Infrared Data Association

Minimal IrDA Protocol Implementation (IrDA Lite)

Version 1.0

November 7, 1996

Counterpoint Systems Foundry, Inc.
Actisys Corporation

Authors:

David Suvak (Counterpoint Systems Foundry, Inc.)
Lichen Wang (Actisys Corporation)

Contributors:

Kevin Dodd (Counterpoint Systems Foundry, Inc.)

Document Status

Version 0.1a: This version is a draft proposal.
Version 0.1b. This version added Ultra Lite
Version 0.1c: This version has minor fixes over version 0.1b, removed Ultra Lite, and is the version submitted to the IrDA for final approval.
Version 01.d: This version has typo fixes over version 01.c and replaces 0.1c as the version submitted to the IrDA for final approval.
Version 1.0: This version has been approved by the IrDA. Only minor typo fixes over version 01.d.

INFRARED DATA ASSOCIATION (IrDA) - NOTICE TO THE TRADE -

SUMMARY:

Following is the notice of conditions and understandings upon which this document is made available to members and non-members of the Infrared Data Association.

- Availability of Publications, Updates and Notices
- Full Copyright Claims Must be Honored
- Controlled Distribution Privileges for IrDA Members Only
- Trademarks of IrDA - Prohibitions and Authorized Use
- No Representation of Third Party Rights
- Limitation of Liability
- Disclaimer of Warranty
- Certification of Products Requires Specific Authorization from IrDA after Product Testing for IrDA Specification Conformance

IrDA PUBLICATIONS and UPDATES:

IrDA publications, including notifications, updates, and revisions, are accessed electronically by IrDA members in good standing during the course of each year as a benefit of annual IrDA membership. Electronic copies are available to the public on the IrDA web site located at irda.org. IrDA publications are available to non-IrDA members for a pre-paid fee. Requests for publications, membership applications or more information should be addressed to: Infrared Data Association, P.O. Box 3883, Walnut Creek, California, U.S.A. 94598; or e-mail address: info@irda.org; or by calling John LaRoche at (510) 943-6546 or faxing requests to (510) 934-5600.

COPYRIGHT:

1. Prohibitions: IrDA claims copyright in all IrDA publications. Any unauthorized reproduction, distribution, display or modification, in whole or in part, is strictly prohibited.

2. Authorized Use: Any authorized use of IrDA publications (in whole or in part) is under NONEXCLUSIVE USE LICENSE ONLY. No rights to sublicense, assign or transfer the license are granted and any attempt to do so is void.

DISTRIBUTION PRIVILEGES for IrDA MEMBERS ONLY:

IrDA Members Limited Reproduction and Distribution Privilege: A limited privilege of reproduction and distribution of IrDA copyrighted publications is granted to IrDA members in good standing and for sole purpose of reasonable reproduction and distribution to non-IrDA members who are engaged by contract with an IrDA member for the development of IrDA certified products. Reproduction and distribution by the non-IrDA member is strictly prohibited.

TRANSACTION NOTICE to IrDA MEMBERS ONLY:

Each and every copy made for distribution under the limited reproduction and distribution privilege shall be conspicuously marked with the name of the IrDA member and the name of the receiving party. Upon reproduction for distribution, the distributing IrDA member shall promptly notify IrDA (in writing or by e-mail) of the identity of the receiving party.

A failure to comply with the notification requirement to IrDA shall render the reproduction and distribution unauthorized and IrDA may take appropriate action to enforce its copyright, including but not limited to, the termination of the limited reproduction and distribution privilege and IrDA membership of the non-complying member.

TRADEMARKS:

1. Prohibitions: IrDA claims exclusive rights in its trade names, trademarks, service marks, collective membership marks and certification marks (hereinafter collectively "trademarks"), including but not limited to the following trademarks: INFRARED DATA ASSOCIATION (wordmark alone and with IR logo), IrDA (acronym mark alone and with IR logo), IR logo, IR DATA CERTIFIED (composite mark), and MEMBER IrDA (wordmark alone and with IR logo). Any unauthorized use of IrDA trademarks is strictly prohibited.

2. Authorized Use: Any authorized use of a IrDA collective membership mark or certification mark is by NONEXCLUSIVE USE LICENSE ONLY. No rights to sublicense, assign or transfer the license are granted and any attempt to do so is void.

NO REPRESENTATION of THIRD PARTY RIGHTS:

IrDA makes no representation or warranty whatsoever with regard to IrDA member or third party ownership, licensing or infringement/non-infringement of intellectual property rights. Each recipient of IrDA publications, whether or not an IrDA member, should seek the independent advice of legal counsel with regard to any possible violation of third party rights arising out of the use, attempted use, reproduction, distribution or public display of IrDA publications.

IrDA assumes no obligation or responsibility whatsoever to advise its members or non-members who receive or are about to receive IrDA publications of the chance of infringement or violation of any right of an IrDA member or third party arising out of the use, attempted use, reproduction, distribution or display of IrDA publications.

LIMITATION of LIABILITY:

BY ANY ACTUAL OR ATTEMPTED USE, REPRODUCTION, DISTRIBUTION OR PUBLIC DISPLAY OF ANY IrDA PUBLICATION, ANY PARTICIPANT IN SUCH REAL OR ATTEMPTED ACTS, WHETHER OR NOT A MEMBER OF IrDA, AGREES TO ASSUME ANY AND ALL RISK ASSOCIATED WITH SUCH ACTS, INCLUDING BUT NOT LIMITED TO LOST PROFITS, LOST SAVINGS, OR OTHER CONSEQUENTIAL, SPECIAL, INCIDENTAL OR PUNITIVE DAMAGES. IrDA SHALL HAVE NO LIABILITY WHATSOEVER FOR SUCH ACTS NOR FOR THE CONTENT, ACCURACY OR LEVEL OF ISSUE OF AN IrDA PUBLICATION.

DISCLAIMER of WARRANTY:

All IrDA publications are provided "AS IS" and without warranty of any kind. IrDA (and each of its members, wholly and collectively, hereinafter "IrDA") EXPRESSLY DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND WARRANTY OF NON-INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS.
IrDA DOES NOT WARRANT THAT ITS PUBLICATIONS WILL MEET YOUR REQUIREMENTS OR THAT ANY USE OF A PUBLICATION WILL BE UN-INTERRUPTED OR ERROR FREE, OR THAT DEFECTS WILL BE CORRECTED. FURTHERMORE, IrDA DOES NOT WARRANT OR MAKE ANY REPRESENTATIONS REGARDING USE OR THE RESULTS OR THE USE OF IrDA PUBLICATIONS IN TERMS OF THEIR CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE. NO ORAL OR WRITTEN PUBLICATION OR ADVICE OF A REPRESENTATIVE (OR MEMBER) OF IrDA SHALL CREATE A WARRANTY OR IN ANY WAY INCREASE THE SCOPE OF THIS WARRANTY.

LIMITED MEDIA WARRANTY:

IrDA warrants ONLY the media upon which any publication is recorded to be free from defects in materials and workmanship under normal use for a period of ninety (90) days from the date of distribution as evidenced by the distribution records of IrDA. IrDA's entire liability and recipient's exclusive remedy will be replacement of the media not meeting this limited warranty and which is returned to IrDA. IrDA shall have no responsibility to replace media damaged by accident, abuse or misapplication. ANY IMPLIED WARRANTIES ON THE MEDIA, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO NINETY (90) DAYS FROM THE DATE OF DELIVERY. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM PLACE TO PLACE.

CERTIFICATION and GENERAL:

Membership in IrDA or use of IrDA publications does NOT constitute IrDA compliance. It is the sole responsibility of each manufacturer, whether or not an IrDA member, to obtain product compliance in accordance with IrDA rules for compliance.

All rights, prohibitions of right, agreements and terms and conditions regarding use of IrDA publications and IrDA rules for compliance of products are governed by the laws and regulations of the United States. However, each manufacturer is solely responsible for compliance with the import/export laws of the countries in which they conduct business. The information contained in this document is provided as is and is subject to change without notice.

TABLE OF CONTENTS

1. Introduction

1.1 Purpose

IrDA Lite is a set of ideas on how to reduce the complexity and code size of the IrDA protocols while maintaining the capability to communicate with existing IrDA devices. The intent of these ideas is to provide a method for devices with simple data communication needs to implement IrDA protocols with a minimum of complexity in a small amount of RAM and ROM.

This document focuses on the minimal set of capabilities required by a device to maintain compatibility with the IrDA specifications. Even though this document focuses on the minimum, implementations are not restricted from adding more capabilities. The IrDA Lite ideas do not need to be taken in their entirety. Implementations are free to incorporate as few or as many IrDA Lite ideas as desired. Implementations that follow the guidelines in this specification will result in the most minimal set of capabilities allowed by IrDA.

1.2 Scope

This specification is intended to be a companion document to the IrDA IrLAP and IrLMP specifications (see section 0 1.3 References). Where as the IrLAP and IrLMP specifications provide the complete description of the respective protocols IrDA Lite discusses how IrLAP and IrLMP can be made as minimal as possible and yet retain the ability to inter-operate with fully functional implementations. This document attempts to duplicate as little information in the other specifications as possible. Therefore, the reader of this specification should be familiar with the IrLAP and IrLMP specifications and be able to refer to those documents when needed.

The rest of this document is divided into sections as follows:

- Section 2 describes the minimum set of IrLAP capabilities including the reduced IrLAP state machines.
- Section 3 describes the minimum set of IrLMP multiplexer capabilities. By taking a different approach in describing the multiplexer this section reduces the IrLMP multiplexer description to two simple state machines.
- Section 4 describes the minimum IrLMP Information Access Service. This includes ideas that have impact on full featured implementations.

1.3 References

IRDALAP	Serial Infrared Link Access Protocol, IrLAP, Version 1.1, Infrared Data Association
IRDALMP	Link Management Protocol, IrLMP, Version 1.1, Infrared Data Association

1.4 State Machine Notation

A number of actions and predicates in the Actions and Events columns of the State Charts make use of set variables. The following set operations are commonly used

$\wedge$	Logical ANDing of predicates
$\vee$	Logical ORing of predicates
$\neg$	Logical NOT of predicate
$\varnothing$	Empty set
$\cup$	Set Union
$\in$	Element of a set
$\notin$	Not an element of a set

2. Minimum IrLAP Capabilities

This section describes the minimum IrLAP capabilities for secondary and primary devices separately. The secondary description is useful for “Secondary Only” devices. The primary description covers the procedures unique to operation initiators and primary devices. Primary devices usually also act as Secondaries especially if devices of the same type are to communicate with each other therefore, a primary device that needs secondary capabilities would combine the two state machines. Before the secondary and primary descriptions are given, general ideas and guidelines common to both are described.

2.1 General NDM Ideas

Below is a list of ideas and guidelines used in the NDM state.

1. Provide no support for optional NDM features including sniffing, Connectionless UI frames, and TEST frames

2. Ignore all frames that do not have the broadcast address.

3. Ignore all frames other than XID, SNRM, and UA that are addressed to the device. Do not send DM response just ignore the frame completely.

4. To save RAM do not store received nicknames in the discovery table.

5. Do not send or maintain a nickname for discovery.

2.2 General NRM Ideas

Below is a list of ideas and guidelines associated with connections

1. Use default parameters:

Baud rate	9600 bps
Data Size	64 bytes
Window size	1
Additional BOFs	0
Max Turn around time	500ms
Disconnect/Threshold time	3 seconds

1. Set the correct min turn around time for your device but always give the other device 10ms. The easiest way to do this is to always send 11 BOFs (see IRDALAP for proper values for 11 BOFs).

2. Because minimal link parameters are used the negotiation parameters can be fixed. No code is needed for negotiation just record the parameters of the other device.

3. In NRM filter out all frames that do not have the connection address including the broadcast address. These can be completely ignored.

4. Do not check for invalid N(s) treat it as unexpected N(s).

5. Do not support the RESET states, just perform a link disconnect.

6. U-Frame handling: Ignore XID, UI frames and any undefined frames (respond to P/F bit by sending S or I frame).

2.3 Minimum Secondary Only

This section describes the minimum secondary only capabilities. First a description of the secondary only ideas and guidelines is given, next the service primitives are described and finally a single state machine is given covering all IrLAP procedures for a secondary only device.

2.3.1 Secondary Only Ideas and Guidelines

Below is a list of guidelines and ideas for Secondary only devices

1. The Lite Secondary does not use the REPLY state nor the query-timer during discovery. It uses a simpler method to detect the beginning/ending of discovery procedure when some of the discovery command slots are missed. Under certain rare circumstances, this may cause it to behave not identically to what IrLAP version 1.1 specifies.

2. The Lite Secondary does not generate a discovery indication.

3. The Lite Secondary ignores all incoming commands with P-bit cleared. It always sends responses with F-bit set. Since the Lite Secondary negotiates to use windows = 1, IrLAP 1.0/1.1 compliant Primary devices should rarely (if ever) send I-Frame commands with P-bit cleared. If a case occurs, then ignored I-Frame commands will automatically be detected and recovered by the sequence checking. Given this, the Lite Secondary only needs to be able to receive one frame at a time.

4. The Lite Secondary ignores all incoming response frames while connected. These frames may indicate another connection exists with the same connection address but this case is very rare so the secondary will just let its watchdog timer catch the problem.

5. The Lite Secondary does not use a DM to refuse an incoming IrLAP connection (SNRM) instead it just ignores the SNRM. Most devices will not refuse an incoming IrLAP connection anyway.

2.3.2 Secondary Only Service Primitives

2.3.2.1 Connect services

- IrLAP_CONNECT.indication()
- IrLAP_CONNECT.response()

Description: The indication primitive indicates that another device is attempting to make a connection. The upper layer will accept the connection via the response primitive or deny the connection with the IrLAP_DISCONNECT.request primitive.

2.3.2.2 Data Services

IrLAP_DATA.request (User-Data)
IrLAP_DATA.indication (User-Data)

Description: The request primitive is used to send reliable User-Data (I-frames). The indication passes received User-Data to the upper layer.

2.3.2.3 Disconnect Services

IrLAP_DISCONNECT.request()
IrLAP_DISCONNECT.indication()

Description: The request primitive is used to refuse an incoming connection (IrLAP_CONNECT.indication) or to disconnect the link. The indication primitive tells the upper layer that the link has been disconnected.

2.3.3 Secondary Only State Machine

2.3.3.1 State Chart

Current State	Event	Action(s)	Next State
OFFLINE (entry state)	link-initialize	NA := Generate-Random-Device-Address; Apply-Default-Connection-Parameters; old_s := 127;	NDM
NDM	link-shutdown		OFFLINE

	Recv Discovery-XID-Cmd:S,s, addressConflictFlag	if s ≤ old_s {
		slot := Generate-Random-Time-Slot(S,s);
		frameSent := false;
		}
		old_s := s;
		if (s ≥ slot) ∧ (s != 0xff) ∧ ¬frameSent {
		if addressConflictFlag = 1 {
		NA := Generate-Random-Device-Address;
		}
		Send-Discovery-XID-Rsp:NA,discovery-info;
		frameSent := true;
		}
			NDM

	Recv End-Discovery-XID-Cmd	old_s := 127;	NDM
	Recv u:snrm:cmd:P:c:d	dest := d; ca := c; Connect-Indication;	CONN
	Recv x:x:x:x		NDM
CONN	Disconnect-Request		NDM
	Connect-Response	Negotiate-Connection-Parameters; Send u:ua:rsp:F:Connection-Parameters; Apply-Connection-Parameters; Initialize-Connection-State; Start-WD-timer for quick disconnect;	NRM
NRM(S)	Data-Request	Queue-Data;	NRM(S)
	Disconnect-Request		SCLOSE

	Recv I:cmd:Ns:Nr:P	if ¬localBusy {
		  if (Ns == Vr) {
		  Vr = Vr + 1 mod 8;
		  Data-Indication;
		  Send-NRM-Response;
		  } else {
		  Send-S-Response;
		  }
		} else {
		  Send-NRM-Response;
		}
		Start-WD-timer
			NRM(S)

	Recv I:cmd:Ns:Nr:P ∧ Invalid-Nr	Send u:rd:rsp:F; Start-WD-timer;	SCLOSE
	Recv s:rr:cmd:Nr:P	remoteBusy := false; Send-NRM-Response; Start-WD-timer	NRM(S)
	Recv s:rnr:cmd:Nr:P	remoteBusy := true; Send-NRM-Response; Start-WD-timer	NRM(S)
	Recv s:rej:cmd:Nr:P ∨ Recv s:srej:cmd:Nr:P	Send-NRM-Response; Start-WD-timer	NRM(S)
	Recv s:x:cmd:Nr:P ∧ Invalid-Nr	Send u:rd:rsp:F; Start-WD-timer;	SCLOSE
	Recv u:snrm:cmd:P	Send u:rd:rsp:F; Start-WD-timer;	SCLOSE
	Recv u:disc:cmd:P	Send u:ua:rsp:F; Apply-Default-Connection-Parameters; old_s := 127; Disconnect-Indication;	NDM
	Recv x:x:cmd:P	Send-S-Response;	NRM(S)
	Recv x:x:x:x		NRM(S)
	WD-timer-expired	Apply-Default-Connection-Parameters; old_s := 127; Disconnect-Indication;	NDM
SCLOSE	Recv u:disc:cmd:P	Send u:ua:rsp:F; Apply-Default-Connection-Parameters; old_s := 127; Disconnect-Indication;	NDM
	Recv x:x:cmd:P	Send u:rd:rsp:F Start-WD-timer	SCLOSE
	WD-timer-expired	Apply-Default-Connection-Parameters; old_s := 127; Disconnect-Indication;	NDM
	Recv x:x:x:x		SCLOSE

2.3.3.2 **State Definitions**

OFFLINE. The station is powered off, not initialized or disabled from operating in the infrared physical medium. (In all other states, the station is powered on, initialized and able to send or receive IrLAP frames.)

NDM. The station is in the normal disconnected mode. It can respond to remote discovery and address resolution procedure requests, and it can respond to remote requests to connect with a remote peer layer.

CONN. A SNRM command has been received from a remote peer layer, the service user has been informed and the local layer is awaiting the service user's refusal or acceptance of the connection.

NRM(S). The station is in the normal response (connected) mode playing the secondary role.

SCLOSE. The station has transmitted an RD frame to the primary and waiting for DISC command from the primary station.

2.3.3.3 Event Descriptions

Link-Initialize. Station user has initialized/enabled the station.

Link-Shutdown. Station user has disabled the station.

Recv Discovery-XID-Cmd:S,s,addressConclictFlag. A discovery command has been received from a remote peer layer. *S* indicates the total number of time slots the discovery procedure will use, *s* indicates the number of the current time slot. addressConflictFlag is bit-2 of the discoveryFlags. It indicates whether the responder needs to generate a new 32-bit device address.

Recv End-Discovery-XID-Cmd. An end of discovery procedure command has been received from a remote peer layer. This command is similar to the general discovery XID command, it uses the general XID command format. However, the slot number field is set to X'FF', and an *info* (hints) field that describes the capabilities of the discovery initiator is included.

Recv u:snrm:cmd:P:c:d. A SNRM command with the P bit set has been received from a remote peer layer. c denotes the connection address and d denotes the destination address.

Recv x:x:x:x. A command or response frame not specifically listed for the state, including unknown or invalid frames, has been received.

Data-Request. The service user has requested that reliable data frame (I-Frame) be sent.

Disconnect-Request. The service user has requested that the connection be terminated.

Connect-Response. The service user has accepted a remote connection request.

Recv i:cmd:Ns:Nr:P. An I command has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

Recv s:rr:cmd:Nr:P. An RR command has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

Recv s:rnr:cmd:Nr:P. An RNR command has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

Recv s:rej:cmd:Nr:P∨ Recv s:srej:cmd:Nr:P. An REJ or SREJ command has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

Recv s:x:cmd:Nr:P. An S frame (REJ, RR, RNR, SREJ) command has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

Recv u:snrm:cmd:P. An SNRM command has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

Recv u:disc:cmd:P. A DISC command has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

Recv x:x:cmd:P. A command frame not specifically listed for the state has been received from the primary station. The P bit was set indicating the secondary may now transmit frames.

WD-timer-expired. The watchdog bit timer has expired.

Invalid-Nr. The Nr field of the received frame is invalid. The following algorithm returns TRUE if the Nr is invalid:

```
If (Nr == Vs) ∨ (Unacked-Frame exists ∧ (Nr == As)) {
   return FALSE
} else {
   return TRUE
}
```

2.3.3.4 Action Descriptions

NA := Generate-Random-Device-Address. Generate a random 32-bit device address and assign it to "NA" as this station's device address.

Apply-Default-Connection-Parameters. Configure IrLAP layer to use the default connection and transmission parameters. This only needs to be done once since defaults are always used.

old_s := 127 . The variable 'old_s' is used to detect the beginning of a new discovery procedure. The condition 's ≤ old_s' is an indication of a new discovery procedure even when some of the slots are not correctly received. 'old_s' is initialized to 127. It is set to s after any slot is correctly received.

slot := Generate-Random-Time-Slot(S,s). Generate a random time slot number between *s* and *S*-1 and save it in 'slot'.

framSent := false. frameSent is a Boolean variable used to indicate whether a discovery XID response frame has already been transmitted.

Send-Discovery-XID-Rsp:NA,discovery-info. The local layer transmits a discovery response XID frame containing its 32-bit device address, NA, and its discovery info string.

Connect-Indication. Inform the service user that a connection has been requested by a remote peer layer.

Send u:dm:rsp:F. Send a DM response frame.

Negotiate-Connection-Parameters. Since the secondary parameters are already the defaults they are ready to send. There is really nothing to do.

Send u:ua:rsp:F:Connection-Parameters. Send a UA response frame with the parameters determined by the last *Negotiate-Connection-Parameters* action executed (parameter string can be fixed).

Apply-Connection-Parameters. Since defaults are used and the secondary has a fixed behavior (e.g. always sends 11 BOFs) there is nothing to do.

Initialize-Connection-State. Initialize the connection state variables:

```
remoteBusy := localBusy := false;
Vr := Vs := As := 0;
```

Queue-Data. Frames from the user layer are placed on a queue to be sent when the station has control of the link.

remoteBusy is a Boolean variable. It is set to true when an RNR command is received, and set to false when an RR command is received.

localBusy is a Boolean variable. It is set to true when the local device is unable to receive I-Frames, and set to false otherwise.

Vr is the expected next I-command sequence number to be received.

Vs is the expected next I-response sequence number to be sent.

As is the I-response sequence number already sent.

Unacked-Frame holds an I-Frame that has been sent but not acknowledged. If it is 0 then there are no unacknowledged I-Frames.

Start-WD-timer for quick disconnect. Start the watchdog timer from zero. Set it to about one second or so.

Data-Indication. Pass the information field of a received I frame to the service user.

Vr := Vr + 1 mod 8. Increment Vr (modulo 8) to get the sequence number of the next I frame expected to receive.

Send-NRM-Response. The following actions are carried out:

```
if ¬remoteBusy {
   if (Nr == As) {
      Send i:rsp:As:Vr:F:Unacked-Frame-data;
      Unacked-Frame = 0;
   } else if (Data in Queue) ∧ (Nr == Vs) {
      Send i:rsp:Vs:Vr:F:data;
      Unacked-Frame := I-Frame Sent
      As := Vs;
      Vs := Vs + 1 mod 8;
   } else {
      Send-S-Response
   }
} else {
   Send-S-Response
}
```

Send-S-Response The following actions are carried out:

```
if localBusy {
   Send s:rnr:rsp:Vr:F;
```

```
} else {
   Send s:rr:rsp:Vr:F;
}
```

Send u:rd:rsp:F. Send a RD response to request disconnect.

Start-WD-timer. Start the watchdog timer from zero. Set it to the negotiated link disconnect time.

2.4 Minimum Primary

This section describes the minimum primary capabilities. First a description of the primary ideas and guidelines is given, next the service primitives are described and finally a single state machine is given covering all IrLAP primary procedures.

2.4.1 Primary Ideas and Guidelines

1. The Lite initiator of discovery uses the same procedures for both discovery and address conflict resolution. The "Generate New Address" field is passed in the IrLAP_DISCOVERY.request.

2. The Lite initiator of discovery limits the discovery operation to 6 slots and it does not record collisions.

3. The Lite Primary only tries once (sends one SNRM) to initiate a connection if that fails to solicit an UA response the connection operation fails. Since it only sends one SNRM it is not possible for SNRM collision to occur because the MediaBusy condition would stop two primaries from both sending SNRMs.

4. The Lite Primary only looks for UA or F-timer expiring in the SETUP state. It ignores all other frames. It does not even look for a DM or DISC which could abort the connection instead it lets the F-Timer expiration catch these.

5. The Lite Primary ignores all incoming responses with F-bit cleared. It always sends commands with P-bit set. Since the Lite Primary negotiates to use windows = 1, IrLAP 1.0/1.1 compliant Secondary devices should rarely (if ever) send I-Frame responses with F-bit cleared. If a case occurs, then ignored I-Frame responses will automatically be detected and recovered by the sequence checking. Given this, the Lite Primary only needs to be able to receive one frame at a time.

2.4.2 Primary Service Primitives

2.4.2.1 Discovery and Address Conflict Services

- IrLAP_DISCOVERY.request(GenAddrBit)
- IrLAP_DISCOVERY.confirm (List-of-Discovery-Logs)

Description: The request primitive is used to initiate a discovery process using 6 slots. The GenAddrBit specifies the actual value of the "Generate New Address" bit of the XID command frame. If set to one an Address Conflict Resolution procedure is performed. The confirm primitive signals completion of the discovery process and returns the list of discovered device as List-of-Discovery-Logs. Each Discovery Log contains the device address and hints field. The nickname is not returned to save RAM.

2.4.2.2 Connect services

- IrLAP_CONNECT.request(Target-Device-Address)
- IrLAP_CONNECT.confirm()

Description: The request primitive is used to request a connection to the device specified by Target-Device-Address. The confirm primitive signals a successful completion of the connection. IrLAP_DISCONNECT.indication signals a failed connection..

2.4.2.3 Data Services

IrLAP_DATA.request (User-Data)
IrLAP_DATA.indication (User-Data)

Description: The request primitive is used to send reliable User-Data (I-frames). The indication passes received User-Data to the upper layer.

2.4.2.4 Disconnect Services

IrLAP_DISCONNECT.request()
IrLAP_DISCONNECT.indication()

Description: The request primitive is used to refuse an incoming connection (IrLAP_CONNECT.indication) or to disconnect the link. The indication primitive tells the upper layer that the link has been disconnected.

2.4.3 Primary State Machine

2.4.3.1 State Chart

Current State	Event	Action(s)	Next State
OFFLINE (entry state)	link-initialize	NA := Generate-Random-Device-Address; Apply-Default-Connection-Parameters;	NDM
NDM	link-shutdown		OFFLINE
	Discovery-Request(Gen) ∧ mediaBusy = false	slotCount := 0 Send Discovery-XID-Cmd:6,Gen,slotCount Start-slot-timer log := {∅}	QUERY
	Discovery-Request(Gen) ∧ mediaBusy = true	Discovery-Confirm(media-busy)	NDM
	Connect-Request(da) ∧ mediaBusy = false	Generate-Random-ConnectionAdr(ca); dest := da Send u:snrm:cmd:P:ca:dest Start-F-timer	SETUP
	Connect-Request(da) ∧ mediaBusy = true	Disconnect-Indication(media-busy)	NDM
	Recv x:x:x:x		NDM
QUERY	slot-timer-expired ∧ slotCount < 6	slotCount := slotCount + 1 Send Discovery-XID-Cmd:6,Gen,slotCount Start-slot-timer	QUERY
	slot-timer-expired ∧ slotCount ≥ 6	Send End-Discovery-XID-Cmd Discovery-Confirm(log)	NDM
	RecvDiscovery-XID-Rsp:sa,info	log := log ∪ {<sa,info>}	QUERY
	Recv x:x:x:		QUERY

SETUP	Recv u:ua:rsp:F	Initialize-Connection-State; Apply-Connection-Parameters Connect-Confirm Send s:rr:cmd:P retryCount := 0 Start-F-timer	NRM(P)
	F-timer-expired	Disconnect-Indication;	NDM
	Recv x:x:x:x		SETUP
XMIT	Data-Request	Queue-Data;	XMIT
	Disconnect-Request	Stop-P-timer Send u:disc:cmd:P Start-F-Timer	PCLOSE
	P-timer expires	Send-NRM-Command	NRM(P)
NRM(P)	Data-Request	Queue-Data;	NRM(P)
	Disconnect-Request	Leave Pending until XMIT	NRM(P)

	Recv I:rsp:Ns:Nr:F	Stop-F-timer
		Process-NR
		if ¬localBusy {
		  if (Ns == Vr) {
		  Vr = Vr + 1 mod 8;
		  Data-Indication;
		  }
		}
		Start-P-Timer
			XMIT

	Recv I:rsp:Ns:Nr:F ∧ Invalid-Nr	Send u:disc:cmd:P; retryCount := 0 Start-F-timer;	PCLOSE
	Recv s:rr:rsp:Nr:F	Stop-F-Timer Process-NR remoteBusy := false; Start-P-Timer	XMIT
	Recv s:rnr:rsp:Nr:F	Stop-F-Timer Process-NR remoteBusy := true; Start-P-Timer	XMIT
	Recv s:rej:rsp:Nr:F ∨ Recv s:srej:rsp:Nr:F	Stop-F-Timer Process-NR Start-P-Timer	XMIT
	Recv s:x:rsp:Nr:F ∧ Invalid-Nr	Send u:disc:cmd:P; retryCount := 0 Start-F-timer;	PCLOSE
	Recv u:rd:rsp:F	Send u:disc:cmd:P; retryCount := 0 Start-F-timer;	PCLOSE
	Recv u:frmr:rsp:F	Send u:disc:cmd:P; retryCount := 0 Start-F-timer;	PCLOSE
	Recv x:x:rsp:F	Send-S-Command Start-F-Timer	NRM(P)
	Recv s:x:cmd:x ∨ Recv i:cmd:x:x:x	Apply-Default-Connection-Parameters; Disconnect-Indication(PrimaryConflict);	NDM

	F-timer expired ∧ retryCount < N1	Send-S-Command retryCount := retryCount + 1 Start-F-Timer	NRM(P)
	F-timer-expired ∧ retryCount ≥ N1	Apply-Default-Connection-Parameters; Disconnect-Indication;	NDM
	Recv x:x:x:x		NRM(P)
PCLOSE	Recv u:ua:rsp:F	Apply-Default-Connection-Parameters; Disconnect-Indication;	NDM
	F-timer expired ∧ retryCount < N2	Send u:disc:cmd:P retryCount := retryCount + 1 Start-F-timer	PCLOSE
	F-timer-expired ∧ retryCount ≥ N2	Apply-Default-Connection-Parameters; Disconnect-Indication;	NDM
	Recv x:x:x:x		PCLOSE

2.4.3.2 State Definitions

OFFLINE. The station is powered off, not initialized or disabled from operating in the infrared physical medium. (In all other states, the station is powered on, initialized and able to send or receive IrLAP frames.)

NDM. The station is in the normal disconnected mode. It will act based on requests to perform discovery and address conflict resolution and establish connections.

QUERY The local layer is currently executing the discovery procedure. It has transmitted a discovery XID command frame and is currently transmitting the time slot XID frames and logging any XID responses that are received within the time slots.

SETUP The SNRM command has been sent to a remote peer layer, the local layer is waiting for a UA response or a timeout.

XMIT. The primary station has the right to transmit an IrLAP command frame. It does not expect to receive any frames because no secondary has the right to transmit.

NRM(P). The station is in the normal response (connected) mode playing the primary role.

PCLOSE. The station has transmitted a DISC frame to the secondary and waiting for an UA response from the secondary station.

2.4.3.3 Event Descriptions

Link-Initialize. Station user has initialized/enabled the station.

Link-Shutdown. Station user has disabled the station.

Discovery-Request(Gen). The service user has requested that a discovery procedure be started. Gen indicates whether the discovery process is normal or an address conflict resolution procedure.

MediaBusy = true. The IR medium is busy. Some other IR activity exists.

MediaBusy = false. The IR medium is available to use.

Connect-Request(da). The service user has requested that a connection be established to the remote device with device address da.

Recv x:x:x:x. A command or response frame not specifically listed for the state, including unknown or invalid frames, has been received.

Slot-timer-expired ∧ slotCount < 6 The slot timer expired and the slotCount is less than the max number of slots 6.

Slot-timer-expired ∧ slotCount ≥ 6 The slot timer expired and the slotCount is equal to or greater than the maximum number of slots 6.

RecvDiscovery-XID-Rsp:sa,info An XID discovery response frame has been received with source device address "sa" and an info field specified by "info".

Recv u:ua:rsp:F. An UA frame has been received from the secondary station.

F-timer-expired - The primary's Final bit timer has expired.

Data-Request. The service user has requested that reliable data frame (I-Frame) be sent.

Disconnect-Request. The service user has requested that the connection be terminated.

P-timer expires. The primary's poll timer has expired. It is time to poll the secondary.

Recv i:rsp:Ns:Nr:F. An I response has been received from the secondary station. The F bit was set indicating the primary may now transmit frames.

Recv s:rr:rsp:Nr:F. An RR response has been received from the secondary station. The F bit was set indicating the primary may now transmit frames.

Recv s:rnr:rsp:Nr:F. An RNR response has been received from the secondary station. The F bit was set indicating the primary may now transmit frames.

Recv s:rej:rsp:Nr:F ∨ Recv s:srej:rsp:Nr:F. An REJ or SREJ response has been received from the secondary station. The F bit was set indicating the primary may now transmit frames.

Recv s:x:rsp:Nr:F. An S frame (REJ, RR, RNR, SREJ) response has been received from the secondary station. The F bit was set indicating the primary may now transmit frames.

Recv u:rd:rsp:F. A RD response has been received from the secondary station. The secondary is requesting that the primary disconnect the link. The F bit was set indicating the primary may now transmit frames.

Recv u:frmr:rsp:F. A FRMR response has been received from the secondary station. The secondary is rejecting something sent by the primary which implies that a problem exists. The primary will disconnect the link.

Recv x:x:rsp:F. A response frame not specifically listed for the state has been received from the secondary station. The P bit was set indicating the primary may now transmit frames.

Recv s:x:cmd:x ∨ Recv i:cmd:x:x:x. An S- or I-command has been received which probably indicates another primary exists.

F-timer expired ∧ retryCount < N1. The F-timer has expired but the disconnect threshold has not been met yet. N1 is the number of times the F-timer expires to reach 3 seconds.

F-timer expired ∧ retryCount ≥ N1. The F-timer has expired and the disconnect threshold has been met. N1 is the number of times the F-timer expires to reach 3 seconds.

Invalid-Nr. The Nr field of the received frame is invalid. The following algorithm returns TRUE if the Nr is invalid:

```
If (Nr == Vs) ∨ (Unacked-Frame exists ∧ (Nr == As)) {
   return FALSE
} else {
   return TRUE
}
```

F-timer expired ∧ retryCount < N2. The F-timer has expired but the retry number has not been met yet. N2 is the number of times to retry sending a DISC frame (3 is a good number).

F-timer expired ∧ retryCount ≥ N2. The F-timer has expired and meets or exceeds the retry number . N2 is the number of times to retry sending a DISC frame (3 is a good number)..

2.4.3.4 Action Descriptions

NA := Generate-Random-Device-Address. Generate a random 32-bit device address and assign it to "NA" as this station's device address.

Apply-Default-Connection-Parameters. Configure IrLAP layer to use the default connection and transmission parameters. This only needs to be done once since defaults are always used.

slotCount := 0 The slotCount variable is set to 0. The slotCount variable is used to keep track of the current slot in a discovery process.

Send Discovery-XID-Cmd:6,Gen,slotCount. Send a discovery XID command indicating 6 slots setting to "Generate Device Address" bit set to Gen with slot number equal to slotCount.

Start-slot-timer. Start a timer to wait for a response to the XID command. A reasonable time for a slot timer is 75 - 85ms.

log := {∅} The discovery log is set to empty.

Discovery-Confirm(media-busy) Tell the service user that the discovery operation was not successful because the IR media was busy.

Generate-Random-ConnectionAdr(ca) The local layer generates a random 8-bit connection address to be used in the address field of all frames in a connection. All values except 0x00 and 0xff are legal connection addresses.

dest := da Save the device address of remote device in the variable dest.

Send u:snrm:cmd:P:ca:dest Send a SNRM command frame indicating a connection address of "ca" with a destination device address of dest.

Start-F-timer Start the F timer.

Disconnect-Indication(media-busy) Indicate that the connection failed because the IR media was busy.

slotCount := slotCount + 1 Increment the slot count by 1

Send End-Discovery-XID-Cmd Send a Discovery XID command frame with 0xff as the slot number which indicates the end of the discovery process. Included with this frame is the local info field.

Discovery-Confirm(log) Signal the end of the discovery process to the service user passing the discovery log.

log := log ∪ {<sa,info>} Add the device address and info field (Hints only) to the discovery log.

Initialize-Connection-State. Initialize the connection state variables:

remoteBusy := localBusy := false;
Vr := Vs := As := 0;

Apply-Connection-Parameters. Since defaults are used and the secondary has a fixed behavior (e.g. always sends 11 BOFs) there is nothing to do.

Connect-Confirm Signal the service user that the connection procedure completed successfully.

Send s:rr:cmd:P Send a RR command frame

retryCount := 0 Set the retryCount variable to 0. The retryCount variable is used to keep track of the number times the F-Timer expires and also the number of times to retransmit disc frames when trying to disconnect the link.

Disconnect-Indication Signal the service user that the link has been disconnected.

Queue-Data. Frames from the user layer are placed on a queue to be sent when the station has control of the link.

Leave Pending until XMIT A disconnect request from the service layer is left pending until the XMIT state is entered.

Stop-P-timer The Poll timer is stopped. The poll timer is used to delay in polling secondary devices. A good algorithm for the poll timer is to set the time to 0 when I frames have received or sent and after a period of no data transfer set it to 90% of the Max Turn Around Time (at 500ms that is 450ms).

Send u:disc:cmd:P Send a DISC command frame.

remoteBusy is a Boolean variable. It is set to true when an RNR command is received, and set to false when an RR command is received.

localBusy is a Boolean variable. It is set to true when the local device is unable to receive I-Frames, and set to false otherwise.

Vr is the expected next I-command sequence number to be received.

Vs is the expected next I-response sequence number to be sent.

As is the I-response sequence number already sent.

Unacked-Frame holds an I-Frame that has been sent but not acknowledged. If it is 0 then there are no unacknowledged I-Frames.

Data-Indication. Pass the information field of a received I frame to the service user.

Vr := Vr + 1 mod 8. Increment Vr (modulo 8) to get the sequence number of the next I frame expected to receive.

Process-NR The following actions are carried out:

```
if ((Nr == Vs) ∧ (Unacked-Frame != 0)) {
   Put-Frame-on-Front-of-Queue;
   Unacked-Frame := 0
}
```

Send-NRM-Command. The following actions are carried out:

```
if ¬remoteBusy {
   if (Data in Queue)) {
      Remove head of Queue
      Send i:cmd:Vs:Vr:P:data;
      Unacked-Frame := I-Frame Sent
      As := Vs;
      Vs := Vs + 1 mod 8;
   } else {
      Send-S-Command
   }
} else {
   Send-S-Command
}
```

Send-S-Command The following actions are carried out:

```
if localBusy {
   Send s:rnr:cmd:Vr:P;
} else {
   Send s:rr:cmd:Vr:P;
}
```

Start-P-Timer Start the poll timer.

Disconnect-Indication(PrimaryConflict). Inform the service user that the local layer has initiated disconnection of the data link connection due to detection of one or more stations behaving as primary stations with the same connection address.

3. Minimum IrLMP Multiplexer Capabilities

This section describes the minimum IrLMP multiplexer capabilities. First the basic ideas are described followed by the service primitives and finally the state machine description. IrLMP is independent of the role (Primary/Secondary) played by IrLAP so a complete IrLMP is described. Some devices can be classified as “Passive Secondary” devices. These devices do not initiate IrLMP connections and therefore can be simplified even further by not implementing the services associated with establishing a connection.

3.1 Basic IrLMP Multiplexer Ideas

Below is a list of ideas and guidelines used to minimize the IrLMP multiplexer.

1. Only one application at a time will be using the IrDA link thus, only one non-IAS IrLMP connection will exist.

2. IrDA Lite primary does not support point-to-multipoint so only one IrLAP connection will exist at a time.

3. Given that only one application at a time is using the link, IrLAP disconnect can be used to disconnect the link. Thus, an IrLMP disconnect does not need to be generated. Even though there may be up to three IrLMP connections (IAS server, IAS client, and application) they all exist for the purpose of allowing the application to communicate so all can be disconnected at once with an IrLAP disconnect. IrLAP disconnect replaces IrLMP disconnect in the following scenarios:
    - Disconnecting the application’s IrLMP connection.
    - Receive a Connect Confirm LM-PDU for a disconnected or non-existent LSAP-Connection.
    - Receive a Connect LM-PDU for a disconnected or non-existent LSAP-Connection.
    - Receive a Data LM-PDU for a disconnected or non-existent LSAP-Connection.
    - Receive a Connect LM-PDU for an active LSAP-Connection (half-open).

4. IrDA Lite will not support exclusive mode and will indicate in the IAS Device Object that no optional features are supported. Thus, IrDA Lite will perform an IrLAP disconnect when an Access Mode LM-PDU is received. This creates an issue for fully functional implementations that wish to use exclusive mode. They should query the IAS to verify that the other device supports the optional exclusive mode capability before issuing the command.

5. The IrLMP specification indicates that the LM-MUX command PDUs, Connect, and Connect Confirm, and the Data PDU, sent to a non-existent LSAP connection will result in the sending of a Disconnect PDU. Disconnect PDU’s sent to non-existent LSAP connections are ignored. Since IrDA Lite will also disconnect when receiving Access Mode PDU’s a simple approach is to disconnect whenever a PDU is received that is destined for a non-existent LSAP connection (DLSAP is not for any of the three LSAP connections). The only difference between this behavior and the standard IrLMP behavior is disconnecting when receiving a Disconnect PDU. This should not be a problem since a well behaved implementation should never send a Disconnect PUD to a non-existent LSAP connection.

6. To simplify the Lite IrLMP state machines and because IrLAP disconnect is used in place of IrLMP disconnect, IrLAP connect and disconnect primitives will be exposed to the IrLMP service user.

3.2 IrLMP Service Primitives

3.2.1 Discovery Services

LM_Discovery.request()
LM_Discovery.confirm(DeviceInfoList)

Description: The request primitive is used to start an IrLAP device discovery procedure. If address conflicts occur they are resolved by issuing another discovery to the broadcast address with the “generate new address” bit set. Any conflicting address still existing will be removed from the result list. The result list is returned via the confirm primitive.

3.2.2 Link Connect Services

- LM_LinkConnect.request(Target-Device-Address)
- LM_LinkConnect.indication()
- LM_LinkConnect.confirm()

Description: The Link Connect services are direct paths to the IrLAP services. This control of the IrLAP link to the application allows the IrLMP state machines to be simpler. There is not a response service because IrDA Lite IrLMP always accepts incoming IrLAP connections.

3.2.3 Connect Services

LM_Connect.request(Called LSAP, Client Data)
LM_Connect.indication(Calling LSAP, Client Data)
LM_Connect.response(Calling LSAP, Client Data)
LM_Connect.confirm(Called LSAP, Client Data)

Description: The request primitive is used to establish a connection with another device at the given LSAP. A successful completion of the connection is signaled with the confirm primitive. An incoming connection is indicated with the indication primitive. An incoming connection is accepted with the response primitive. An IrLAP connection must exist before an IrLMP connection can be made.

3.2.4 Disconnect Services

LM_LinkDisconnect.request()
LM_LinkDisconnect.indication()
LM_Disconnect.indication()

Description: The LinkDisconnect.request primitive is used to disconnect the IrLAP connection which will result in the disconnection of the IrLMP connections (Application and IAS). The LinkDisconnect.indication is given to the IrLMP service user when the IrLAP connection is disconnected by the other side and when it is disconnected locally. The Disconnect.indication is given to the particular LSAP connection user when its peer sends an IrLMP disconnect.

3.2.5 Data Services

LM_Data.request(Data)
LM_Data.indication(Data)

Description: The request primitive is used to send data and the indication primitive indicates received data.

3.3 IrLMP State Machines

Two state machines are used to precisely describe the IrLMP portion of IrDA Lite. The first is known as Station Control and it manages the IrLAP link and the discovery process. The second is know as Connection Control and it manages individual LSAP connections. There are at most three LSAP connections in IrDA Lite: the application, IAS server and IAS Client.

For simplicity sake the Discovery and IrLAP Link services are handled by the Station Control State Machine. All LSAP connection services are handled by the LSAP connection state machine.

3.3.1 IrLMP Station Control State Machine

3.3.1.1 State Chart

Current State	Event	Action(s)	Next State
SDISC	LM_LinkConnect.request	IrLAP_CONNECT.request	SSETUP
	LM_Discovery.request	GenAddrBit = 0 IrLAP_DISCOVERY.request(GenAddrBit) ConflictFlag := FALSE	DISCOVER
	IrLAP_CONNECT.indication	IrLAP_CONNECT.response LM_LinkConnect.indication	SACTIVE
	LM_LinkDisconnect.request	Error /* No IrLAP connection */	SDISC
SSETUP	IrLAP_CONNECT.confirm	LM_LinkConnect.confirm	SACTIVE
	IrLAP_DISCONNECT.indication	LM_LinkDisconnect.indication	SDISC
DISCOVER	IrLAP_DISCOVERY.confirm $\wedge$ AddressConflicts(Log) = $\varnothing$	LM_Discovery.confirm(Log)	SDISC
	IrLAP_DISCOVERY.confirm $\wedge$ AddressConflicts(Log) $\neq \varnothing$ $\wedge$ ConflictFlag = FALSE	GenAddrBit := 1 IrLAP_DISCOVERY.request(GenAddrBit) ConflictFlag := TRUE	DISCOVER
	IrLAP_DISCOVERY.confirm $\wedge$ AddressConflicts(Log) $\neq \varnothing$ $\wedge$ ConflictFlag = TRUE	RemoveConflicts(Log) LM_Discovery.confirm(Log)	SDISC
SACTIVE	LM_LinkDisconnect.request	LM_LinkDisconnect.indication	SDISC
	IrLAP_DISCONNECT.indication	IrLAP_DISCONNECT.request LM_LinkDisconnect.indication	SDISC
	IrLAP_DATA.indication (Not LM-Access) $\wedge$ DLSAP $\in$ LM-MUX-CONS	Forward (IrLAP_DATA.indication)	SACTIVE
	IrLAP_DATA.indication $\wedge$ DLSAP $\notin$ LM-MUX-CONS	IrLAP_DISCONNECT.request LM_LinkDisconnect.indication	SDISC
	IrLAP_DATA.indication (LM-Access)	IrLAP_DISCONNECT.request LM_LinkDisconnect.indication	SDISC
	LM_LinkConnect.request	Error /* Connection already exists */	SACTIVE

3.3.1.2 State Definitions

SDISC The station is in the disconnected mode. It is not performing a discovery and an IrLAP connection does not exists. It will act based on requests to perform discovery and establish connections.

SSETUP The station is in the process of establishing a connection. A LinkConnect.request was issued and the station is waiting for the peer device to respond.

DISCOVER The station has started a discovery process and is waiting for the results.

SACTIVE An IrLAP connection is active.

3.3.1.3 Event Descriptions

LM_LinkConnect.request LM-MUX user has requested that an IrLAP connection be established

LM_Discovery.request LM-MUX user has requested a discovery procedure

IrLAP_CONNECT.indication indicates an incoming IrLAP connection from another device.

LM_LinkDisconnect.request LM-MUX user has requested that the IrLAP connection be disconnected.

IrLAP_CONNECT.confirm indicates that a requested IrLAP connection has completed successfully.

IrLAP_DISCONNECT.indication indicates that the IrLAP connection has been disconnected.

IrLAP_DISCOVERY.confirm indicates that the IrLAP discovery process has completed returning a log of results

AddressConflicts(Log) = ∅ indicates that no conflicts exist in the discovery log

AddressConflicts(Log) ≠ ∅ indicates that there are some conflicts in the discovery log

ConflictFlag = FALSE ConflictFlag is equal to FALSE

ConflictFlag = TRUE ConflictFlag is equal to TRUE

IrLAP_DATA.indication (Not LM-Access) An IrLAP I-Frame is received that is not an LM Access command/response

DLSAP ∈ LM-MUX-CONS The DLSAP of the received LM-MUX PDU matches one of the LSAP connections (Application, IAS Server, IAS client).

DLSAP ∉ LM-MUX-CONS The DLSAP of the received LM-MUX PDU does not match any of the LSAP connections.

IrLAP_DATA.indication (LM-Access) An IrLAP I-Frame is received that is an LM-Access mode command/response.

3.3.1.4 Action Descriptions

IrLAP_CONNECT.request request an IrLAP connection,

GenAddrBit = 0 Set the GenAddrBit variable to 0 indicating that a normal IrLAP discovery is desired.

IrLAP_DISCOVERY.request(GenAddrBit) Request an IrLAP discovery with the requested value of the GenAddrBit

ConflictFlag := FALSE The ConflictFlag is used to keep track of whether the discovery process is a doing conflict address resolution or not. The flag is set to FALSE to indicate that it is a normal discovery.

IrLAP_CONNECT.response Tell IrLAP to accept the incoming IrLAP connection.

LM_LinkConnect.indication Indicate that an IrLAP connection started from another device is up. This indication is passed to all the LSAP connections and the LM-MUX service user.

LM_LinkConnect.confirm Indicate that the requested IrLAP connection is up. This confirm message is passed to all the LSAP connections and the LM-MUX service user.

Error The requested operation is not allowed at this time. Implementations can return a failure.

LM_LinkDisconnect.indication Indicate that the IrLAP connection has been disconnected. This message is passed to all the LSAP connections and the LM-MUX service user.

LM_Discovery.confirm(Log) Tell the LM-MUX service user that the discovery process is finished and pass the discovery log.

GenAddrBit := 1 Set the GenAddrBit variable to 1 indicating that a conflict address resolution procedure is desired.

ConflictFlag := TRUE Set the ConflictFlag to TRUE indicating that address conflict resolution is taking place.

RemoveConflicts(Log) Remove the conflicting addresses from the Discovery Log.

Forward (IrLAP_DATA.indication) Forward the IrLAP_DATA.indication to the appropriate LSAP connection state machine. The match is made on the DLSAP of the packet.

IrLAP_DISCONNECT.request Request that the IrLAP connection be disconnected.

3.3.2 IrLMP Connection Control State Machine

3.3.2.1 State Chart

Current State	Event	Action(s)	Next State
NOT_READY	LM_Connect.request	Error /* No IrLAP Connection */	NOT_READY
	LM_Data.request	Error /* No IrLAP Connection */	NOT_READY
	LM_LinkConnect.indication		READY
	LM_LinkConnect.confirm		READY
	LM_LinkDisconnect.indication		NOT_READY
READY	LM_Connect.request	IrLAP_DATA.request (LM Connect)	CSETUP
	LM_Data.request	Error /* No LM-MUX connection */	READY
	IrLAP_DATA.indication (LM Connect)	LM_Connect.indication	INCOMING
	IrLAP_DATA.indication (LM Connect confirm)	LM_LinkDisconnect.request	NOT_READY
	IrLAP_DATA.indication (LM Data)	LM_LinkDisconnect.request	NOT_READY
	IrLAP_DATA.indication (LM-Disconnect)		READY
	LM_LinkDisconnect.indication		NOT_READY
CSETUP	IrLAP_DATA.indication (LM Connect confirm)	LM_Connect.confirm	ACTIVE
	IrLAP_DATA.indication (LM Disconnect)	LM_Disconnect.indication	READY
	IrLAP_DATA.indication (LM Connect)	LM_Disconnect.indication	READY
	IrLAP_DATA.indication (LM Data)		CSETUP
	LM_LinkDisconnect.indication		NOT_READY
INCOMING	LM_Connect.response	IrLAP_DATA.request (LM Connect Confirm)	ACTIVE
	LM_LinkDisconnect.indication		NOT_READY
ACTIVE	LM_Data.request(data)	IrLAP_DATA.request (LM Data)	ACTIVE
	IrLAP_DATA.indication (LM Data)	LM_Data.indication(data)	ACTIVE
	IrLAP_DATA.indication (LM Connect)	LM_LinkDisconnect.request	NOT_READY
	IrLAP_DATA.indication (LM Connect Confirm)		ACTIVE
	IrLAP_DATA.indication (LM Disconnect)	LM_Disconnect.indication	READY
	LM_LinkDisconnect.indication		NOT_READY

3.3.2.2 State Definitions

NOT_READY There is no IrLAP connection. The LSAP connection is not ready and is waiting for an IrLAP connection to exist.

READY An IrLAP connection exists. The LSAP connection is in a disconnected state.

CSETUP An LM connect PDU has been sent and the LSAP connection is waiting for the response from the other side.

INCOMING An incoming LM connect PDU has been received and the LSAP connection is waiting for a response (Connect Response or Disconnect) from the LM-MUX user.

CACTIVE The LSLAP connection is active and connected. Data can be transferred.

3.3.2.3 Event Descriptions

LM_Connect.request LM-MUX user is requesting that an LSAP connection be established.

LM_Data.request LM-MUX user is requesting that data be sent over an LSAP connection.

LM_LinkConnect.indication Indication that an IrLAP connection initiated by another device is up. This indication is also passed to the LM-MUX user by the Station Control state machine.

LM_LinkConnect.confirm Indication that the IrLAP connection initiated by the this device is up. This confirmation is also passed to the LM-MUX user by the Station Control state machine.

LM_LinkDisconnect.indication Indicates that the IrLAP link is down. This indication is also passed to the LM-MUX user by the Station Control state machine.

IrLAP_DATA.indication (LM Connect) IrLAP I-Frame is received which contains an LM Connect PDU.

IrLAP_DATA.indication (LM Connect confirm) IrLAP I-Frame is received which contains an LM Connect Confirm PDU.

IrLAP_DATA.indication (LM Data) IrLAP I-Frame is received which contains an LM Data PDU.

IrLAP_DATA.indication (LM-Disconnect) IrLAP I-Frame is received which contains an LM Disconnect PDU.

LM_Connect.response LM-MUX user accepts an incoming LSAP connection request.

LM_Data.request(data) LM-MUX user sends data over the LSAP connection.

3.3.2.4 Action Descriptions

Error An error condition occurred. The event is ignored.

IrLAP_DATA.request (LM Connect) Send an IrLAP I-frame containing an LM connect command PDU.

LM_Connect.indication Tell the LM-MUX user that the peer device it attempting to establish an incoming LSAP connection.

LM_LinkDisconnect.request Tell the Station Control state machine to disconnect the IrLAP connection. The LM-MUX user and all other LSAP connections will be notified by the Station Control state machine.

LM_Connect.confirm Tell the LM-MUX user that the requested LSAP connection is confirmed (connection is now active).

LM_Disconnect.indication Tell the LM-MUX user that the LSAP connection is disconnected.

IrLAP_DATA.request (LM Connect Confirm) Send an IrLAP I-frame containing an LM connect confirm command PDU.

IrLAP_DATA.request (LM Data) Send an IrLAP I-frame containing an LM data PDU.

LM_Data.indication(data) Pass a data PDU to the LM-MUX user.

4. Minimum IrLMP IAS Capabilities

This section describes some IAS guidelines that are recommended for both fully functional and Lite implementations. By following these guidelines IAS queries can be reduced to a single packet exchange and interoperability between different IrDA implementations will be maximized.

1. Do not allow multiple objects in the IAS database to have the same name. If this guideline is followed then a list will never be returned for a GetValueByClass query. In most cases this is the best method even for fully functional implementations. When two objects have the same name it is difficult to determine which object to use. An InstanceName attribute is one way but many times it is hard for software to understand instance names and also hard for the user to make the proper choice. Some PC implementations of IrCOMM allow for two objects with the same name. These objects are distinguished by the LPT/COM bit in the parameters attribute. In most devices only one IrCOMM service is needed and even in the PC only the COM service is commonly tied to an application acting like a server so in practice there is usually only one IrCOMM service registered at a time. Another exception is Plug and Play which is pretty much confined to the PC and is not a service to which applications attempt to connect. We can also look at examples from the Internet where only one FTP or HTTP service exists on a single machine.

2. In constructing IAS objects the only necessary attribute is one that specifies the LSAP Selector value. The two most common attributes for doing this have already been defined. For services built directly on the IrLMP multiplexer the attribute name is “IrDA:IrLMP:LsapSel”. For services built on Tiny TP the attribute name is “IrDA:TinyTP:LsapSel. These attributes are of type integer. If this guideline is followed then the only IAS query needed to connect to a service is a GetValueByClass query for the LSAP selector value. IrCOMM is an exception in that it also has an attribute named “Parameters”. Therefore, it is highly recommended that services build IAS objects with a single attribute for the LSAP selector using the names given above and use the connection to send other parameters and information.

3. Keep IAS object and attributes names small so that a GetValueByClass query can fit in a single 64 byte packet. Again the only necessary attribute is the LSAP selector and the two most common attribute names have already been defined. The length of the longest LSAP attribute name is 19 (IrDA:TinyTP:LsapSel). A GetValueByClass IAS query consumes 2 bytes for IrLMP plus 1 byte for IAS plus 1 byte for the length of the class name plus the actual bytes of the class name plus 1 byte for the length of the attribute name plus the actual bytes of the attribute name. The fixed portion is 5 bytes leaving 59 bytes for the name of the attribute and the name of the class. If we assume 19 bytes for the attribute name then 40 bytes are left for the class name. Thus, it is highly recommended that IAS objects names be kept at 40 bytes or less. IrCOMM fits within these guidelines.

4. If other attributes are necessary then restrict the size of the value of the attribute so that the result for a GetValueByClass query can fit in a single 64 byte packet. The fixed overhead of a GetValueByClass result is 6 bytes (2 IrLMP, 1 IAS opcode, 1 return code, 2 list length). Assuming that there are no classes with the same name, an additional 2 bytes is taken up by the object identifier, leaving 56 bytes for the actual result. Thus, attribute values should be kept at 56 bytes or less.

5. IrDA Lite implementations can disconnect IrLAP if queries or results take more than one frame but should deal properly with ACK frames. Since ACK frames are allowed, even in single packet queries it is possible that an implementation exists that will send them, IrDA Lite implementations that want to be robust should ignore ACK frames and not disconnect IrLAP.

If the guidelines above are followed a typical IAS query will be a single packet exchange. The query itself will be a single packet containing a GetValueByClass query for the LSAP selector value of the desired application or service. The result will be a single packet containing the 4 byte result. Even if IrCOMM is implemented IAS queries can be kept to a single packet exchange as long as multiple instances of IrCOMM do not exist. Even on the PC this will more often than not be the case even though the PC allows the possibility of two IrCOMM entries in the IAS.