unit.

CLASSINFO

SYNTAX: CLASSINFO: Class1;

A mechanism is needed to identify information unique to each device class. One example is the case where a different command / data stream is required for each functional unit. Another example is the vendor unique case of defining new keys and values for a specific device class.

Keys which are specific to a unique functional unit should be listed after the CLASSINFO entry for that unit.

Encoding

Current and future implementations can utilize the ASCII Device ID string as specified in IEEE Std 1284-1994. This can be used in conjunction with the binary values using IEEE Registration Authority keys is specified for IEEE Std 1394-1995.

Accessing

There is no guarantee that the intermediate protocol stack is in a working state at the time this information is required. The minimal requirement is that this information must be available via Read4 transactions.

Issues

Generic 1394 Node Discovery

Not all 1394 nodes will have the same Unit Directory definition. A node which is attempting to enumerate the bus may not be able to identify every device if the target device contains a Unit Directory definition which was not known at the time of the enumerators implementation. This is a typical problem with complex systems composed of components supplied by various manufacturers.

While it may seem reasonable to ignore nodes that contain new and unknown encodings, it ignores the possibility of providing some base level of information to a user to help identify the node. In a bus environment where there can be one or more unknown devices, it would definitely assist in problem isolation.

The recommendation to other working groups would be to consider the usage of at least the minimal ASCII device ID string as a textual descriptor following the Model key in the Root Directory.

Applicability to Other Interfaces

Since the current ASCII device ID string is also used on Parallel Port and USB devices, it is desirable for any definitions adopted here be considered for implementation on devices which are equipped with those interfaces.

Access to Device ID Information Via other means

The larger working group may decide to make this information available via command / response mechanism in the Transport Command Set.

A vendor may opt to provide vendor specific mechanisms to access this Device ID information.

Interface Specific Information

A mechanism is needed to identify unique devices on a specific interface. Some interface standards provide for this information within the specification. An example of this is the EUI-64 value in the Configuration ROM of a 1394 node.

Devices may implement one or more physical interfaces that may or may not be of the same type. It would be useful to understand which interface was used to acquire the string.

The implementation details to meet these requirements should be addressed by interface specific software drivers and not in the device identification string.

Recommendation

Compliance

Encoding as per IEEE Std 1284-1994 Clause 7.6.

Minimal Requirements

Single Function Devices

MANUFACTURER, MODEL, COMMAND SET and CLASS keys

Multiple Function Devices

MANUFACTURER and MODEL keys with CLASS and CLASSINFO keys used to identify COMMAND SET information for each functional unit.

Abbreviations

MANUFACTURER – MFG
MODEL – MDL
CLASS – CLS
CLASSINFO - CLI
COMMAND SET – CMD

Sample Device ID String

Single Function Devices

LENGTH – two bytes
MFG: ACME Manufacturing;
MDL: New Standard Unit;
CLS: PRINTER;
CMD: GenericPDL;

Multiple Function Devices

LENGTH – two bytes
MFG: ACME Manufacturing;
MDL: New Standard Unit;
CLS: PRINTER, SCANNER, FAX;
CLI: PRINTER;
CMD: GenericPCL
CLI: SCANNER;
CMD: GenericSCL
CLI: FAX;
CMD: GenericFCL