Introduction to the IrDA™ Protocol

Table of Contents

Introduction

The popularity of handheld devices such as laptop computers, Personal Digital Assistants (PDAs), digital cameras and medical diagnostics is increasing rapidly. This proliferation of portable devices has brought the issue of wireless data communications to the forefront of many embedded systems designs.

The Infrared Data Association (IrDA) was formed in 1993 to address some of these issues. The IrDA's stated mission is "to create an interoperable, low cost IR data interconnection standard that supports a walk-up, point to point user model for a wide range of appliances and devices." To that end, the IrDA has created the Serial Infrared Data Link Standard, which is currently at Rev. 1.1, and supports data transfer rates up to 4.0 Mbps.

Many industry segments can benefit from wireless communications. It is not inconceivable that in the future, you will be able to walk up to your local ATM machine obtain some cash (or reload your debit card) and balance your electronic checkbook using IrDA. You can use that new found money to pay off the auto mechanic who just performed some drive up point and shoot diagnostics on your car. On your way home, as the toll booths automatically debit your account when you car drives through them, you can stop by the local store and print out some of those pictures you've taken with your digital camera. All this could be done without plugging in a single cable.

Infrared communications provide several advantages over the other wireless option, which is RF based.

- Compact hardware
- Low Cost
- No antennas
- Unregulated
- Relatively secure because of directionality

Of course, nothing is perfect and Infrared communication is no exception. Some of the drawbacks of using IR are as follows

- Limited Range (specification dictates 1 meter minimum range)
- Line of Sight (Beam can be blocked)
- Directional (15 degree half angle infrared cone)

The following article presents an overview of the IrDA protocol, as well as some details on some of the performance issues associated with it.

The IrDA Protocol

As with most other communications protocols, the IrDA protocol consists of a physical layer and several layers of software protocol. Some of the layers are required, while others are optional. The standards specify what services are to be provided; they do not specify the implementation or any type of API. A full implementation of the protocol stack might appear as in Figure 1.

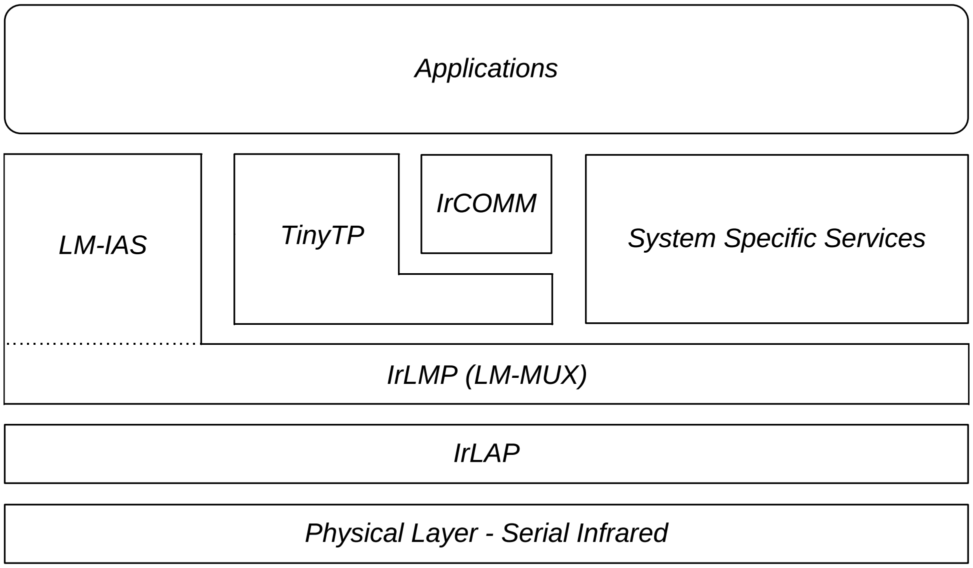


Figure 1

The following sections of this article will discuss each of the elements of the protocol in more detail.

IrDA Physical Layer

There are actually three different physical layers associated with the IrDA protocol. The original specification (Rev. 1.0), which supports rates from 9600 to 115.2 kbits/s, uses a standard UART (8 bits, no parity, 1 start, 1 stop) to drive a simple IR encoder/decoder which in turn drives the IR transceiver. Considering that most microcontrollers have built in UARTs, this can be a very low cost configuration.

All three physical layers have some common characteristics. They each support point to point and point to multipoint configurations. Communications are all half duplex using carrier sense with no collision detection. When connections are established, the communications follow a single master (or primary) to one or more slaves (or secondary). Of course, a device may be involved in multiple connections and may act as a primary in one connection, while being a secondary in another. This will be explained in more detail in a later section of this article.

The encoding of the data for the Rev 1.0 spec. (up to 115Kbps) uses Return to Zero Inverted (RZI) modulation. It calls for a logic “0” to consist of a light pulse that is a minimum of 1.6 usec and a maximum of 3/16 of a bit time, and a logic “1” to consist of no pulse occurring in the requisite time frame. Figure 2 shows an example of an IrDA Rev 1.0 physical layer frame transmission. Each byte is transmitted using a start bit of a logic “0” and a stop bit of a logic “1”.

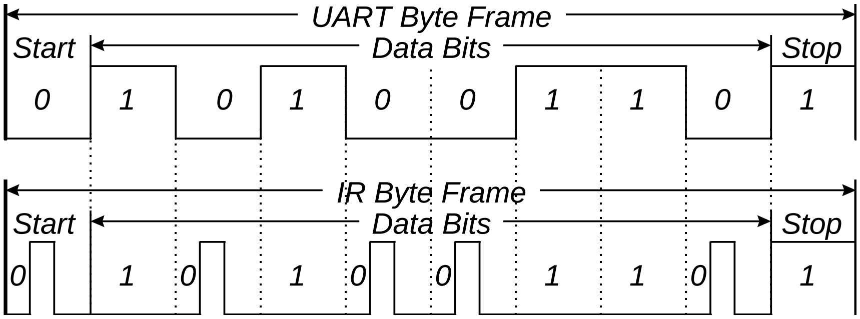


Figure 2

To accommodate the higher transmission rates that really make the IrDA very attractive, the IrDA Rev 1.1 High Speed extensions define two more physical layers. The first of these layers supports data rates of 0.576 Mbps and 1.152 Mbps. This layer uses the same RZI encoding scheme that Rev 1.0 does, but the framing and pulse durations are somewhat different. The IR pulse that represents a logic zero is between ¼ and 3/16 of a bit time. More importantly, each byte is not transmitted separately, with it's own start and stop bits. Transmitted frames use standard HDLC format with bit stuffing after five consecutive logic ones and a 16 bit CRC. The other difference from the lower rate is that the start and stop flags both become 0x7E. An example of a transmitted data stream is as shown in Figure 3.

RAW DATA ^{LSB} *00110011 10101111 10001111 00011010 11111011 10001010* ^{MSB}
CRC
TRANSMITTED DATA ^{LSB} *00110011 10101111 100001111 00011010 111110011 10001010* ^{MSB}

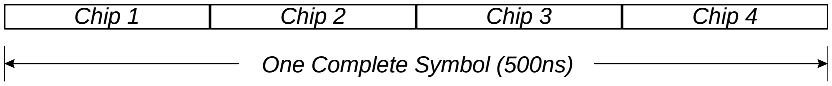
Figure 3

Note that all data, including the CRC, is bitstuffed, and that data is transmitted LSB first. Not shown in Figure 3 are the beginning and ending flags. Each frame must have at least two beginning flags and one ending flags. A complete frame for the 0.56 and 1.152 Mbps physical layers would look as follows in Figure 4.

0x7E	0x7E	Addr	Control	Up to 2045 Bytes Data	16 Bit CRC	0x7E

Figure 4

The final physical layer transmits data at 4.0Mbps rate. This physical layers uses Pulse Position Modulation (PPM) encoding, which defines a specific data symbol duration (500ns) that is broken up into 4 chips per symbol. As shown in Figure 5 , each symbol can represent 2 bits.



Data Bit Pair	4PPM Data Symbol
00	1000
01	0100
10	0010
11	0001

All other symbols are illegal for data

Figure 5

Each transmission frame consists of a preamble of 16 repeated transmissions of 4 specific symbols (1000 0000 1010 1000) which are not valid for data, a Start Flag, Stop flag and a 32 bit CRC. A typical frame is shown in Figure 6

Preamble	*Start Flag*	*Data (Up to 2045 Bytes)*	*32 Bit CRC*	*Stop Flag*

Figure 6

When designing an IrDA capable device, it is not necessary to support all three physical layers. However all discovery and capability negotiations occur at 9600 bps, so the original Rev. 1.0 physical layer must be supported, while the high speed extensions are optional. This has the unfortunate effect of complicating the design and increasing the cost of a device that chooses to use the higher speed extensions, because at least two different encoder/decoder modules must be included in the design. A typical IrDA capable communications section that supports the higher speed extensions is shown in Figure 7.

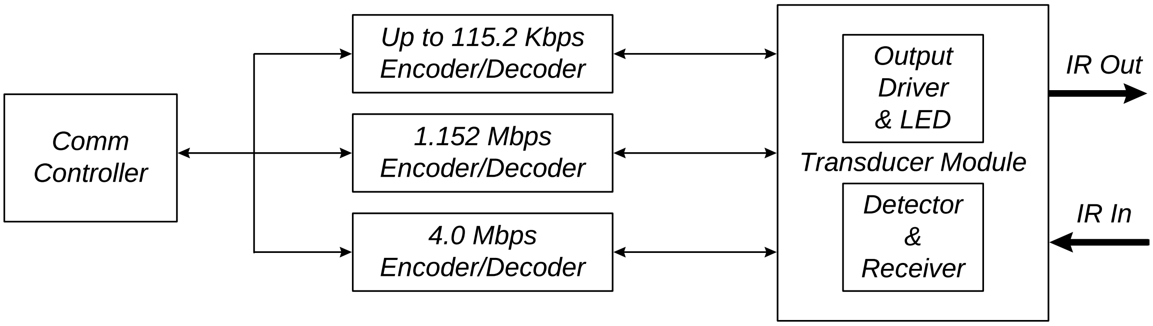


Figure 7

The IrDA Link Access Protocol (IrLAP)

The Data Link Layer (Layer 2) is referred to as the Link Access Protocol (IrLAP). As mentioned previously, the IrLAP specification defines services that must be provided by the data link layer to the upper layer of the protocol. The specification does not provide an implementation or API.

As with many protocols, the IrLAP supports both connectionless and connection-oriented services. There are four generic types of service primitives defined in the IrLAP. They are as follows:

- Request - passed ***from*** the upper layer to invoke a service.
- Indication - passed ***to*** the upper layer to indicate an event or to notify the upper layer of an IrLAP initiated action.
- Response - passed ***from*** an upper layer to acknowledge some procedure invoked by an indication primitive.
- Confirm - passes ***to*** the upper layer to convey the results of the previous request.

The interaction of the IrLAP primitives can be shown graphically as shown in Figure 8.

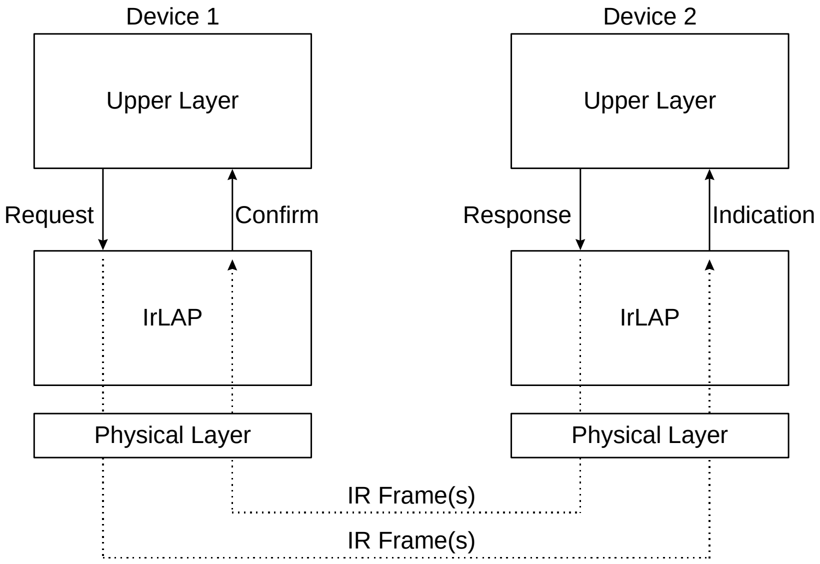


Figure 8

The connectionless services available in the IrLAP consist of Discovery services, which find available devices; Address Conflict services, that resolve any address conflicts; and Unit Data services, which send out broadcast data. There are six connection oriented services supported by the IrLAP. They are Status services, which perform quality of link reporting; Reset services, where both ends must agree to reset; Sniffing services, which perform a special low power connect service; Data services, which can be reliable, sequenced or unreliable, expedited; and Disconnect services, which terminates the logical connection.

In order to provide all these services, the IrLAP has its own data frame structure that rides on top of the physical layer frames. There is a Beginning of Frame flag (0xC0), an address (A) field, a control (C) field an information (I) field, a 16 bit CRC that is calculated using the A,C and I fields, and an End of Frame flag (0xC1). A typical IrLAP frames appears as shown in Figure 9.

BOF	*A*	*C*	*I*	*16 Bit CRC*	*EOF*

Figure 9

Note that the IrLAP is an unbalanced data link. For each connection that exists, there is a primary device and one or more secondary devices. The primary is the master and controls the data flow, while the secondaries are slaves and only speak when allowed to. All devices must have the capability to become a secondary, while primary capabilities are optional. The address field of an IrLAP frame is always that of the secondary device that is involved. It is the destination when the primary is transmitting, and it is the source address when a secondary is transmitting. The control field specifies the type of each frame. These may be Unnumbered, which are used for data link management; Supervisory, which are used for acknowledgments, error reporting , etc.; and Information Transfer. Data can be segmented among multiple frames, with the send and receive counts and the sequence numbers embedded in the control field of information and supervisory frames.

The IrLAP specification defines a series of Finite State Machines (FSM) that outline the operating procedures of the link layer. Figure 10 shows the flow of the states of the IrLAP.

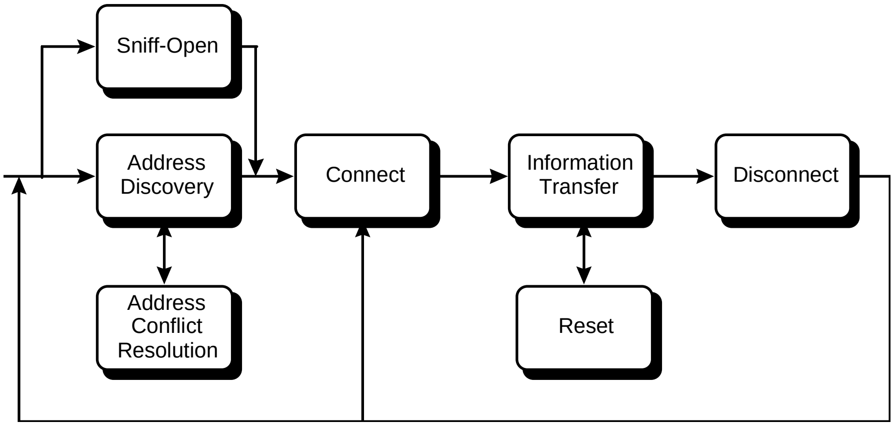


Figure 10

Discovery

The discovery procedure is used to determine the physical addresses of all devices that are within range. An initiator starts the discovery procedure by sending out an Exchange Station Information (XID) discovery frame. All devices within range will generate a random address and respond with an XID response frame. The initiator will collect the pertinent information from all responding devices, such as address, communication parameters and other capabilities, and report it to the upper layers in a discovery log. The one obvious side effect of using random address selection during the discovery sequence is that two devices could select the same address. This situation is handled by the initiator , which sends out an XID frame to the address in question with the address conflict flag of the control field set to TRUE. The conflicting devices will then randomly select another address and send an XID response. This process continues until all address conflicts are resolved.

The Sniff-Open procedure allows a device to begin the discovery process while conserving power. A Sniffing device will periodically wake-up and check for any IR traffic. If any IR traffic is detected, the device will go back to sleep. If there currently no IR traffic detected, the device will send out an XID ***response*** frame to address 0xFFFFFFFF. The device will the wait for an XID command, an XID discovery frame or a Set Normal Response Mode (SNRM) frame, which begins normal connect procedures. If no frames are forthcoming, the device will go back to sleep and repeat the process later. As mentioned earlier, all discovery procedures are performed at 9600 baud, which must be supported by all IrDA compliant devices.

Connection

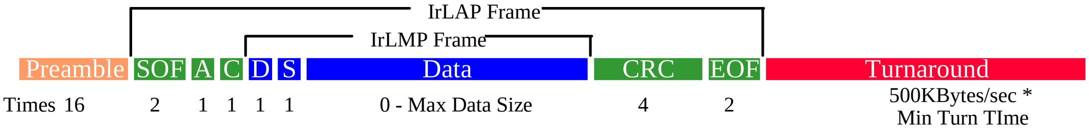
Once the discovery procedures are complete, a connection procedure can begin. A connection is attempted by the initiator sending out an SNRM frame which includes the communications parameters that the requesting device supports. The receiving device either sends an Unnumbered Acknowledgment (UA) frame containing the accepted communications parameters or else it sends a Disconnect Mode (DM) frame, rejecting the connection attempt. The parameters that are negotiated are as follows:

- Baud Rate - From 9600bps to 4Mbps
- Maximum Turnaround time - a trade off between responsiveness and power consumption during idle periods.
- Data Size - maximum number of bytes in a packet
- Window Size - the maximum number of packets that can be transmitted consecutively by a device before allowing the receiving device to acknowledge.
- Addition BOFs - Additional BOF bytes transmitted at the start of a frame. This parameter is ignored for 4Mbps connections.
- Minimum Turnaround time - time needed to account for any receiver latency in the transceiver circuit.

- Link Disconnect / Threshold time - can be set to anywhere from 3 seconds to 40 seconds.

The negotiated parameters have a significant effect on the transport efficiency of the connection. This effect is especially dramatic when a connection is very one sided, such as with a laptop to a printer, and the data rate is 4Mbps.

Data transfers between devices will contain an IrLMP frame embedded within an IrLAP frame as shown in Figure 11 which shows a 4Mbps IrLAP frame. Note that some low level IrLAP transfers, such as an acknowledge (UA) frame, will not have the IrLMP portion of the frame.



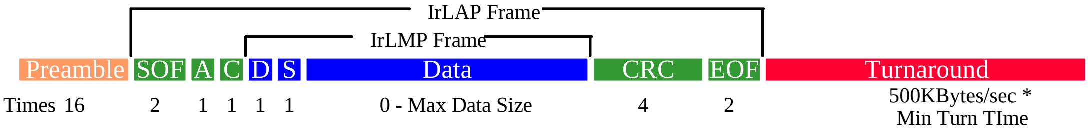


Figure 11 IrLAP Transfer Frame

Also note that the turnaround time at the end of a frame is only necessary if the transmitting device is allowing the receiving device a chance to respond (the P/F bit is set in the IrLAP Control byte). If the devices are communicating with a window size of 1, the transmitting device (the laptop) must allow the receiving device (the printer) a chance to acknowledge after each frame. The sequence would look like that in Figure 12.

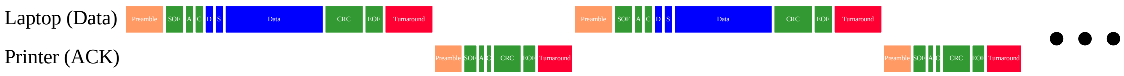


Figure 12 Window Size = 1

The turnaround time can be the biggest deterrent to efficient transfer if the Window size is too small and the receiver latency is large. In the above example each transfer of the maximum packet size would incur two turnaround time penalties - one for the data frame and one for the UA frame. With a Window size of seven, the turnaround time delay would only occur after seven packets as shown in Figure 13.

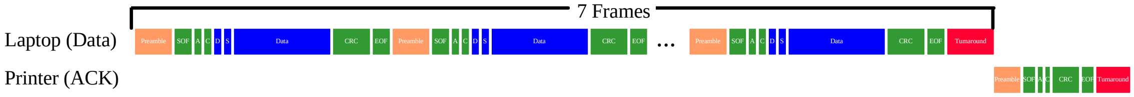


Figure 13 Window Size = 7

Using the information discussed above, a graph of the transfer efficiency versus turnaround time and Window size can be generated and is shown in Figure 14. Note that the following caveats apply to the graph in Figure 14.

- Baud rate = 4Mbps.
- Bit stuffing was ignored - in practice, bit stuffing overhead is typically < 2%.
- All IrLMP packets transmit the maximum data allowed.
- Any upper level protocol overhead (such as TinyTP is ignored).

Analyzing Figure 14 reveals that, as the turnaround time decreases, the Window size becomes less and less of a factor in transfer efficiency. For example, at a maximum packet size of 2048 and turnaround time of 10 msec, the transfer efficiency is 16% with a Window size of 1 and 59% with a Window size of 7. However, if the turnaround time is reduced to 100 microseconds, the transfer efficiencies become 93% and 97%, respectively.

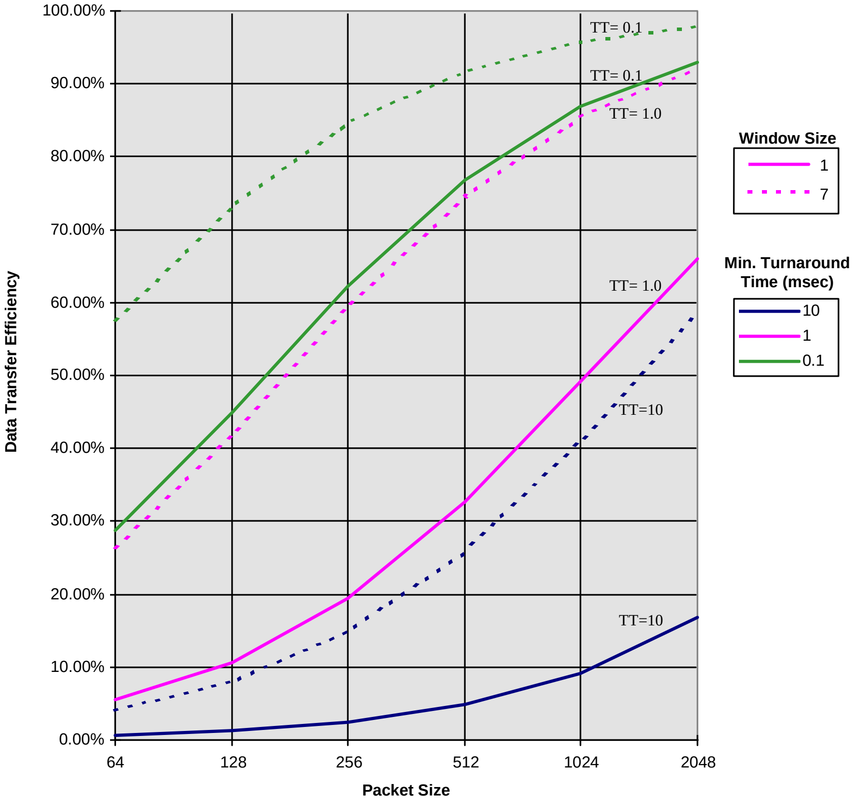


Figure 14 IrDA Transfer Efficiency at 4Mbps

Minimum turnaround time is determined by the hardware, in particular the receiver latency of the IrDA transceiver that is used. Receiver latencies of 1msec are typical, although devices with latencies less that 100 microseconds are available. At speeds of 115Kbps or less, this effect is much less pronounced, a turnaround time of 1msec is only 14 byte times, compared to 500 byte times at 4Mbps.

Information Transfer

Once connected, devices can either exchange data, reset or disconnect. Data can be sent using reliable, sequenced frames. Using the reliable (I) mode, data can span multiple frames with counters keeping track of the number of frames sent and the number of the next frame expected to be received. Data can also be sent using unreliable, expedited (U) service. In this case the data cannot span multiple frames.

If both devices agree, the connection can be reset, which clears the frame counters and retry counters but keeps the connection open. A connection can also be disconnected by either device. The disconnection does not require approval from both devices.

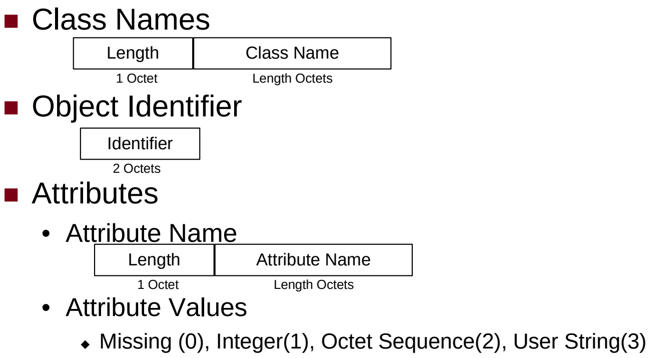
The IrDA Link Management Protocol (IrLMP)

The IrLMP actually consists of two parts, the Link Management Information Access Service (LM-IAS) and the Link Management Multiplexer (LM-MUX).

The Information Access Service (LM-IAS)

The LM-IAS maintains an information base of what services this particular device has to offer to other devices. The LM-IAS also provides a mechanism to discover the services provided by the other devices. This mechanism is similar to how an SNMP capable device would go about obtaining object information from a devices Management Information Base (MIB) on an Ethernet network.

Objects in the information base consist of a name, an identifier and up to 256 attributes. The format for each of those entities is shown in Figure 15.



- Class Names
- Object Identifier
- Attributes
  - Attribute Name
  - Attribute Values
    - Missing (0), Integer(1), Octet Sequence(2), User String(3)

Figure 15

All devices must have at least one object in its information base. Object 0 contains the device name and its IrDA feature support properties. Other objects can contain specific information about the function or functions supported by the device. An example information base is shown in Figure 16.

Device Class			
Attribute Name	**Attribute Type**	**Value**	**Description**
DeviceName	IDString	“My Digital Camera”	The name of the product.
IrLMPSupport	Octet Sequence	0x01 0x37 0x00	IrLMP Rev, IAS and LM-MUX support
IrDA:IrLMP:LsapSel	Integer	0x03	LSAP selector number for IrLMP
IrDA:TinyTP:LsapSel	Integer	0x04	LSAP selector number for TinyTP
IrCOMM Class			
Attribute Name	**Attribute Type**	**Value**	**Description**
IrDA:IrLMP:LsapSel	Integer	0x05	LSAP selector for IrLMP
IrDA:TinyTP:LsapSel	Integer	0x06	LSAP selector number for TinyTP
Parameters	Octet Sequence	0x00 0x01 0x0F	3 byte sequence describing IrCOMM services
PnP Class			
Attribute Name	**Attribute Type**	**Value**	**Description**
DeviceID	IDString	“CAM001”	EISA manufacturer’s device ID
Name	VendorString(90)	“My Digital Camera”	Device description (EISA Name field)
Manufacturer	VendorString(30)	“Cameras R Us”	Manufacturer’s name (EISA MFR field)
Category	VendorString(3)	CAM	Functional Category of Device (EISA CAT field)

Version	TwoBytes	1,0	IrPnP specification version. (Currently 1.0)
Status	Integer	0	Bit field conveying status to host. 0 = Available.
CompCnt	Integer	0	Number of devices this device is compatible with.

Figure 16

Communications between two LM-IAS entities uses the Information Access Protocol (IAP). This protocol defines six service primitives including:

- LM_GetInfoBaseDetails
- LM_GetObjects
- LM_GetValue
- LM_GetValueByClass
- LM_GetObjectInfo
- LM_GetAttributeName

The specification for the IAP consists of an independent client and server Finite State Machine for each open connection on a device. The Finite State Machine either initiates or services requests through the IrLAP. Figure 17 shows that the upper layers of the protocol need not be involved in the service discovery procedure. Note that there can only be one call in progress between a client and a server.

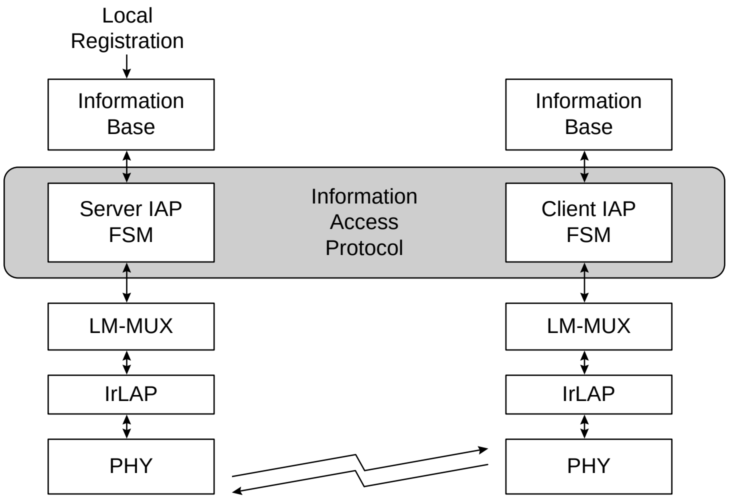


Figure 17

The Link Management Multiplexer

The current IrDA specification implements a point to point protocol, an optional point to multipoint specification is being developed. Two devices can have only one IrLAP connection between them. The Link Management Multiplexer (LM-MUX) allows multiple logical data connections over the single IrLAP connection between devices.

The LM-MUX provides services to the upper layers through Link Service Access Points (LSAPs). The LM-MUX services the IrLAP connections through IrLAP Connection Endpoints. Figure 18 illustrates the LM-MUX service boundaries.

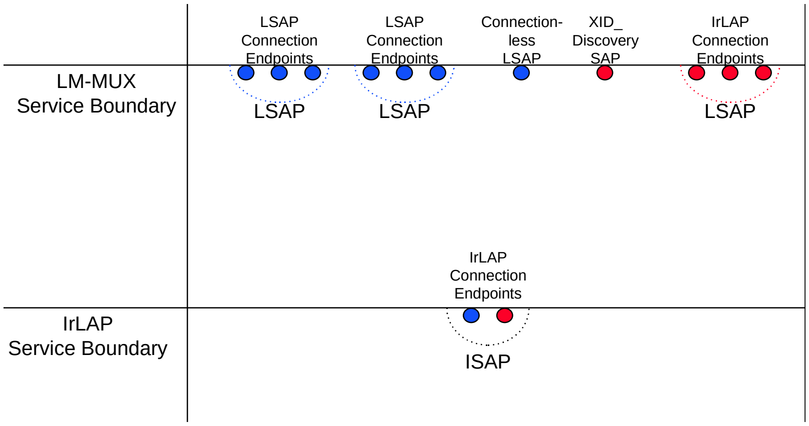

LSAP Connection Endpoints
LSAP Connection Endpoints
Connection-less LSAP
XID_ Discovery SAP
IrLAP Connection Endpoints
LM-MUX Service Boundary
LSAP
LSAP
LSAP
IrLAP Connection Endpoints
IrLAP Service Boundary
ISAP

Figure 18

There are a few things to note about the LM-MUX's treatment of the LSAPs. There can be multiple logical connection endpoints in each LSAP, the LM-MUX does not provide any kind of flow control among it's LSAPs, there can only be one connection between any one pair of LSAPs and finally, connections between LSAPs do not necessarily have to go outside the device. These concepts are shown in

Figure 19.

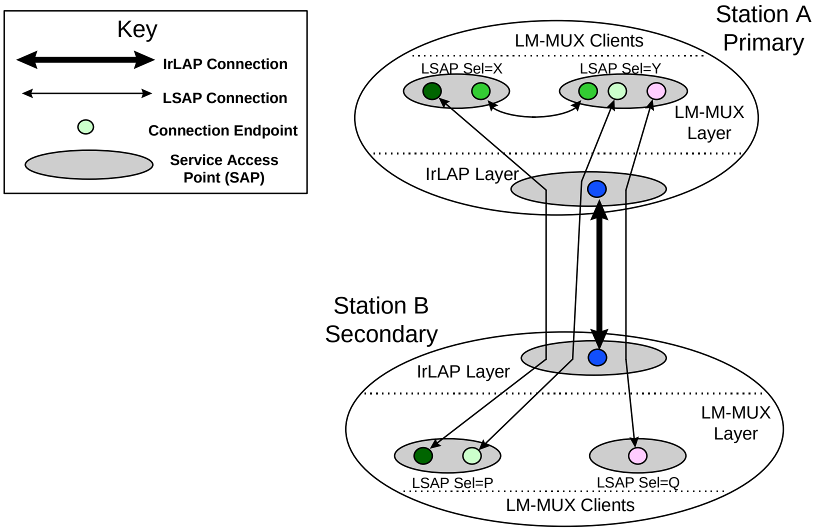


Figure 19

Internally, the LM-MUX consists of three main parts. There is one LSAP Connection control Finite State Machine for each LSAP connection endpoint. There is a single receive demultiplexer that takes data indications from the IrLAP and routes it to the proper connectionless endpoint or connection oriented endpoint. There is also a single Station Control Entity that is responsible for the transmission of connectionless data, operating the XID_Discovery process, makes and breaks IrLAP connections, assigns LSAP connections to IrLAP connections and allows for the transition between exclusive and multiplexed LM-MUX modes. Figure 20 LM-MUX Internals and Figure 21 Station Control FSM show the internal workings of the LM-MUX.

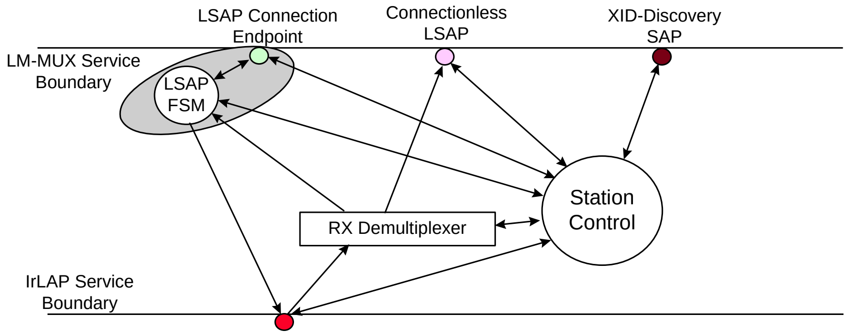


Figure 20 LM-MUX Internals

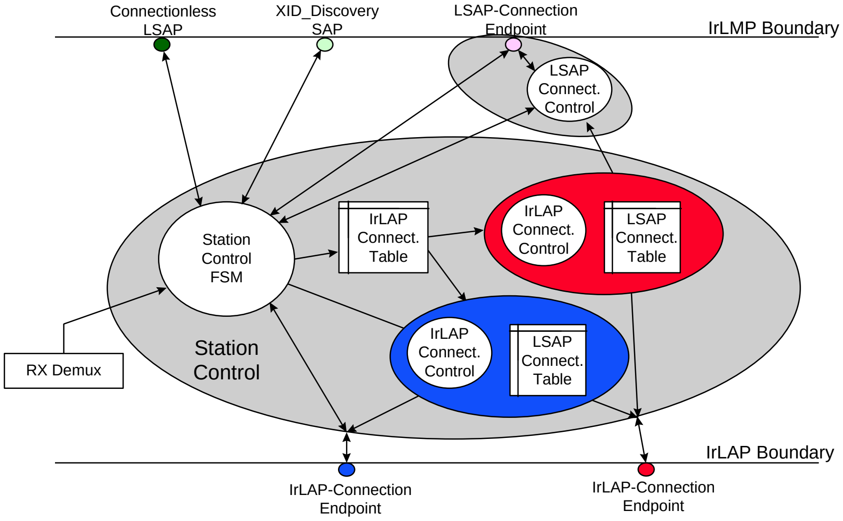


Figure 21 Station Control FSM

From a service standpoint, the IrLMP has three types of external service primitives which can be called by the upper layers of the protocol stack; Discover services, Link Control services, and Data Transfer services.

The Discover service primitives include LM_DiscoverDevices and LM_Sniff. These services create connectionless XID_Discovery LSAPs.

The Link Control Services consist of 5 service primitives. They are as follows:

- LM_Connect - makes a logical connection between two LSAPs
- LM_Disconnect - breaks LSAP connections
- LM_Status - provides link status indications for each LSAP
- LM_Idle - Locally marks an LSAP as idle
- LM_AccessMode - change between exclusive and multiplex mode

Finally, the IrLMP has three types of Data Transfer Service Primitives. They are:

- LM_Data - sends reliable I frames to the remote LSAP
- LM_UData - sends expedited UI frames to the remote LSAP
- LM_ConnectionlessData - sends out of connection UI frames

The TinyTP

The TinyTP is a flow control mechanism for use with the IrLMP. When multiplexed LM-MUX mode is being used (multiple LSAP's), the only flow control in the system exists within the IrLAP level. Without flow control at the IrLMP level, the logical connections between LSAP's could become deadlocked or lose data in certain cases. The TinyTP provides for independently flow controlled transport connections as well as for segmentation and re-assembly. The TinyTP uses Data Credits as it's method of flow control. The connect TinyTP Protocol Data Unit (PDU) specifies the initial data credit count and any optional transfer parameters. Each successive data transfer contains a number of delta credits that specifies the number of PDUs that may be sent in the reverse direction. Figure 22 shows the typical initial and subsequent PDUs used by the TinyTP.

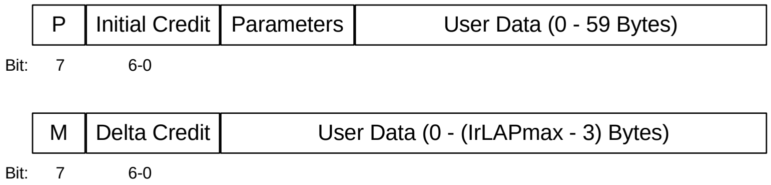


Figure 22

The TinyTP is not a required part of the IrDA specification and can be omitted, particularly if any upper layers of the protocol stack contain a flow control mechanism.

The IrCOMM

The IrCOMM is an optional portion of the IrDA specification that provides for emulation of the standard serial and parallel ports for legacy applications. Using the IrCOMM does not provide the most efficient use of the IrDA protocol, but in cases where there are older applications that used existing serial or parallel ports it may be the quickest way to get the application up and running. The IrCOMM provides four classes of services:

- 3-Wire Raw (IrLPT) - cannot be multiplexed, uses the IrLAP flow control mechanism.
- 3-Wire - emulates simple RS232, uses TinyTP for flow control.
- 9-Wire - emulates a full RS-232 interface, uses TinyTP.
- Centronics -emulates a standard Centronics interface, uses TinyTP.

Along with the normal connect, disconnect and data services, the IrCOMM provides control services that perform the emulation of the hardware handshaking and flow control (DTR, DSR, etc.) associated with the RS232 and Centronics interfaces. One final note about the IrCOMM, when IrCOMM is used there must be an associated entry in the IAS information base with the Class Name of IrDA:IrCOMM along with some specific object attributes.

The IrLAN

IrLAN is a recently approved extension to the IrDA specification. It is designed to provide transparent attachment to networks for portable devices. The IrLAN specification defines two LSAPs, a control LSAP and a data LSAP. The control LSAP is used for configuration purposes, while the data LSAP is used to pass the actual network traffic. There are three modes for IrLAN. An Access point mode allows several IrDA devices to connect to a LAN through a device, while each IrDA device retains a unique network (IP) address. The Hosted mode is similar to the Access point mode except that the bridging device presents a single IP address to the network, which all attached IrDA devices must share. This complicates the software in the bridging device because it must act a a router for all the connected IrDA devices. Finally there is the Peer to Peer mode, which allows devices to communicate using the network protocol, but not necessarily be connected to any other wired LAN. Figure 23 shows a simple Access point mode connection.

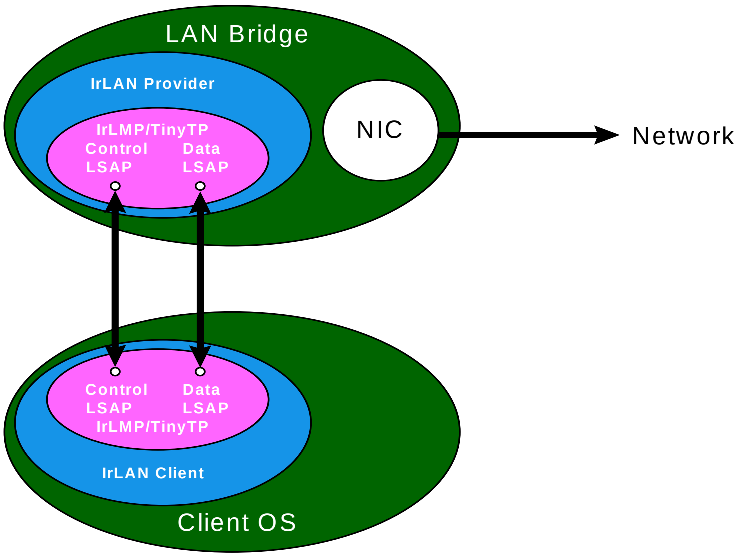


Figure 23 IrLAN Access Point

IrDA Object Exchange Protocol

IrDA Object Exchange Protocol (IrOBEX) is an extension to the IrDA protocol that is designed to ease the sometimes painful task of transferring data of varying types between devices. IrOBEX treats data that needs to be transferred as objects - it could be a file, an image, some business card information, or anything else. It uses an HTTP like mechanism of sending headers that describe the object to be transferred as well as the data itself. IrOBEX also defines a session protocol, which consists of a set of “opcodes” that define specific actions such as Put and Get. These opcodes are used in an request/response protocol that allows an orderly transfer of data.

IrDA Lite

The full implementation of the IrDA protocol can be an extremely significant undertaking for designs that have only simple data communications needs. To satisfy those needs the IrDA has approved IrDA Lite. IrDA Lite is a subset of the full IrDA protocol that reduces the complexity and code size of the implementation. IrDA Lite also maintains upward compatibility with devices that implement the full protocol stack. So instead of being an extremely significant undertaking to implement the protocol, it becomes just a significant undertaking.

The IrDA Lite implementation of the IrLAP limits the types of services that are provided and some of the Quality of service parameters that are negotiated in the connection procedure. In particular, IrLite sets the Window size to 1 and the maximum packet size to 64 bytes. Because the IrDA Lite cannot support multiple secondaries in a connection and it is defined with a Window size of 1, some assumptions can be made about incoming frames that greatly simplify the state machine design. An IrDA Lite Secondary ignores all incoming response frames and all incoming commands with the P (poll) bit cleared. The Lite Secondary also does not support the Sniff procedure nor does it generate a Discovery indication to the upper layers. The following services are provided by a Lite Secondary:

- IrLAP_CONNECT.indication()
- IrLAP_CONNECT. response()
- IrLAP.DATA.request(User-Data)
- IrLAP_DATA.indication(User-Data)
- IrLAP_DISCONNECT.request()
- IrLAP_DISCONNECT.indication()

The Lite Primary incorporates all the services of the Lite Secondary and adds the capability of initiating a Discovery sequence.

The Lite IrLMP multiplexer is greatly simplified by limiting the link to only one non-IAS connection. Therefore, at most there will be one IAS client, one IAS server and one application connection. Also, since the IrDA Lite primary does not support multiple logical connections, there will be only one non-IAS IrLMP connection at a time. One result of this simplification is that the IrLMP will not need any Disconnect service. Any disconnect received from the IrLAP will result in disconnection of all three IrLMP connections simultaneously since they use the same IrLAP connection. The IrLMP Lite Services provided are as follows:

- LM_Discovery.request()
- LM_Discovery.confirm(DeviceInfoList)
- LM_LinkConnect.request(Target-Device-Address)
- LM_LinkConnect.indication()
- LM_LinkConnect.confirm()
- 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)
- LM_LinkDisconnect.request()
- LM_LinkDisconnect.indication()
- LM_Disconnect.indication()
- LM_Data.request(Data)
- LM_Data.indication(Data)

Remember that IrLite is a *guideline* for a minimum implementation. There are some commercially available IrLite performance protocol stacks that keep the simplicity of only one IrLMP connection, but improve the Quality of Service parameters - such as increasing the Window size to 7 and the maximum packet size to 2048.

Implementing the IrDA Protocol

To implement the IrDA Lite protocol, several state machines must be designed. The actual state machine descriptions are all part of the IrDA Lite specification. There are three main state machines that we must deal with: the IrLAP Secondary State Machine (we are implementing a secondary only device), the IrLMP Station State Machine and the IrLMP Connection Control State Machine. The state diagrams are shown in Figures 26, 27 and 28. The state and transition definitions are those specified in the respective tables in the IrDA Lite specification. The numbering system for the transitions directly correlate to the order in which each transition appears in the state tables of the IrDA Lite specification.

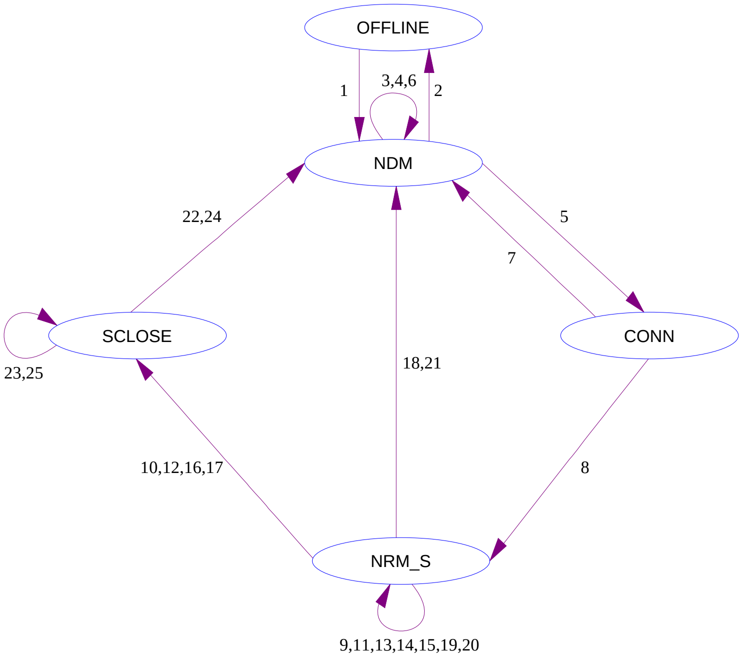


Figure 24 IrLAP Secondary State Machine

The IrLAP Secondary State Machine is relatively straightforward. The station in the offline state when it is powered off, not initialized or the IR port is disabled. Once it is initialized it moves to the Normal Disconnect Mode (NDM). When in the NDM a station can perform discovery and address resolution procedures and can accept requests to connect to other devices via the Set Normal Response Mode (SNRM) command. If a connect request is received the station enters the Connect (CONN) state and passes the request to the upper layers. This connection is either accepted, which will transition the device to the Normal Response Mode (NRM(S)), or it can be refused, which will send the station back to the NDM state.

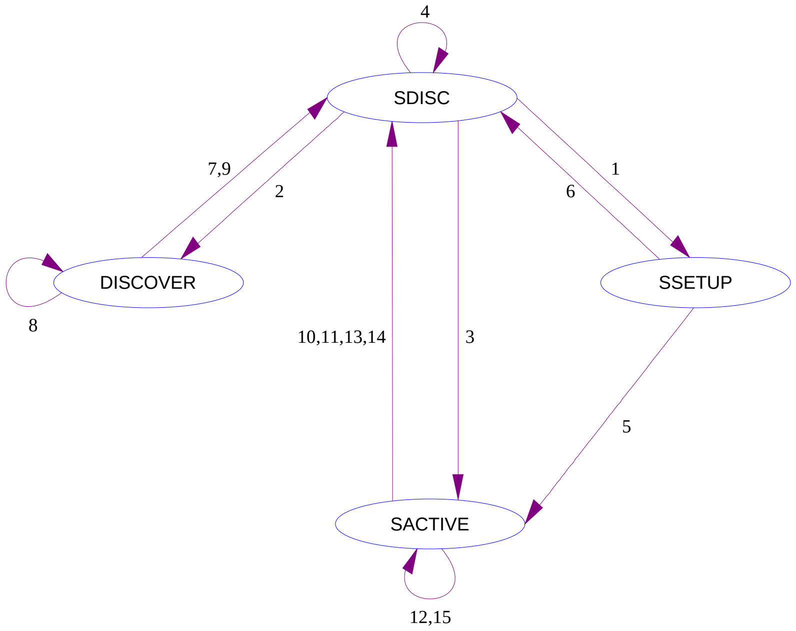


Figure 25   IrLMP Station Control State Machine

The IrLMP Station Control state machine manages the IrLAP link and the discovery process. It is a very simple machine with only four states with two of those states being somewhat transient. In the disconnected state (SDISC) the station is waiting to for a discovery process to be initiated or a connection to be requested. The DISCOVERY state is a transient state that is entered when a discovery request is received and exited when a DISCOVERY.confirm packet is received. Note that the confirm packet can be either a successful completion of the event or an indication of an address conflict. From the SDISC state, a station can also enter the SSETUP state which is also a transient state that is entered when a LM_LinkConnect.equest is received. This state is exited with either an IrLAP_CONNECCT.confirm or an IrLAP.DISCONNECT.indication is received. When a connection has been successfully established the station enters the SACTIVE state. The station will remain in this state until a disconnect it initiated.

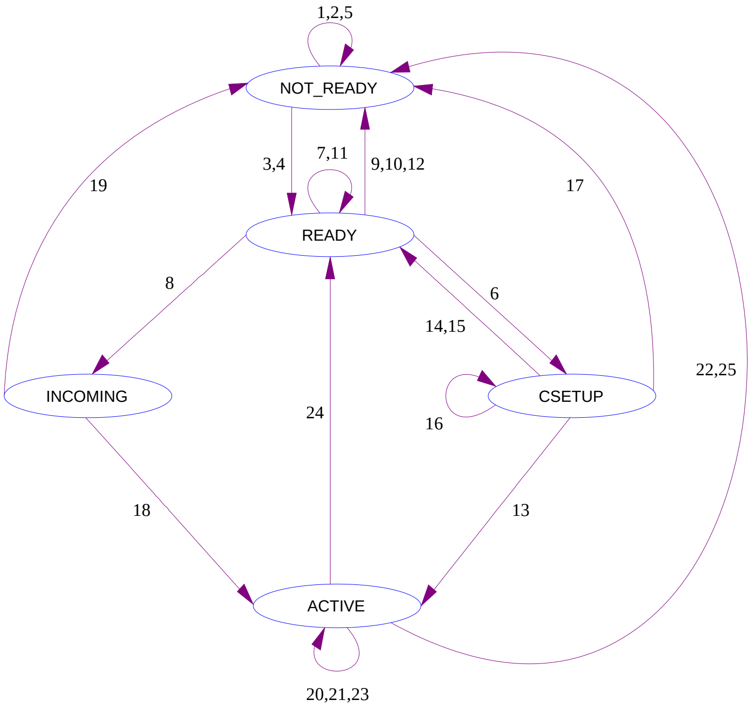


Figure 26   LSAP Connection Control State Machine

The final IrDA related state machine we will discuss is the IrLMP Connection Control state machine. This state machine controls the individual LSAP connections. There is one Connection Control state machine for each LSAP that exists in the device, with a maximum of three for an IrDA Lite device. While this state diagram looks more complex that the others, it really is not that much more involved. There are 5 states with two of them being transient. In the NOT_READY state there is no IrLAP connection and the LSAP connection is waiting for one to be established. Once an IrLAP connection is established, the device is READY for either an LM_Connect.request from the upper layers or for an indication that a peer want to establish a connection. In the first case, the device enters the CSETUP state and waits for a connect confirmation to be received from the IrLAP. In the second case, the device enters the INCOMING state and waits for a response from the application layer. Once a connection is successfully established, the device enters the CACTIVE state and data can be transferred between LSAPs.

As with any complex networking protocol, the best way to implement it is to buy it. There are several vendors, including Counterpoint Systems Foundry and Actisys, that offer a completely tested implementation on several different platforms. Also, an IrDA stack is becoming a key feature of many RTOS vendors, such as Integrated Systems, Wind River Systems and Microware. If you happen to have a combination of RTOS and microprocessor that doesn’t have an IrDA protocol stack available, some of the vendors mentioned will undertake a porting effort for you.

Conclusion

The IrDA protocol, along with USB, Ethernet and IEEE 1394 (Firewire™) will be the data channels of the future. Any embedded system that needs to share information (and what applications won't?) will use at least one of the aforementioned communication methods. IrDA is the only one of those protocols that is wireless. It recognition factor is growing rapidly with consumers and end users and is on its way to becoming a must have feature in portable devices such as laptops, digital cameras, PDAs and portable phones. This article only brushes the surface of the IrDA protocol, but hopefully it gives a basic understanding that can be used as a platform to build on.

References & Useful Sites

http://www.irda.org - The Infrared Data Association Home Page - the main source of information about the IrDA including the actual specification. Note that IrDA, IrLAP, IrLMP and pretty much anything else in this article that starts with Ir are trademarks of the Infrared Data Association.

http://www.microsoft.com/windows/software/irda.htm - IrDA drivers for Win95.

http://www.countersys.com - Counterpoitn Systems Foundry

http://www.actisys.com - Actisys corporation