TO: T10 Membership
FROM: Paul A. Suhler, Quantum Corporation
DATE: 13 June 2008
SUBJECT: T10/07-469r5, ADT-2: Internet ADT (iADT)

Revisions

0 Initial revision (2 November 2007)

1 First revision (9 March 2008)
Changed name to Network ADT (iADT).
Added registered port number.
Allowed iADT ports to use any port number.
Removed iADT-specific baud rate and $Timeout_{ACK}$.

2 Second revision (11 April 2008)
Deleted the ABORT service request and the ABORTED service indication.
Added analysis of existing state machines, link services, and frame header fields.
Added analysis of physical layer connections.

3 Third revision (30 April 2008)
Added discussion of including legacy ADT signals in new Custom Connector section.
Added proposed connector signal name and pinout.

4 Fourth revision (29 May 2008)
Separated the concept of an "ADT port" from an "ADT interconnect port."
Added link layer protocol services generic to all physical layers, as well as mappings to RS-422, TCP, and UDP.
Added requirement for ADT ports using Ethernet to ignore negotiated baud rate in computing acknowledgement timeout.

4 Fifth revision (13 June 2008)
Enhanced model section.
Removed LED connections from ADT Ethernet bus description.
Added descriptions of LED blinking.
Specified a fixed $Timeout_{ACK}$ in seconds for ADT TCP connections.
Deleted ADT UDP interconnect.

General

To allow future data transfer devices to have improved and alternate means to communicate with automation devices, Ethernet is proposed as an ADT port. One possible configuration would be an isolated subnet with the library controller and all drives attached. These ports will typically be 10/100BaseT, so there will be a great increase in bandwidth above the fastest existing RS422-based ADI ports.

Implementing an ADI Ethernet port could be done in two ways. One would be to use iSCSI to carry SCSI commands, data, and status and then to invent a new protocol for VHF data. A simpler approach would be to transport the entire ADT protocol over a networking protocol. This proposal is to do the latter, and is named Internet ADT (iADT).

A straightforward implementation of iADT would be to open a TCP connection between the automation device and the data transfer device. A TCP connection (also known as a stream) provides bi-directional reliable delivery of a stream of bytes. The existing ADT link layer protocol provides the necessary framing. While TCP error correction would prevent framing errors and parity errors from reaching the ADT layer, it would still be possible for acknowledgement timeouts to occur.

To avoid the need to modify ADT-2 to specify mapping of TCP connections to I_T nexuses, this proposal sidesteps the issue by stating that one ADT port connects to one other ADT port, without reference to the interconnect layer. At the interconnect layer, this proposal defines ADT interconnect ports through which ADT ports connect. There are two types of ADT interconnect ports, serial and Ethernet. One ADT serial port can connect only to one other ADT serial port, while multiple ADT Ethernet ports can connect to one another. Nevertheless, when ADT ports connect via ADT Ethernet ports, each ADT port can connect to only one other ADT port.

This organization of the standard avoids changes to the clauses for link, transport, and SCSI application layers.

Technical issues

The following are technical issues which must be considered in developing this proposal:

Timeouts

- After discussion in the May 2008 working group meeting, it was decided that the acknowledgement timeout should be used. While its use in detecting corrupted frames is not necessary when using TCP, it can still be used in recovering from a skip in frame numbers in at least one observed case. See the discussion below under ADT link layer analysis.

Negotiated Parameters

- Of the parameters in the Login IU, only Major/Minor Revision, Maximum Payload Size, and Maximum Ack Offset seem to be needed in iADT. Baud rate is unnecessary.

Port Numbers

- The original intent of this proposal was to use a fixed port number for the iADT port on both ends (sockets) of the TCP connection. A registered port number (4169) was obtained from the Internet Assigned Numbers Authority (IANA). However, existing Sockets implementations appear to dynamically assign the port number of the port performing a TCP active OPEN, so this requirement is relaxed. Instead, the only socket required to use 4169 is one in the device performing a passive OPEN (Listen). I.e, a DTD will do a passive OPEN on port 4169 and the library will connect to that port. Similarly, the library could do a passive OPEN on 4169 if it is desired for the DTDs to initiate the connection.
- If the network segment inside the library connects to a router that connects outside the library, then the drive can be protected by requiring the router not to pass packets with the iADT port number in either the Source Port or Destination Port field of the TCP header. Requiring the receiving end of a connection request to use the iADT port number will facilitate this protection.

I_T Nexuses and TCP Connections

- This revision of the proposal removes the concept of the socket, although those could still exist at a lower level in implementations. By keeping the current concept of an ADT port's connecting only with one other ADT port, the I_T nexus can now be defined explicitly in terms of the X_ORIGIN bit (as it is now) and implicitly in terms of the ADT port identifier. This means that there is no change from the current standard.
- There is a new case in the I_T nexus loss case list, for closing of the TCP connection while still logged in at the ADT level.
- There was a question whether the TCP ABORT could map to a device reset. David Black has since advised against this, saying "…an attempt to use this sort of TCP feature as a carrier of SCSI level function/semantics is not a good idea in general." Moreover, it is not clear (1) what events in a host already cause a TCP ABORT, and (2) whether the OS function to reset a storage device could be made to send an ABORT. Finally, RFC 793

specifies that an ABORT causes release of the TCB (control block), as does a CLOSE. This implies that an ABORT should also cause an I_T nexus loss.

Physical Layer

- This revision of the proposal separates the ADT port from the physical port.
- References to physical layer signals outside the physical layer clause have been qualified so that it will be obvious that they do not apply to iADT ports.
- The actual physical layer mandates Ethernet autonegotiation without mentioning specific speeds.

Custom Connector

The working group decided not to pursue a standard connector to include Ethernet. Instead, an ADT Ethernet "bus" is specified to list those connections which would be mandatory and optional.

- 1000BaseT requires four pairs of wires; usually all are wired in RJ-45 connectors and in Ethernet cables. However, 10 and 100BaseT only require two pair, so we discard the other two. There is no forecast need for an ADT Ethernet port to support Gigabit Ethernet.
- The ADT Ethernet bus will include the ADT $Sense_a$ line. Standalone DTDs may use Ethernet. Examples of how to discover presence in a library include a jumper or an extra pin on the Ethernet connector. If the DTD is not installed in a library, then it will enable its primary port(s) regardless of the saved setting of the port enable (PE) bit.
- In this revision of the proposal, support for the $Reset_a$ connection is optional. In Ethernet, this will cause a re-initialization of the port, which would imply re-acquiring the IP address, re-establishing connections, etc.
- In this revision of the proposal, support for the $Sense_d$ connection is optional.
- In this revision of the proposal, support is added for one or two LED connections to indicate Ethernet signal sense and activity. The connection will directly drive an LED which is pulled up via a resistor. The current and voltage characteristics of the connections are specified, but not those of the LED or resistor. This is intended to give designers maximum flexibility.
- The working group decided not to specify serial diagnostic connections in the ADT Ethernet bus

Discovery

- The working group wishes to specify how to discover the IP address of the library's and DTD's iADT ports.
- One possible means of discovery would be to use the Discovery and Description steps of the Universal Plug and Play (UPnP) protocol. This uses broadcast of UDP datagrams and does not require a server to track service locations. This would require the DTD to support an HTTP server. Proposal T10/08-198r0 describes how UPnP could be used. Discovery will not be a part of this proposal.

ADT link layer analysis

This section examines ADT's link-level specification for areas that are irrelevant to iADT, including frame header fields, information units, and state machines. While the current revision of the proposal makes no changes to the link layer, this information is retained for reference.

Much of the error recovery in ADT is to detect and correct physical-layer corruption of frames; these can be corrected by retransmitting the corrupted frame and are termed recoverable errors. Other errors, such as specifying an invalid protocol, setting a reserved bit, and sending a too-long packet can be due to firmware errors at a higher level. Simple retransmission cannot fix these errors and they are termed

unrecoverable. TCP's reliable delivery will eliminate the recoverable errors, but cannot fix the unrecoverable errors.

State machines

The Transmitter Error and Receiver Error state machines are only used to recover from out of order or lost frames. TCP makes them unnecessary, and along with them the Initiate Recovery IUs.

Frame header fields

All of the frame header fields in ADT appear to be necessary in iADT. The following table summarizes the reasons.

Table 1 – Applicability of ADT frame header fields

Field	Comments
PROTOCOL	Needed to differentiate SCS Encapsulation, Fast Access, etc.
FRAME TYPE	Needed for various protocols
X_ORIGIN	Needed to distinguish exchanges originated by library from those originated by the DTD. This is effectively a part of the EXCHANGE ID field.
EXCHANGE ID	Needed to differentiate overlapped commands, etc.
FRAME NUMBER	Needed to associate ACKs and NAKs with frames.
PAYLOAD SIZE	Needed to help trap errors in frame assembly.

Timeouts

The original intent of the acknowledgement IU timeout in ADT was to recover from lost or corrupted (and thus discarded) frames. TCP should protect against both of these, so the only possible causes for this timeout would be slow processing in the receiver of the frame to be acknowledged or slow network transmission. However, a case was presented in which the acknowledgement timeout was used to recover from a malformed ACK IU. As a result, this revision of the proposal retains the acknowledgement timeout.

Link service IUs

Following is a summary of which ADT Link Service IUs are needed and which are not needed.

Table 2 – Applicability of ADT link service IUs

IU type	Comments
Login IU	Yes – Need a mechanism to agree on Major Revision, Minor Revision, Maximum Payload Size, and Maximum Ack Offset.
Logout IU	Yes – Need to provide logout duration and reason code.
Pause IU	No – If no receive() is performed on the connection, then data will not be lost. (This was originally intended to prevent dropping bytes on an RS-422 connection that was being ignored.)
NOP IU	No – Does anyone feel that this is needed?
Initiate Recovery IU	No – TE/RE state machines are not required.
Initiate Recovery ACK IU	No – TE/RE state machines are not required.
Initiate Recovery NAK IU	No – TE/RE state machines are not required.
Device Reset IU	Yes
Timeout IU	Yes
ACK IU	Yes – While the flow control function of the ACK IU may not be needed, it still serves the purpose of indicating that a frame did not have non-recoverable errors. See the discussion below of the NAK IU.
NAK IU	Yes – See the following discussion of status codes.

The NAK IU is necessary to report certain errors that are due to an incorrectly-assembled frame; they are not related to corrupted or out-of-order frames. All of these errors are non-recoverable, i.e., they cannot be fixed by retransmission. For example, the upper layer assembling the frame may exceed the maximum payload length or may have a mismatch between the payload length field and the actual payload length.

Table 3 – Applicability of NAK IU status codes

Status code	Comments
OVER-LENGTH	Yes – This error can occur and cannot necessarily be corrected by retransmission.
UNDER-LENGTH	Yes – This error can occur and cannot necessarily be corrected by retransmission.
UNEXPECTED FRAME NUMBER	Yes – The ACK may be malformed.
AWAITING INITIATE RECOVERY IU	No
HEADER RESERVED BIT SET	Yes – This error can occur.
INVALID EXCHANGE ID	Yes – This error can occur.
UNSUPPORTED PROTOCOL	Yes – This error can occur.
OUT OF RESOURCES	Yes – This error can occur.
LOGIN IN PROGRESS	Yes – This error can occur.
INVALID OR ILLEGAL IU RECEIVED	Yes – This error can occur.
REJECTED, PORT IS LOGGED OUT	Yes – Unless we can abolish logins.
MAXIMUM ACK OFFSET EXCEEDED	Yes – This error can occur.
MAXIMUM PAYLOAD SIZE EXCEEDED	Yes – This error can occur.
UNSUPPORTED FRAME TYPE FOR SELECTED PROTOCOL	Yes – This error can occur.
NEGOTIATION ERROR	Yes – Unless we can abolish logins.
Vendor specific	Yes.

Items Not Specified

The following technical issues have not been addressed in this proposal:

- While the maximum payload size decided on in ADT negotiation will continue to be driven by device resources, can it be kept independent of the TCP Maximum Segment Size (MSS), which is typically 1500 bytes in IPv4? An ADT frame split across multiple TCP segments might be handled inefficiently. (The MSS is the largest amount of data that can be sent in an unsegmented piece. The Maximum Transmission Unit (MTU) is the largest packet (header, data, and trailer) that can be sent. Because data is a component of a packet, MTU > MSS.)
- If a DTD is installed with both Ethernet and RS-422 ADI ports connected to the automation device, there could be confusion, although this would not be a new issue as currently nothing prohibits having two ADI ports. There is a practical issue, i.e., implementations may have taken shortcuts that would make the behavior of the ADC device server non-SAM-compliant with respect to multiple I_T nexuses. This is not a standards issue, and this proposal will not address the question of multiple ADI ports.
- Sockets APIs typically include an “out-of-band” channel that can be processed separately from regular data. This can be used to allow some data to bypass data sent earlier. This feature is not specified in this proposal, as it has no clear advantages and could potentially cause problems.

Changes to ADT-2 rev. 5

Markup conventions

Proposed additions are in blue, removed text is in ~~crossed out red~~.

Editor's notes in green provide information on decisions to be made and actions to be performed before this proposal can be integrated into the standard.

Change to clause 2

Add the following subclauses:

2.1.4 IETF references

RFC 791, *Internet Protocol – DARPA Internet Program – Protocol Specification*

RFC 793, *Transmission Control Protocol (TCP) – DARPA Internet Program – Protocol Specification*

RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification*

2.1.5 IEEE references

IEEE 802.3-2005, *Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*

Changes to clause 3

Add the following definition:

3.1.x LLC: link layer control.

3.1.x MAC: media access control.

3.1.x MDI: medium dependent interface.

3.1.x PHY: physical layer.

Changes to clause 4

Add the following at the end of clause 4.1:

The ADT protocol defines communication between two ADT ports. The ADT protocol consists of five layers. These are the ADT physical layer, the ADT interconnect layer, the ADT link layer, the ADT transport layer, and the SCSI application layer. ADT interconnect port implements the ADT interconnect layer and the ADT physical layer. An ADT port implements both the ADT transport layer and the ADT link layer.

The ADT physical layer (see clause 5) provides two alternative physical connections for data, RS-422 and Ethernet, as well as sense, signal, and LED connections.

The ADT interconnect layer (see clause 6) provides transmission of encoded characters between ADT ports. Two alternative transmission methods are defined, ADT serial and ADT TCP. The ADT serial protocol provides transmission over an RS-422 physical layer. The ADT TCP protocol provides transmission over a TCP connection. The TCP connection provides transmission over an Ethernet physical layer.

The ADT link layer (see clause 7) provides reliable transmission of ADT frames between ADT ports. The ADT frames are represented as encoded characters.

The ADT transport layer (see clause 8) provides transmission of two categories of information units (IUs), SCSI encapsulation IUs and fast access IUs, between ADT ports. The information units are represented as ADT frames.

The SCSI application layer (see clause 9) provides transport protocol services for processing SCSI commands and task management requests.

Figure 4 shows the communication between ADT ports and between ADT interconnect ports at the different layers of the protocol, from the physical layer to the transport layer.

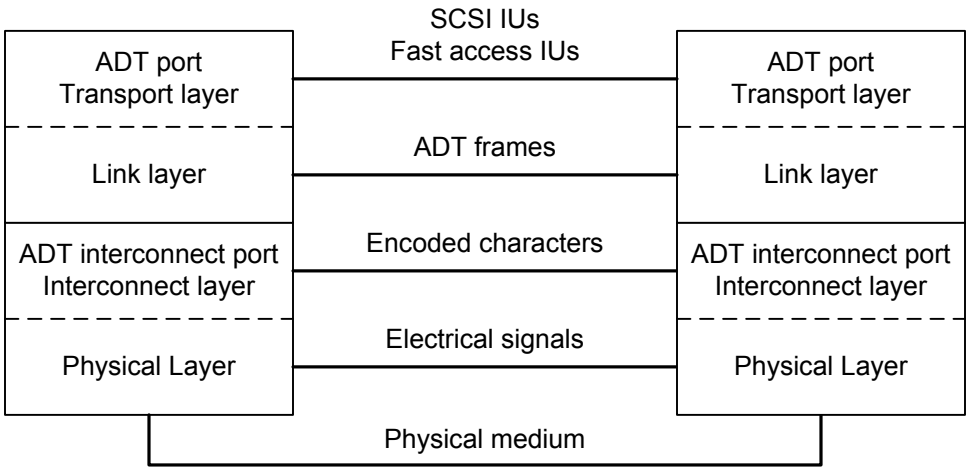

Figure 4 – Communication model

Figure 5 shows the hierarchy of protocols which may be used to implement ADT on the physical layers.

ADT transport layer SCSI encapsulation | Fast access		Transport layer
ADT link layer		Link layer
ADT serial	ADT TCP	Interconnect layer
	TCP	
	IP	
	Ethernet LLC	
	Ethernet MAC	
RS-422	Ethernet PHY	Physical layer

Figure 5 – ADT protocol hierarchy

Editor's Note 1: The above figure is done using a table. I can render it as a Visio diagram if desired.

The term ADT serial port refers to an ADT interconnect port using the ADT serial bus (see 5.3) and the ADT serial interconnect layer (see clause 6). A single ADT port may use an ADT serial port for connection with another device. Figure 6 shows connections corresponding to Figure 3. ADT port A is connected with ADT port B, and ADT port C is connected with ADT port D.

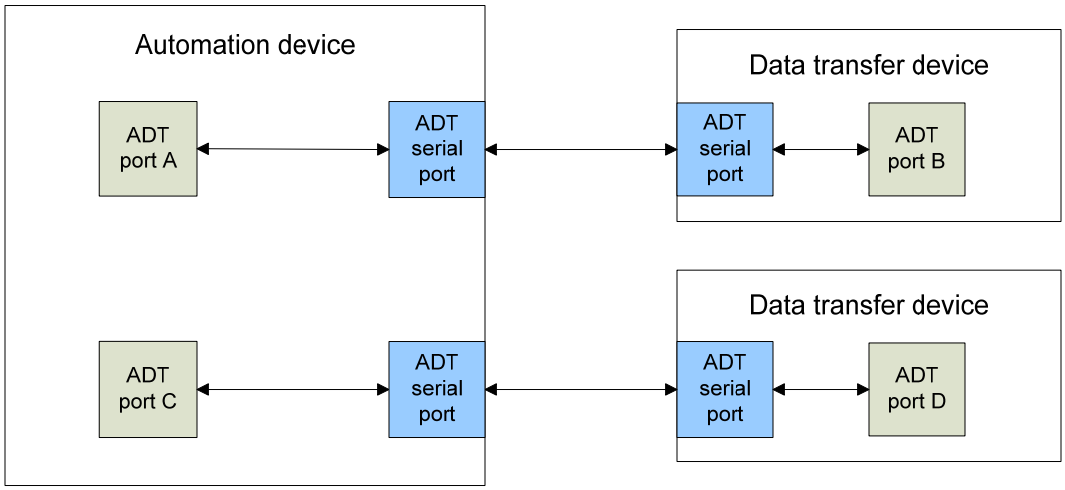

Figure 6 – ADT serial port example

The term ADT TCP port refers to an interconnect port using the ADT Ethernet bus (see clause 5.3) and the ADT TCP interconnect layer (see clause 6). Multiple ADT ports in one device may share a single ADT TCP port for connection to other devices. Figure 7 shows two pairs of ADT ports connected by ADT TCP ports, corresponding to the connections shown in Figure 3. ADT port A is connected to ADT port B, and ADT port C is connected to ADT port D.

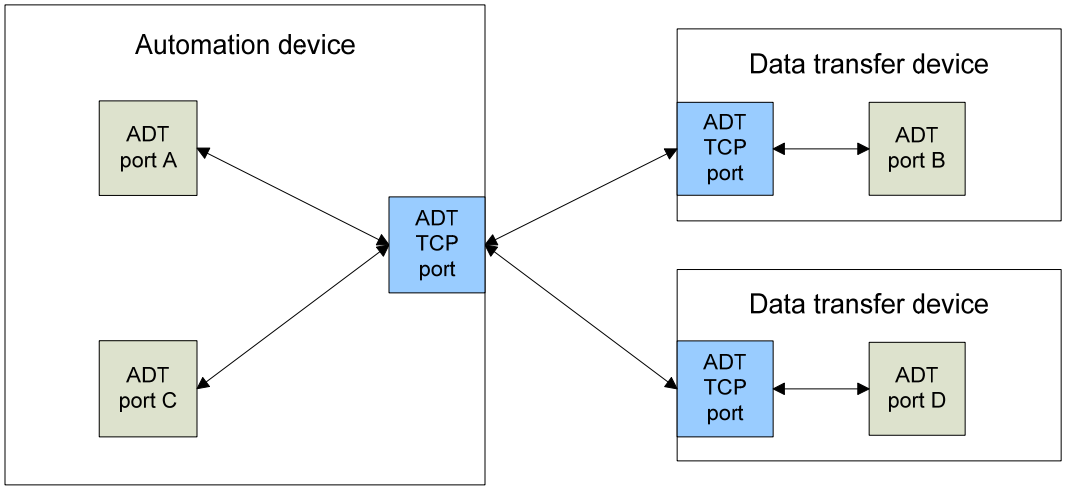

Figure 7 – ADT TCP port example

Figure 4 – Port State Machine Diagram

Renumber to figure 8. Add an arrow labeled "Close Event (to all states, causing transition to P0:Initial)" to upper left of figure.

Figures 5 – 8

Renumber to figures 9 – 12.

4.8 I_T nexus loss

An I_T nexus loss event shall occur if an ADT port:

a) sends a Port Login IU with the AOE bit set to one;
b) receives a Port Login IU with the AOE bit set to one;
c) receives an ACK IU in response to a Device Reset IU;
d) detects the change of state of the Sense line from presence to absence (i.e., $Sense_a$ for DT device port and ~~$Sense_b$~~ $Sense_d$ for automation device port (see figure ~~11~~ ??); ~~or~~
~~e) detects the assertion of the $Reset_a$ line (see table 13).~~
e) detects the invocation of the Reset service indication (see 6.6.10); or
f) receives a Closed service indication while the Port State Machine is in either P1:Login or P2:Logged In state (see 6.3.2).

Editor's Note 2: The original reference in item d) to Figure 11 does not appear to be correct. We need to find where this actually should go.

…

4.10.1 Acknowledgement time-out period calculation

When changing operating parameters (see 3.1.32), ~~a port~~ an ADT port connecting via an ADT serial port shall calculate a new acknowledgement IU time-out period using the formula in figure ~~9~~ 13. The port shall apply the new acknowledgement IU time-out period to every frame transmitted after changing operating parameters

[Figure 9 is unchanged, other than renumbering to 13]

An ADT port connecting via an ADT Ethernet port shall use an initial acknowledgement time-out period of 2.5 seconds. This may be changed if the ADT port processes a time-out IU.

Changes to clause 5

Add a new clause 5.1, renumber {5.1, 5.2, 5.3} to {5.2, 5.3, 5.4}, renumber and modify clause 5.1.5, add a new clause 5.2.6, and add two paragraphs and a table to 5.3:

5 Physical layer

5.1 Physical layer introduction

The ADT physical layer defines a number of connection types. Some of these connections are used in all ADT buses, some are used only in ADT buses which transmit and receive data using RS-422 signals, and some are used only in ADT buses which transmit and receive data using Ethernet signals. A connector is defined for use only in ADT ports which transmit and receive data using RS-422 signals.

~~5.1~~ 5.2 Electrical characteristics

Clauses {5.1.1, …, 5.1.4} are renumbered to {5.2.1, …, 5.2.4}.

Figures 10 and 11 are renumbered to 14 and 15.

Modify clause 5.1.5 as follows:

~~5.1.5~~ 5.2.5 Transmit-receive connection

A Transmit-Receive (Tx-Rx) connection is a complete simplex signal path from one ADT serial port to a second ADT serial port. A Tx-Rx connection includes:

a) a signal generator connected to the output compliance point of one ADT serial port;
b) a pair of transmission media from the output compliance point of one ADT serial port to the input compliance point of a second ADT serial port; and
c) a signal receiver connected to the input compliance point of the second ADT serial port.

[…]

Add the following clause:

5.2.6 LED connections

LED connections are used by a DT device to drive light-emitting diodes (LEDs) to indicate the status of the Ethernet connections. Table u describes the electrical characteristics of an LED connection at the output compliance point. The description assumes that:

a) the output is an open-collector type;
b) an LED and a resistor are connected in series between the output and the positive supply voltage.

Table u – LED connection output characteristics

Signal State	Current	Voltage
Asserted	-25 mA < I_{OL}	0 V < V_{OL} < 0.4 V
Negated	I_{OL} < 20 µA	V_{OH} < 5.5 V

Table v defines the LED connections used by the DT device.

Table v – LED connections

Connection Name	O/M[a]	Connection Type	Driven By
LED_{active}	O	LED	DT device port
LED_{signal}	O	LED	DT device port
[a] O indicates support is optional, M indicates support is mandatory.			

A DT device supporting both the LED_{signal} and LED_{active} connections may signal in the following manner:

a) if signal presence is detected, the LED_{signal} connection is asserted. If no signal presence is detected, the LED_{signal} connection is deasserted; and
b) if activity is detected on the TX_D1 or RX_D2 connections, the LED_{active} connection is alternately asserted and deasserted. If no activity is detected on the TX_D1 or RX_D2 connections, the LED_{active} connection is deasserted.

A DT device supporting only the LED_{signal} connection may signal in the following manner:

a) if no signal presence is detected, the LED_{signal} connection is deasserted;
b) if signal presence is detected and no activity is detected on the TX_D1 and RX_D2 connections, the LED_{signal} connection is asserted; and
c) if activity is detected on the TX_D1 or RX_D2 connections, the LED_{signal} connection is alternately asserted and deasserted.

~~5.2~~ 5.3 Bus composition

This standard defines two sets of bus connections. The ADT serial bus applies to implementations using the transmit-receive connections defined in 5.2.5. The ADT Ethernet bus applies to implementations using Ethernet connections (see IEEE 802.3-2005).

Table 7 defines the connections that make up the ADT serial bus. With the exception of $Sense_a$ and $Sense_d$ this standard defines the behavior of these connections only when an initiator port asserts $Sense_a$ and a target port asserts $Sense_d$.

Table 7 — ADT serial bus connections

Connection Name	O/M[a]	Connection Type	Driven By	Connection Definition
$Reset_a$	O	Signal	automation	An automation device may use this connection to

			device port	signal a reset request to a DT device by invoking the Reset service request. A DT device shall treat the receipt of a signal on this connection as an invocation of the Reset service indication in the ADT port attached to the ADT serial bus (see 6.6.10). ~~a) as a port logout (see 6.5.5); or~~ ~~b) as a hard reset (see 4.7).~~
$Sense_a$	M	Sense	automation device port	A DT device shall use this connection to sense the presence or absence of an automation device on the ADT bus.
$Sense_{aux}$	O	Sense		This standard does not define the use of this connection.
$Sense_d$	M	Sense	DT device port	An automation device shall use this connection to sense the presence or absence of a DT device on the ADT bus.
$Signal_{aux}$	O	Signal		This standard does not define the use of this connection.
Tx_a - Rx_d	M	Tx-Rx	automation device port	An automation device shall use this connection to send serialized data. A DT device shall receive serialized data on this connection.
Tx_d - Rx_a	M	Tx-Rx	DT device port	A DT device shall use this connection to send serialized data. An automation device shall receive serialized data on this connection.
[a] O indicates support is optional, M indicates support is mandatory				

Table w defines the connections that make up the ADT Ethernet bus. With the exception of $Sense_a$, this standard defines the behavior of these connections only when an initiator port asserts $Sense_a$ and a target port asserts $Sense_d$.

Table w - ADT Ethernet bus connections

Connection Name	O/M[a]	Connection Type	Driven By	Connection Definition
$Reset_a$	O	Signal	automation device port	See Table 7.
$Sense_a$	M	Sense	automation device port	See Table 7.
$Sense_d$	O	Sense	DT device port	See Table 7.
TX_D1+	M	MDI [b]	[c]	See IEEE 802.3-2005.
TX_D1-	M	MDI [b]	[c]	See IEEE 802.3-2005.
RX_D2+	M	MDI [b]	[c]	See IEEE 802.3-2005.
RX_D2-	M	MDI [b]	[c]	See IEEE 802.3-2005.

[a] O indicates support is optional, M indicates support is mandatory.
[b] Medium Dependent Interface (MDI) and alternate MDI (MDI-X) are defined in IEEE 802.3-2005. An MDI connection shall support autonegotiation of link speed.
[c] In the MDI configuration, the local port drives the TX_D1 pair. In the MDI-X configuration, the local port drives the RX_D2 pair.

~~5.3~~ 5.4 Connector pin-out

ADT serial ports shall use the plug connector defined in SFF-8054. Table 8 defines the pinout for the ADT port connector on the DT device.

Table 8 is renumbered, but otherwise unchanged.

New clause 6

Insert a new clause 6 between 5 (Physical layer) and 6 (Link layer):

6 Interconnect layer

6.1 Interconnect layer introduction

The ADT interconnect layer provides protocol services for transmitting and receiving sequences of encoded characters between ADT ports.

An ADT port may either initiate a connection to a specific interconnect port, or await a connection from any interconnect port. An ADT port initiates a connection by invoking the **Connect** service request. An ADT port awaits a connection by invoking the **Listen** service request. When the connection is established, both ADT ports receive a **Connected** service indication.

An ADT port sends encoded characters on a connection by invoking the **Send** service request. The **Send** service request specifies the connection, the buffer containing the characters to be sent, and the number of characters to be sent. When the **Send** service request completes, the characters have been accepted for delivery and the buffer may be reused.

An ADT port receives encoded characters on a connection by invoking the Receive service request. The **Receive** service request specifies the connection, the buffer to contain the received characters, and the maximum number of characters to be placed in the buffer. When characters have been placed in the buffer, the **Received** service indication is invoked. The **Received** service indication indicates the number of characters that have been placed in the buffer. To receive more characters on the connection, the ADT port must invoke the **Receive** service request again.

An ADT port closes a connection by invoking the **Disconnect** service request. Any characters that have been submitted for delivery by earlier **Send** service requests will be transmitted before the connection is closed. When an ADT port receives a **Disconnected** service indication, the connection is closed and no more characters shall be received.

An ADT port in an automation device resets an ADT port in a DT device by invoking the **Reset** service request. This causes the ADT port in the DT device to receive a **Reset received** service indication.

Table 4 – Interconnect layer protocol services

Protocol service	Interconnect service protocol interaction	Invoked by
Listen	Request	Either
Connect	Request	Either
Connected	Indication	Either
Send	Request	Either
Receive	Request	Either
Received	Indication	Either
Reset	Request	Automation device
Reset received	Indication	DT device

6.2 Interconnect layer protocol service definitions

6.2.1 Connect service request

An ADT port uses the **Connect** protocol service request to initiate a connection between a local interconnect port and a specific remote interconnect port. One ADT port may establish not more than one concurrent connection.

Service Response = Connect (IN (Local Port, Remote Port, OUT (Connection))

Input arguments:

Local Port: An identifier for the local interconnect port.

Remote Port: The identifier for the remote interconnect port. If the **Remote Port** argument is not specified, then

Output arguments:

Connection: The identifier for the connection.

Service Response assumes one of the following values:

GOOD: The request completed successfully.

INVALID LOCAL PORT: The request failed because the **Local Port** argument did not specify a valid local interconnect port.

LOCAL PORT IN USE: The request failed because the **Local Port** argument specified a local interconnect port that was unable to support any more connections.

INVALID REMOTE PORT: The request failed because the **Remote Port** argument did not specify a valid remote interconnect port.

CONNECTION REFUSED: The request failed because the **Remote Port** did not accept the connection. This service response shall be reported if no ADT port had performed a **Listen** service request on the remote interconnect port.

6.2.2 Listen service request

An ADT port uses the **Listen** protocol service request to await a connection. One ADT port may await one connection on a local ADT serial port. Multiple ADT ports may await one connection each on a local ADT Ethernet port.

Service Response = Listen (IN (Local Port, OUT (Connection))

Input arguments:

Local Port: An identifier for the local interconnect port.

Output arguments:

Connection: The identifier for the connection.

Service Response assumes one of the following values:

GOOD: The request completed successfully.

INVALID LOCAL PORT: The request failed because the **Local Port** argument did not specify a valid local interconnect port.

LOCAL PORT IN USE: The request failed because the **Local Port** argument specified a local interconnect port that was unable to support any more connections.

6.2.3 Connected service indication

An interconnect port uses the **Connected** service indication to notify the ADT port that the requested connection has been established. The **Connected** service indication shall not be invoked if the ADT port has not invoked either a **Connect** or a **Listen** service request.

Connected (IN (Connection))

Input arguments:

Connection: The identifier for the connection.

6.2.4 Send service request

An ADT port uses the **Send** service request to send data on a connection. If a subsequent **Send** service request is invoked before all of the data in the buffer specified by a previous **Send** service request, then all of the data in the buffer for the previous invocation shall be sent before any data in the buffer of the subsequent invocation is sent.

If the **Send** service request returns a service response of **OK**, then the contents of the buffer may be modified without affecting the data to be transmitted.

When the **Send** service request returns a service response of **OK**, then the characters may or may not have been transmitted on the physical connection.

Service Response = **Send (IN (Connection, Buffer, Buffer Size))**

Input arguments:

Connection: The identifier for the connection.

Buffer: A buffer containing data to be transmitted. The data in the buffer shall be encoded (see 7.2).

Buffer Size: The number of characters of encoded data to be transmitted on the connection.

Service Response assumes one of the following values:

GOOD: The request completed successfully.

INVALID CONNECTION: The request failed because the **Connection** argument did not specify an established connection.

INVALID BUFFER: The request failed because the **Buffer** argument did not specify a valid buffer.

OUT OF RESOURCES: The request failed because the interconnect port lacked resources to accept more characters for transmission.

6.2.5 Receive service request

An ADT port invokes the **Receive** service request to receive data from a connection. The data received shall be processed as specified in clause 7.

If the **Receive** service request is invoked a second time before the **Received** service indication has been invoked, then the second **Receive** service request shall be rejected with a service response of **RECEIVE PENDING**.

Service Response = **Receive (IN (Connection, Buffer, Buffer Size))**

Input arguments:

Connection: The identifier for the connection.

Buffer: A buffer to contain received data.

Buffer Size: The maximum number of characters of encoded data to be placed in the buffer.

Service Response assumes one of the following values:

GOOD: The request completed successfully.

INVALID CONNECTION: The request failed because the **Connection** argument did not specify an established connection.

INVALID BUFFER: The request failed because the **Buffer** argument did not specify a valid buffer.

RECEIVE PENDING: The request failed because the ADT port has invoked the **Receive** service request and the interconnect port has not yet invoked the **Received** service indication.

6.2.6 Received service indication

The **Received** service indication notifies the ADT port that a number of characters have been received.

There is not a one-to-one correspondence between invocations of **Send** in one ADT port and invocations of **Received** in the other ADT port, i.e., the characters delivered in one invocation of Received may have been sent by one or more invocations of **Send**. Similarly, the characters sent in one invocation of Send may be delivered in one or more invocations of **Received**.

Received (IN (Connection, Buffer, Received Character Count))

Input arguments:

Connection: The identifier for the connection.

Buffer: A buffer containing data received. The data in the buffer shall be encoded (see 6.2).

Received Character Count: The number of characters received and placed in the buffer.

6.2.7 Close service request

An ADT port invokes the **Close** service request to close a connection.

If the **Send** service request is invoked after the **Close** service request, then the **Send** service request shall be rejected with a service response of **INVALID CONNECTION**.

Service Response = Close (IN (Connection))

Input arguments:

Connection: The identifier for the connection.

Service Response assumes one of the following values:

GOOD: The request completed successfully.

INVALID CONNECTION: The request failed because the **Connection** argument did not specify an established connection.

6.2.8 Closed service indication

The **Closed** service indication notifies the ADT port that the connection has been closed. The **Received** service indication shall not be invoked after the **Closed** service indication has been invoked and before the **Connected** service indication has been invoked. When the **Closed** service indication is invoked, the ADT port shall send a Close Event to the port state machine.

Closed (IN (Connection))

Input arguments:

Connection: The identifier for the connection.

6.2.9 Reset service request

An automation device uses the **Reset** service request to reset the ADT port in a DT device.

Reset (IN (Local Port))

Local Port: An identifier for the local interconnect port.

6.2.9 Reset received service indication

The **Reset received** service indication resets all connected ADT ports.

Reset received (IN (Local Port))

Local Port: An identifier for the local interconnect port.

6.3 ADT serial port support of link layer protocol services

6.3.1 Connection establishment

When an ADT port invokes either the **Connect** or the **Listen** service request, the connection is considered to be established immediately. Invocation of either service request shall cause no transmission of data on the physical link. When either service request is invoked, the interconnect port shall invoke the **Connected** service indication. The **Connected** service indication may be invoked before the **Connect** or **Listen** service request has returned.

The ADT serial port shall support only one connection. If a **Connect** or **Listen** service request has completed successfully and disconnection has not occurred, then a subsequent **Connect** or **Listen** request shall be rejected with a status of LOCAL PORT IN USE.

Table x shows how the arguments to the **Connect** service request are used by the ADT serial port.

Table x – Connect service request usage by ADT serial port

Argument	ADT serial implementation
Local Port	Used to select the interconnect port
Remote Port	Ignored
Connection	Assigned by the interconnect port and used by subsequent service requests and indications

Table x+1 shows how the arguments to the **Listen** service request are used by the ADT serial port.

Table x+1 – Listen service request usage by ADT serial port

Argument	ADT serial implementation
Local Port	Used to select the interconnect port
Connection	Assigned by the interconnect port and used by subsequent service requests and indications

Table x+2 shows how the argument to the **Connected** service indication is set by the ADT serial port.

Table x+2 – Connected service indication usage by ADT serial port

Argument	ADT serial implementation
Connection	The value of the **Connection** argument returned by the **Connect** or **Listen** service request

6.3.2 Data transmission

Table x+3 shows how the arguments to the **Send** service request are used by the ADT serial port.

Table x+3 – Send service request usage by ADT serial port

Argument	ADT serial implementation
Connection	Assigned by the interconnect port and used by subsequent service requests and indications
Buffer	The buffer containing data to be transmitted
Buffer Size	The number of characters in the buffer to be sent. The characters are encoded, i.e., the number includes Escape characters

6.3.3 Data reception

Table x+4 shows how the arguments to the **Receive** service request are used by the ADT serial port.

Table x+4 – Receive service request usage by ADT serial port

Argument	ADT serial implementation
Connection	Assigned by the interconnect port and used by subsequent service requests and indications
Buffer	The buffer to contain received data
Buffer Size	The maximum number of characters to be placed in the buffer

Table x+5 shows how the arguments to the **Received** service indication are set by the ADT serial port.

Table x+5 – Received service indication usage by ADT serial port

Argument	ADT serial implementation
Connection	Assigned by the interconnect port and used by subsequent service requests and indications
Buffer	The buffer containing the received data. The buffer shall be the same buffer specified in the previous invocation of the **Receive** service request.
Buffer Size	The number of characters placed in the buffer

6.3.4 Closing a connection

When an ADT port successfully invokes the **Close** service request:

a) the interconnect port shall transmit all characters which had been delivered to the ADT serial port by previous invocations of the **Send** service request which completed successfully;

b) the ADT serial port may discard any characters received on the physical connection after the invocation of the **Close** service request; and

c) if any characters have been received by the ADT serial port and not yet transferred to the ADT port, then the ADT serial port shall accept **Receive** service requests and invoke the **Received** service indication until all received characters have been transferred.

When all characters received on the ADT serial port have been transferred to the ADT port, then the ADT serial port shall invoke the **Closed** service indication. The **Closed** service indication may be invoked before the **Close** service request completes.

Table x+6 shows how the argument to the **Close** service request is set by the ADT port.

Table x+6 – Close service request usage by ADT serial port

Argument	ADT serial implementation
Connection	The value of the **Connection** argument returned by the **Connect** or **Listen** service request

Table x+7 shows how the argument to the **Closed** service request is set by the ADT serial port.

Table x+7 – Closed service request usage by ADT serial port

Argument	ADT serial implementation
Connection	The value of the **Connection** argument returned by the **Connect** or **Listen** service request

6.3.5 Reset received

A DT device shall treat the invocation of the **Reset received** service indication either:

a) as a port logout (see 6.5.5); or

b) as a hard reset (see 4.7).

6.4 ADT TCP port support of link layer protocol services

6.4.1 Connection establishment

When an ADT port invokes the **Connect** service request, an ADT interconnect supporting TCP (i.e., an ADT TCP port) shall perform an active **OPEN** call (see RFC 793) to the remote interconnect port whose IP address is specified in the **Remote Port** argument and using a remote port number of 4169. The means by which the ADT port learns the IP address of the remote interconnect port is beyond the scope of this standard.

When an active **OPEN** call has successfully completed, each ADT TCP port shall invoke the **Connected** service indication to its corresponding ADT port.

The ADT TCP port may support more than one connection.

Table y shows how the arguments to the Connect service request are used by the ADT TCP port.

Table y – Connect service request usage by ADT TCP port

Argument	ADT TCP port implementation
Local Port	**local port** argument to the **OPEN** call
Remote Port	IP address component of the **foreign socket** argument to the **OPEN** call
Connection	Connection identifier created by the ADT TCP port. This identifier shall correspond to the **local connection name** returned by the **OPEN** call.

When an ADT port invokes the **Listen** service request, an ADT TCP port shall perform a passive **OPEN** call on the local ADT interconnect port specified in the **Local Port** argument and using a local port number of 4169.

Table y+1 shows how the arguments to the **Listen** service request are used by the ADT TCP port.

Table y+1 – Listen service request usage by ADT TCP port

Argument	ADT TCP port implementation
Local Port	**local port** argument to the **OPEN** call
Connection	Connection identifier created by the ADT TCP port. This identifier shall correspond to the **local connection name** returned by the **OPEN** call.

Table y+2 shows how the argument to the **Connected** service indication is set by the ADT TCP port.

Table y+2 – Connected service indication usage by ADT TCP port

Argument	ADT TCP port implementation
Connection	Connection identifier returned by the **Connect** service request

6.4.2 Data transmission

When the Send service request is invoked, the ADT TCP port shall invoke the **SEND** call (see RFC 793) with the **PUSH flag** argument set. Table y+3 shows how the arguments to the **Send** service request are used by the ADT TCP port.

Table y+3 – Send service request usage by ADT TCP port

Argument	ADT TCP port implementation
Connection	Connection identifier returned by the **Connect** service request
Buffer	**buffer address** argument to **SEND** call
Buffer Size	**byte count** argument to **SEND** call

6.4.3 Data reception

Table y+4 shows how the arguments to the **Receive** service request are used by the ADT TCP port.

Table y+4 – Receive service request usage by ADT TCP port

Argument	ADT TCP port implementation
Connection	Connection identifier returned by the **Connect** service request
Buffer	**buffer address** argument to **RECEIVE** call

Buffer Size	**byte count** argument to **RECEIVE** call

Table y+5 shows how the arguments to the **Received** service indication are used by the ADT TCP port.

Table y+5 – Received service indication usage by ADT TCP port

Argument	ADT TCP port implementation
Connection	Connection identifier returned by the **Connect** service request
Buffer	**buffer address** argument to **RECEIVE** call
Buffer Size	The number of characters placed in the buffer

6.4.4 Closing a connection

When an ADT port successfully invokes the **Close** service request, then the ADT TCP port shall invoke the **CLOSE** call (see RFC 793). TCP guarantees that characters previously transferred with the **SEND** call shall be delivered before the connection is closed.

Table y+6 shows how the argument to the **Close** service request is used by the ADT TCP port.

Table y+6 – Close service request usage by ADT TCP port

Argument	ADT TCP port implementation
Connection	Connection identifier returned by the **Connect** service request

When an ADT TCP port enters the CLOSED state (see RFC 793), it shall invoke the **Closed** service indication. Table y+7 shows how the argument to the **Closed** service request is set by the ADT TCP port.

Table y+7 – Closed service request usage by ADT TCP port

Argument	ADT TCP port implementation
Connection	Connection identifier returned by the **Connect** service request

6.4.5 Reset received

A DT device shall treat the invocation of the **Reset received** service indication either:

a) as a **Closed** service indication (see 6.6.9) and may open a new connection; or

b) as a hard reset (see 4.7).

Renumbering

Clauses presently numbered 6 and higher are renumbered to 7 and higher. Notes, figures, and tables will also need to be renumbered.

Figures 12 and higher are renumbered to 16 and higher.