# Infrared Data Association

# Plug and Play Extensions

**to**

# Link Management Protocol

**Version 1.1**

**Monday, January 08, 1996**

**Intel Corporation**

**Microsoft Corporation**

## 1. Introduction

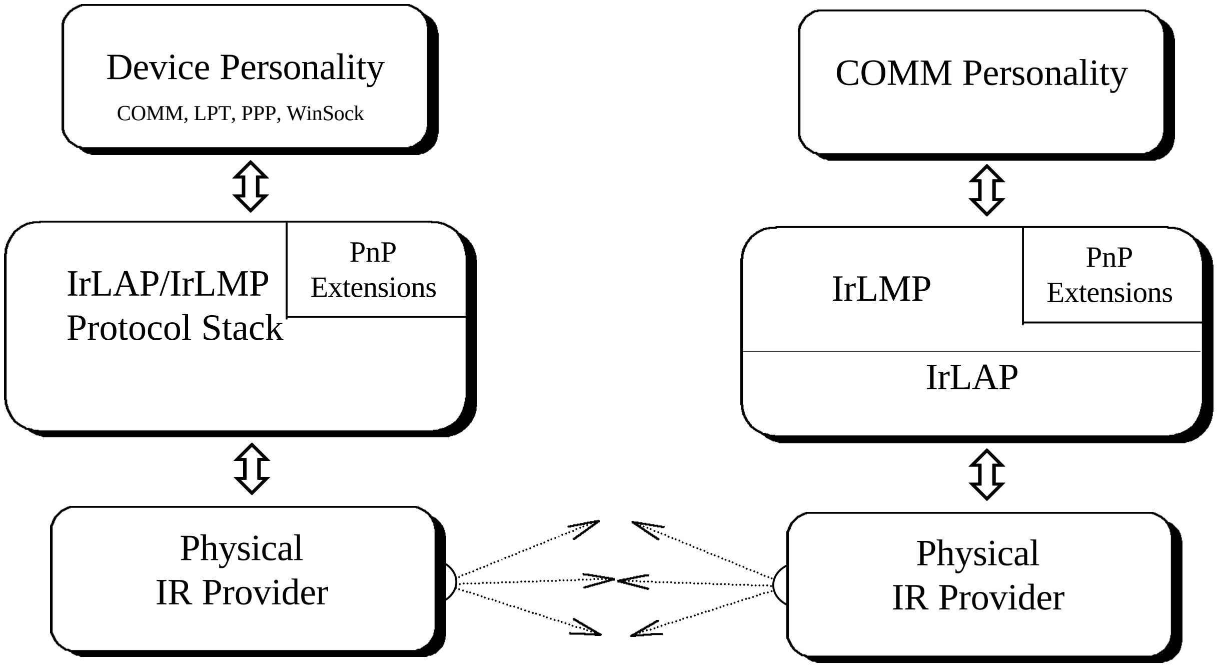
There are two scenarios in IR device inter-operability:

- Communication between a host (PC) and a peripheral.
- Communication between two hosts.

The differentiating feature in the two scenarios is that in the first case, an IR peripheral is logically a part of the host (as far as the user is concerned), while in the other, the two hosts are logically independent units. There are Plug and Play issues in both cases.

### Scope

The software layers proposed by IrDA enable two IR devices to communicate with each other. Communication capability alone does not imply Plug and Play. This document focuses on Plug and Play issues pertaining to host and peripheral inter-operability. This document does not address Plug and Play issues supporting the physical infrared provider. The physical infrared provider is similar to any other physical device and PnP issues of this device are addressed in the standard PnP world.



The above diagram illustrates the software stack on a typical IrDA approved PnP compatible device. The stack pictured on the left portrays a PnP ready OS and the layers required to support a general IrDA PnP implementation.  The stack on the right depicts an IrDA approved device which only needs software enough to support it’s own device.

Plug and Play issues between two hosts will be addressed at a later time. Currently, we do not address Plug and Play with IR peripherals needed at boot, for example, a keyboard.

## References

| | |
|---|---|
| [EISA] | EISA specification version 3.12 |
| [WinID] | Windows Generic Device ID specification |
| [PnP] | Plug and Play Device Driver Developer's Guide |
| [IrLAP] | Infrared Data Association Link Access Protocol |
| [IrLMP] | Infrared Data Association Link Management Protocol |

## 2. Terminology

**Device ID** A Device ID is an uncompressed EISA Device ID.

**EISA Device ID** An EISA device ID is a seven character field consisting of a three-character manufacturer code, a three-character hexadecimal product identifier and a one character hexadecimal revision number.

**Host** In the context of this document, a host is the entity that has discovered an IR device.

**IAS** Information Access Service. This is an information service that must be provided by every IrDA device. The information available via the IAS contains a description of the services being offered by an IrDA device to other IrDA devices.

**IAP** Information Access Protocol, IrLMP defined function set required to access the IAS data.

**IrDA** Infrared Data Association. The Standards body responsible for infrared PC standards.

**IrLAP** Infrared Link Access Protocol

**Initiator** Station performing the IrLAP discovery procedure.

**IrLMP** Infrared Link Management Protocol

**LSAP** Link Service Access Point

**IR Device** An IR entity that responds to the XID discovery process.

**IR Peripheral** An IR device that provides one or more peripheral functions to a host via an IR connection.

**Multi-function Device** A peripheral device that supports more than one logical function. For example a device that provides modem access plus printer services.

**Peripheral** A device external to a Host that provides a specific function to that host.

**Windows Device ID** A Windows Device ID is a PnP Device ID. This list of PnP device ID’s is currently maintained by Microsoft.

**Responder** A station that responds to an IrLAP discovery procedure.

# 3. Plug and Play Issues

Many PnP add-in device issues are not relevant for IR devices. IR devices have no need to contend for physical resources such as IRQ's and IO mappings. However, the IrLAP physical communication device is a standard PnP device consuming physical resources. IR peripherals do need a mechanism for device identification.

For transparency and ease-of-use reasons, connecting to an IR peripheral must be logically similar to connecting to a physically attached peripheral. Once discovered and integrated, the IR peripheral should appear in the host's view of the underlying hardware. In a like manner, when the user removes the peripheral, it should be removed from the host's view. In current Microsoft products, this view typically appears in the Registry.

If an IR device is discovered and needs to be incorporated into the system, the Plug and Play concerns could be classified as:

- device identification
- operational status of the peripheral

There are additional OS Plug and Play issues that are beyond the scope of this document.

### Device Discovery

As part of the device discovery process, the device passes a DeviceInfo field to the host.[IrLAP] The link management layer controls the content of that DeviceInfo field in the IrLAP discovery process. The DeviceInfo field contains a service hint mask which has a single bit field defined that when set, states that the responding IrDA compliant device is Plug and Play ready. See [IrLMP] for DeviceInfo data definition and in particular for the service hint mask definitions. Note that since the service hint field is advisory a non PnP capable device could state that is was a PnP device and the PnP aware software layers must handle the device properly. However a device specifying, via the service hint mask, that it is a PnP device will incur additional overhead due to the software layers attempting to read the *PnP* object.

### Device Identification

After device discovery, the host must determine the capabilities of the device. The IrLMP specification defines an Information Access Service (IAS) to allow devices to exchange information about device capabilities. This specification adds an IAS object for Plug and Play IR devices. The IAS PnP object is defined later in this document. The IAS PnP object contains two attributes that are needed to identify the discovered IR device.

The DeviceID attribute is the key to recognition of devices in the IrPnP arena. A Device ID is an uncompressed EISA device ID. EISA ID's include but are not limited to the Windows generic Device ID's maintained by Microsoft specifically for Plug and Play compatibility. Windows Generic Device ID's are EISA ID's using the reserved "PnP" manufacturer's code. In cases where there exists a definition for a Windows Device ID and a definition for a manufacturer's specific ID, the Windows Device ID takes precedence.

The Category attribute defines the specific class of function supported by the PnP IR Device. This field is semantically the same as the EISA defined CAT field. See [EISA] for known values for the Category attribute.

In order to support multi-function devices two approaches are available:

1. As in the IrPnP 1.0 specification, a singular device ID can be used to imply multiple functions. This approach requires that a unique PnP enumerator be written specifically to support that device. It is then up to the enumerator for that device to load or cause the load of the appropriate drivers for the device at all levels. It is due to the complexity of this problem that the new approach is introduced for this 1.1 version of the IrPnP specification
2. Multiple PnP objects can be stored in the IAS of the device. Using this approach, the standard infrared enumerator can query all the PnP objects on the device and perform the loading of the appropriate devices.

In the case of option number 2, there are various complications:

1. The IrCOMM specification currently provides no way of supporting multiple concurrent IrCOMM channels. As a result a multi-function device that provides both modem access and printer access concurrently can not be supported by the IrCOMM protocol. To alleviate this problem IrCOMM must be extended to support multiple concurrent channels.
2. When a multifunction device includes IrCOMM support, there is currently no way to distinguish between a PnP Id that is referring to a device that is accessed through IrCOMM, such as a modem or a printer, or a PnP Id that is refering to some other non-IrCOMM function, such as LAN access or a disk drive. To alleviate this problem, 3 new attributes are added to the PnP object. These attributes describe the relationship between PnP Ids and functions on the device.

### Operational Status of the Peripheral

During discovery a device can inform the host of device status. This status may reflect the fact that the device is currently busy or in some way unable to support a service connection at this time. Device status values and their meanings are discussed in the ”IAS PnP Objects for Peripherals“ section that follows.

## 4. IAS PnP Objects for Peripherals

The IrLMP specifies Information Access Service as the means to access information about services provided by IR devices. This information is stored as Objects with Object 0, of class *Device*, being the only object required by the IrLMP specification. For Plug and Play purposes, we define a set of distinguished objects, of class *PnP*. These new class object are required to be present on IR devices supporting Plug and Play.

The IAS PnP Object does not contain information about IRQ's, DMA channels and I/O ports. This information is meaningless since the device described by the object is not physically connected to the host.

An Object is composed of a number of name-value pairs. Each name-value pair is an Attribute. The IrLMP specification defines the data format of Attributes in Section 4.3.3. The Attribute Type definitions used in the following object definition are restricted subsets of the IrLMP Attributes. The attribute names are case sensitive, they must appear exactly as listed below with no trailing blanks.

In the case of a single function PnP Class object, the attributes DeviceID, Name, Manufacturer, Category, Version and Status must be included. In the case of a multifunction device, depending on the object, the Function, ParentFunction and ParentInstance may also be required. See the description below for details. An example PnP object is included in Appendix A. The following table defines the Attributes associated with the PnP Object.

Object 1: Class *PnP*

| Attribute Name | PnP Attribute Type |
|---|---|
| DeviceID | IDString |
| Name | VendorString (90) |
| Manufacturer | VendorString (30) |
| Category | VendorString (3) |
| Version | TwoBytes |
| Status | Integer |
| Function | String |
| ParentFunction | String |
| ParentInstance | Integer |
| CompCnt | Integer |
| Comp#01 | IDString |
| . | |
| Comp#nn | IDString |

### *Attribute Types*

The Attribute types described here are all based upon the Attribute value definitions defined in the IrLMP specification (section 4.3.3). They are defined here so that the IrPnP can impose limitations upon IrLMP defined Attributes. The limitations imposed are length restrictions to reduce overhead.

### IDString

The IDString type is a “User String” as defined in the IrLMP Specification limited to 14 Bytes of character data. Notice that the limitation is placed upon the length of the field (or the number of octets in the value). This implies a 14 character limit on ASCII strings and a 7 character limit on Unicode strings. Currently ASCII is the character set used in EISA Device ID’s by convention, but it costs very little to allow for 16 Bit character sets in possible future implementations.

### VendorString (xx)

The VendorString type is a “User String” as defined in the IrLMP Specification limited to the number of bytes of character data specified by the xx parameter. Notice that the limitation is placed upon the length of the field (or the number of octets in the value). This means that usage of 16 bit characters would halve the effective amount of data.

### Integer

This is the “Integer” type as described in the IrLMP specification.

### TwoBytes

This is a restricted case of the “Octet Sequence” described in the IrLMP specification. This is an “Octet Sequence” restricted to exactly 2 Octets.

## *Attributes*

This section describes the usage of the attributes of the PnP Class object.

### DeviceID

This is the DeviceID of the IrDA Play and Play device. The Device ID specified here must be used by the higher level software to determine details describing the peripheral, or peripherals in the case of a multi-function device, supported by that device. Multi function devices are identified by a singular DeviceID. The existence of multiple functions on the device must be gleaned by the host from that singular DeviceID. Multi-function devices must also specify the “Category” attribute to be “MFC”, the EISA multi-function moniker.

IrDA Plug and Play device providers are expected to procure a three letter EISA manufacturer’s identifier and assign IDs for their devices. The specific EISA Device ID is constructed from the EISA manufacturer’s ID and the manufacturer’s device number as described in [EISA]. EISA identifiers may be obtained from:

BCPR Services, Inc
P.O. Box 11137
Spring, Texas 77391-1137
(713)251-4770 (phone)
(713)251-4832 (fax)

In the event that there are no device specific device drivers, the DeviceID provided by the manufacturer may alternatively be a Windows Generic Device ID. This allows the vendor to report hardware-level compatibility with standard devices defined by the Plug and Play Association.

### Name

This is the device specific description. It is identical to the EISA defined field of the same name. This attribute can include up to 90 ASCII characters of text useful for identifying the product. This field is a textual description suitable for user presentation.

The intended use of this field is to provide a mechanism such that the OS, or other software layer, may present to the user a description of a device that has been detected. This will be useful in cases where the OS does not have support for the newly detected device and the user must supply a software update. For example, an OEM supplied driver diskette.

### Manufacturer

Manufacturer's name. It is identical to the EISA defined field MFR. This attribute can include up to 30 ASCII characters of text. This field is a textual description suitable for user presentation. This field is used for the same function as the NAME attribute.

### Category

This attribute contains a three upper case character text field. This field identifies the functional category of the device. This field is semantically the same as the EISA defined CAT field. See [EISA] for known values for the Category attribute.

### Version

The version number of the IrPnP specification supported by the device. This field is provided for compatibility with future versions. This field is made up of exactly 2 octets. The first Octet is the Major Version number and the second octet is the minor version number. For example IrPnP Specification Version 1.0 would consist of the pair (1,0).

### Status

This is a bit field that is used by a device to convey status to the host at discovery time. This is the only mechanism defined to acquire this type of device status. Therefore, in order for a host to determine if a PnP device's status has changed the host must request the IAS PnP Class information.

| Meaning | Bit Mask | Description |
|---|---|---|
| Unavailable | `0x00000001` | If this bit is set to one, then the device is not currently available for use by requesting node. If this bit is set to zero, then the device is available to the requester. |
| Reserved | `0xfffffffe` | Reserved for future use. |

### Function

This attribute reflects the function within the device this PnP Id refers to. The format of this field is the same as the format used in the HintBits field during device discovery. This attribute is only necessary for multi-function devices.

### ParentFunction

This attribute is required in multifunction devices where some of the devices are “children” of other devices. A typical example would be a modem. In this case the modem is a “child” of IrCOMM and this filed would have the COMM support hint bit set. The format of this attribute is the same as the format used in the HintBits field during device discovery.

### ParentInstance

This attribute is required in multifunction devices where some of the devices are “children” of other devices and there is more than one instance of the parent function. A typical example would be a modem. In this case the modem is a “child” of IrCOMM. Since there may be multiple instances of a parent function, such as multiple concurrent COMM connections, this attribute is used to distinguish the parents. If the parent is the first IrCOMM instance than this value will be 0. If the parent is the second IrCOMM instance, then this value will be 1 etc…

### CompCnt

This attribute defines the number of devices that this device is compatible with. The Compatibility ID’s immediately follow this attribute. If the device has no supported compatibility modes then, this attribute must be zero and there will be no Compatibility ID’s in the PnP Class object.

There can be no more than sixteen compatibility modes for any IrDA PnP device. This restriction is imposed to ensure that the discovery process is reasonable performant.

### Comp#nn

This is the PnP DeviceID for the nth standard device that this device is compatible with. A compatible device is used by the OS to identify other device drivers that may be used with the discovered IrPnP device. There must be exactly *CompCnt* Compatibility ID’s listed in the PnP Class object. By returning a compatibility ID, the device is declaring itself to be operational using the device driver of the specified device.

It is important to note that the Comp#nn name includes a variable number. This number is encoded as a two digit decimal number. For example, the third device listed in the compatibility list would have a name encoded as “Comp#03”. Note that there is no “Comp#00” attribute.

## 5. Other Relevant Issues

The following issues are relevant in Plug and Play and must be addressed at implementation time:

### Performance Issues

It is significant to note that the delay imposed upon a system by the discovery mechanism can be substantially affected by the length of the PnP Class Object defined by the vendor. The vendor can add unnecessary data overhead by specifying more compatabilities than needed and by using an excessively long "Name" attribute.

Device manufacturers must ensure that devices respond as quickly as possible since the IrLAP model specifies that a device can send responses only when it is polled by a host. Also, once polled, the host cannot transmit to the device until the device gives permission to the host (by setting the *Final* bit). The permission to transmit must be exchanged such that it is not held at one end for a long time.

### PnP Device ID Management

The EISA Device ID's are industry wide resources that must be managed via a standard clearing house. Microsoft is the current owner of the PnP Device ID pool. Other EISA device ID assignments are made via the EISA standards committee.

# 6. Sequence of Operations

During the normal discovery process, a host will detect various IR devices. The detected devices may or may not be PnP IR devices. In order to determine if a discovered device is a PnP device, the host must query the PnP compatibility bit in the DeviceInfo data returned by the device during the discovery process [IrLMP]. When it determines that a device says it is PnP compatible, the host software must issue an IAP LM-GetValueByClass request for the “DeviceID” attribute of the “PnP” class object. The device is a PnP device only if the request for the “PnP” object returns one or more valid PnP objects.

The steps to install a specific device on a system is implementation dependent, and as such is not discussed in detail here. However, it is important to note that a device’s status will provide an indication to the host that the device, while detected, is not available for use at discovery time.

The Version 1.1 extensions allow the system attaching to the PnP device to understand the relationship between device functions. This eliminates the need for each device class to have its own method of enumerating its children devices. For instance, a device that supports a COM port attached to a modem and concurrently supports LAN access provides 2 PnP objects, one of which is the LAN access PnP Id and another which describes a modem as a child of IrCOMM.

The transition of logical device connections is an implementation issue and must be addressed at another level.

**Other**

The device driver for the detected peripheral is responsible for handling other information such as distinguishing different event types on the peripheral. For example, an IR mouse will have two different types of events: movement of the mouse, and the button clicks, only the device driver need be aware of these different event types.

## 7. Example PnP Class Object

This section contains an example PnP Class object. The data is presented in a manner intended to portray the data that must be transferred for each object. The attribute ordering is not important to the “PnP” class object. The example uses ASCII encoding only.

The table below lists both the names and the values of the device attributes since for each attribute both the name and the value is transmitted. The number below each element of an attribute is the number of bytes of data that must be transmitted for that element. The two columns at the right is the total number of bytes transmitted for that attribute of information and of data. For example the DeviceID attribute consists of nine characters for the user string “DeviceID” (length byte = 8, and 8 characters), the one byte Type field (type=4), the one byte character set (ASCII=0) and eight characters for the EISA ID (length byte=7 and 7 characters) for a total of 9+1+1+8 = 19 bytes of transmitted data.

| Len | Characters | Type | Char Set | Len | Characters | Bytes Transmitted |
|---|---|---|---|---|---|---|
| 8 | DeviceID | 3 | 0 | 7 | ZAP0403 | |
| 1 | 8 | 1 | 1 | 1 | 7 | 19 |
| 4 | Name | 3 | 0 | 55 | Zappa's Special Midi Device, Version 2.7, IrDA Approved | |
| 1 | 4 | 1 | 1 | 1 | 55 | 63 |
| 12 | Manufacturer | 3 | 0 | 17 | Loud Systems Inc. | |
| 1 | 12 | 1 | 1 | 1 | 17 | 33 |
| 8 | Category | 3 | 0 | 3 | OTH | |
| 1 | 8 | 1 | 1 | 1 | 3 | 15 |
| 7 | Comp#01 | 3 | 0 | 7 | PNPB002 | |
| 1 | 7 | 1 | 1 | 1 | 7 | 18 |
| 7 | Comp#02 | 3 | 0 | 7 | PNPB007 | |
| 1 | 7 | 1 | 1 | 1 | 7 | 18 |
| 7 | Comp#03 | 3 | 0 | 7 | PNPB011 | |
| 1 | 7 | 1 | 1 | 1 | 7 | 18 |

| Len | Characters | Type | Integer Value | |
|---|---|---|---|---|
| 6 | Status | 1 | 0 | |
| 1 | 6 | 1 | 4 | 12 |
| 8 | CompCnt | 1 | 2 | |
| 1 | 8 | 1 | 4 | 12 |

| Len | Characters | Type | Len | Octet 1 | Octet 2 | |
|---|---|---|---|---|---|---|
| 7 | Version | 2 | 2 | 1 | 0 | |
| 1 | 7 | 1 | 2 | 1 | 1 | 13 |
| | | | | | Total | 221 |