Mitel Technical Specification 21

MiLAP/MiLAP-S Specification

Neil Sipkes
Systems Design and Standards

Version B02 (Dave Walker & Scott Walker)

31 October 1996

The one-octet address format is the only one currently implemented in MiLAP and MiLAP-S.

The UA unnumbered response is used by a data link layer entity to acknowledge the receipt and acceptance of the mode-setting commands SABM and DISC. Received mode-setting commands are not processed until the UA response is transmitted. No information field is permitted with the UA response. The transmission of the UA response does not indicate the clearance of any busy condition that was reported by the earlier transmission of an RNR frame by the same data link layer entity.

This is in variance with Q.921 [10], which does allow the transmission of a UA response to clear a busy condition.

Document History

Version	Date	Description	Author(s)
A00	15-JUL-1992	First draft review	Neil Sipkes
A00	15-SEP-1992	Second draft review	Neil Sipkes
A01	06-NOV-1992	First release of MTS21	Neil Sipkes
B01	01-JUN-1995	Added Section D, corected state tables and many problems in section C	Steve Duhn & François Audet
B02	31-OCT-1996	Corrections to Section D	Scott Walker & Dave Walker
B03	12-July 1999	MTS21 converted from frame to word Doc format	K.Steeden

This is to acknowledge the people who provided input or reviewed version B01:

Roland Michaud Communicating Objects
Dave McNeil Migration & Peripherals Software
Pierre Lavallée Migration & Peripherals Software
Chris Nason Migration & Peripherals Software
Kim Currie System Control & Digital Terminals
Radovan Prodanovic System Control & Digital Terminals
Roger Bastin System Control & Digital Terminals
Normand Cyr Network Enhancement Products
Tom McGuinness Peripherals and Interfaces
Dave Perry System Control & Digital Terminals
Steve Duhn Automated Testing Solution

Andy Weatherson Network Enhancement Products
François Audet Systems Design and Standards

1.0 Reviewers for MTS21

This is to acknowledge the people who have reviewed version A01 this document:

Roland Michaud Peripheral Firmware
Dave Perry Peripheral Firmware
Bob Armstrong Product Verification
Steve Duhn Product Test Automation
Tom McGuinness Peripherals and Interfaces
Ken Duesling Peripheral Firmware
Normand Cyr Peripherals and Interfaces
Gary Lam NTDD
John Boyd GX5000

The conformance testing (section D) was written by Steve Duhn.

2.0 Introduction

Mitel Technical Specifications for DNIC, MiLINK, MiLAP, and MiNET
Mitel Technical Specifications (MTS) for DNIC, MiLINK, MiLAP, and MiNET define the physical, electrical, and procedural requirements (corresponding to Layers 1, 2, and 3 of the OSI Reference Model) for digital sets and peripherals. In addition, the scope of the specifications for MiLAP and MiNET extends to the system level as well.
The purpose of the specifications is to ensure correct functioning and consistency of implementation. These MTSs are based on internal documents generated by various development groups, and also reflect the latest status of the Protocols Working Group (PWG). Relevant test principles are included wherever possible to verify conformance to each specification. Wherever appropriate, safety or overvoltage requirements, electromagnetic compatibility limitations or other compatibility or regulatory requirements are also included for completeness.
The interpretation of these MTSs is the sole responsibility of the Technical Requirements and Standards Department. Periodic updates will be issued to ensure, amongst other things, compliance with the most recent decisions of the PWG.
The Mitel Technical Specifications for DNIC, MiLINK, MiLAP, and MiNET are structured as follows:
MTS20 MiLINK Specification
MTS21 MiLAP/MiLAP-S Specification
MTS22 MiNET Specification
MTS23 DNIC Specification

Section A - Overview

1 Document Format of MTS21

This Mitel Technical Specification (MTS21) is divided into four sections:
Section A Provides an overview of MiLAP and MiLAP-S.
Section B Provides a general introduction to the concepts of the data link layer protocol.

Section C Provides the protocol definition for the data link layer. The information in this section is based on Mitel documents PS.7 [1] and FAD.26 [2].
Section D Details the conformance testing to be used in verifying the protocol definition of Section C.

Text in a dashed-line box provides additional explanatory information.

Text in a shaded box highlights differences between MiLAP, MiLAP-S and Q.921.

2 Abbreviations and Acronyms Used in MTS21

ATS Abstract Test Suite
CCITT Comité Consultatif International Télégraphique et Téléphonique
CEI Connection Endpoint Identifier
CES Connection Endpoint Suffix
CME Connection Management Entity
C/R Command/Response (bit)
DC Disconnected (state)
DISC Disconnect (command)
DLCI Data Link Connection Identifier
DM Disconnected Mode (response)
DS Disconnect send (state)
EA Extended Address (bit)
F Final (bit)
FCS Frame Check Sequence
FRMR Frame reject (response)
I Information (field, frame, format)
IDU Interface Data Unit
IUT Implementation Under Test
k Maximum number of outstanding I frames
LAP Link Access Procedure
LAPB Link Access Procedure Balanced
LAPD Link Access Procedure on the D-channel
LB Local Busy (state)
LME Layer Management Entity
M Modifier (bits)
MiLAP Mitel Link Access Procedure
MiLINK Mitel proprietary physical layer
MiNET Mitel Network layer
MTS Mitel Technical Specification
NM Normal (state)
NT2 Network Termination 2 (e.g., a PBX)
N(R) Receive sequence number
N(S) Send sequence number
N200 Maximum number of frame retransmissions
N200A Maximum number of frame retransmissions (B-channel link establishment)
N201 Maximum number of I-field octets
OSI Open System Interconnection
P Poll (bit)
PDU Protocol Data Unit
PICS Protocol Implementation Conformance Statement
PIXIT Protocol Implementation Extra Information for Testing
P/F Poll/Final (bit)
PWG Mitel Protocol Working Group

RB Remote Busy (state)
REJ Reject (response)
RJ Reject send (state)
RNR Receive Not Ready (command, response)
RR Receive Ready (command, response)
RS SABM reset (state)
S Supervisory (frame)
SABM Set Asynchronous Balanced Mode (command)
SAP Service Access Point
SAPI Service Access Point Identifier
SDU Service Data Unit
TE Terminal Equipment
TEI Terminal Endpoint Identifier
T200 Retransmission timer
T210 D-channel access timer
U Unnumbered (frame)
UA Unnumbered Acknowledge (response)
UI Unnumbered Information (frame)
V(A) Acknowledge state variable
V(R) Receive state variable
V(S) Send state variable
X-Notation Symbolic representation of message flow state transitions, show in order

3 References

Mitel PS.7, *MiLAP - Mitel Link Layer Access Procedure*, Version A04, March 1992 (in SMS STD library)
Mitel FAD.26, *Software Specification for the Implementation of DESKBUS*, Version A01, March 1992 (in SMS PERAPP library)
Mitel MTS20, *MiLINK Specification*, Version A01
Mitel MTS22, *MiNET Specification*, Version A07
CCITT Recommendation X.200, *Reference Model of Open Systems Interconnection for CCITT Applications*
CCITT Recommendation X.210, *OSI Layer Service Conventions*
CCITT Recommendation X.211, *Physical Service Definition of Open Systems Interconnection for CCITT Applications*
CCITT Recommendation X.212, *Data Link Service Definition of Open Systems Interconnection for CCITT Applications*
CCITT Recommendation Q.920 (I.440), *ISDN User-Network Interface Data Link Layer - General Aspects*
ITU-T Recommendation Q.921, *ISDN User-Network Interface Data Link Layer Specification*, March 1993
ITU-T Recommendation Q.921 bis, *Digital Subscriber Signalling System No. 1 - Abstract Test Suite for LAPD Conformance Testing*, March 1993
ISO DIS 9646, Parts 1-5, *OSI Conformance Testing Methodology and Framework, SC21/WG1 Sydney Meeting*, 01 February 1989

Section B - General Concepts

1.0 General

This specification (MTS21-B) describes in general terms the procedures for the data link layer (layer 2). The concepts described are based on Q.920 [9]. MTS21-C provides the detailed procedures specific to MiLAP and MiLAP-S.

The purpose of layer 2 is to convey information between layer 3 (network layer) entities.

The concepts and procedures take into consideration the principles and terminology of CCITT Recommendations X.200 [5] and X.210 [6] - the reference model and layer service conventions for Open System Interconnection (OSI).

- One example of an applicable physical layer (MiLINK) can be found in MTS20 [3]. One example of an applicable layer 3 protocol (MiNET) can be found in MTS22 [4].
- The term data link layer is used in the main text of this specification. However, mainly in figure and tables, the terms layer 2 and L2 are used as abbreviations. Furthermore, the term layer 3 is used to indicate the layer above the data link layer.
- In the context of this specification, the term “network” refers to a switching system (eg. PBX), whereas the term “user” refers to station-side devices (eg. telephone sets, peripherals, etc.).

2.0 Concepts and terminology

The basic structuring technique in the OSI reference model is layering. According to this technique, communication among application processes is viewed as being logically partitioned into an ordered set of layers represented in a vertical sequence as shown in Figure 1/MTS21-B.

A data link layer Service Access Point (SAP) is the point at which the data link layer provides services to layer 3. Associated with each data link layer SAP is one or more data link connection endpoint (see Figure 2/MTS21-B). A data link connection endpoint is identified by a data link Connection Endpoint Identifier (CEI) as seen from layer 3 and by a Data Link Connection Identifier (DLCI) as seen from the data link layer.

Entities exist in each layer. Entities in the same layer, but in different systems which must exchange information to achieve a common objective are called “peer entities”. Entities in adjacent layers interact through their common boundary.

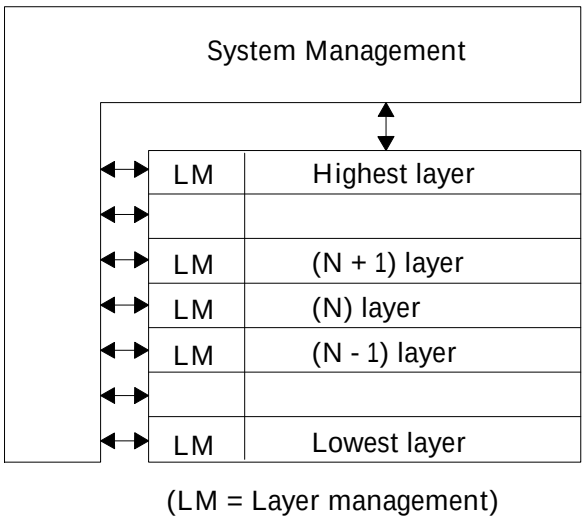


Figure 1/MTS21-B
Layering

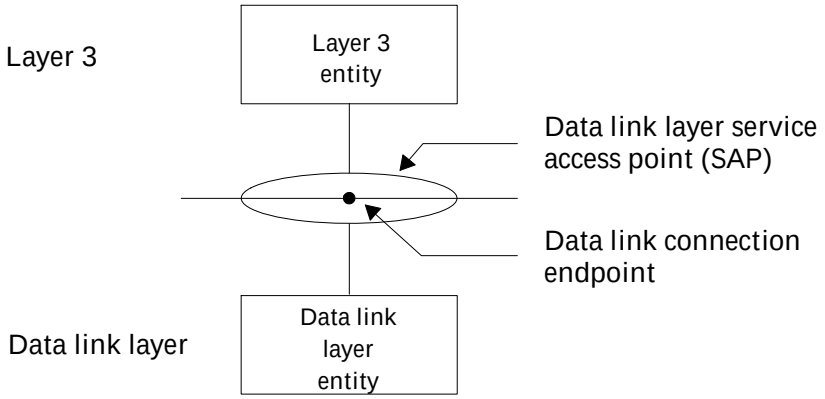


Figure 2/MTS21-B
Entities, service access points and endpoints

Cooperation between data link layer entities is governed by a peer-to-peer protocol specific to the layer. In order for information to be exchanged between two or more layer 3 entities, an association must be established between the layer 3 entities via the data link layer using a data link layer protocol. This association is called a data link connection. Data link connections are provided by the data link layer between two or more SAPs (see Figure 3/MTS21-B).

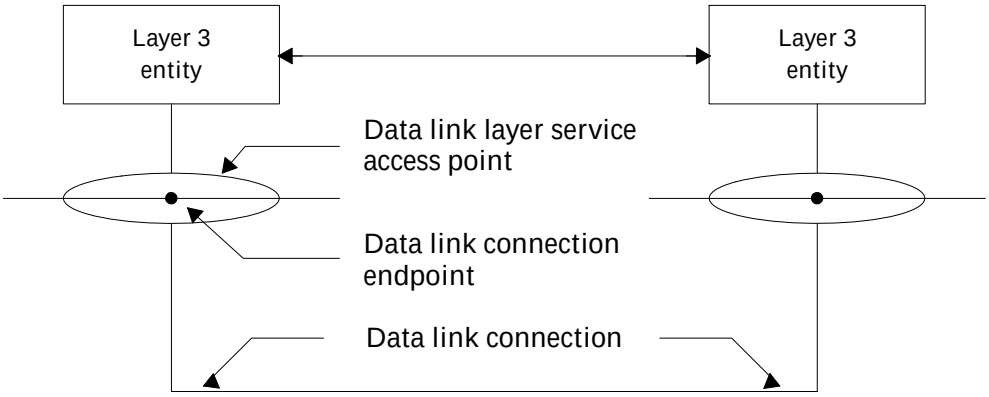


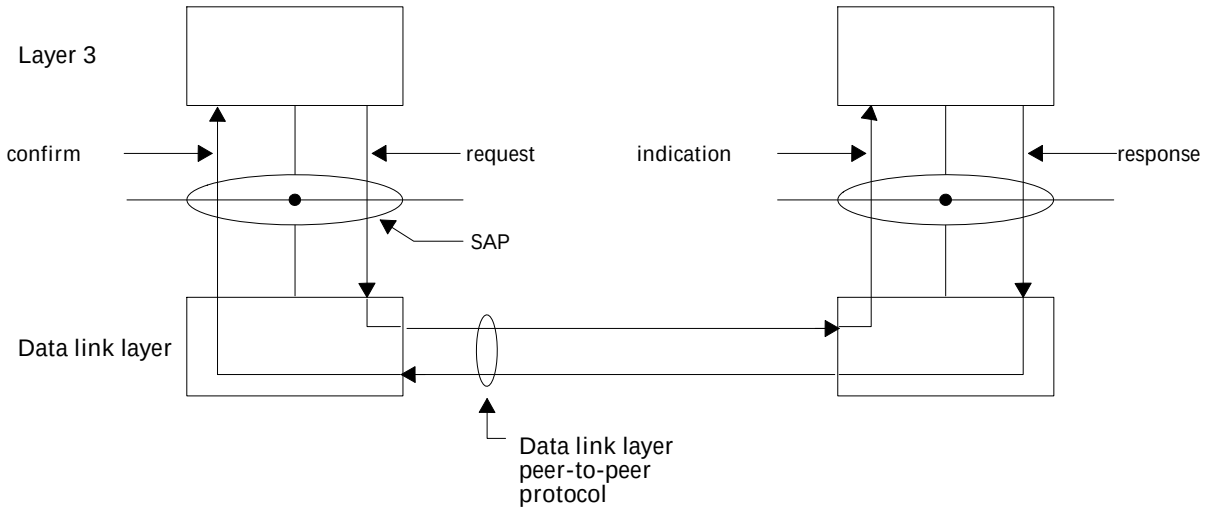
Figure 3/MTS21-B
Peer-to-peer relationship

Data link layer protocol data units (PDUs) are conveyed between data link layer entities by means of a physical connection, making use of physical service data units (SDUs).

Layer 3 requests services from the data link layer via service primitives. The same applies for the interaction between the data link layer and the physical layer. The primitives represent, in an abstract way, the logical exchange of information and control between the data link layer and adjacent layers. They do not specify or constrain implementation.

The primitives that are exchanged between the data link layer and adjacent layers are of the following four types (see also Figure 4/MTS21-B):

a) request;
b) indication;
c) response; and
d) confirm.



Note - The same principle applies for data link layer-physical layer interactions.

Figure 4/MTS21-B
Primitive action sequence

The *request* primitive type is used when a higher layer is requesting a service from the next lower layer.

The *indication* primitive type is used by a layer providing a service to notify the next higher layer of any specific activity which is service related. The *indication* primitive may be the result of an activity of the lower layer related to the primitive type *request* at the peer entity.

The *response* primitive type is used by a layer to acknowledge receipt, from a lower layer, of the primitive type *indication*.

The *confirm* primitive type is used by the layer providing the requested service to confirm that the activity has been completed.

Not all primitives require all four action sequences to be used.
See Table 5/MTS21-C for more detailed information on which types are applicable to specific primitives.

Layer-to-layer interactions are specified in MTS21-C.

Information is transferred, in various types of data units carried in primitives, between peer entities and between entities in adjacent layers that are attached to a specific SAP (see X.200 [5]). The data units are of two types:

- protocol data units (PDUs) of a peer-to-peer protocol; and
- interface data units (IDUs). These may contain information relevant to the peer entities at the ends of the connection or information of local significance such as layer-to-layer information concerning status and specialized service requests.

The PDUs of the layer 3 peer-to-peer protocol are carried by the data link connection in the form of service data units (SDUs). The contents of IDUs containing layer-to-layer information concerning status and specialized service requests, which are of local significance (i.e., not peer-to-peer), are never conveyed over a data link connection or a physical connection.

This specification defines (see also Figure 5/MTS21-B):

a) the peer-to-peer protocol for the transfer of information and control between any pair of data link layer service access points; and

b) the interactions between the data link layer and layer 3, and between the data link layer and the physical layer.

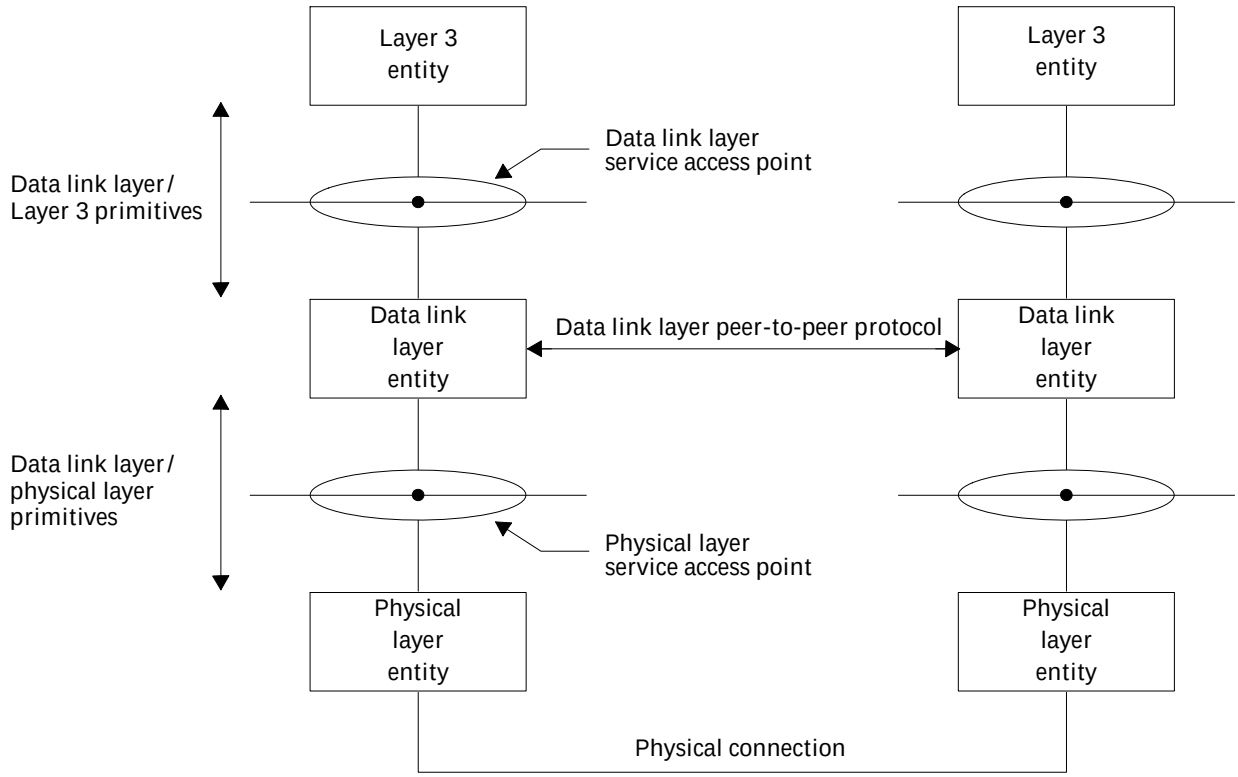


Figure 5/MTS21-B
Data link layer reference model

3.0 Overview description of data link layer functions and procedures

3.1 General

The purpose of the data link layer is to convey information between layer 3 entities. Specifically, the data link layer will support multiple terminal installations at the user-network interface.
All data link layer PDUs are transmitted in frames which are delimited by flags (a flag is a unique bit pattern). The frame structure is defined in MTS21-C.

The data link layer includes functions for:
a) the provision of one or more data link connections on a channel. Discrimination between the data link connections is by means of a data link connection identifier (DLCI) contained in each frame;
b) frame delimiting, alignment and transparency, allowing recognition of a sequence of bits transmitted
over a channel as a frame;
c) sequence control, to maintain the sequential order of frames across a data link connection;
d) detection of transmission, format and operational errors on a data link connection, and notification to higher layers of unrecoverable errors;
e) recovery from detected transmission, format, and operational errors;
f) flow control.

Data link layer functions provide the means for information transfer between multiple combinations of data link connection endpoints. The information transfer may be via point-to-point data link connections or via broadcast data link connections. In the case of point-to-point information transfer, a frame is directed to a single endpoint, while in the case of broadcast information transfer, a frame is directed to one or more endpoints.

Two types of operation of the data link layer are defined for layer 3 information transfer, namely unacknowledged and acknowledged. They may coexist on a single channel.
Figure 6/MTS21-B shows three examples of point-to-point information transfer. Figure 7/MTS21-B shows an example of broadcast information transfer. For the purpose of these figures, the terms user side and TE are used to denote station-side entities (e.g., sets, MiLINK peripherals, etc.), while the terms network side and NT2 are used to denote the switching system (e.g., PBX, etc.).

Note - Parts a) and c) of Figure 6/MTS21-B are most representative of MiLAP and MiLAP-S.

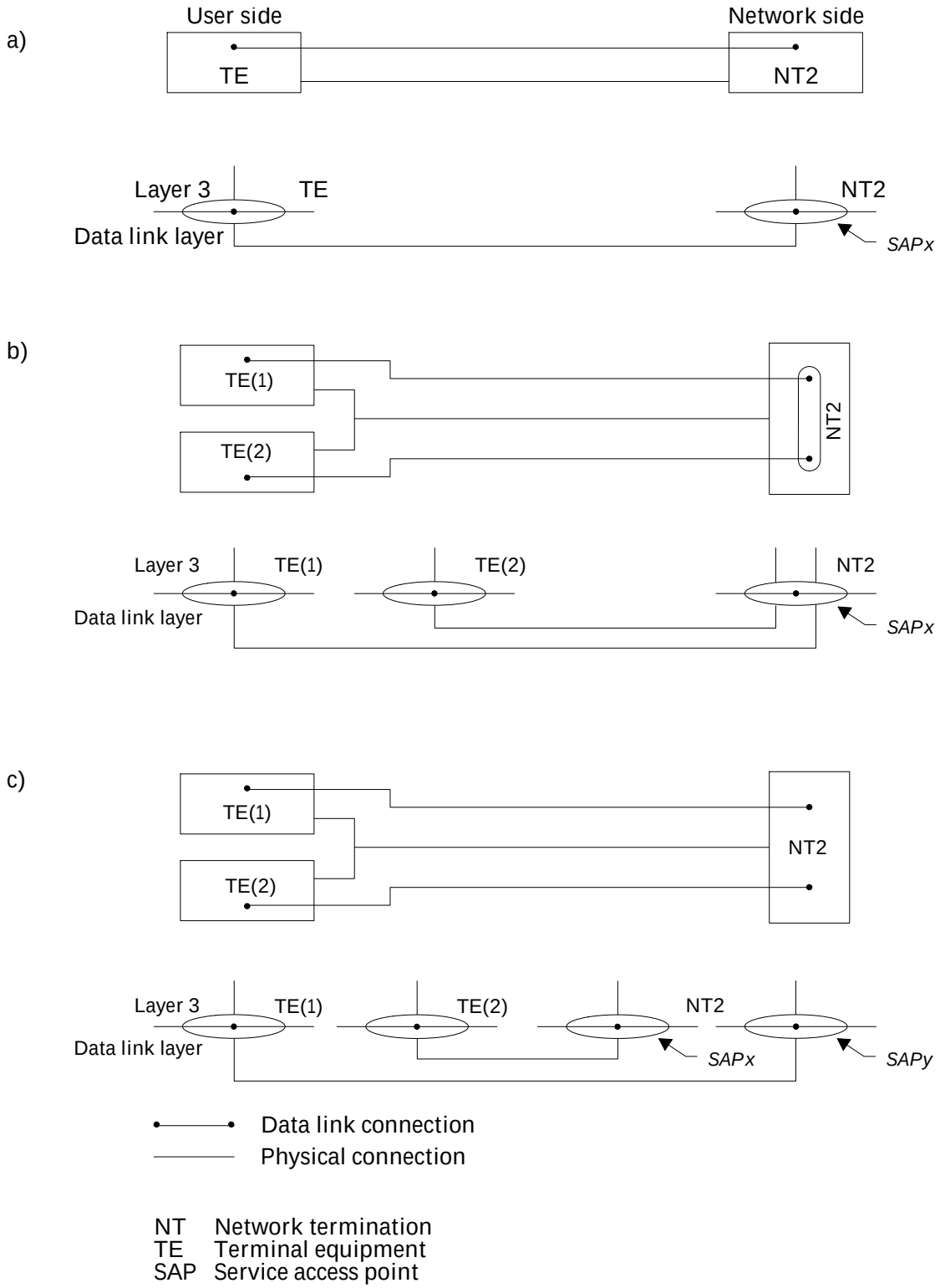


Figure 6/MTS21-B
Point-to-point data link connections

Part a) of Figure 6/MTS21-B depicts a simple point-to-point arrangement (i.e., one user terminal per network interface). Part b) of Figure 6/MTS21-B depicts a network interface that supports two terminals using the same SAP (e.g. voice call) but is still using a point-to-point arrangement (i.e., no broadcast information transfer). Part c) of Figure 6/MTS21-B depicts a network interface that supports two terminals in a point-to-point arrangement but also is using two SAPs (e.g., one voice call and one data call).

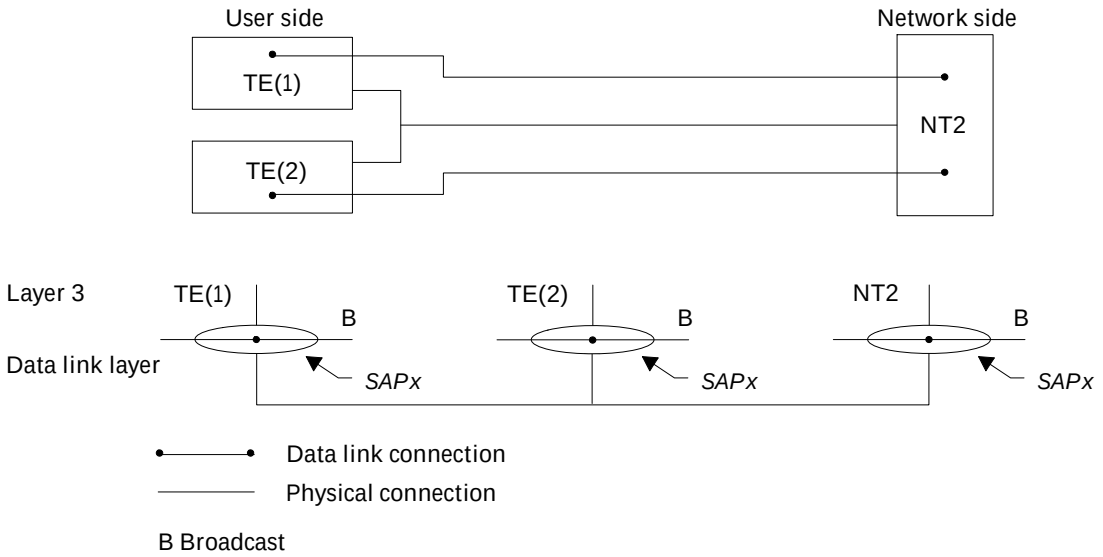


Figure 7/MTS21-B
Broadcast data link connection

Figure 7/MTS21-B depicts a user-network interface that supports broadcast information transfer. This allows the network to broadcast a message to all terminals on a specific interface.

3.2 Unacknowledged operation

With this type of operation layer 3 information is transmitted in Unnumbered Information (UI) frames. At the data link layer the UI frames are not acknowledged. Even if transmission and format errors are detected, no error recovery mechanism is defined. Flow control mechanisms are not defined.

Unacknowledged operation is applicable for point-to-point and broadcast information transfer; that is, a UI frame may be sent to a specific endpoint or broadcast to multiple endpoints associated with a specific Service Access Point Identifier (SAPI).

UI frames are not supported in MiLAP-S. Although defined in MiLAP, UI frames are not currently implemented.

3.3 Acknowledged operation

With this type of operation, layer 3 information is transmitted in frames that are acknowledged at the data link layer.

Error recovery procedures, based on retransmission of frames not yet acknowledged, are specified. In the case of errors which cannot be corrected by the data link layer, a report to the management entity is made. Flow control procedures are also defined.

Acknowledged operation is applicable for point-to-point information transfer.

One example of acknowledged operation transfer is multiple frame operation.

The term ?multiple frame operation? may be meaningless without some background history. Early data link layers would operate in single frame mode, whereby they would send out one frame and wait for an acknowledgement from the remote peer. Modern data link layer protocols can send multiple frames before expecting acknowledgements. For MiLAP and MiLAP-S, this ?window? may be 1 to 7 frames.

Layer 3 information is sent in numbered Information (I) frames. A number of I frames may be outstanding at the same time. Multiple frame operation is initiated by a multiple frame establishment procedure using a Set Asynchronous Balanced Mode (SABM) command.

3.4 Establishment of information transfer modes

3.4.1 Data link connection identification

Note - The concepts below, while valid in a general sense, are simplified in MiLAP and MiLAP-S. See MTS21-C.

A data link connection is identified by a Data Link Connection Identifier (DLCI) carried in the address field of each frame.

The DLCI is associated with a Connection Endpoint Identifier (CEI) at the two ends of the data link connection.

The CEI is used to identify interface data units (IDUs) passed between the data link layer and Layer 3. It consists of the SAPI and the Connection Endpoint Suffix (CES).

The DLCI consists of two elements: the SAPI and the Terminal Endpoint Identifier (TEI).

One addressing format defined for MiLAP and MiLAP-S uses a single-octet address. This address format does not differentiate SAPI and TEI, and appears as a single-value address. A format using separate SAPI and TEI is defined, but not yet implemented. See MTS21-C.

The SAPI is used to identify the service access point on the network side or the user side of the user-network interface.

The TEI is used to identify a specific connection endpoint within a service access point.

The TEI (or address) is assigned by the network, if the user equipment is of the automatic assignment category, or it is entered into the user equipment, for example, by the user or the manufacturer, if the user equipment is of the non-automatic assignment category (see § 3.4.2).

Note - In MiLAP and MiLAP-S, non-automatic addresses are not entered by the user.

The DLCI is a pure data link layer concept. It is used internally by the data link layer entity and is not known by the layer 3 entity or management entity. In these latter entities, the concept of CEI is used instead.

The CEI is composed of the SAPI information and the CES reference value. The CES is a value selected by the layer 3 or management entity to address the data link layer entity. When the relevant TEI is known by this entity, it will internally associate the DLCI to the CEI. The Layer 3 and management entities will use this CEI to address its peer entity.

3.4.2 Address administration

Note - For this section, the term address is used for both the single-octet address format and the separate TEI format.

The purpose of the address assignment procedure is to allow a user equipment to obtain an address value that the data link layer entities within the user equipment will use in subsequent communications over the data link connections.

The assigned address value is typically common to all SAPs (if more than one) in a user equipment. The procedure is conceptually located in the management entity.

Although the above is generally true for data link layer entities, for MiLAP and MiLAP-S the procedure is accomplished jointly between layer 3 and the data link layer. See MTS21-C.

When an address has been assigned, the user equipment establishes an association between the address and a CES in each SAP (that is, the DLCI is associated with a CEI). In the network, the corresponding association is made upon reception of the first frame containing the assigned address, or at the time of address assignment.

At that point in time, a data link layer peer-to-peer association has been formed.
The association between the DLCI and CEI will be removed by the address removal procedures on request from the management entity when recognizing that the address value is no longer valid.

For MiLAP and MiLAP-S this removal procedure is initiated by the data link layer, generally as a result of detection of a physical layer discontinuity. The switching system can change, as well as remove, an address. The removal process consists of forcing the user equipment back to the default address.

3.4.3 Establishment of multiple frame operation

Before point-to-point acknowledged information transfer can start, an exchange of a SABM frame and an Unnumbered Acknowledgement (UA) frame must take place.
The multiple frame establishment procedure is specified in detail in MTS21-C.

4.0 Service characteristics

4.1 General

The data link layer provides services to layer 3 and to the layer 2 management entities, and utilizes the services provided by the physical layer and layer management. A formal description of the data link layer service provided to layer 3 and layer management is given in §4.2 and §4.3, respectively. The layer management service provided to the data link layer is given in §4.3.

Note - Communication between different layers in the OSI reference model makes use of primitives which
are passed across the layer boundaries. The data link layer primitives defined in this specification represent, in an abstract way, the logical exchange of information and control between the data link layer and adjacent layers. They do not specify nor constrain implementations.

4.2 Services provided to layer 3

The specification of the interactions with layer 3 (primitives) provides a description of the services that the data link layer (plus the physical layer) offer to layer 3, as viewed from layer 3.

Two forms of information transfer service are associated with layer 3. The first is based on unacknowledged information transfer at the data link layer while the second service is based on acknowledged information transfer at the data link layer.

Layer 3 PDUs are passed to layer 2 in the form of layer 2 SDUs.

4.2.1 Unacknowledged information transfer service

Note 1 - In this case, the information transfer is not acknowledged at the data link layer. Acknowledgement procedures may be provided at higher layers.
Note 2 - Unacknowledged information transfer is defined, but not yet implemented for MiLAP. It is not supported in MiLAP-S.

The information transfer is via broadcast or point-to-point data link connections.

The characteristics of the unacknowledged information transfer service are summarized in the following:
a) provision of a data link connection between layer 3 entities for unacknowledged information transfer of PDUs within layer 2 to convey SDUs available to layer 3;
b) identification of data link connection endpoints; and
c) no verification of PDU arrival within the peer data link layer entity.
The primitives associated with the unacknowledged information transfer service are:
i) Data transfer
DL-UNIT-DATA-*request*/*indication*
The DL-UNIT-DATA-*request* primitive is used to request that a SDU be sent using the procedures for unacknowledged information transfer service. The DL-UNIT-DATA-*indication* primitive indicates the arrival of a SDU received by means of an unacknowledged information transfer service.

4.2.2 Acknowledged information transfer service

One example for mode of operation is multiple frame.

The characteristics of the acknowledged information transfer service are summarized in the following:
a) provision of a data link connection between layer 3 entities for acknowledged information transfer of PDUs within layer 2 to convey SDUs available to Layer 3;
b) identification of data link connection endpoints;
c) sequence integrity of data link layer SDUs in the absence of malfunctions;
d) notification to the peer entity in the case of errors, for example, loss of sequence;
e) notification to the management entity of unrecoverable errors detected by the data link layer; and
f) flow control.
The primitives associated with the acknowledged information transfer services are:
i) Data transfer
DL-DATA-*request*/*indication*
The DL-DATA-*request* primitive is used to request that a SDU be sent using the procedures for the acknowledged information transfer service. The DL-DATA-*indication* primitive indicates the arrival of a SDU received by means of the acknowledged information transfer service.
ii) Establishment of multiple frame operation
DL-ESTABLISH-*request*/*indication*/*confirm*
These primitives are used, respectively, to request, indicate and confirm the establishment of multiple frame operation between two service access points.
iii) Termination of multiple frame operation
DL-RELEASE-*request*/*indication*/*confirm*
These primitives are used, respectively, to request, indicate and confirm an attempt to terminate multiple frame operation between two service access points.

4.3 Administrative services

The characteristic of the administrative service currently recognized is:
a) error reporting.
The primitive associated with this service is:
i) Notification of error
MDL-ERROR-*indication*

This primitive is used to report error situations between layer 2 management and the data link layer entity.

4.4 Model of the data link service

4.4.1 General

The ability of the data link layer to execute a service request by layer 3 depends on the internal state of the data link layer. For the layer 3 entity, the internal state of the data link layer is represented by the state of that data link connection endpoint within a data link service access point which is used by this layer 3 entity to invoke a service.

Consequently, the data link service may be defined by means of data link connection endpoint states, whereby the capabilities provided by the data link layer and the service primitives may be related to these states.

In order to allow a data link service user to invoke a service by making use of primitives, the DL-primitives defined in MTS21-C have to be related to: point-to-point data link connections (acknowledged or unacknowledged transfer of information) and/or broadcast data link connections (unacknowledged transfer of information). See Table 1/MTS21-B.
An unconfirmed service is defined as a service which does not result in an explicit confirmation. A confirmed service is defined as a service which results in an explicit confirmation from the service-provider. There is not necessarily any relationship to a response from the peer service-user.
Table 1/MTS21-B

Applicability of DL-Primitives to information transfer modes

Generic name of the DL- primitive	POINT-TO-POINT INFORMATION TRANSFER MODE		BROADCAST INFORMATION TRANSFER MODE
	ACKNOWLEDGED	UNACKNOWLEDGED	
ESTABLISH	CONFIRMED SERVICE		
RELEASE	CONFIRMED SERVICE		
DATA	UNCONFIRMED SERVICE		
UNIT-DATA		UNCONFIRMED SERVICE	UNCONFIRMED SERVICE

4.4.2 Data link layer representation as seen by layer 3

4.4.2.1 Data link connection endpoint states

The states of a data link connection endpoint may be derived from the internal states of the data link layer entity supporting this type of a data link connection.

4.4.2.2 Broadcast data link layer connection services

A broadcast data link connection provides an unacknowledged information transfer service.
The broadcast data link connection endpoint is always in the information transfer state.

4.4.2.3 Point-to-point data link connection endpoint services

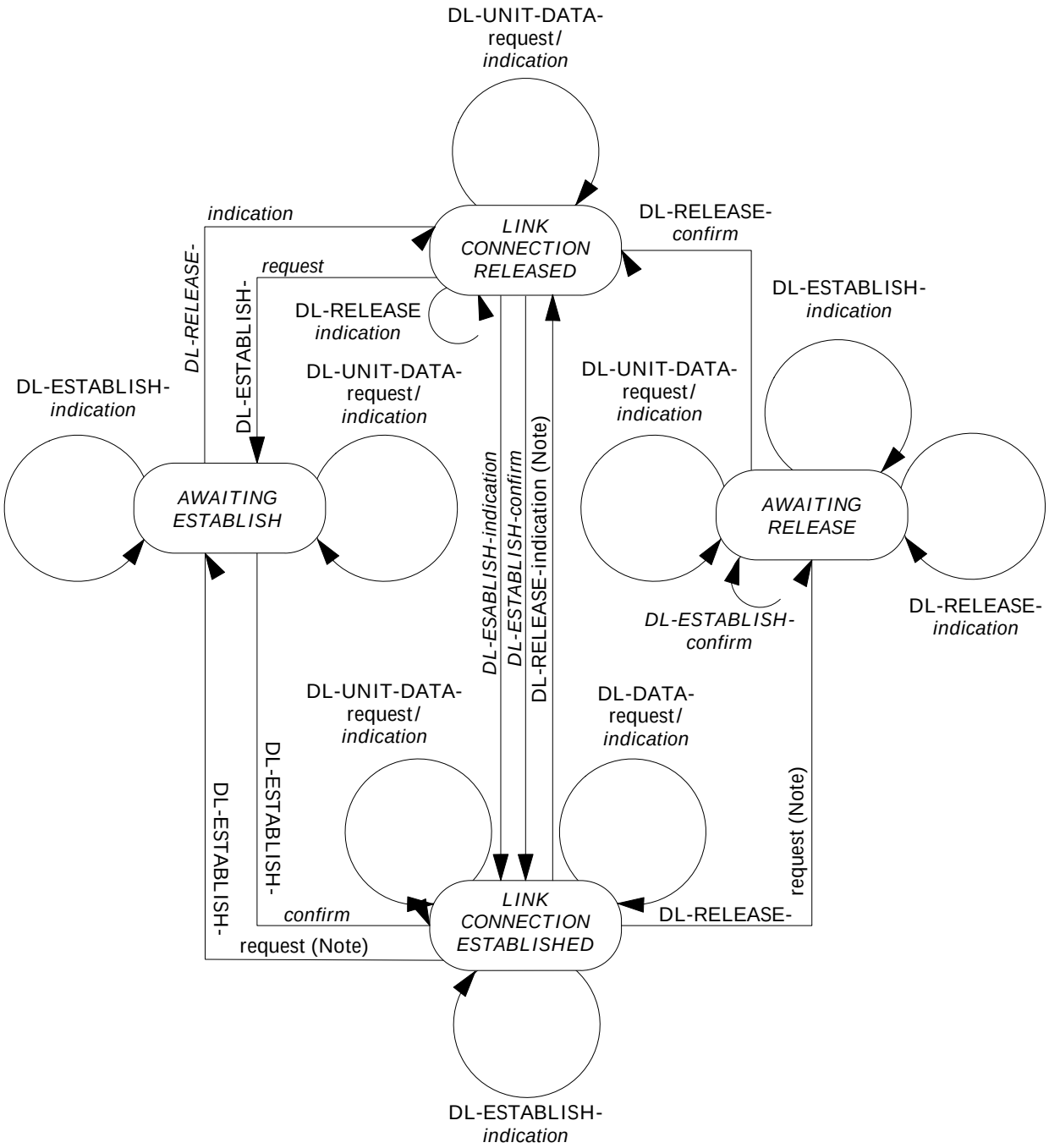
A point-to-point data link connection provides both an unacknowledged and acknowledged information transfer service. Within each data link service access point, one or more data link connection endpoint may be present, each identified by a CES.
The acknowledged information transfer service, in addition, implies the presence of the link establishment, link re-establishment and link release services.
The point-to-point data link connection endpoint states are:
- *link connection released state;*
- *awaiting establish state;*
- *awaiting release state;*
- *link connection established state.*

Refer to Table 7/MTS21-C for a mapping between the above states and the data link layer states.

4.4.2.4 Sequences of primitives at one point-to-point data link connection endpoint
The primitives provide the procedural means to specify conceptually how a data link service user can invoke a service.

This section defines the constraints on the sequence in which the primitives may occur. The sequences are related to the states at one point-to-point data link connection endpoint.

The possible overall sequences of primitives at a point-to-point data link connection endpoint are defined in the state transition diagram, Figure 8/MTS21-B. The link connection released and link connection established states are stable states whilst the awaiting establish and awaiting release are transition states.



Note - Possible loss of information

Figure 8/MTS21-B
State transition diagram for sequences of primitives at a point -to-point data link connection as seen by layer 3

4.5 Services required from the physical layer

The services provided by the physical layer are summarized in the following:

a) physical layer connection for the transparent transmission of bits in the same order in which they are submitted to the physical layer;

b) indication of the physical status of the D-channel; and
c) transmission of data link layer PDUs, passed to the physical layer in the form of physical layer SDUs, according to their respective data link layer priority.
Some of the above services may be implemented in the management entity on the user side or network side.

The method of describing these services is by means of service primitives. The primitives between the data link layer and the physical layer for the example case of MiLINK (see MTS20 [3]) are:
i) PH-DATA-*request*/*indication*
These primitives are used to request that a SDU be sent and to indicate the arrival of a message unit.
ii) PH-ACTIVATE-*indication*
This primitive is used to indicate cessation of a discontinuity in the physical layer connection.
iii) PH-DEACTIVATE-*indication*
This primitive is used to indicate a discontinuity in the physical layer connection.
iv) PH-HOLD-*request/confirm*
These primitives are used to request and confirm that the physical layer access and hold the D-channel for address negotiation.
v) PH-RELEASE-*request/confirm*
These primitives are used to request and confirm that the physical layer release the D-channel after completion of address negotiation.
vi) PH-ARBITRATE-*confirm*
This primitive is used to confirm that the physical layer has completed procedures necessary to gain access to the D-channel. For example, this primitive is issued by a MiLINK physical layer when go-ahead procedures (see MTS20) have been completed. Although this primitive should be issued by the physical layer each time arbitration is completed, it is used by the data link layer only during address negotiation.

Note - Some of the above primitives are specific to the MiLINK physical layer. Other physical layers may employ a different set of primitives.

Section C - Protocol Definition

1.0 Introduction

This specification (MTS21-C) specifies the frame structure, elements of procedure, format of fields and procedures for the proper operation of MiLAP and MiLAP-S.
The concepts, terminology, overview description of functions and procedures, and the relationship to other specifications are described in general terms in MTS21-B.

Note 1 - As stated in MTS21-B, the term data link layer is used in the main text of this specification. However, mainly in figures and tables, the terms layer 2 and L2 are used as abbreviations. Furthermore, the term layer 3 is used to indicate the layer above the data link layer.

Note 2 - All references within this document to layer management entity and/or connection management entity refer to those entities at the data link layer.
The purpose of the data link layer is to convey information between layer 3 entities (e.g., MiNET). Generally, this information transfer takes place on a D-channel, however, MiLAP and MiLAP-S can also be used to establish data link connections over a B-channel. The protocols are independent of transmission bit rate, and require a duplex bit-transparent channel.
MiLAP represents a subset of ITU-T Recommendation Q.921 [10]. One of the major differences is that MiLAP does not differentiate between network side and user side. MiLAP-S represents a further subset of MiLAP. Notes have been added to this section to highlight differences.

1.1 Differences between MiLAP and MiLAP-S

MiLAP-S, representing the most reduced form of the data link layer protocol, is generally used in devices which are subject to severe memory and cost limitations. Examples of such devices are the SS410, SS420 and SS430 digital telephone sets.

The following lists the major differences between MiLAP-S and MiLAP:

- MiLAP-S does not support the U-frame commands DISC or UI;
- MiLAP-S supports the reception, but not the transmission, of RR-poll, RNR and RNR-poll;
- MiLAP-S assumes the data link to be active at all times, therefore there are no MiLAP-S states corresponding to link connection released or awaiting release.
- a MiLAP-S data link layer entity is assumed to never be in an own receiver busy condition; and
- MiLAP-S assumes an effective transmit window size of 1.

2.0 Frame structure for peer-to-peer communication

2.1 General

All data link layer peer-to-peer exchanges shall be in frames conforming to one of the formats shown in Figure 1/MTS21-C. Two formats, one containing a one-octet address and one containing a two-octet address, are defined. The latter format is not currently implemented in MiLAP or MiLAP-S.

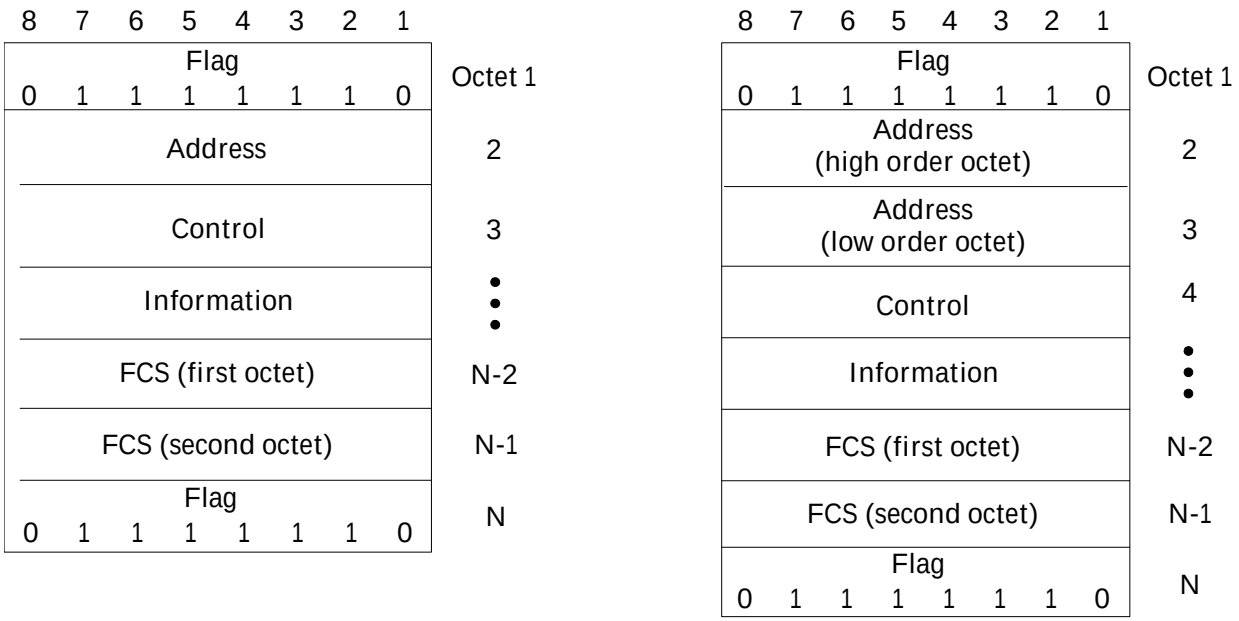


Figure 1/MTS21-C
Frame formats

2.2 Flag sequence

All frames shall start and end with a flag sequence consisting of one 0 bit followed by six contiguous 1 bits followed by one 0 bit. The flag preceding the address field is defined as the opening flag. The flag following the Frame Check Sequence (FCS) is defined as the closing flag. The closing flag of one frame may also serve as the opening flag of the next frame in some applications. However, all receivers must be able to accommodate receipt of more than one contiguous flag.

2.3 Address field

The address field shall follow the opening flag and consists of one or two octets as shown in Figure 1/MTS21-C. The formats of the address field are defined in § 3.2.

The one-octet address field is similar to that referenced in Q.921 [10] for LAPB operation over a D-channel. The two-octet address field is similar to that defined in Q.921 for LAPD operation. The one-octet address field was adopted for MiLAP and MiLAP-S to simplify addressing requirements. The two-octet address field is defined, but not currently implemented.

2.4 Control field

The control field shall follow the address field and consists of a single octet as shown in Figure 1/MTS21-C. The format of the control field is defined in § 3.4.

The one-octet control field is in variance with Q.921 [10], which uses two-octet control fields for I-frames and S-frames.

2.5 Information field

The information field shall follow the control field. The contents of the information field shall consist of an integral number of octets. The maximum number of octets in the information field is defined in § 5.8.5.

2.6 Transparency

A transmitting data link layer entity shall examine the frame content between the opening and closing flag sequences (address, control, information and FCS fields) and shall insert a 0 bit after all sequences of five contiguous 1 bits (including the last five bits of the FCS) to ensure that a flag or abort sequence is not simulated within the frame. A receiving data link layer entity shall examine the frame contents between the opening and closing flag sequences and shall discard any 0 bit which directly follows five contiguous 1 bits.

2.7 Frame check sequence (FCS) field

The FCS field shall be a 16-bit sequence. It shall be the ones complement of the sum (modulo 2) of:

a) the remainder of x^k ($x^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$) divided (modulo 2) by the generator polynomial $x^{16} + x^{12} + x^5 + 1$, where k is the number of bits in the frame existing between, but not including, the final bit of the opening flag and the first bit of the FCS, excluding bits inserted for transparency; and

b) the remainder of the division (modulo 2) by the generator polynomial $x^{16} + x^{12} + x^5 + 1$, of the product of x^{16} by the content of the frame existing between, but not including, the final bit of the opening flag and the first bit of the FCS, excluding bits inserted for transparency.

As a typical implementation at the transmitter, the initial content of the register of the device computing the remainder of the division is preset to all 1s and is then modified by division by the generator polynomial (as described above) on the address, control and information fields. The ones complement of the resulting remainder is transmitted as the 16-bit FCS.

As a typical implementation at the receiver, the initial content of the register of the device computing the remainder is preset to all 1s. The final remainder, after multiplication by x^{16} and then division (modulo 2) by the generator polynomial $x^{16} + x^{12} + x^5 + 1$ of the serial incoming protected bits and the FCS, will be 0001110100001111 (x^{15} through x^0, respectively) in the absence of transmission errors.

2.8 Format convention

2.8.1 Numbering convention

The basic convention used in this MTS is illustrated in Figure 2/MTS21-C. The bits are grouped into octets. The bits of an octet are shown horizontally and are numbered from 1 to 8. Multiple octets are shown vertically and are numbered from 1 to n.

2.8.2 Order of bit transmission

The octets are transmitted in ascending numerical order. Inside an octet, bit 1 is the first bit to be transmitted.

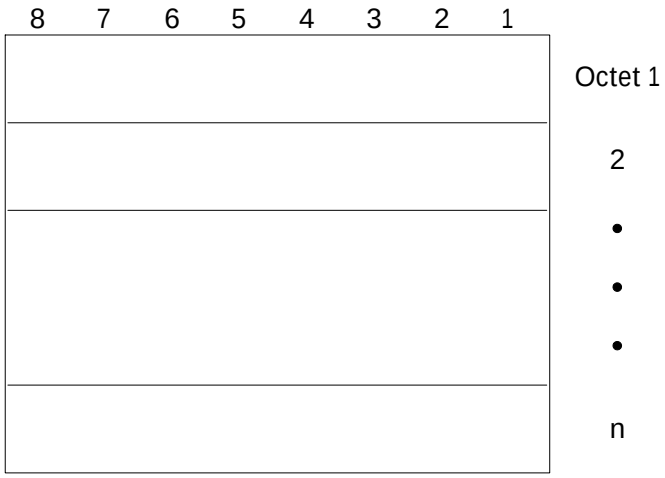


Figure 2/MTS21-C
Format convention

2.8.3 Field mapping convention

When a field is contained within a single octet, the lowest bit number of the field represents the lowest order value.

When a field spans more than one octet, the order of bit values within each octet progressively decreases as the octet number increases. The lowest bit number associated with the field represents the lowest order value.

For example, a bit number can be identified as a couple (o, b) where o is the octet number and b is the relative bit number within the octet. Figure 3/MTS21-C illustrates a field that spans from bit (1,3) to bit (2,7). The high order bit of the field is mapped on bit (1,3) and the low order bit is mapped on bit (2,7).

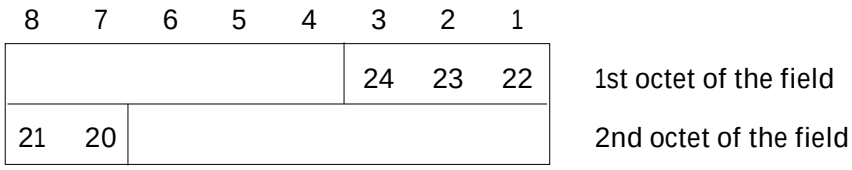


Figure 3/MTS21-C
Field mapping convention

An exception to the preceding field mapping convention is the data link layer FCS field, which spans two octets. In this case, bit 1 of the first octet is the high order bit and bit 8 of the second octet is the low order bit (see Figure 4/MTS21-C).

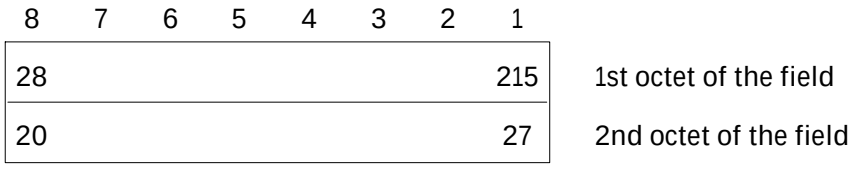


Figure 4/MTS21-C
FCS mapping convention

2.9 Invalid frames

An invalid frame is a frame which:

a) is not properly bounded by two flags; or

b) has fewer than four octets between flags of one-octet address frames; or

c) has fewer than five octets between flags of two-octet address frames; or

c) does not consist of an integral number of octets prior to zero bit insertion or following zero bit extraction; or

d) contains a frame check sequence error; or

e) contains a service access point identifier (see § 3.3.4.1) which is not supported by the receiver.

Invalid frames shall be discarded without notification to the sender. No action is taken as a result of such a frame.

2.10 Undefined frames

An undefined frame is a frame which:

a) is error-free, but which is unknown as a command or a response; or

b) has an invalid frame format; or

c) has an invalid N(R); or

d) has the final bit set when there is no poll outstanding.

Undefined frames shall be ignored, and an SABM shall be sent in response.

2.11 Frame abort

Receipt of seven or more contiguous 1 bits shall be interpreted as an abort and the data link layer shall ignore the frame currently being received.

3.0 Elements of procedures and formats of fields for data link layer peer-to-peer communication

3.1 General

The elements of procedures define the commands and responses that are used on the data link connections. Procedures are derived from these elements of procedures and are described in § 5.

3.2 Address field formats

The two address field formats are shown in Figure 5/MTS21-C. Both formats contain the address field extension bits and a command/response indication bit. In the one-octet format, the distinction between SAPI and TEI does not exist, resulting in a single-byte address (equivalent to the DLCI). In the two-octet format, separate fields are defined for the SAPI and TEI (DLCI is the concatenation of SAPI and TEI).

Note - Specific allocations for the SAPI and TEI subfields in the two-octet address field are for further study. Furthermore, the two-octet address format is defined, but not currently implemented.

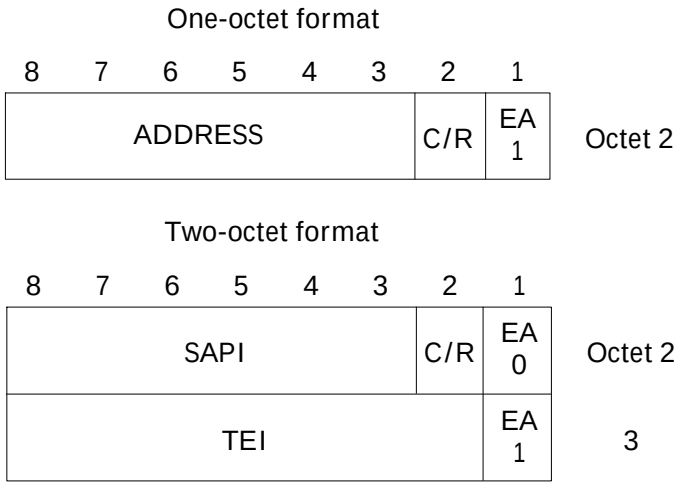


Figure 5/MTS21-C
Address field format

3.3 Address field variables

3.3.1 Address field extension bit (EA)

The address field range is extended by reserving the first transmitted bit of the address field octets to indicate the final octet of the address field. The presence of a 1 in the first bit of an address field octet signals that it is the final octet of the address field. The two-octet address field shall have bit 1 of the first octet set to 0 and bit 1 of the second octet set to 1, otherwise the frame shall be ignored.

3.3.2 Command/response field bit (C/R)

The C/R bit identifies a frame as either a command or a response. A frame shall be a command when the C/R bit is set to a 1, otherwise the frame shall be a response.

This interpretation is in variance with Q.921 [10], which defines the C/R bit differently depending on a context of ?user? and ?network?. Since MiLAP and MiLAP-S can be used on a B-channel, such distinctions are unnecessarily confusing.

3.3.3 One-octet address field

The one-octet address format is the only one currently implemented in MiLAP and MiLAP-S.

The address field allows 64 addresses to be specified, where bit 3 of the address field octet is the least significant binary digit and bit 8 is the most significant binary digit. The address values are currently allocated as shown in Table 1/MTS21-C.

Note - Address values currently unused shall be considered reserved, and are for further study.

3.3.4 Two-octet address field

3.3.4.1 Service access point identifier (SAPI)
The SAPI identifies a point at which data link layer services are provided by a data link layer entity to a layer 3 or system management entity. Consequently, the SAPI specifies a data link layer entity that should process a data link layer frame, and also a layer 3 or management entity which is to receive information carried by the data link layer frame. The SAPI allows 64 service access points to be specified, where bit 3 of the address field octet containing the SAPI is the least significant binary digit and bit 8 is the most significant. Specific SAPI values are for further study.

Note - The concept of management entities has been simplified by MiLAP and MiLAP-S. The term " system management" will be used in this specification, as opposed to management entities at each layer in addition to a system management function. System management should be considered a combination of the two management types.
Table 1/MTS21-C
One-octet address allocation

Address (decimal)	Function
0	Fixed-address devices using B1 channel
1	Fixed-address devices using B2 channel
2,3	Reserved
4 - 29	Unused
30	Auto-address negotiation
31	?Dead? address
32 - 62	Unused
63	Broadcast

3.3.4.2 Terminal endpoint identifier (TEI)
A TE may contain one or more TEIs used for point-to-point data transfer. The TEI for a broadcast data link connection is associated with all user side data link layer entities containing the same SAPI. The TEI subfield allows 128 values where bit 2 of the address field containing the TEI is the least significant binary digit and bit 8 is the most significant binary digit.
The TEI subfield bit pattern 1111111 (=127) is defined as the group TEI. The group TEI is assigned permanently to the broadcast data link connection associated with the addressed service access point (SAP). Other TEI values are for further study.

3.4 Control field formats

The control field identifies the type of frame which will be either a command or response. The control field will contain sequence numbers where applicable.
Three types of control field formats are specified: numbered information transfer (I format), supervisory functions (S format), and unnumbered information transfers and control functions (U format). For the one-octet address frame format, the control field occupies octet 3. For the two-octet address frame format, the control field occupies octet 4.
The control field formats are shown in Table 2/MTS21-C.
Table 2/MTS21-C

Control field formats

Control field bits	8	7	6	5	4	3	2	1
I format	N(R)			0	N(S)			0
S format	N(R)			P/F	S		0	1
U format	M			P/F	M		1	1

N(S) = transmitter send sequence number

N(R) = transmitter receive sequence number

3.4.1 Information transfer (I) format

The I format shall be used to perform an information transfer between layer 3 entities. The functions of N(S) and N(R) are independent; that is, each I frame has an N(S) sequence number, and an N(R) sequence number which may or may not acknowledge additional I frames received by the data link layer entity. The use of N(S) and N(R) is defined in § 5.

In variance with Q.921 [10], MiLAP and MiLAP-S do not support poll in I frames.

3.4.2 Supervisory (S) format

The S format shall be used to perform data link supervisory control functions such as the following:

a) acknowledge I frames;
b) request retransmission of I frames; and
c) request a temporary suspension of transmission of I frames.

The functions of N(R) and P/F are independent, that is each supervisory frame has an N(R) sequence number which may or may not acknowledge additional I frames received by the data link layer entity, and a P/F bit that may be set to 0 or 1.

The use of N(R) and P/F is defined in § 5.

3.4.3 Unnumbered (U) format

The U format shall be used to provide additional data link control functions and unnumbered information transfers for unacknowledged information transfer. This format does not contain sequence numbers. It does include a P/F bit that may be set to 0 or 1.

In variance with Q.921 [10], MiLAP and MiLAP-S do not support poll in UI frames.

3.5 Control field parameters and associated state variables

The various parameters associated with the control field formats are described in this section. The coding of the bits within these parameters is such that the lowest numbered bit within the parameter field is the least significant bit.

3.5.1 Poll/Final (P/F) bit

All frames contain the Poll/Final (P/F) bit. The P/F bit serves a function in both command frames and response frames. In command frames the P/F bit is referred to as the P bit. In response frames it is referred to as the F bit. The P bit set to 1 is used by a data link layer entity to solicit (poll) a response frame from the peer data link layer entity. The F bit set to 1 is used by a data link layer entity to indicate the response frame transmitted as a result of a soliciting (poll) command.
The use of the P/F bit is described in § 5.

3.5.2 Multiple frame operation - variables and sequence numbers

3.5.2.1 Modulus

Each I frame is sequentially numbered and may have the value 0 through n minus 1 (where n is the modulus of the sequence numbers). For MiLAP and MiLAP-S, the modulus equals 8 and the sequence numbers cycle through the entire range, 0 through 7.
Note - All arithmetic operations on state variables and sequence numbers contained in this MTS are affected by the modulus
operation.

The MiLAP/MiLAP-S modulus is in variance with Q.921 [10], which uses a modulus of 128.

3.5.2.2 Send state variable V(S)

Each point-to-point data link connection endpoint shall have an associated V(S) when using I frame commands. V(S) denotes the sequence number of the next I frame to be transmitted. The V(S) can take on the value 0 through n minus 1. The value of V(S) shall be incremented by 1 with each successive I frame transmission, and shall not exceed V(A) by more than the maximum number of outstanding I frames (k). The value of k may be in the range of 1 through
7.

It is recommended that MiLAP-S implementations restrict the transmit window to 1 to avoid potential retransmission problems. However, the value of k may exceed 1. The receive window size is not affected by this restriction.

3.5.2.3 Acknowledge state variable V(A)

Each point-to-point data link connection endpoint shall have an associated V(A) when using I frame commands and supervisory frame commands/responses. V(A) identifies the last I frame that has been acknowledged by its peer [V(A) minus 1 equals the N(S) of the last acknowledged I frame]. V(A) can take on the value 0 through n minus 1. The value of V(A) shall be updated by the valid N(R) values received from its peer (see § 3.5.2.6). A valid N(R) value is one that is in the range $V(A) \leq N(R) \leq V(S)$.

3.5.2.4 Send sequence number N(S)

Only I frames contain N(S), the send sequence number of transmitted I frames. At the time that an in-sequence I frame is designated for transmission, the value of N(S) is set equal to V(S).

3.5.2.5 Receive state variable V(R)

Each point-to-point data link connection endpoint shall have an associated V(R) when using I frame commands and supervisory frame commands/responses. V(R) denotes the sequence number of the next in-sequence I frame expected to be received. V(R) can take on the value 0 through n minus 1. The value of V(R) shall be incremented by 1 with the receipt of an error-free, in-sequence I frame whose N(S) equals V(R).

3.5.2.6 Receive sequence number N(R)

All I frames and supervisory frames contain N(R), the expected send sequence number of the next received I frame. At the time that a frame of the above types is designated for transmission, the value of N(R) is set equal to V(R). N(R) indicates that the data link layer entity transmitting the N(R) has correctly received all I frames numbered up to and including N(R) - 1.

3.5.3 Unacknowledged operation - variables and parameters
No variables are defined. One parameter (N201) is defined (see § 5.8.5).

3.6 Frame types

3.6.1 Commands and responses
The following commands and responses are supported by MiLAP and MiLAP-S. Table 3/MTS21-C shows commands and responses for MiLAP, while Table 4/MTS21-C shows commands and responses for MiLAP-S. Each data link connection shall support the full set of commands and responses for the protocol intended (MiLAP or MiLAP-S).
For purposes of the MiLAP and MiLAP-S procedures, those frame types not identified in the corresponding table are identified as undefined frames. The actions to be taken are specified in § 5.7.5.
Table 3/MTS21-C

MiLAP commands and responses

Format	Command	Response	Control field encoding							
			8	7	6	5	4	3	2	1
I-frame	I (information)		N(R)			0	N(S)			0
S-frame	RR (receive ready)	RR (receive ready)	N(R)			P/F	0	0	0	1
S-frame	RNR (receive not ready)	RNR (receive not ready)	N(R)			P/F	0	1	0	1
S-frame		REJ (reject)	N(R)			0	1	0	0	1
U-frame	SABM (set async. bal. mode)		0	0	1	P	1	1	1	1
U-frame	DISC (disconnect)		0	1	0	P	0	0	1	1

Table 4/MTS21-C
MiLAP-S commands and responses

Format	Command	Response	Control field encoding 8 7 6 5 4 3 2 1
I-frame	I (information)		N(R) 0 N(S) 0
S-frame	RR (receive ready)	RR (receive ready)	N(R) P/F 0 0 0 1
S-frame	RNR (receive not ready)		N(R) P 0 1 0 1
S-frame		RNR (receive not ready)	N(R) 0 0 1 0 1
S-frame		REJ	N(R) 0 1 0 0 1

MiLAP-S does not support U-frame commands DISC (disconnect) and UI (unnumbered information), or U-frame responses DM (disconnected mode) and FRMR (frame reject). MiLAP-S supports reception, but not transmission of RR-poll, RNR and RNR-poll.

3.6.2 Information (I) command

The function of the information (I) command is to transfer, across a data link connection, sequentially numbered frames containing information fields provided by layer 3. This command is used in the multiple frame operation on point-to-point data link connections.

3.6.3 Set asynchronous balanced mode (SABM) command

The SABM unnumbered command is used to place the addressed user side or network side into modulo 8 multiple frame acknowledged operation.

No information field is permitted with the SABM command. A data link layer entity confirms acceptance of an SABM command by the transmission at the first opportunity of a UA response. Upon acceptance of this command, the data link layer entity's V(S), V(A) and V(R) are set to 0. The transmission of an SABM command indicates the clearance of all exception conditions.

Previously transmitted I frames that are unacknowledged when this command is processed remain unacknowledged and are discarded. It is the responsibility of a higher level (e.g., layer 3) or the management entity to recover from the possible loss of the contents of such I frames.

3.6.4 Disconnect (DISC) command

The DISC unnumbered command is used to terminate the multiple frame operation.

Note - This command is not supported by MiLAP-S. A MiLAP-S data link shall be assumed to be active at all times.

No information field is permitted with the DISC command. The data link layer entity receiving the DISC command confirms the acceptance of the DISC command by transmission of a UA response. The data link layer entity sending the DISC command terminates the multiple frame operation when it receives the acknowledging UA
response.

In variance with Q.921 [10], neither MiLAP nor MiLAP-S support the DM (disconnected mode) response. The DM response would normally also be a valid response to a DISC command.

Previously transmitted I frames that are unacknowledged when the DISC command is processed remain unacknowledged and are discarded. It is the responsibility of a higher level (e.g., layer 3) or the management entity to recover from the possible loss of the contents of such I frames.

3.6.5 Unnumbered information (UI) command

When a layer 3 entity requests unacknowledged information transfer, the UI unnumbered command is used to send information to its peer without affecting data link layer variables. UI command frames do not carry a sequence number and therefore, the UI frame may be lost without notification.

3.6.6 Receive ready (RR) command/response

The RR supervisory frame is used by the data link layer entity to:

a) indicate it is ready to receive an I frame;

b) acknowledge previously received I frames numbered up to and including N(R) - 1 (as defined in § 5); and

c) clear a busy condition that was indicated by the earlier transmission of an RNR frame by that same data link layer entity.

In addition to indicating the status of a data link layer entity, the RR command with the P bit set to 1 may be used by the data link layer entity to ask for the status of its peer data link layer
entity.

MiLAP-S supports the reception, but not the transmission, of the above RR-poll command.

3.6.7 Reject (REJ) command/response

The REJ supervisory frame is used by a data link layer entity to request retransmission of I frames starting with the frame numbered N(R). The value of N(R) in the REJ frame acknowledges I frames numbered up to and including N(R) - 1. New I frames pending initial transmission shall be transmitted following the retransmitted I frame(s).

Only one REJ exception condition for a given direction of information transfer is established at a time. The REJ exception condition is cleared (reset) upon the receipt of an I frame with an N(S) equal to the N(R) of the REJ frame.

The transmission of a REJ frame shall also indicate the clearance of any busy condition within the sending data link layer entity that was reported by the earlier transmission of an RNR frame by that same data link layer entity.

Note - Timer T200 is not started on transmission of an REJ command.

In variance with Q.921 [10], the REJ command cannot have the P bit set to 1 in MiLAP or MiLAP-S. This poll would normally be used by a data link layer entity to query the status of its peer data link layer entity.

3.6.8 Receive not ready (RNR) command/response

The RNR supervisory frame is used by a data link layer entity to indicate a busy condition; that is, a temporary inability to accept additional incoming I frames. The value of N(R) in the RNR frame acknowledges I frames numbered up to and including N(R) - 1.
In addition to indicating the status of a data link layer entity, the RNR command with the P bit set to 1 may be used by the data link layer entity to ask for the status of its peer data link layer entity.

Note - MiLAP-S supports the reception, but not the transmission, of RNR and RNR-poll. If a MiLAP entity has sent an RNR command (indicating a local busy condition) to a MiLAP-S entity, the MiLAP entity shall subsequently clear the condition by transmitting an RR-poll to the MiLAP-S entity.

3.6.9 Unnumbered acknowledgement (UA) response

The UA unnumbered response is used by a data link layer entity to acknowledge the receipt and acceptance of the mode-setting commands SABM and DISC. Received mode-setting commands are not processed until the UA response is transmitted. No information field is permitted with the UA response. The transmission of the UA response does not indicate the clearance of any busy condition that was reported by the earlier transmission of an RNR frame by the same data link layer entity.

This is in variance with Q.921 [10], which does allow the transmission of a UA response to clear a busy condition.

4.0 Elements for layer-to-layer communication

4.1 General

Communications between layers and (for this MTS) between the data link layer and the layer management are accomplished by the means of primitives.
Primitives represent, in an abstract way, the logical exchange of information and control between the data link layer and adjacent layers. They neither specify nor constrain implementations.
Primitives consist of commands and their respective responses associated with the services requested of a lower layer. The general syntax of a primitive is:
XX - Generic name - Type: Parameters
where XX designates the interface across which the primitive is active.
For this MTS, the following definitions for XX are used:
1) DL for communication between layer 3 and the data link layer;
2) PH for communication between the data link layer and the physical layer; and
3) MDL for communication between the layer management and the data link layer.

Data link service definitions and primitives are defined in CCITT Recommendation X.212 [8]. It may prove helpful to consult this document for a better understanding of primitives. Some variances are noted in ? 4.1.1 following.

4.1.1 Generic names

The generic name of a primitive specifies the activity that should be performed. Table 5/MTS21-C illustrates the primitives defined in this MTS. Note that not all primitives have associated parameters.

The primitive generic names that are defined in this MTS are described in §§ 4.1.1.1 to 4.1.1.13.

4.1.1.1 DL-ESTABLISH
The DL-ESTABLISH primitives are used to request, indicate and confirm the outcome of the procedures for establishing multiple frame operation.

4.1.1.2 DL-RELEASE
The DL-RELEASE primitives are used to request, indicate and confirm the outcome of the procedures for terminating a previously established multiple frame operation, or for reporting an unsuccessful establishment attempt.
Table 5/MTS21-C

Primitives associated with MTS21

Generic Name	Type			Parameters		Parameter Data Contents
	Request	Indication	Confirm	Priority indicator	Parameter Data	
L3 ? L2						
DL-ESTABLISH	X	X	X	-	-	
DL-RELEASE	X	X	X	-	-	
DL-DATA	X	X	-	-	X	Layer 3 PDU (peer-to-peer message)
DL-UNIT DATA	X	X	-	-	X	Layer 3 PDU (peer-to-peer message)
DL-ASSIGN	X	-	-	-	X	Address value
DL-REMOVE	-	X	-	-	-	
M ? L2						
MDL-ERROR	-	X	-	-	X	Reason for error message
L2 ? L1						
PH-DATA	X	X	-	X	X	Data link layer PDU (peer-to-peer frame)
PH-ACTIVATE	-	X	-	-	-	
PH-DEACTIVATE	-	X	-	-	-	
PH-HOLD	X	-	X	-	-	
PH-RELEASE	X	-	X	-	-	
PH-ARBITRATE	-	-	X	-	-	

L3 ? L2: Layer 3/data link layer boundary

M ? L2: System management entity/data link layer boundary

L2 ? L1: Data link layer/physical layer boundary

4.1.1.3 DL-DATA

The DL-DATA primitives are used to request and indicate SDUs containing layer 3 PDUs which are to be transmitted, or have been received, by the data link layer using the acknowledged information transfer service.

4.1.1.4 DL-UNIT-DATA

The DL-UNIT-DATA primitives are used to request and indicate SDUs containing layer 3 PDUs which are to be transmitted, or have been received, by the data link layer using the unacknowledged information transfer service.

4.1.1.5 DL-ASSIGN
The DL-ASSIGN primitive is used by layer 3 to request that the data link layer use the address or TEI value contained within the parameter data of the primitive.

4.1.1.6 DL-REMOVE
The DL-REMOVE primitive is used by the data link layer to notify layer 3 that the current address or TEI should no longer be considered valid. This condition normally occurs as a result of a physical layer discontinuity.

The use and definition of the DL-ASSIGN primitive in MiLAP and MiLAP-S is in variance with Q.921 [10]. In Q.921, TEI negotiation takes place strictly at the data link layer, involving the data link layer and layer management entities. In MiLAP and MiLAP-S, the process for acquiring a new address involves both layer 3 (i.e., MiNET) and the data link layer entity. Furthermore, MiLAP and MiLAP-S do not provide mechanisms for layer management peer-to-peer communication. This is accomplished in Q.921 via unacknowledged information transfer using a specific SAPI (63).

Note - The procedures for address negotiation for the two-octet address format are for further study.

4.1.1.7 MDL-ERROR
The MDL-ERROR primitive is used to indicate to the connection management entity that an error has occurred.

4.1.1.8 PH-DATA
The PH-DATA primitives are used to request and indicate SDUs containing frames used for data link layer peer-to-peer communications passed to and from the physical layer.

4.1.1.9 PH-ACTIVATE
The PH-ACTIVATE primitive is used to indicate to the data link layer that the physical layer has been activated (e.g., cessation of a MiLINK discontinuity).

4.1.1.10 PH-DEACTIVATE
The PH-DEACTIVATE primitive is used to indicate to the data link layer that the physical layer has been deactivated (e.g., a MiLINK discontinuity has occurred).

4.1.1.11 PH-HOLD
The PH-HOLD primitive is used to request and confirm that the physical layer has established D-channel access hold. This mode allows the data link layer to transmit more than one frame without relinquishing the D-channel.

Note - This primitive is used for address negotiation (see § 5.3).

4.1.1.12 PH-RELEASE
The PH-RELEASE primitive is used to request and confirm that the physical layer has released D-channel access hold.
Note - This primitive is used for address negotiation (see § 5.3).

4.1.1.13 PH-ARBITRATE
The PH-ARBITRATE primitive is used to confirm that the physical layer has completed necessary D-channel access procedures (i.e., Dbusy and go-ahead arbitration - see MTS20 [3]).

Note - This primitive is used for address negotiation (see § 5.3).

4.1.2 Primitive types

The primitive types defined in this MTS are:

1) The *request* primitive type is used when a higher layer or layer management is requesting a service from the lower layer.

2) The *indication* primitive type is used by a layer providing a service to inform the higher layer or Layer management.

3) The *confirm* primitive type is used by the layer providing the requested service to confirm that the activity has been completed.

4.1.3 Parameter definition

A parameter consists of data, such as: SAPI/TEI or reasons. The parameter data is associated with the primitive and contains information related to the service. In the case of DATA primitives, the parameter data contains the SDU which allows the service user to transmit its PDU to the peer service user entity. For example, the DL-DATA parameter data contains layer 3 (i.e., MiNET) information. The PH-DATA parameter data contains the data link layer frame.

Note - The operations across the data link layer/layer 3 boundary shall be such that the layer sending a primitive can assume a temporal order of the bits within the parameter data and that the layer receiving the primitive can reconstruct the information with its assumed temporal order.

4.2 Primitive procedures

4.2.1 General

Primitive procedures specify the interactions between adjacent layers to invoke and provide a service. The service primitives represent the elements of the procedures.
In the scope of this MTS the interactions between layer 3 and the data link layer are specified.

4.2.2 Layer 3 - data link layer interactions

The states of a data link connection endpoint may be derived from the internal states of the data link layer entity supporting this type of a data link connection.
Data link connection endpoint states are defined as follows:

a) Broadcast data link connection endpoint
   - information transfer state.

b) Point-to-point data link connection endpoint
   - link connection released state;
   - awaiting establish state;
   - awaiting release state; and
   - link connection established state.

Note - For mappings between the states of the data link connection endpoint and the data link layer entity, refer to Table 7/MTS21-C.
The primitives provide the procedural means to specify conceptually how a data link service user can invoke a service.

This section defines the constraints on the sequences in which the primitives may occur. The sequences are related to the states at one point-to-point data link connection endpoint.
The possible overall sequences of primitives at a point-to-point data link connection endpoint are defined in the state transition diagram, Figure 6/MTS21-C. The link connection released and link connection established states are stable states whilst the awaiting establish and awaiting release states are transition

states.

The figure is specific to MiLAP. MiLAP-S does not have states equivalent to Link Connection Released or Awaiting release. A MiLAP-S primitive-driven state change typically occurs only with the DL-RELEASE-request primitive (see Table 12/MTS21-C).

The model illustrates the behaviour of layer 2 as seen by layer 3. The model assumes that the primitives passed between layers are implemented using a first-in/first-out queue. In this model, collisions of *request* and *indication* primitives can occur, thereby illustrating actions that seem to be in conflict with the actual layer 2 protocol description. In some implementations, these collisions could occur.

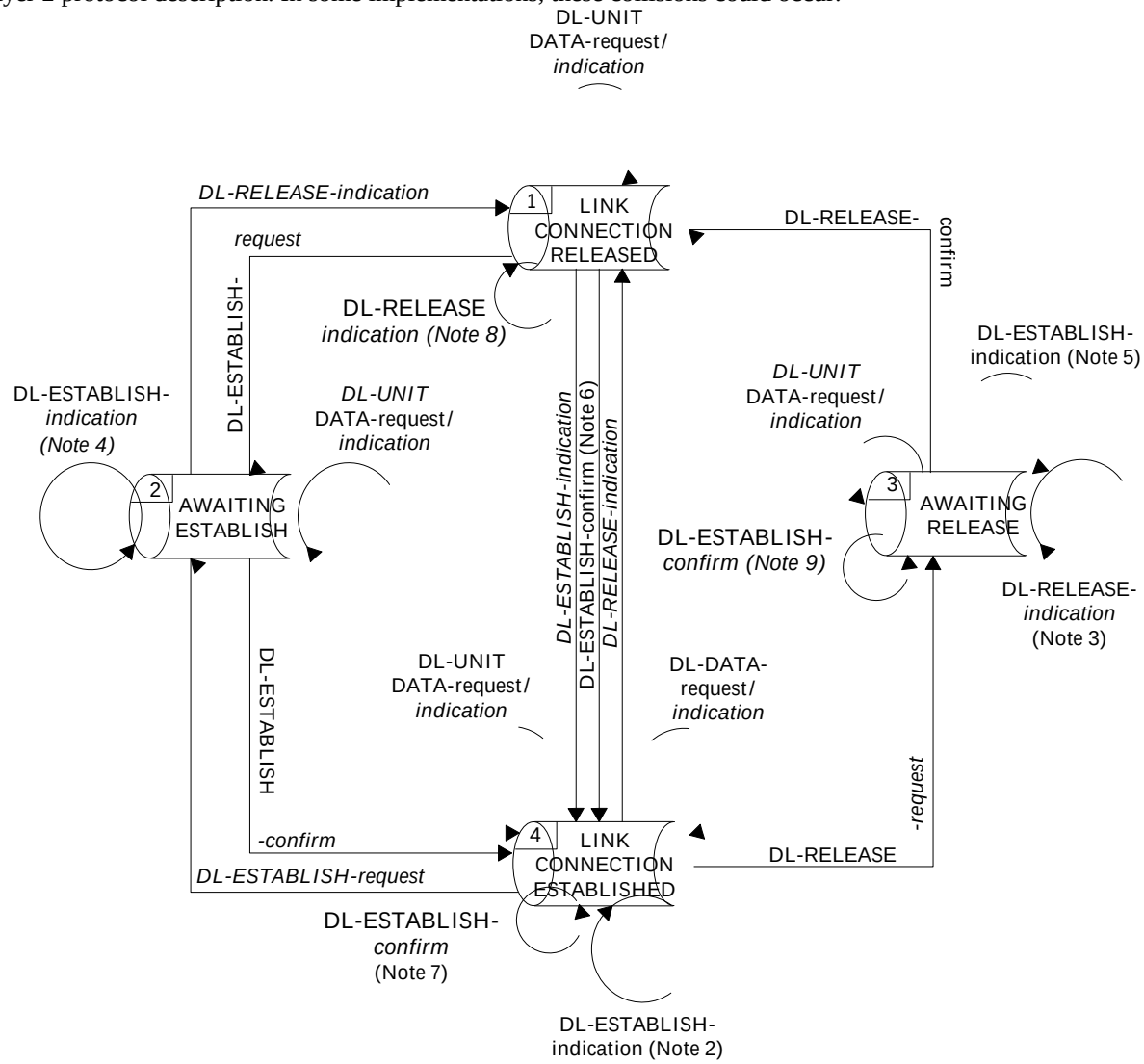


Figure 6/MTS21-C
State transition diagram for sequences of primitives

Note 1 - If the data link layer entity issues a DL-ESTABLISH *indication* (this applies to the case of a peer system initiated re-establishment) or DL-RELEASE *indication*, this indicates the discard of all the data link service data units representing DL-DATA *requests*.

Note 2 - This primitive notifies layer 3 of link re-establishment.

Note 3 - This primitive will occur if a DL-RELEASE *request* collides with a DL-RELEASE *indication*.

Note 4 - This primitive will occur if a DL-ESTABLISH *request* collides with a DL-ESTABLISH *indication*.

Note 5 - This primitive will occur if a DL-RELEASE *request* collides with a DL-ESTABLISH *indication*.

Note 6 - This primitive will occur if a DL-ESTABLISH *request* (this applies to the case of layer 3 initiated re-establishment) collides with a DL-RELEASE *indication*. Since this DL-RELEASE *indication* is not related to the DL-ESTABLISH *request*, the data link layer will establish the link and issue a DL-ESTABLISH *confirm*.

Note 7 - This primitive will occur as a result of multiple collisions of primitives. If a first DL-ESTABLISH *request* collides with a DL-RELEASE *indication*, the data link layer will establish the link and issue a DL-ESTABLISH *confirm* (see Note 6). This DL-ESTABLISH *confirm* (related to the first DL-ESTABLISH *request*) would collide with a subsequent DL-ESTABLISH *request*, which may be issued since layer 3 is not aware that the DL-RELEASE *indication* was not related to the first DL-ESTABLISH *request*. Since layer 3 relates this DL-ESTABLISH *confirm* to the subsequent DL-ESTABLISH *request*, it assumes the data link layer is in the link connection established state, but the data link layer will re-establish the link and issue again a DL-ESTABLISH *confirm*.

Note 8 - This primitive will occur if a DL-ESTABLISH-*request* (this applies to the case of Layer 3 initiated re-establishment) collides with a DL-RELEASE-*indication*. Since this DL-RELEASE-*indication* is not related to the DL-ESTABLISH-*request*, the data link layer will try to establish the link and if this is not possible, it issues a DL-RELEASE-*indication*.

Note 9 - This primitive will occur as a result of multiple collisions of primitives. If a first DL-ESTABLISH-*request* collides with a DL-RELEASE-*indication*, the data link layer will establish the link and issue a DL-ESTABLISH-*confirm* (see Note 6). This DL-ESTABLISH-*confirm* may collide with a subsequent DL-ESTABLISH-*request* and the data link layer will re-establish the link and issue again a DL-ESTABLISH-*confirm* (see Note 7). This second DL-ESTABLISH-*confirm* (it is related to the second DL-ESTABLISH-*request*) may collide with a subsequent DL-ESTABLISH-*request* which may be issued since Layer 3 is not aware that the DL-RELEASE-*indication* was not related to the first DL-ESTABLISH-*request*. Since Layer 3 relates this first DL-ESTABLISH-*confirm* to the subsequent DL-ESTABLISH-*request* it assumes the data link layer is in the link connection established state, but the data link layer will re-establish the link and issue again a DL-ESTABLISH-*confirm* (see Note 7).

4.3 Block interaction diagram of the data link layer

MTS21-C § 4.1 defines the primitives associated with MiLAP and MiLAP-S, and § 4.2 defines the primitive procedures between layer 3 and the data link layer.
This section clarifies how the primitives defined in this MTS apply to the various functional blocks.
A block interaction diagram relates the service primitives to those functional blocks which have to interact (see Figure 7/MTS21-C). Additional signals are needed for internal use within the data link layer for the communication between point-to-point link procedures or broadcast link procedures, respectively, and the multiplex procedure.

The figure is an aid to illustrate the relationship between the various functional blocks. It is not intended to constrain implementation. The primitives contained in Figure 7/MTS21-C are those defined in § 4.1.

Note - Not all functions associated with defined primitives are currently used. For example, address negotiation (DL-ASSIGN) and unacknowledged information transfer (DL-UNIT-DATA) are not yet implemented.

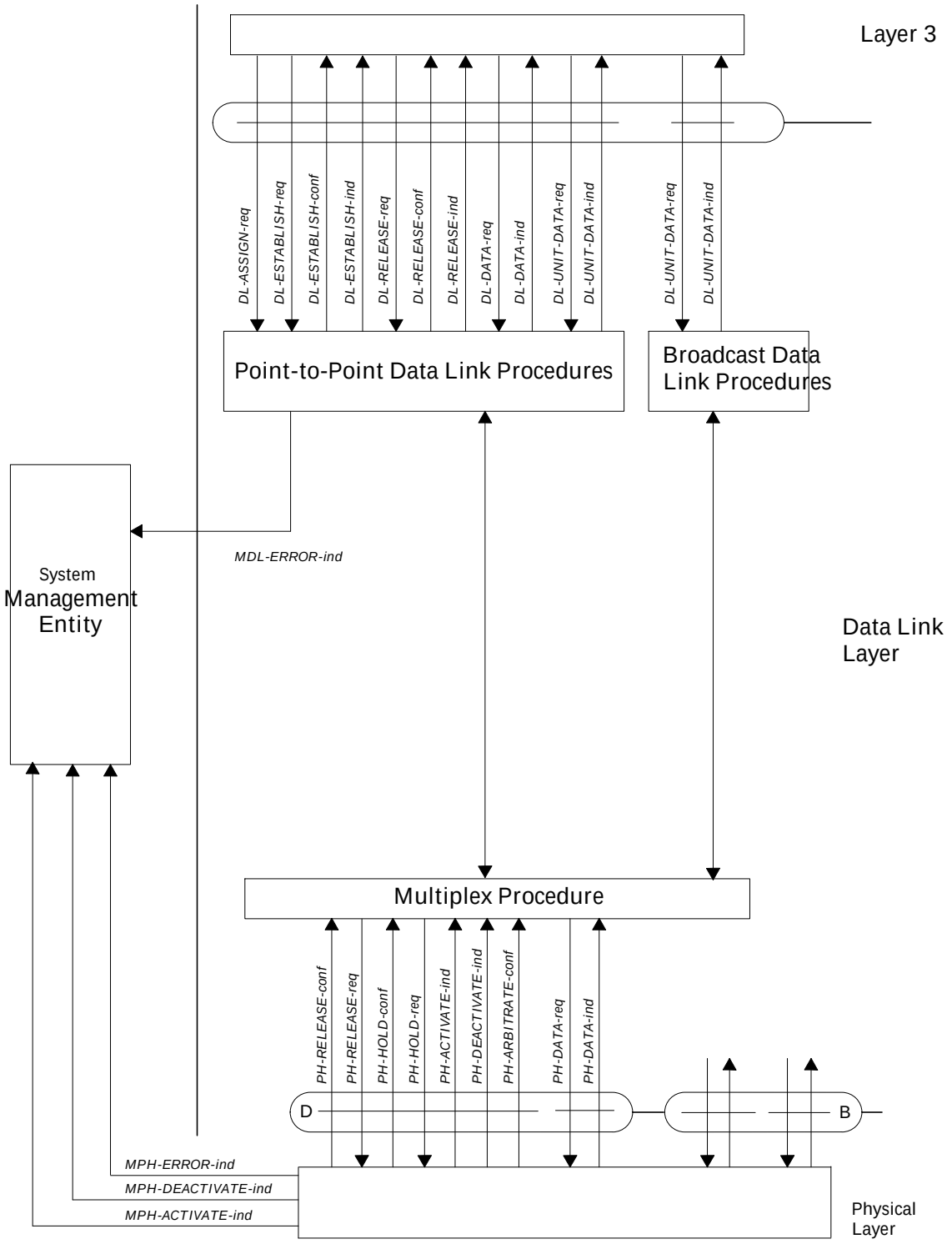


Figure 7/MTS21-C
Block interaction diagram

5.0 Definition of the peer-to-peer procedures of the data link layer

The procedures for use by the data link layer are specified in the following sections.
The elements of procedure (frame types) which apply are:

a) for unacknowledged information transfer (§ 5.2):
UI-command;

Note - MiLAP-S does not support UI-command.

b) for multiple frame acknowledged information transfer (§§ 5.4 to 5.7):
SABM-command,
UA-response,
DISC-command,

Note - MiLAP-S does not support DISC-command.
RR-command/response,

Note - MiLAP-S supports reception, but not transmission, of RR-poll.
RNR-command/response,

Note - MiLAP-S supports reception, but not transmission, of RNR and RNR-poll.
REJ-command/response,
I-
Command.

In variance with Q.921 [10], neither MiLAP nor MiLAP-S support DM-response or FRMR-response.

5.1 Procedure for the use of the P/F bit

5.1.1 Unacknowledged information transfer

For unacknowledged information transfer, the P/F bit is not used and shall be set to 0.

5.1.2 Acknowledged multiple frame information transfer

A data link layer entity receiving an SABM, DISC, RR or RNR frame with the P bit set to 1 shall set the F bit to 1 in the next response frame it transmits. For an SABM-command or a DISC-command (MiLAP only) with the P bit set to 1, the response shall be a UA-response with the F bit set to 1. In the normal state, for an RR-command or an RNR-command with the P bit set to 1, the response shall be an RR-response with the F bit set to
1.

In variance with Q.921 [10], neither MiLAP nor MiLAP-S support the poll bit in I frames or REJ frames. In these frames, bit position 5 of each frame shall be set to 0.

5.2 Procedures for unacknowledged information transfer

5.2.1 General

The procedures which apply to the transmission of information in unacknowledged operation are defined below.
No data link layer error recovery procedures are defined for unacknowledged operation.

Note - Unacknowledged transfer is supported only in MiLAP, but is not currently implemented.

5.2.2 Transmission of unacknowledged information

Note - The term transmission of a UI frame refers to the delivery of a UI frame by the data link layer to the physical layer.
SDUs to be conveyed by means of unacknowledged information transfer are passed to the data link layer by layer 3 entities using the primitive DL-UNIT DATA *request*. The SDUs passed by layer 3 shall be transmitted in UI command frames.
For broadcast operation, the address value in the UI command address field (single destination) shall be set to 63 (binary 11 1111), the group value.
For point-to-point operation, the appropriate address value shall be used.

Note - Unacknowledged point-to-point operation in MiLAP, and the appropriate TEI value, are for further study.
The P bit shall be set to 0.
In the case of persistent layer 1 deactivation (e.g., MiLINK discontinuity), the data link layer will be informed by an appropriate indication. Upon receipt of this indication, all UI transmission queues shall be discarded.

5.2.3 Receipt of unacknowledged information

On receipt of a UI command frame with an address supported by the receiver, the contents of the information field shall be passed to the layer 3 entity using the data link layer to layer 3 primitive DL-UNIT DATA *indication*. Otherwise, the UI command frame shall be discarded.

5.3 Address management procedures

Note - The following procedures have been defined for MiLAP and MiLAP-S but are not currently implemented. Procedures for the two-octet address format are for further study.

5.3.1 General

This section defines the address management procedures for address values to be used for point-to-point data link connections (address value is in the range from 2 through 31).

Note - Address values 0 and 1 are fixed and used in current product implementations. Address values 30 and 31 have special significance (see Table 1/MTS21-C) and are described in § 5.3.2 and § 5.3.4, respectively.

For MiLAP and MiLAP-S, address management procedures are limited to assignment procedures only. Address check and removal procedures are not supported.

In variance with Q.921 [10], MiLAP and MiLAP-S address management procedures involve interactions at the physical layer (i.e., MiLINK), data link layer and layer 3 (i.e., MiNET).

A MiLINK peripheral in the address-unassigned state shall use the address assignment procedures to enter the address-assigned state. The purpose of these procedures is to allow a peripheral to request the system to assign an address value that the data link layer in the requesting peripheral will use in subsequent communications.
A MiLINK peripheral's data link layer entity shall use the negotiation address (30) on power up.
The data link layer entity shall remove the address value (i.e., return to the negotiation address) when it is notified that a persistent physical layer discontinuity has occurred. The physical layer shall use the PH-DEACTIVATE-*indication* primitive for this purpose.
The initiation of address assignment procedures occurs whenever the peripheral data link layer entity determines that its current address is the negotiation address and physical layer continuity has been established. The physical layer shall use the PH-ACTIVATE-*indication* primitive to indicate continuity.

Note - These procedures allow for only a single negotiated address per data link layer, although fixed addresses could also be

used.

In variance with Q.921 [10], MiLAP and MiLAP-S address negotiation begins immediately. It is not postponed until a layer 2 service is requested by layer 3.

5.3.2 Address assignment procedure

Note - The following procedures do not conform to negotiation procedures found in Q.921 [10].

Upon initiation of the address assignment procedure (i.e., after power-up or physical link discontinuity, e.g., loss of synchronization), the peripheral data link layer entity shall notify the physical layer of the need to arbitrate for and then hold the D-channel (see MTS20 [3]). The data link layer entity shall use the PH-HOLD-*request* primitive for this purpose.

Upon successful D-channel arbitration, the physical layer shall notify the data link layer entity. The physical layer shall use the PH-HOLD-*confirm* primitive for this purpose. The data link layer entity shall immediately start a timer (T210 - see § 5.8.2) and establish a data link connection with its peer using the negotiation address (30).

Upon successful link establishment, the data link layer entity shall inform layer 3 of the need to negotiate a new address. The data link layer entity shall use the DL-REMOVE-*indication* primitive for this purpose. Negotiation for actual address assignment is then conducted between the peripheral's layer 3 entity and its peer (see MTS22 [4]).

Upon receipt of the new address assignment, layer 3 shall notify the data link layer entity of the new value. Layer 3 shall use the DL-ASSIGN-*request* primitive for this purpose.

The data link layer entity shall then request the physical layer to terminate the D-channel hold state. The data link layer shall use the PH-RELEASE-*request* primitive for this purpose. When the physical layer has terminated the D-channel hold state it shall notify the data link layer entity. The physical layer shall use the PH-RELEASE-*confirm* primitive for this purpose. Upon receipt of this primitive, the data link layer shall clear timer T210 and inform layer 3. The data link layer shall use the DL-RELEASE-*indication* primitive for this purpose.

Following this state, the data link layer shall once again establish a data link connection with its peer, this time using the new address value. Normal procedures shall be employed (i.e., the D-channel hold mode shall not be employed).

Figure 8/MTS21-C shows an example sequence of primitives and actions during address negotiation.

5.3.3 Expiry of timer T210

If timer T210 expires before the data link layer receives a new address from layer 3 (via the DL-ASSIGN-*request* primitive), then the D-channel shall be released (via the PH-RELEASE-*request* primitive). Upon acknowledgement of release by the physical layer (via the PH-RELEASE-*confirm* primitive), the data link layer shall restart the address negotiation procedures.

The value of T210 is specified in § 5.8.2.

5.3.4 Link access denial

During address negotiation (or at any time), the system may, via the contents of a layer 3 message (see MTS22 [4]), indicate that the peripheral shall use a new address of value of 31. This is known as the “dead” address, and denies the peripheral further access to the link. Upon notification of this address from layer 3 (via the DL-ASSIGN-*request* primitive), the data link layer shall move its link address to the new value and shall attempt no further transmissions over the D-channel. Any I or UI queues shall be discarded.

Note - During address negotiation, this state will follow the normal RR response with address 30 (see Figure 8/MTS21-C).

Receipt by the data link layer entity of any subsequent frame with a link address value 31 shall cause the peripheral to restart all procedures in a manner equivalent to power-up. Frames with any other address value shall be ignored.

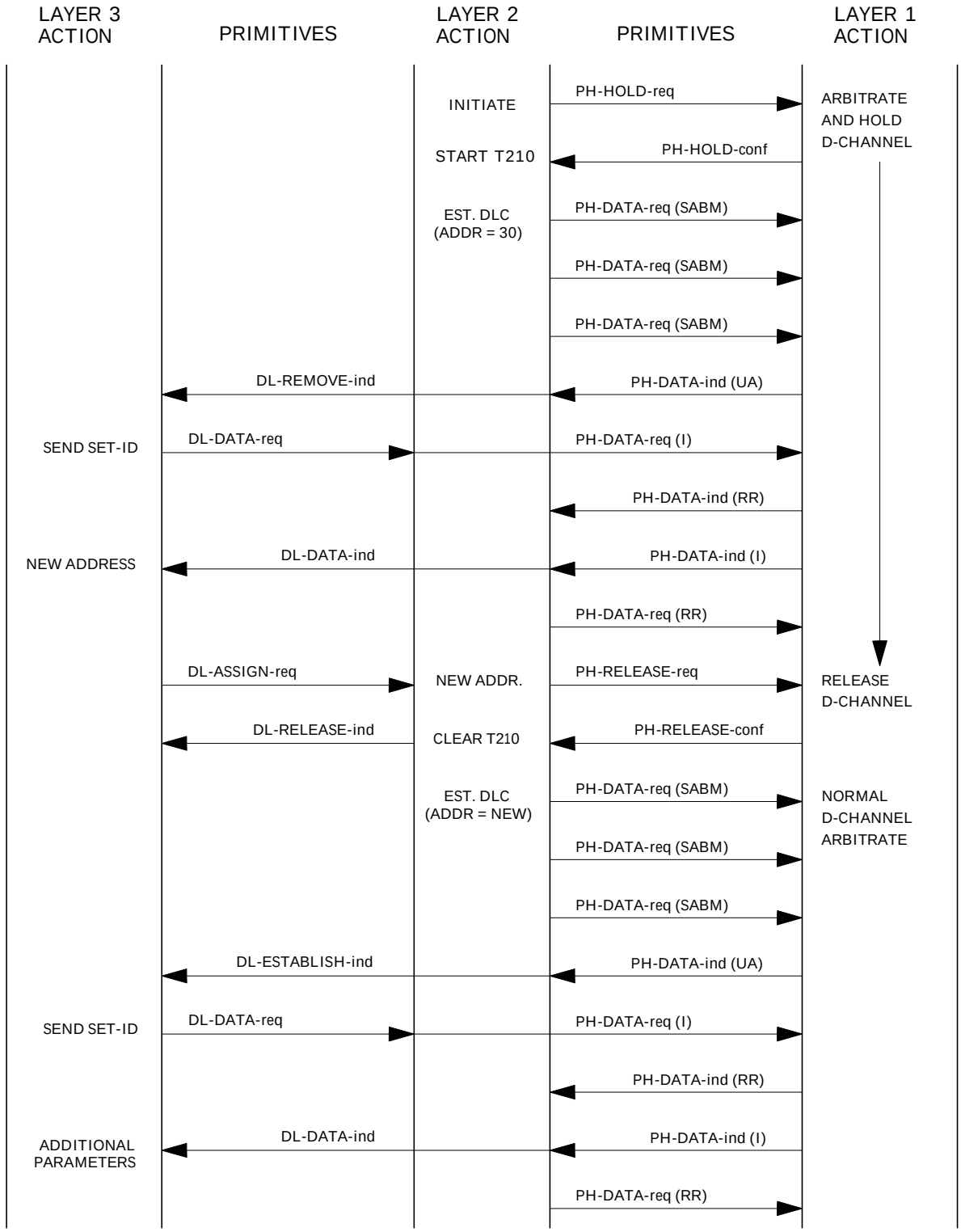


Figure 8/MTS21-C
Primitives and actions sequences example for address negotiation

Note 1 - SET_ID is the first layer 3 message sent during address negotiation, however, this message does not actually form part of the address negotiation procedures.

Note 2 - The D-channel hold procedure is only required for the first address negotiation sequence. The subsequent multiple-frame establish on the new address shall not use the D-channel hold procedure.

5.4 Procedures for establishment and release of multiple frame operation

Note - MiLAP-S assumes the data link to be active at all times. SABM/UA exchanges are used only to reset the data link under exception conditions. Procedures in this section are not applicable to MiLAP-S.

5.4.1 Establishment of multiple frame operation

5.4.1.1 General

These procedures shall be used to establish multiple frame operation between the system and a MiLAP-supporting peripheral.

Layer 3 will request establishment of the multiple frame operation by use of the DL-ESTABLISH-*request* primitive. Re-establishment may be initiated as a result of the data link layer procedures defined in § 5.6. All frames other than unnumbered frame formats received during the establishment procedures shall be ignored.

5.4.1.2 Establishment procedures

A data link layer entity shall initiate a request for the multiple frame operation to be set by transmitting the SABM command. All existing exception conditions shall be cleared, the retransmission counter shall be reset, and timer T200 shall be started. Timer T200 is defined in § 5.8.1. All mode setting commands shall be transmitted with the P bit set to 1.

Layer 3 initiated establishment procedures imply the discard of all outstanding DL-DATA-*request* primitives and all I frames in queue.

A data link layer entity receiving an SABM command shall:

- respond with a UA response with the F bit set to the same binary value as the P bit in the received SABM command;
- set V(S), V(R) and V(A) to 0;
- enter the multiple-frame-established state and inform layer 3 using the DL-ESTABLISH-*indication* primitive;

Note - Use of the DL-ESTABLISH-*indication* primitive in the initial address negotiation multiple-frame-establish state (address = 30) is not mandatory, as establishment of multiple frame operation is implied in this case by the DL-REMOVE-*indication* primitive.

- clear all existing exception conditions; and
- clear any existing peer receiver busy condition.

Q.921 [10] indicates an optional timer, T203, which determines the maximum time allowed without frames being transmitted. This timer is not used in MiLAP and MiLAP-S.

Upon reception of the UA response with the F bit set to 1, the originator of the SABM command shall:

- reset timer T200;
- set V(s), V(R) and V(A) to zero; and
- enter the multiple-frame-established state and inform layer 3 using the DL-ESTABLISH-*confirm* primitive.

A DL-RELEASE-*request* primitive received during data link layer initiated re-establishment shall be serviced on completion of the establishment mode-setting operation.

5.4.1.3 Procedure on expiry of timer T200

If timer T200 expires before the UA response with the F bit set to 1 is received, the data link layer entity shall:

- retransmit the SABM command as above;
- restart timer T200; and
- increment the retransmission counter N200.

After retransmission of the SABM command N200 times, the data link layer entity shall indicate this to layer 3 and the management entity by means of the DL-RELEASE-*indication* and MDL-ERROR-*indication* primitives, respectively. The data link layer entity shall then enter the disconnect send state, after discarding all outstanding DL-DATA-*request* primitives and all I frames in queue.
The value of N200 is defined in § 5.8.4.

5.4.2 Information transfer
Having either transmitted the UA response to a received SABM command or received the UA response to a transmitted SABM command, I frames and supervisory frames shall be transmitted and received according to the procedures described in § 5.5.
If a SABM command is received while in the multiple-frame-established (normal) state, the data link layer entity shall respond with a UA frame and remain in the multiple-frame-established state. The I queue shall not be
discarded.

This procedure is in variance with Q.921 [10], which requires the data link layer entity to follow link re-establishment procedures.

On receipt of a UI command, the procedures defined in § 5.2 shall be followed.

Note - The procedures defined in § 5.2 for unacknowledged information transfer are defined, but not currently implemented in MiLAP and MiLAP-S.

5.4.3 Termination of multiple frame operation

5.4.3.1 General
These procedures shall be used to terminate the multiple frame operation between the system and a MiLAP-supporting peripheral.
Layer 3 will request termination of the multiple frame operation by use of the DL-RELEASE-*request* primitive.
All frames other than unnumbered frames received during the release procedures shall be ignored. All outstanding DL-DATA-*request* primitives and all I frames in queue shall be discarded.
In the case of a layer 1 deactivation (indicated to the data link layer entity via the PH-DEACTIVATE-*indication* primitive), the data link layer entity shall discard all I frame queues and deliver to layer 3 a DL-RELEASE-*confirm* primitive if a DL-RELEASE-*request* primitive is outstanding, or otherwise a DL-RELEASE-*indication* primitive.

5.4.3.2 Release procedure
A data link layer entity shall initiate a request for release of the multiple frame operation by transmitting the disconnect (DISC) command with the P bit set to 1. Timer T200 shall then be started and the retransmission counter N200 reset.
A data link layer entity receiving a DISC command while in the multiple-frame-established (normal) or timer recovery (SABM reset) state shall transmit a UA response with the F bit set to the same binary value as the P bit in the received DISC command. A DL-RELEASE-*indication* primitive shall be passed to layer 3, and the disconnected state shall be entered.
If the originator of a DISC command receives a UA response with the F bit set to 1, it shall enter the disconnected state and reset timer T200. The data link layer entity shall now notify layer 3 by means of the DL-RELEASE-*confirm* primitive. The conditions relating to this state are defined in § 5.4.4.

5.4.4 Address-assigned state
While in the address-assigned state:
- the receipt of a DISC command shall result in the transmission of a UA response;
- on receipt of an SABM command, the procedures defined in § 5.4.1 shall be followed;
- on receipt of UI commands, the procedures defined in § 5.2 shall be followed; and
- all other frame types shall be discarded.

5.4.5 Collision of unnumbered commands and responses

5.4.5.1 Identical transmitted and received commands

If the transmitted and received unnumbered commands (SABM or DISC) are the same, the data link layer entity shall send a UA response at the earliest possible opportunity. The indicated state shall be entered only after receiving the UA response. The data link layer entity shall notify layer 3 by means of the appropriate confirm primitive.

5.4.5.2 Different transmitted and received commands

If the received unnumbered command does not have the P bit set to 1 and does not cause a transition to a primary state, it shall be ignored.

If the received command has the P bit set to 1 and does not cause a transition to a primary state, the earlier transmitted command shall be repeated with the P bit set to 1.

If the received command requires a transition to a primary state, the data link layer entity shall send a UA response and transition to the state normal to reception of the command. Any timer recovery conditions shall be cancelled after the transition. Any frames received prior to this event and not yet processed shall be discarded.

This procedure is in variance with Q.921 [10], which requires the data link layer entity to issue a DM response. This response is not supported in MiLAP or MiLAP-S.

5.5 Procedures for information transfer in multiple frame operation

The procedures which apply to the transmission of I frames are defined below.

Note - The term “transmission of an I frame” refers to the delivery of an I frame by the data link layer to the physical layer.

5.5.1 Transmitting I frames

Information received by the data link layer entity from layer 3 by means of a DL-DATA-*request* primitive shall be transmitted in an I frame. The control field parameters N(S) and N(R) shall be assigned the values of V(S) and V(R), respectively. V(S) shall be incremented by 1 at the end of the transmission of the I frame.

If timer T210 is not running at the time of transmission of an I frame, it shall be started. The timer shall be reset on receipt of the PH-ARBITRATE-*confirm* primitive from layer 1.

Note - This primitive signals the successful acquisition of the D-channel by layer 1 (see, for example, MTS20 [3]). In particular, when the Mitel “go-ahead” procedure is required, T200 must not be started until the go-ahead indication is received. T200 is not started on transmission of a REJ command.

If timer T200 is not running, it shall be started on receipt of the PH-ARBITRATE-*confirm* primitive from layer 1.

If timer T200 expires, the procedures defined in § 5.5.8 shall be followed.

If V(S) is equal to V(A) plus k (where k is the maximum number of outstanding I frames - see § 5.8.6), the data link layer entity shall not transmit any new I frames, but may retransmit an I frame as a result of the error recovery procedures as described in §§ 5.5.5 and 5.5.8.

When the network side or user side is in the own receiver busy condition, it may still transmit I frames, provided that a peer receiver busy condition does not exist.

Note - Any DL-DATA-*request* primitives received while in the timer recovery condition shall be queued.

5.5.2 Procedure on expiry of timer T210

If timer T210 expires prior to the receipt of the PH-ARBITRATE-*confirm* primitive from layer 1, the data link layer entity shall:

- increment counter N200;
- re-transmit the frame; and
- restart timer T210.

5.5.3 Receiving I frames

Independent of a timer recovery condition, when a data link layer entity is not in an own receiver busy condition and receives a valid I frame whose N(S) is equal to the current V(R), the data link layer entity shall:

- pass the information field of this frame to layer 3 using the DL-DATA-*indication* primitive;
- increment by 1 its V(R); and
- act as indicated below.

If, on receipt of this I frame, the data link layer entity is still not in an own receiver busy condition:

- if no I frame is available for transmission, or if an I frame is available for transmission but a peer receiver busy condition exists, the data link layer entity shall transmit an RR response with the F bit set to 0; or

I-frames may be acknowledged by the N(R) sequence number in RR, RNR, REJ or another Iframe. To improve efficiency, a device may elect to wait for a period of time of ACKTMR before transmitting the acknowledgement. This may perhaps eliminate the need to send two acknowledgements. The use of this technique is straightly implementation dependent and does

- if an I frame is available for transmission and no peer receiver busy condition exists, the data link layer entity shall transmit the I frame with the value of N(R) set to the current value of V(R) as defined in §

5.5.1.

If, on receipt of this I frame, the data link layer entity is now in an own receiver busy condition, it shall transmit an RNR response with the F bit set to 0.

When the data link layer entity is in an own receiver busy condition, it shall process any received I frame in accordance with § 5.5.7.

Note - MiLAP-S entities are always considered ready to receive. Therefore, procedures above which relate to own receiver busy conditions do not apply to MiLAP-S entities.

The above procedures are a subset of those defined in Q.921 [10], since MiLAP and MiLAP-S do not support I frames with the P bit set to 1.

5.5.4 Sending and receiving acknowledgements

5.5.4.1 Sending acknowledgements

Whenever a data link layer entity transmits an I frame or a supervisory frame, N(R) shall be set equal to V(R).

5.5.4.2 Receiving acknowledgements

On receipt of a valid I frame or supervisory frame (RR, RNR, or REJ), even in the own receiver busy or timer recovery conditions, the data link layer entity shall treat the N(R) contained in this frame as an acknowledgement for all the I frames it has transmitted with an N(S) up to and including the received N(R) - 1. V(A) shall be set equal to N(R). When not in the timer recovery condition, the data link layer entity shall reset the timer T200 on receipt of a valid I frame or supervisory frame with the N(R) higher than V(A) (actually acknowledging some I frames), or an REJ frame with an N(R) equal to V(A).

Note1 - A MiLAP-S data link layer entity is assumed to never be in an own receiver busy condition.

Note2 - If a supervisory frame with the P bit set to 1 has been transmitted and not acknowledged by a supervisory frame response with the F bit set to 1, timer T200 shall not be reset.

Note3 - Upon receipt of a valid I frame, timer T200 shall not be reset if the data link layer entity is in the peer receiver busy condition.

If timer T200 has been reset by the receipt of an I, RR, or RNR frame, and if there are outstanding I frames still unacknowledged, the data link layer entity shall restart timer T200. If timer T200 then expires, the data link layer entity shall follow the recovery procedure as defined in § 5.5.8 with respect to the unacknowledged I frames.
If timer T200 has been reset by the receipt of an REJ frame, the data link layer entity shall follow the retransmission procedures defined in § 5.5.5.

5.5.5 Receiving REJ frames
On receipt of a valid REJ frame, the data link layer entity shall act as follows:
a) if it is not in the timer recovery condition:
- clear an existing peer receiver busy condition;
- set its V(S) and its V(A) to the value of the N(R) contained in the REJ frame control field;
- stop timer T200; and
- transmit the corresponding I frame as soon as possible, as defined in § 5.5.1, taking into account items 1) to 3) below and the paragraph following item 3).

b) if it is in the timer recovery condition:
- clear an existing peer receiver busy condition; and
- set its V(A) to the value of the N(R) contained in the REJ frame control field.

The above procedures are a subset of those defined in Q.921 [10], since MiLAP and MiLAP-S do not support REJ frames with the poll/final bits.

Transmission of I frames shall take into account the following:
1) if the data link layer entity is transmitting a supervisory frame when it receives the REJ frame, it shall complete that transmission before commencing transmission of the requested I frame;
2) if the data link layer entity is transmitting an SABM command, a DISC command, or a UA response when it receives the REJ frame, it shall ignore the request for retransmission; and
3) if the data link layer entity is not transmitting a frame when the REJ is received, it shall immediately commence transmission of the requested I frame.

MiLAP-S does not support transmission of the DISC command.

Note - MiLAP-S does not support transmission of the DISC command.
All outstanding unacknowledged I frames, commencing with the I frame identified in the received REJ frame, shall be transmitted. Other I frames not yet transmitted may be transmitted following the retransmitted I frames.

5.5.6 Receiving RNR frames
After receiving a valid RNR command or response, if the data link layer entity is not engaged in a mode-setting operation, it shall set a peer receiver busy condition and then:
- if it was an RNR command with the P bit set to 1, it shall respond with an RR response with the F bit set to 1if the data link layer entity is not in an own receiver busy condition, and shall respond with an RNR response with the F bit set to 1 if the data link layer entity is in an own receiver busy condition; and
- if it was an RNR response with the F bit set to 1, an existing timer recovery condition shall be cleared and the N(R) contained in this RNR response shall be used to update V(S).

Note- A MiLAP-S data link layer entity is assumed to never be in an own receiver busy condition.
The data link layer entity shall take note of the peer receiver busy condition and not transmit any I frames to the peer which has indicated the busy condition.

Note- The N(R) in any RR or RNR command frame (irrespective of the setting of the P bit) shall not be used to update the V(S).
The data link layer entity shall then:

- treat the N(R) contained in the received RNR frame as an acknowledgement for all the I frames that have been (re)transmitted with an N(S) up to and including N(R) - 1, and set its V(A) to the value of the N(R) contained in the RNR frame; and
- restart timer T200, unless a supervisory response frame with the F bit set to 1 is still expected.

If timer T200 expires, the data link layer entity shall:

- if it is not yet in a timer recovery condition, enter the timer recovery condition and reset the retransmission count variable; or
- if it is already in a timer recovery condition, add 1 to its retransmission count variable.

The data link layer entity shall then:

a) if the value of the retransmission count variable is less than N200:
   - transmit an appropriate supervisory command (see note 4) with the P bit set to 1; and
   - restart timer T200.

b) if the value of the retransmission count variable is equal to N200:
   - initiate a re-establishment procedure as defined in § 5.6; and
   - indicate this to the management entity by means of the MDL-ERROR-*indication* primitive.

The data link layer entity receiving the supervisory frame with the P bit set to 1 shall respond, at the earliest opportunity, with a supervisory response frame (see note 4) with the F bit set to 1, to indicate whether or not its own receiver busy condition still exists.

Note 3 - A MiLAP-S data link layer entity is assumed to never be in an own receiver busy condition.

Upon receipt of the supervisory response with the F bit set to 1, the data link layer entity shall reset timer T200, and:

- if the response is an RR or REJ response, the peer receiver busy condition is cleared and the data link layer entity may transmit new I frames or retransmit I frames as defined in §§ 5.5.1 or 5.5.5, respectively; or
- if the response is an RNR response, the data link layer entity receiving the response shall proceed according to § 5.5.6, first paragraph.

If a supervisory command (RR, RNR, or REJ) with the P bit set to 0 or 1, or a supervisory response frame (RR, RNR, or REJ) with the F bit set to 0 is received during the enquiry process, the data link layer entity shall:

- if the supervisory frame is an RR or REJ command frame or an RR or REJ response frame with the F bit set to 0, clear the peer receiver busy condition and if the supervisory frame received was a command with the P bit set to 1, transmit the appropriate supervisory response frame (see note 4) with the F bit set to 1. However, the transmission or retransmission of I frames shall not be undertaken until the appropriate supervisory response frame with the F bit set to 1 is received or until expiry of timer T200; or
- if the supervisory frame is an RNR command frame or an RNR response frame with the F bit set to 0, retain the peer receiver busy condition and if the supervisory frame received was an RNR command with the P bit set to 1, transmit the appropriate supervisory response frame (see note 4) with the F bit set to 1.

Upon receipt of an SABM command, the data link layer entity shall clear the peer receiver busy condition.

Note 4 - The appropriate supervisory frame for the circumstances indicated is defined below:

- If the data link layer entity is not in an own receiver busy condition and is in a reject exception condition (that is, an N(S) sequence error has been received, and an REJ frame has been transmitted, but the requested I frame has not been received), the appropriate supervisory frame is the RR frame.
- If the data link layer entity is not in an own receiver busy condition but is in an N(S) sequence error exception condition (that is, an N(S) sequence error has been received but an REJ frame has not been transmitted), the appropriate supervisory frame is the REJ frame.
- If the data link layer entity is in an own receiver busy condition, the appropriate supervisory frame is the RNR frame.
- Otherwise, the appropriate supervisory frame is the RR frame.

5.5.7 Data link layer own receiver busy condition

Note - The procedures in this section apply only to MiLAP, as MiLAP-S is assumed to never be in an own receiver busy condition.

When the data link layer entity enters an own receiver busy condition, it shall transmit an RNR frame at the earliest opportunity.

The RNR frame may be:
- an RNR response with the F bit set to 0; or
- if this condition is entered on receiving a command frame with the P bit set to 1, an RNR response with the F bit set to 1; or
- if this condition is entered on expiry of timer T200, an RNR command with the P bit set to 1.

All received I frames shall be discarded, after updating V(A).

The above procedure is a subset of that defined in Q.921 [10], since MiLAP and MiLAP-S do not support I frames with the P bit set to 1. Normally, an RNR response frame would also be sent if the I frame had the P bit set to 1.

All received supervisory frames with the P/F bit set to 0 shall be processed, including updating V(A).
All received supervisory frames with the P bit set to 1 shall be processed including updating V(A). An RNR response with the F bit set to 1 shall be transmitted.
To indicate to the peer data link layer entity the clearance of the own receiver busy condition, the data link layer entity shall transmit an RR frame or, if a previously detected N(S) sequence error has not yet been reported, an REJ frame with the N(R) set to the current value of V(R).

Note - A MiLAP-S entity will not transmit the RR frame case above, as it is assumed to be never in the own receiver busy case.
The transmission of an SABM command or a UA response (in reply to an SABM command) also indicates to the peer data link layer entity the clearance of the own receiver busy condition.

5.5.8 Waiting acknowledgement
The data link layer entity shall maintain an internal retransmission count variable.
If timer T200 expires, the data link layer entity shall:
- if it is not yet in the timer recovery condition, enter the timer recovery condition and reset the transmission count variable; or
- if it is already in the timer recovery condition, add 1 to its retransmission count variable.

The data link layer entity shall then:
a) if the value of the retransmission count variable is less than N200:
   - restart timer T200; and
   - retransmit the last transmitted I frame [V(S) - 1] or send an RR-poll

This procedure is in variance with Q.921 [10], since the retransmitted I frame is normally sent with the P bit set to 1. The equivalent bit position in MiLAP and MiLAP-S is always set to 0.

b) if the value of the retransmission count variable is equal to N200:
   - initiate a re-establishment procedure as defined in § 5.6; and
   - indicate this to the system management entity by means of the MDL-ERROR-*indication* primitive.

Note - When a data link layer entity is first establishing a data link connection over a B-channel, counter N200A shall be used (see § 5.8.4). Once the B-channel connection is established, counter N200 shall be used.
The following paragraph applies only for a data link layer entity which is in the timer recovery condition, since the case of receiving acknowledgement in the multiframe established state is described in § 5.5.4.2.
The timer recovery condition is cleared only if the data link layer entity receives a valid supervisory frame response with the F bit set to 1. If the N(R) of this received supervisory frame is within the range from its current V(A) to its current V(S) inclusive, it shall set its V(S) to the value of the received N(R). Timer T200 shall be reset if the received supervisory frame response is an RR or REJ response with the F bit set to 1. The data link layer entity shall resume then with I frame transmission or retransmission, as appropriate.
Timer T200 shall be reset and restarted if the received supervisory response is an RNR response with the F bit set to 1, to proceed with the enquiry process according to § 5.5.6.

5.6 Re-establishment of multiple frame operation

5.6.1 Criteria for re-establishment

The criteria for re-establishing the multiple-frame mode of operation are defined in this section by the following conditions:

- the receipt, while in the multiple-frame mode of operation, of an SABM;
- the receipt of a DL-ESTABLISH-*request* primitive from layer 3 (see § 5.4.1.1);
- the occurrence of N200 retransmission failures while in the timer recovery condition (see § 5.5.8);
- the occurrence of a frame rejection condition as identified in § 5.7.5;

This procedure is in variance with Q.921 [10], which includes FRMR and DM criteria. These responses are not supported by MiLAP and MiLAP-S.

5.6.2 Procedures

In all re-establishment situations, the data link layer entity shall follow the procedures defined in § 5.4.1. All locally generated conditions for re-establishment shall cause the transmission of the SABM.

In the case of data link layer and peer initiated re-establishment, the data link layer entity shall also:

- issue an MDL-ERROR-*indication* primitive to the management entity; and
- if V(S)>V(A) prior to re-establishment, issue a DL-ESTABLISH-*indication* primitive to layer 3, and discard all I queues.

In case of layer 3 initiated re-establishment, or if a DL-ESTABLISH-*request* primitive occurs pending re-establishment, the DL-ESTABLISH-*confirm* primitive shall be used.

5.7 Exception condition reporting and recovery

Exception conditions may occur as the result of physical layer errors or data link layer procedural errors. The error recovery procedures which are available to effect recovery following the detection of an exception condition at the data link layer are defined in this section.

5.7.1 N(S) sequence error

An N(S) sequence error exception condition occurs in the receiver when a valid I frame is received which contains an N(S) value which is not equal to the V(R) at the receiver. The information field of all I frames whose N(S) does not equal V(R) shall be discarded.

The receiver shall not acknowledge (nor increment its V(R)) the I frame causing the sequence error, nor any I frames which may follow, until an I frame with the correct N(S) is received.

A data link layer entity which receives one or more I frames having sequence errors but otherwise error-free, or subsequent supervisory frames (RR, RNR, and REJ), shall use the control field information contained in the N(R) field and the P or F bit to perform data link control functions; for example, to receive acknowledgement of previously transmitted I frames and to cause the data link layer entity to respond if the P bit is set to 1. Therefore, the retransmitted I frame may contain an N(R) field value that is updated from, and therefore different from, the one contained in the I frame transmitted originally.

In variance with Q.921 [10], MiLAP and MiLAP-S do not support poll in I frames.

The REJ frame is used by a receiving data link layer entity to initiate an exception condition recovery (retransmission) following the detection of an N(S) sequence error.

Only one REJ exception condition for a given direction of information transfer shall be established at a time.

A data link layer entity receiving an REJ command or response shall initiate sequential transmission (retransmission) of I frames starting with the I frame indicated by the N(R) contained in the REJ frame.

An REJ exception condition is cleared when the requested I frame is received or when an SABM or DISC command is received.

Note - MiLAP-S does not support the DISC command.

5.7.2 N(R) sequence error

An N(R) sequence error exception condition occurs in the transmitter when a valid supervisory frame or I frame is received which contains an invalid N(R) number.

A valid N(R) is one that is in the range V(A) $\leq$ N(R) $\leq$ V(S).

The information field contained in an I frame which is correct in sequence and format may be delivered to layer 3 by means of the DL-DATA-*indication* primitive.

The data link layer entity shall inform the management entity of this exception condition by means of the MDL-ERROR-*indication* primitive, and shall initiate re-establishment according to § 5.6.2.

5.7.3 Timer recovery condition

If a data link layer entity, due to a transmission error, does not receive a single I frame or the last I frame(s) in a sequence of I frames, it will not detect an out-of-sequence exception condition and therefore will not transmit an REJ frame.

The data link layer entity which transmitted the unacknowledged I frame(s) shall, on the expiry of timer T200, take appropriate recovery action (as defined in § 5.5.8) to determine at which I frame retransmission must begin.

5.7.4 Invalid frame condition

Any frame received which is invalid (as defined in § 2.9) shall be discarded, and no action shall be taken as a result of that frame.

5.7.5 Frame rejection condition

A frame rejection condition results from one of the following conditions:

- the receipt of an undefined frame (see § 2.10 and § 3.6.1, second paragraph);
- the receipt of a supervisory or unnumbered frame with incorrect length;
- the receipt of an invalid N(R); or
- the receipt of a frame with an information field which exceeds the maximum established length.

Upon occurrence of a frame rejection condition whilst in the multiple frame operation, the data link layer entity shall:

- issue an MDL-ERROR-*indication* primitive; and
- initiate re-establishment (see § 5.6.2).

Upon occurrence of a frame rejection condition during establishment or release from the multiple frame operation, or whilst a data link is not established, the data link layer entity shall:

- issue an MDL-ERROR-*indication* primitive; and
- discard the frame.

Note - For satisfactory operation it is essential that a receiver is able to discriminate between invalid frames (as defined in § 2.9), and frames with an I-field which exceeds the maximum established length. An unbounded frame may be assumed, and thus discarded, if two times the longest permissible frame plus two octets are received without a flag detection.

5.7.6 Unsolicited response frames

The action to be taken on receipt of an unsolicited response frame is defined in Table 6/MTS21-C.

Table 6/MTS21-C

Actions taken on receipt of unsolicited response frames

Unsolicited response frame	Address-assigned	Awaiting establishment	Awaiting release	Multiple frame modes of operation	
				Established mode	Timer recovery
UA response F = 1	MDL-ERROR *-indication*	Solicited	Solicited	MDL-ERROR *-indication*	MDL-ERROR *-indication*
UA response F = 0	MDL-ERROR *-indication*	MDL-ERROR *-indication*	MDL-ERROR *-indication*	MDL-ERROR *-indication*	MDL-ERROR *-indication*
Supervisory response F = 1	Ignore	Ignore	Ignore	MDL-ERROR *-indication*	Solicited
Supervisory response F = 0	Ignore	Ignore	Ignore	Solicited	Solicited

Q.921 [10] specifies actions for DM responses. The DM response is not supported in MiLAP or MiLAP-S.

5.8 List of system parameters

The service parameters listed below are associated with each individual SAP.

MiLAP and MiLAP-S employ fixed parameter values, thus do not support the parameter negotiation option of Q.921 [10].

5.8.1 Timer T200

The value of timer T200, at the end of which transmission of a frame may be initiated according to the procedures described in § 5.5, is product dependant.

In current implementations, T200 is normally set to 150 ms between two PBXs, 600 ms for voice sets and 300 ms for datasets.

For data link layer entities operating on a B-channel, the value of T200 shall be 50 ms.

5.8.2 Timer T210

The value of timer T210, the maximum time for D-channel hold, shall be less than 800 ms.

Note - This value is recommended, but is still for further study.

5.8.3 Timer ACKTMR

The value of ACKTMR shall be smaller than T200. As a general guideline, it should be about a third of the value of T200.

Note- This timer is implementation dependant and does not form an integral part of MTS21. Its usage is not required. It is only used in some implementations to reduce traffic.

5.8.4 Maximum number or retransmissions (N200)

The maximum number of retransmissions of a frame (N200) shall be 7.

Note1 - Current implementations typically use an N200 value of 5.

Note2 - For data link layer entities attempting to establish on a B-channel, intervening switching entities may require times longer than N200 times T200 to establish a circuit switch connection. Therefore, a separate counter (N200A), whose minimum value shall be 60 and whose maximum value should not exceed 100 (3 seconds to 5 seconds), shall be used in such an instance. The default value for N200A should be 64. Once the data link layer has been established, N200 shall be used.

5.8.5 Maximum number of octets in an information field (N201)

The maximum number of octets in an information field shall be 260.

Note - The use of large I fields may impact system performance. Some current products limit the I field to values such as 86 and 14.

5.8.6 Maximum number of outstanding I frames (k)

The maximum number (k) of sequentially numbered I frames that may be outstanding (i.e., unacknowledged) at any given time shall be no greater than 7.

Note - Current MiLAP-S implementations limit the value of k to a maximum of 3. Some implementations use a value of 1.

- State transition tables of the point-to-point procedures of the data link layer

General

States

For both MiLAP and MiLAP-S, primary and secondary states are defined.

The primary states for MiLAP are:

- DISC send (DS);
- disconnected (DC); and
- SABM reset (RS).

The secondary states for MiLAP are:

- normal (NM);
- local busy (LB);
- reject sent (RJ); and
- remote busy (RB).

The primary state for MiLAP-S is:

- SABM reset (RS).

The secondary states for MiLAP-S are:

- normal (NM);
- reject sent (RJ); and
- remote busy (RB).

Table 7/MTS21-C illustrates the mapping between the above states and the connection endpoint states as defined in MTS21-B § 4.4.2.3.

Events

Events which may cause state transitions, and which are addressed in the state transition tables, are composed of the following:

- primitives;
- repertoire of frames to be received (see Table 3/MTS21-C and Table 4/MTS21-C); and
- internal events (e.g., servicing of queues and expiry of timers).

Table 7/MTS21-C

Mapping to connection endpoint

Connection endpoint state	MiLAP	MiLAP-S
Link connection released	DC	-
Awaiting establish	RS	RS
Awaiting release	DS	-
Link connection established	NM, LB, RJ, RB	NM, RJ, RB

states

Actions

The actions which may be taken when an event occurs whilst in a specific state comprise:

- transition to another state;
- transmitting peer-to-peer frames;
- issuing primitives;
- setting timers;
- setting counters;
- updating variables;
- setting P/F bit;
- discarding contents of queues; and
- ignoring received frames.

Key to the state transition table

Each cell in the state transition table occurs at the intersection of a current state and a stimulus received by the data link layer entity. The cell itself defines the actions to be taken (see § A.1.3) for the given conditions.
The first entry in a given cell defines the state to be assumed as a result of the conditions. This state may or may not involve a state transition (i.e., may remain in the current state). Subsequent entries define other actions to be taken.
The stimulus T200 refers to the expiry of timer T200. The action T200 refers to the starting of timer T200. It shall be started when a command is sent.
"Data to I queue" and "Data to UI queue" refer to data being transferred from layer 3 to layer 2, while the " Tx I frame" and "Tx UI frame" refer to data being transferred from layer 2 to layer 1.
The Master side refers to the system (or network) side, while the Slave side refers to the user side (e.g., telephones and datasets). The slave side is the side that is responsible for initiating communication via a SABM.

Table 8/MTS21-C
MiLAP primary state table (Master side)

Stimulus	DS DISC send	DC Disconnected	RS SABM reset
DISC (P = 0, 1)	DS Tx UA	DC Tx UA	DC Tx UA (F = 0, 1)
SABM	DS	NM Tx UA DL-ESTABLISH-ind	NM Tx UA
SABM (P = 1)	DS Tx DISC (P = 1) T200	NM Tx UA (F = 1) DL-ESTABLISH-ind	NM Tx UA (F = 1)
UA (F = 0, 1) (expected)	DC DL-RELEASE-conf	DC	NM DL-ESTABLISH-ind
UI	DS	DC	RS
T200 expires retransmissions < N200	DS Tx DISC (P = 1) T200	DC	RS Tx SABM (P = 1) T200
T200 expires retransmissions = N200	DC (note 1)	DC	DS Tx DISC (P = 1) T200
RR, RNR (P = 1)	DS Tx DISC (P = 1) T200	DS Tx DISC (P = 1) T200	RS Tx SABM (P = 1) T200
I frame; bad cmd/resp; unsolicited F bit; S frame (RR, RNR, REJ)	DS	DC	RS
DL-ESTABLISH-req	RS	RS	RS
DL-RELEASE-req	DS	DL-RELEASE-conf	DS Tx DISC; T200
DL-DATA-req	DS	DC	RS Discard I queue
DL-UNIT-DATA-req	DS Data to UI queue	DC Data to UI queue	RS Discard UI queue
DL-ASSIGN-req	RS Store address	RS Store address	RS Store address

Note 1 - May also involve a change to the D-channel, if operating on a B-channel.

Note 2- The DC state is a transient state and may involve a transition to another channel.

Table 9/MTS21-C
MiLAP primary state table (Slave side)

Stimulus	DS DISC send	DC Disconnected	RS SABM reset
DISC (P = 0, 1)	DS Tx UA	DC Tx UA	DC Tx UA (F = 0, 1)
SABM	DS	NM Tx UA DL-ESTABLISH-ind	RS Tx UA
SABM (P = 1)	DS Tx DISC (P = 1) T200	NM Tx UA (F = 1) DL-ESTABLISH-ind	RS Tx UA (F = 1)
UA (F = 0, 1) (expected)	DC DL-RELEASE-conf	DC	NM DL-ESTABLISH-ind
UI	DS	DC	RS
T200 expires retransmissions < N200	DS Tx DISC (P = 1) T200	DC	RS Tx SABM (P = 1) T200
T200 expires retransmissions = N200	DC (note 1)	DC	DS Tx DISC (P = 1) T200
RR, RNR (P = 1)	DS Tx DISC (P = 1) T200	DS Tx DISC (P = 1) T200	RS Tx SABM (P = 1) T200
I frame; bad cmd/resp; unsolicited F bit; S frame (RR, RNR, REJ)	DS	DC	RS
DL-ESTABLISH-req	RS Tx SABM; T200	RS Tx SABM; T200	RS
DL-RELEASE-req	DS	DL-RELEASE-conf	DS Tx DISC; T200
DL-DATA-req	DS	DC	RS Discard I queue
DL-UNIT-DATA-req	DS Data to UI queue	DC Data to UI queue	RS Discard UI queue
DL-ASSIGN-req	RS Store address T200	RS Store address T200	RS Store address T200

Note 1 - May also involve a change to the D-channel, if operating on a B-channel.
Note 2- The DC state is a transient state and may involve a transition to another channel.

Table 10/MTS21-C
MiLAP secondary state table

Stimulus	NM Normal	LB Local Busy	RJ Reject sent	RB Remote busy	RB and RJ
DISC	DC Tx UA DL-RELEASE-ind	DC TxUA DL-RELEASE-ind	DC TxUA DL-RELEASE-ind	DC TxUA DL-RELEASE-ind	DC TxUA DL-RELEASE-ind

	Discard queues	Discard queues	Discard queues	Discard queues	Discard queues
SABM (P = 0, 1)	NM Tx UA DL-ESTABLISH-ind	NM Tx UA DL-ESTABLISH-ind	NM Tx UA DL-ESTABLISH-ind	NM Tx UA DL-ESTABLISH-ind	NM Tx UA DL-ESTABLISH-ind
I frame (good)	NM Tx RR DL-DATA-ind	LB Tx RNR Discard	NM Tx RR DL-DATA-ind	RB (not LB also) Tx RR DL-DATA-ind (LB also) Tx RNR Discard	RB Tx RR DL-DATA-ind
I frame (bad N(S))	RJ Tx REJ Discard	LB Discard	RJ Discard	RB and RJ TX REJ Discard	RB and RJ Discard
UI frame	NM DL-UNIT-DATA-ind	LB Discard	RJ DL-UNIT-DATA-ind	RB DL-UNIT-DATA-ind	RJ
RR (P = 1)	RB Tx RR (F = 1)	LB Tx RNR (F = 1)	RB and RJ Tx RR (F = 1)	NM (not LB also) Tx RR (F = 1) LB (LB also) Tx RNR (F = 1)	RJ Tx RR (F =1)
RNR (P = 1)	RB Tx RR (F = 1)	LB and RB Tx RNR (F = 1)	RB and RJ Tx RR (F = 1)	RB Tx RR (F = 1)	RB and RJ Tx RR (F = 1)
RR	NM	LB	RJ	LB or NM	RJ
RNR	RB T200	LB and RB	RB and RJ	RB	RB and RJ
REJ	NM	LB	RJ	LB or NM	RJ
RR (F = 1) (expected)	NM Stop T200	LB	RJ Stop T200	NM Stop T200	RJ Stop T200
RNR (F =1) (expected)	RB	LB and RB	RB and RJ	RB	RB and RJ
T200 expires; retransmissions < N200	NM Tx RR (P = 1) T200	LB Tx RNR (P = 1) T200	T200 RJ (not LB also) Tx RR (P = 1) (LB also) Tx RNR (P = 1)	T200 RB (not LB also) Tx RR (P = 1) (LB also) Tx RNR (P = 1)	RB and RJ Tx RR (P = 1) T200
T200 expires; retransmissions = N200	RS T200 Tx SABM (P = 1) DL-RELEASE-ind	RS T200 Tx SABM (P = 1) DL-RELEASE-ind	RS T200 Tx SABM (P = 1) DL-RELEASE-ind	RS T200 Tx SABM (P = 1) DL-RELEASE-ind	RS T200 Tx SABM (P = 1) DL-RELEASE-ind
Bad N(R); UA; bad cmd/resp; unsolicited F bit	RS Tx SABM DL-RELEASE-ind	RS Tx SABM DL-RELEASE-ind	RS Tx SABM DL-RELEASE-ind	RS Tx SABM DL-RELEASE-ind	RS Tx SABM DL-RELEASE-ind
No buffers	LB Tx RNR	LB	RJ and LB Tx RNR	RB and LB Tx RNR	Rb and RJ and LB Tx RNR
Buffers	NM	NM Tx RR (P = 1)	RJ	RB	RB and RJ
DL-ESTABLISH-	RS Tx SABM	RS Tx SABM	RS Tx SABM	RS Tx SABM	RS Tx SABM

req	T200 Discard I queue	T200 Discard I queue	T200 Discard I queue	T200 Discard I queue	T200 Discard I queue
DL-RELEASE-req	DS Tx DISC T200 Discard I queue	DS Tx DISC T200 Discard I queue	DS Tx DISC T200 Discard I queue	DS Tx DISC T200 Discard I queue	DS Tx DISC T200 Discard I queue
DL-DATA-req	NM Data to I queue	LB Data to I queue	RJ Data to I queue	RB Data to I queue	RB and RJ Data to I queue
DL-UNIT-DATA-req	NM Data to UI queue	LB Data to UI queue	RJ Data to UI queue	RB Data to UI queue	RB and RJ Data to UI queue
DL-ASSIGN-req	RS Store address Tx SABM T200 Discard I queue	RS Store address Tx SABM T200 Discard I queue	RS Store address Tx SABM T200 Discard I queue	RS Store address Tx SABM T200 Discard I queue	RS Store address Tx SABM T200 Discard I queue

Note 1- LB, RB and RJ states may coexist.
Note 2- When RB exists, the local data link layer entity shall conduct status inquiry until the RB is cleared.
Note 3- When leaving LB due to "buffers" stimulus, the poll bit in the RR need only be set when the remote data link layer entity is MiLAP-S.

Table 11/MTS21-C
MiLAP-S primary state table

Stimulus	RS SABM reset
SABM	RS Tx UA
SABM (P = 1)	RS Tx UA (F = 1)
UA	NM DL-ESTABLISH-ind
T200 expires	RS Tx SABM T200
I frame bad command/response unsolicited F bit S frame (RR, RNR, REJ)	Discard
DL-ESTABLISH-req	RS
DL-RELEASE-req	RS DL-RELEASE-conf
DL-DATA-req	RS Discard I queue
DL-UNIT-DATA-req	RS Discard UI queue
DL-ASSIGN-req	RS Store address Tx SABM T200

Table 12/MTS21-C
MiLAP-S secondary state table

Stimulus	NM Normal	RJ Reject sent	RB Remote busy	RB and RJ
SABM (P = 0, 1)	NM Tx UA (F=0, 1) DL-ESTABLISH-ind	NM Tx UA (F=0, 1) DL-ESTABLISH-ind	NM Tx UA (F=0, 1) DL-ESTABLISH-ind	NM Tx UA (F=0, 1) DL-ESTABLISH-ind
UA	RS Tx SABM; T200 DL-RELEASE-ind	RS Tx SABM; T200 DL-RELEASE-ind	RS Tx SABM; T200 DL-RELEASE-ind	RS Tx SABM; T200 DL-RELEASE-ind
I frame (good)	NM Tx RR DL-DATA-ind	NM Tx RR DL-DATA-ind	RB Tx RR DL-DATA-ind	RB Tx RR DL-DATA-ind
I frame (bad N(S))	RJ Tx REJ	RJ Discard	RB and RJ TX REJ	RB and RJ Discard
RR (P = 0)	NM	RJ	NM	RJ
REJ (P = 0)	NM Tx I frame; T200	RJ Tx I frame; T200	NM Tx I frame; T200	RJ Tx I frame; T200
RR (P = 1)	NM Tx RR (F = 1)	RJ Tx RR (F = 1)	NM Tx RR (F = 1)	RJ Tx RR (F = 1)
RNR (P = 0)	RB	RB and RJ	RB	RB and RJ
RNR (P = 1)	RB Tx RR (F = 1)	RB and RJ Tx RR (F = 1)	RB Tx RR (F = 1)	RB and RJ Tx RR (F = 1)
T200 expires retransmissions < N200	NM Tx I frame; T200	RJ Tx I frame; T200	RB Note	RB and RJ Note
T200 expires retransmissions = N200	RS Tx SABM T200 DL-RELEASE-ind	RS Tx SABM T200 DL-RELEASE-ind	RB Note	RB and RJ Note
Bad N(R); bad cmd/resp; unsolicited F bit	RS Tx SABM; T200 DL-RELEASE-ind	RS Tx SABM; T200 DL-RELEASE-ind	RS Tx SABM; T200 DL-RELEASE-ind	RS Tx SABM; T200 DL-RELEASE-ind
DL-ESTABLISH-req	RS Tx SABM; T200 Discard I queue	RS Tx SABM; T200 Discard I queue	RS Tx SABM; T200 Discard I queue	RS Tx SABM; T200 Discard I queue
DL-RELEASE-req	RS Tx SABM; T200 Discard I queue DL-RELEASE-conf	RS Tx SABM; T200 Discard I queue DL-RELEASE-conf	RS Tx SABM; T200 Discard I queue DL-RELEASE-conf	RS Tx SABM; T200 Discard I queue DL-RELEASE-conf
DL-DATA-req	NM Data to I queue	RJ Data to I queue	RB Data to I queue	RB Data to I queue
DL-UNIT-DATA-req	NM Data to UI queue	RJ Data to UI queue	RB Data to UI queue	RB Data to UI queue
DL-ASSIGN-req	RS Store address Tx SABM; T200	RS Store address Tx SABM; T200	RS Store address Tx SABM; T200	RS Store address Tx SABM; T200

Note - The number of retransmissions shall not be incremented while in RB state.

Section D - Conformance Testing

1.0 Introduction

1.1 Purpose

This section defines the abstract test suite for conformance testing MiLAP user or network side equipment, or MiLAP-S user side equipment.

1.2 Scope

It is possible that some tests will not be applicable to all IUT's. A test selection procedure has to be performed to determine this. Such selection shall be based on the Protocol Implementation Conformance Statement (PICS) and the Protocol Implementation Extra Information for Testing (PIXIT). In doing such a selection, interoperability among MITEL DNIC devices shall be of prime importance.

1.3 Background

This test suite uses valid, invalid, undefined and inopportune frames to test the IUT behaviour. These terms are defined as follows:

Valid frames: A valid frame is an expected frame which arrives at the correct state and does not belong to any of the categories listed under invalid frames.

Invalid frames: An invalid frame as described in Section C, is a frame which:

is not properly bounded by two flags; or

has fewer than four octets between flags of one-octet address frames; or

does not consist of an integral number of octets prior to zero bit insertion or following zero bit extraction; or

contains a frame check sequence error; or

contains a service access point identifier which is not supported by the receiver.

Undefined frames: An undefined frame is a frame which:

is error free, but which is unknown as a command or a response; or

has an invalid frame format; or

has an invalid N(R); or

has the final bit set when there is no poll outstanding.

Inopportune frames: An inopportune frame is a syntactically valid frame arriving at a time (IUT's state) when it should be considered irrelevant by the IUT.

1.4 General aspects

As per ISO DIS 9646, “a complete and independent specification of the actions required to achieve a specific test purpose” is called an abstract test suite. These test cases, along with the test body, include a preamble and a postamble, which are defined below, to ensure starting and ending in a stable state and involve one or more consecutive or concurrent connections. However, as per ISO DIS 9646 Part 2, it can be useful to use other stable states for starting and ending abstract test cases, in order to concatenate test cases in a manner which permits efficient execution.

1.5 Preamble

The preamble of a test case consists of the steps required to bring the IUT to the appropriate initial state. There may be alternate sequences of test steps which can be performed to initialize the IUT. These test steps in the preamble have been carefully chosen, considering the test methodology and other state coordination procedures that are available. In general, it is preferred that these steps be built on an “idle state” which is stable and “most likely” under many testing situations.

1.6 Test body

The test body is the sequence of steps within a test case, that is essential to achieve the test purpose, followed by the verification of the IUT's ending state. Verdicts are assigned to the possible outcomes of the test case.

It is important to test the observable behaviour of the IUT, which includes state transitions and protocol data unit (PDU) responses. If one assumes that all states are implemented, it is not possible to obtain unique traces or signatures which would guarantee that the IUT is in the expected state. Also, some of the MiLAP/MiLAP-S states are transitional and may not be implemented.

1.7 Postamble

At the end of the execution of a test body, the IUT may not be in an “idle state”. A postamble is required to return the IUT from its current state to an idle state. It will ensure that the IUT is in the SABM reset (RS) state.

1.8 Timers

MiLAP-S timers include: T200, T201, T210.
Tester timer TWAIT: Specifies the maximum duration the tester waits before concluding the IUT is not transmitting an unexpected message.

Note - The tester will use the T200 timer as the maximum duration that the tester will wait for before concluding that the IUT is not transmitting an expected frame.

1.9 Layer 2 information frame content

This test suite requires the use of Layer 3 initiated I frames. However, the contents of the information field shall not force a Layer 3 response from the upper entity.

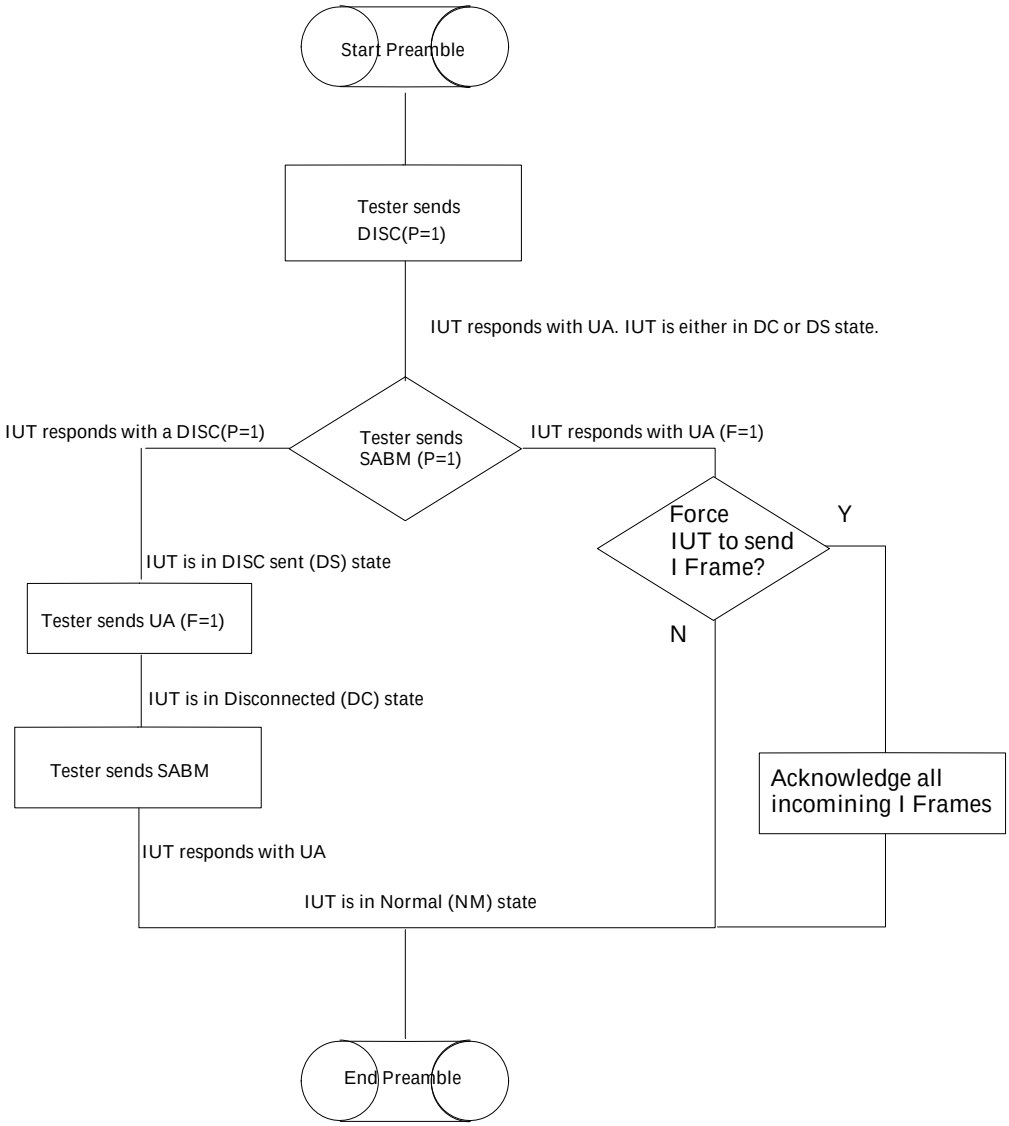
1.10 PICS/PIXIT relationship to the abstract test suite

There are instances when the execution of a test case depends on the answer to a PICS or PIXIT question. This makes the execution of the test case conditional.

2.0 Abstract test suite for MiLAP D-channel

2.1 Preamble

The following preamble will be used for this ATS. This will ensure that the IUT is in the normal (NM) state prior to the start of each test. The tester will first send a DISC (P=1) frame. The IUT will respond with a UA frame and providing it is not in the Disconnect sent (DS) state, migrate to the Disconnected (DC) state. The tester sends a SABM frame (P=1). If the IUT responds with an UA frame then the IUT will migrate from the DC state to the Normal(NM) state. If the IUT responds with a DISC frame (P=1) then it must be in the Disconnect sent (DS) state. In order to get the IUT from the DS state to the NM state the tester sends a UA frame(F=1). This causes the IUT to migrate to the DC state. The tester then sends a SABM frame. The IUT will respond with a UA frame and migrate to the Normal (NM) state.



2.2 Postamble

At the end of the execution of a test body, the IUT may not be in an “idle state”. A postamble is required to return the IUT from its current state to an idle state. The following postamble will be used for this ATS. This will ensure that the IUT is in the DISC sent (DS) state.

TESTER IUT
DISC >
< UA
>DC or DS

RR (P=1) >
< DISC
>DS

2.3 PIXIT Proforma

There are instances when the execution of a test case depends on the answer to a PICS or PIXIT question. This makes the execution of the test case conditional.

Value of k (transmit)
Value of k (receive)
N201 maximum octets in an I frame.
N200 maximum re-transmissions.
Set PBX Type
Date of testing
Limitations of IUT
Tester serial number
Test software version number
Tester operator's name
Can IUT send SABM frame on demand?
Can the IUT be forced to send an I frame on demand?
Can the IUT be forced to send more than 1 I frame on demand?
Which I frame will IUT send for these tests?
Which I frame can the tester send to the IUT (programmable field of I frame)?

2.4 PICS Proforma

There are instances when the execution of a test case depends on the answer to a PICS or PIXIT question. This makes the execution of the test case conditional.

Value of T200
Address of IUT
What set ID (if applicable) will be sent to the IUT by the tester

2.5 Test Cases

The following test cases are written using X- Notation. The Conformance tester sends the commands shown by the right arrows and the IUT sends the messages shown by the left arrows. State transitions are shown with a > symbol. This means the IUT is expected to migrate to the state shown after the > symbol. Note that although not shown, the preamble is executed at the beginning of each test and the postamble at the end of each test. The preamble puts the IUT into the normal (NM) state. The postamble puts the IUT into DISC sent (DS) state. Even if the test case fails the postamble will be executed.

2.5.1 Go-Ahead procedure Test when the IUT is the PBX

This test ensures the physical layer Go-Ahead procedure is working. The test operator plugs in the IUT. The tester sends idle flags and then switches to continuous flags. When the IUT receives the continuous flags pattern, it should send a Go-Ahead to the tester. The tester then sends a SABM frame followed by an idle ones pattern. The IUT should respond with a UA frame. The tester then sends a continuous flags pattern. The IUT should respond with a Go-Ahead. The tester then sends a Report Set ID I frame to the IUT followed by an idle ones pattern. The IUT should then respond with an RR frame.

TESTER IUT
plug in IUT
< idle ones or flags
idle ones pattern >
Continuous flags >
< Go-Ahead(7F,0)
SABM >
dle ones pattern >
< UA
Continuous flags >

< Go-Ahead (7F,0)
I frame (report Set ID) >
Idle ones pattern >
<- RR

2.5.2 Go-Ahead procedure Test when the IUT is connected to the PBX (e.g.,Set/Dataset)
This test ensures the physical layer Go-Ahead procedure is working. The test operator plugs in the IUT. The IUT should send idle ones and then switch to continuous flags. When the tester receives the continuous flags pattern, it sends multiple Go-Aheads to the IUT. The IUT is then expected to send a SABM frame and then send an idle ones pattern. The tester responds with a UA frame. The IUT is then expected to send a continuous flags pattern. The tester will then send a Go-Ahead to the IUT. The IUT is expected to send a Report Set ID I frame to the tester followed by an idle ones pattern. The tester will then respond with an RR frame.
TESTER IUT
plug in IUT
idle ones or flags >
< Idle ones pattern
<- Continuous flags
Multiple Go-Aheads (7F, 0) >
<- SABM
>RS, start T200
<- Idle ones pattern
UA >
>NM,stop T200
<- Continuous flags
Go-Ahead (7F, 0) >
<- I frame (report Set ID)
start T200
<- Idle ones pattern
RR >
stop T200

2.5.3 DISC frame received while in the DISC send (DS) state
The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a DISC frame and the IUT is expected to send a UA frame and remain in the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.
TESTER IUT
DISC >
< UA
>DC
RR(P=1) >
<- DISC(P=1)
>DS
DISC >
<- UA
SABM (P=1) >
<- DISC (P=1)

2.5.4 DISC frame received while in the Disconnected (DC) state
The tester sends a DISC frame to force the IUT into the DC state. The tester then sends another DISC frame and the IUT is expected to send a UA frame and remain in the DC state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a UA frame (F=1) and migrate to the NM state. The tester sends an I frame to verify this state transition. The IUT is expected to respond with a RR frame.
TESTER IUT
DISC >

```
<       UA
        >DC
DISC    >
<       UA
SABM (P=1)      >
<-      UA (F=1)
        >NM
I frame (good)  >
<-      RR
```

2.5.5 DISC frame received while in the SABM reset (RS) state

The tester sends an I frame with a bad N(R) to force the IUT into SABM reset (RS) state. The tester then sends a DISC frame. The IUT is expected to send a UA frame and migrate to the Disconnected (DC) state. The tester then sends an RR (P=1) frame to verify that the IUT is in the Disconnected (DC) state. The IUT should transmit a DISC frame (P=1) and migrate to the DISC send (DS) state. The tester then verifies this state transition by sending a SABM frame (P=1). The IUT is expected to send a DISC frame (P=1)and remain in the DS state.

```
TESTER          IUT
I frame (bad N(R))          >
<       SABM
        >RS, start T200
DISC    >
<       UA
        >DC
RR (P=1)        >
<       DISC (P=1)
        >DS
SABM (P=1)      >
<       DISC (P=1)
```

2.5.6 SABM frame received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a SABM frame and the IUT is expected to ignore it and remain in the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

```
TESTER          IUT
DISC    >
<-      UA
        >DC
RR (P=1)        >
<-      DISC (P=1)
        >DS,start T200
SABM    >
        ignore
SABM (P=1)      >
<-      DISC (P=1)
```

2.5.7 SABM frame received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a SABM frame. The IUT should respond with a UA frame and migrate to the NM state. The tester verifies this by sending a good I frame the IUT should respond with an RR frame.

```
TESTER          IUT
DISC    >
<       UA
        >DC
SABM    >
```

```
<       UA
        >NM
I frame (good)  >
<       RR
```

2.5.8 SABM frame received by IUT while in SABM reset (RS) state

The tester sends an I frame with a bad N(R) to force the IUT into SABM reset (RS) state. The tester then sends a SABM frame. The IUT is expected to send a UA frame and remain in SABM reset state. The tester then sends an I frame to verify that the IUT is in SABM reset state. The IUT should ignore this I frame.

```
TESTER          IUT
I frame (bad N(R))      >
<       SABM
        >RS, start T200
SABM    >
<       UA
I frame (good)  >
        ignore
```

2.5.9 SABM frame (P=1) received by IUT while in DISC send (DS) state

Already tested in § 2.5.2.

2.5.10 SABM frame (P=1) received by IUT while in Disconnected (DC) state

Already tested in § 2.5.3.

2.5.11 SABM frame (P=1) received by IUT while in SABM reset (RS) state

The tester sends an I frame with a bad N(R) to force the IUT into SABM reset state. The tester then sends a SABM frame with poll bit set to 1. The IUT is expected to send a UA frame with the final bit set to 1 and the IUT should remain in the SABM reset state. This is tested by sending an I frame. The I frame should be ignored by the IUT.When T200 expires the IUT should send a SABM frame(P=1). The tester responds with a UA frame(F=1). The IUT should then migrate to the Normal state. The tester verifies this by sending an I frame. The IUT should respond with a RR frame.

```
TESTER          IUT
I frame (bad N(R))      >
<       SABM
        >RS, start T200
SABM (P = 1)    >
<-      UA (F = 1)
I frame (good)  >
        T200 expires
<       SABM(P=1)
        start T200
UA (F=1)        >
        >NM,stop T200
I frame (good)  >
<-      RR
```

2.5.12 UA frame received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to the IUT. The IUT is expected to respond with a UA frame and migrate to the DC state. The tester then sends an RR frame (P=1). The IUT is expected to respond with a DISC frame (P=1) and migrate to the DS state. The tester then sends a UA frame and the IUT is expected to ignore it since the final bit is not set. When T200 expires the IUT is expected to send a DISC frame(P=1). The tester sends a SABM frame(P=1). The IUT should respond with a UA frame (F=1) and migrate to the NM state. The tester then sends a good I frame and expects the IUT to respond with a RR frame.

```
TESTER          IUT
DISC    >
<       UA
```

```
          >DC
 RR (P=1)          >
<         DISC (P=1)
          >DS,start T200
UA        >
          ignore
          T200 Expires
<         DISC (P=1)
          restart T200
SABM (P=1)      >
<-        UA (F=1)
          >NM,stop T200
I frame (good)    >
<-        RR
```

2.5.13 UA frame received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a UA frame and the IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame (P=1) to the IUT. The IUT should respond with a UA frame (F=1) and migrate to the NM state.Th e tester verifies this by sending an I frame. The IUT should respond with an RR frame.

```
TESTER            IUT
DISC      >
<         UA
          >DC
UA        >
          ignore
SABM (P=1)      >
<         UA (F=1)
          >NM
I frame (good)    >
<         RR
```

2.5.14 UA frame received by IUT while in SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset (RS) state. The tester then sends a UA frame which should cause the IUT to enter the normal state (NM). The tester will wait TWAIT seconds to ensure the IUT has stopped its T200. T he tester verifies that the IUT is in the Normal state by sending an I frame. The IUT should then respond with an RR frame.

```
TESTER            IUT
I frame (bad N(R))          >
<         SABM
          >RS, start T200
          T200 Expires
<         SABM
          restart T200
UA        >
          >NM, stop T200
TWAIT to ensure IUT has stopped T200
I frame (good)    >
<-        RR
```

2.5.15 UA frame (F=1) received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a UA frame(F=1) and the IUT is expected to migrate to the DC state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a UA frame (F=1) and migrate to the NM state.The tester then sends a good I frame and expects the IUT to respond with an RR frame.

```
TESTER          IUT
DISC    >
<       UA
        >DC
 RR (P=1)       >
<       DISC (P=1)
        >DS
UA (F=1)        >
        >DC
SABM (P=1)      >
<-      UA (F=1)
        >NM
I frame (good)  >
<-      RR
```

2.5.16 UA frame (F=1) received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a UA frame and the IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame (P=1) to the IUT. The IUT should respond with a UA frame (F=1) and migrate to the NM state.Th e tester verifies this by sending an I frame. The IUT should respond with an RR frame.

```
TESTER          IUT
DISC    >
<       UA
        >DC
UA (F=1)        >
        ignore
SABM (P=1)      >
<-      UA (F=1)
        >NM
I frame (good)  >
<-      RR
```

2.5.17 UA frame(F=1) received by IUT while in SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset (RS) state. The tester then waits T200 seconds for the IUT to transmit a SABM frame (P=1).The tester then sends a UA frame (F=1) which should cause the IUT to enter the normal state (NM). The tester verifies this by sending an I frame. The IUT should then respond with an RR frame.

```
TESTER          IUT
I frame (bad N(R))      >
<       SABM
        >RS, start T200
        T200 Expires
<       SABM (P=1)
        start T200
UA (F=1)        >
        >NM, stop T200
TWAIT to ensure IUT has stopped T200
I frame (good)  >
<-      RR
```

2.5.18 UI frame received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a UI frame and the IUT is expected to send nothing and remain in the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

```
TESTER          IUT
```

DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
UI >
ignore
SABM (P=1) >
< DISC (P=1)

2.5.19 UI frame received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a UI frame and the IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame (P=1) to the IUT. The IUT should respond with a UA frame (F=1) and migrate to the NM state.Th e tester verifies this by sending an I frame. The IUT should respond with an RR frame.

TESTER IUT
DISC >
< UA
>DC
UI >
ignore
SABM (P=1) >
<- UA (F=1)
>NM
I frame (good) >
< RR

2.5.20 UI frame received by IUT while in the SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends a UI frame. The IUT is expected to ignore this frame and remain in the SABM reset state.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
UI >
ignore
T200 expires
<- SABM (P=1)

2.5.21 T200 expires (RC< N200) while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The IUT is expected to transmit a DISC frame (P=1) when T200 expires.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
T200 expires
< DISC (P=1)
T200 expires
< DISC (P=1)

2.5.22 T200 expires (RC< N200) while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state.When T200 expires nothing should be transmitted and the IUT should remain in the DC state. The tester verifies this by sending a SABM frame. The IUT should respond with a UA frame and migrate to the Normal state. The tester verifies this by sending a good I frame. The IUT is expected to send a RR frame.

TESTER IUT
DISC >
< UA
>DC
T200 expires
TWAIT to ensure IUT does not send a frame
SABM >
< UA
>NM
I frame (good) >
< RR

2.5.23 T200 expires (RC< N200) while in the SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset state. When T200 expires the IUT should send a SABM frame (P=1).

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
T200 expires
<- SABM (P=1)

2.5.24 T200 expires (RC = N200) while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. Each expiry of T200 should cause the IUT to transmit a DISC frame (P=1). This will continue until RC = N200. Then the IUT is expected to migrate to the DC state. The tester verifies this by sending a SABM frame. The IUT should respond with a UA frame and migrate to the NM state. The tester verifies this by sending an I frame. The IUT is expected to respond with a RR frame.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
T200 expires
< DISC (P=1)
T200 expires
< DISC (P=1)
T200 expires (RC = N200)
>DC
Wait TWAIT seconds
SABM >
<- UA
>NM
I frame (good) >
< RR

2.5.25 T200 expires (RC = N200) while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. Since T200 does not run in the DC state nothing should be transmitted by the IUT.The tester waits a period of time greater than (RC multiplied by T200) to ensure the IUT does not transmit a frame.

TESTER IUT
DISC >
< UA
>DC
TWAIT to ensure IUT does not send a frame

2.5.26 T200 expires (RC = N200) while in the SABM reset (RS) state
In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset state. On each expiry of T200 the IUT is expected to transmit a SABM frame (P=1). When T200 expires N200 times, the IUT is expected to send a DISC frame (P=1) and migrate to the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT is expected to respond with a DISC frame (P=1).
TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
T200 expires
< SABM (P=1)
T200 expires (RC = N200)
< DISC (P=1)
>DS
SABM (P=1) >
< DISC (P=1)

2.5.27 RR frame (P=1) received by IUT while in the DISC send (DS) state
The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a RR frame (P=1) and the IUT is expected to transmit a DISC frame (P=1) and remain in the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.
TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
RR (P=1) >
< DISC (P=1)
SABM (P=1) >
< DISC (P=1)

2.5.28 RNR frame (P=1) received by IUT while in the DISC send (DS) state
The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a RNR frame (P=1) and the IUT is expected to transmit a DISC frame (P=1) and remain in the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.
TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
RNR (P=1) >
< DISC (P=1)
SABM (P=1) >
< DISC (P=1)

2.5.29 REJ frame (P=1) received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a REJ frame (P=1) and the IUT is expected to send a DISC frame (P=1) and remain in the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
REJ (P=1) >
< DISC (P=1)
SABM (P=1) >
< DISC (P=1)

2.5.30 RR frame (P=1) received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a RR frame(P=1) and the IUT is expected to send a DISC frame (P=1) and migrate to the DS state. The tester verifies this by sending a SABM frame (P=1) to the IUT. The IUT should respond with a DISC frame (P=1) and remain in the DS state.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
SABM (P=1) >
< DISC (P=1)

2.5.31 RNR frame (P=1) received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a RNR frame (P=1) and the IUT is expected to send a DISC frame (P=1) and migrate to the DS state. The tester verifies this by sending a SABM frame (P=1) to the IUT. The IUT should respond with a DISC frame (P=1) and remain in the DS state.

TESTER IUT
DISC >
< UA
>DC
RNR (P=1) >
< DISC (P=1)
>DS
SABM (P=1) >
< DISC (P=1)

2.5.32 REJ frame (P=1) received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a REJ frame (P=1) and the IUT is expected to ignore it and remain in the DC state. The tester verifies this by sending a SABM frame (P=1) to the IUT. The IUT should respond with a UA frame (F=1) and migrate to the NM state.

TESTER IUT
DISC >
< UA
>DC

REJ (P=1) >
ignore
SABM (P=1) >
< UA (f=1)
>NM

2.5.33 RR frame (P=1) received by IUT while in the SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends a RR frame (P=1). The IUT is expected to send a SABM frame (P=1) and remain in the RS state. This is verified by waiting T200 seconds for another SABM frame(P=1) to be transmitted by the IUT.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
RR (P=1) >
stop T200
<- SABM (P=1)
start T200
T200 expires
<- SABM (P=1)
start T200

2.5.34 RNR frame (P=1) received by IUT while in the SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends a RNR frame (P=1). The IUT is expected to send a SABM frame (P=1) and remain in the RS state.This is verified by waiting T200 seconds for another SABM frame(P=1) to be transmitted by the IUT.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
RNR (P=1) >
stop T200
<- SABM (P=1)
start T200
T200 expires
<- SABM (P=1)
start T200

2.5.35 REJ frame (P=1) received by IUT while in the SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends a REJ frame (P=1). The IUT is expected to respond with a SABM frame (P=1) and remain in the RS state.This is verified by waiting T200 seconds for another SABM frame(P=1) to be transmitted by the IUT.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
REJ (P=1) >
stop T200
< SABM (P=1)
start T200
T200 expires
< SABM (P=1)
start T200

2.5.36 Bad command received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a bad command and the IUT is expected to send nothing and remain in the DS state. when T200 expires the IUT is expected to send a DISC frame(P=1). The tester then sends a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

```
TESTER          IUT
DISC    >
<       UA
        >DC
RR (P=1)        >
<       DISC (P=1)
        >DS,start T200
bad command     >
        ignore
        T200 expires
<       DISC (P=1)
        restart T200
SABM (P=1)      >
<       DISC (P=1)
        restart T200
```

2.5.37 Bad command received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a bad command. The IUT is expected to send nothing and remain in the DC state. The tester verifies this by waiting TWAIT and then sending a SABM frame (P=1). The IUT is expected to respond with a UA frame (F=1) and migrate to the NM state. The tester verifies this by sending an I frame. The IUT is expected to send an RR frame.

```
TESTER          IUT
DISC    >
<       UA
        >DC
bad command     >
        TWAIT
SABM (P=1)      >
<-      UA (F=1)
        >NM
I frame (good)  >
<       RR
```

2.5.38 Bad command received by IUT while in the SABM reset (RS) state

The tester forces the IUT into the SABM reset state by sending it an I frame with a bad N(R). The tester then sends a bad command. The IUT should ignore this message and remain in the SABM reset state. The tester verifies this by waiting TWAIT seconds. During the TWAIT period the IUT will transmit SABM frames each time its T200 timer expires.

```
TESTER          IUT
I frame (bad N(R))      >
<       SABM
        >RS, start T200
bad command     >
        ignore
        T200 expires
<       SABM
        >RS, start T200
TWAIT
```

2.5.39 Unsolicited F bit command received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends an unsolicited F bit and the IUT is expected to send nothing and remain in the DS state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS
unsolicited f bit >
SABM (P=1) >
< DISC (P=1)

2.5.40 Unsolicited F bit command received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends a unsolicited F bit. The IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame (P=1). The IUT is expected to respond with a UA frame (F=1) and migrate to the NM state.The tester verifies this by sending an I frame. The IUT should respond with a RR frame.

TESTER IUT
DISC >
< UA
>DC
unsolicited F bit >
SABM (P=1) >
<- UA (F=1)
>NM
I frame (good) >
< RR

2.5.41 Unsolicited F bit command received by IUT while in SABM reset (RS) state

The tester forces the IUT into the SABM reset state by sending an I frame with a bad N(R). The tester then sends an unsolicited F bit. The IUT should ignore this and remain in SABM reset state. When T200 expires the IUT should send another SABM frame(P=1).

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
RR (F=1) >
<- SABM(P=1)
start T200

2.5.42 I frame received by IUT while in the DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends an I frame and the IUT is expected to send nothing and remain in the DS state. When T200 expires the IUT is expected to transmit a DISC frame(P=1).The tester then sends a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)

```
          >DS,start T200
I frame   >
          ignore
          T200 expires
<         DISC (P=1)
          restart T200
SABM (P=1)      >
<         DISC (P=1)
          restart T200
```

2.5.43 I frame received by IUT while in the Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an I frame and the IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a UA frame (F=1) and migrate to the NM state. The tester verifies this by sending an I frame. The IUT should respond with a RR frame.

```
TESTER          IUT
DISC    >
<         UA
          >DC
I frame   >
          ignore
SABM (P=1)      >
<-        UA (F=1)
          >NM
I frame (good)     >
<         RR
```

2.5.44 I frame received by IUT while in SABM reset (RS) state

In this test, the tester sends an I frame with a bad N(R) to force the IUT in the SABM reset state. The tester then sends an I frame. This I frame should be ignored by the IUT.

```
TESTER          IUT
I frame (bad N(R))          >
<         SABM
          >RS, start T200
I frame   >
          ignore
          T200 expires
<         SABM(P=1)
```

2.5.45 RR supervisory frame received by IUT while in DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a RR frame and the IUT is expected to send nothing and remain in the DS state. When T200 expires the IUT is expected to send a DISC frame (P=1).The tester then sends a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

```
TESTER          IUT
DISC    >
<         UA
          >DC
RR (P=1)          >
<         DISC (P=1)
          >DS,start T200
RR        >
          ignore
          T200 expires
<         DISC (P=1)
```

restart T200
SABM (P=1) >
< DISC (P=1)
restart T200

2.5.46 RR supervisory frame received by IUT while in Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame and the IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a UA frame (F=1) and migrate to the NM state. The tester verifies this by sending an I frame. The IUT is expected to respond with an RR frame.

TESTER IUT
DISC >
< UA
>DC
RR >
SABM (P=1) >
<- UA (F=1)
>NM
I frame (good) >
< RR

2.5.47 RR supervisory frame received by IUT while in SABM reset (RS) state

This test verifies that the IUT ignores a supervisory frame while in the SABM reset state. The tester first sends an I frame with a bad N(R) to force the IUT into the SABM reset state. Then the tester sends an RR frame. The IUT should ignore it.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
RR >
ignore
T200 expires
< SABM(P=1)

2.5.48 RNR frame received by IUT while in DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a RNR frame and the IUT is expected to send nothing and remain in the DS state.When T200 expires the IUT is expected to send a DISC frame (P=1). The tester then sends a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS,start T200
RNR >
ignore
T200 expires
< DISC (P=1)
restart T200
SABM (P=1) >
< DISC (P=1)
restart T200

2.5.49 RNR frame received by IUT while in Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RNR frame and the IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a UA frame (F=1) and migrate to the NM state The tester verifies this by sending an I frame. The IUT is expected to respond with an RR frame.

```
TESTER          IUT
DISC    >
<       UA
        >DC
RNR     >
SABM (P=1)      >
<-      UA (F=1)
        >NM
I frame (good)  >
<       RR
```

2.5.50 RNR frame received by IUT while in SABM reset (RS) state

This test verifies that the IUT ignores a supervisory frame while in the SABM reset state. The tester first sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends an RNR frame. The IUT should ignore it.

```
TESTER          IUT
I frame (bad N(R))          >
<       SABM
        >RS, start T200
RNR     >
        ignore
        T200 expires
<       SABM(P=1)
```

2.5.51 REJ frame received by IUT while in DISC send (DS) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an RR frame (P=1) to force the IUT into the DS state. The tester then sends a REJ frame and the IUT is expected to send nothing and remain in the DS state. When T200 expires the IUT is expected to send a DISC frame (P=1).The tester sends a SABM frame(P=1). The IUT should respond with a DISC frame (P=1) and remain in the DS state.

```
TESTER          IUT
DISC    >
<       UA
        >DC
RR (P=1)        >
<       DISC (P=1)
        >DS,start T200
REJ     >
        ignore
        T200 expires
<       DISC (P=1)
        restart T200
SABM (P=1)      >
<       DISC (P=1)
        restart T200
```

2.5.52 REJ frame received by IUT while in Disconnected (DC) state

The tester sends a DISC frame to force the IUT into the DC state. The tester then sends an REJ frame and the IUT is expected to send nothing and remain in the DC state. The tester verifies this by sending a SABM frame(P=1). The IUT should respond with a UA frame (F=1) and migrate to the NM state. The tester verifies this by sending an I frame. The IUT is expected to respond with a RR frame.

TESTER IUT
DISC >
< UA
>DC
REJ >
SABM (P=1) >
<- UA (F=1)
>NM
I frame (good) >
< RR

2.5.53 REJ frame received by IUT while in SABM reset (RS) state
This test verifies that the IUT ignores a supervisory frame while in the SABM reset state. The tester first sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends an REJ frame. The IUT should ignore it.
TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
REJ >
ignore
T200 expires
< SABM(P=1)

2.5.54 Frame exceeding maximum number of bytes (N201) received by IUT while in normal (NM) state
This test sends an I frame that is too long for the IUT. The IUT should send a SABM frame and migrate to the reset state.
TESTER IUT
I (too long) >
< SABM
>RS, start T200

2.5.55 Frame shorter than the minimum number of bytes received by IUT while in normal (NM) state
This test verifies that the IUT will ignore a frame that is too short. A frame less than 4 octets (including 2 FCS bytes) is considered too short.The tester sends a 2 octet frame followed by a 3 octet frame. Both frames should be ignored by the IUT. The tester then sends a good I frame and the IUT is expected to respond with a RR frame.
TESTER IUT
Frame (2 octets) >
ignore
Frame (3 octets) >
ignore
I (good) >
< RR

2.5.56 I frame (P=1) received by IUT while in normal (NM) state
The tester sends an I frame with the poll bit set to 1. The IUT should send a SABM frame and migrate to the reset state.
TESTER IUT
I frame (P = 1) >
< SABM
>RS, start T200
Address test while in NM state.
This test checks that the IUT will only respond to its set address X.
TESTER IUT

I (good) address = X >
< RR
I (good) address = Y >
ignore
I (good) address = X >
< RR

2.5.57 DISC frame received by IUT while in normal (NM) state

The tester sends a DISC frame. The IUT is expected to respond with a UA frame and migrate to the DC state. The tester verifies this by sending an RR frame (P=1). The IUT should respond with a DISC frame (P=1) and migrate to the DS state.

TESTER IUT
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS

2.5.58 DISC frame received by IUT while in Reject sent (RJ) state

This test sends an I frame with a bad N(S) to force the IUT into the reject sent state. Another I frame with a bad N(S) is then sent to ensure the IUT is in the RJ state. The IUT should ignore this frame. A DISC frame is then sent by the tester and the IUT is expected to respond with a UA frame and migrate to the DC state. The tester then sends an RR frame (P=1) and the IUT is expected to respond with a DISC frame (P=1).

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS

2.5.59 DISC frame received by IUT while in Remote busy (RB) state

The tester sends an RNR frame to force the IUT into the remote busy (RB) state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to perform some action on the IUT that would normally send an I frame to the tester. This I frame should not be transmitted since the IUT is in the RB state.The tester then sends a DISC frame and the IUT is expected to respond with a UA frame and migrate to the DC state. The tester verifies this by sending an RR frame(P=1). The IUT is expected to send a DISC frame (P=1) and migrate to the DS state.

TESTER IUT
RNR >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)

>DS

2.5.60 DISC frame received by IUT while in Remote Busy and Reject state (RJ and RB) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a DISC frame and expects the IUT to respond with a UA frame and migrate to the DC state. The tester verifies this by sending a RR frame (P=1). The IUT is expected to respond with a DISC frame (P=1) and migrate to the DS state.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
I frame (bad N(S)) >
< REJ
>RJ and RB
DISC >
< UA
>DC
RR (P=1) >
< DISC (P=1)
>DS

2.5.61 SABM frame received by IUT while in normal (NM) state

The IUT should respond to the tester's SABM frame with a UA frame and remain in the normal state. The I frame is sent to verify that the IUT has not gone into the SABM reset state.

TESTER IUT
SABM >
< UA
I frame (good) >
< RR

2.5.62 SABM frame received by IUT while in reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. Another I frame with a bad N(S) is then sent to ensure the IUT is in the RJ state. The IUT should ignore this frame. A SABM frame is then sent by the tester and the IUT is expected to respond with a UA frame and migrate to the normal state. The tester then sends a good I frame. The IUT should respond with a RR frame. The tester then sends another I frame with a bad N(S) to verify that the IUT is in the Normal (NM) state. The IUT should respond with a REJ frame and migrate to the reject sent (RJ) state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
SABM >
< UA
>NM
I frame (good) >
< RR
I frame (bad N(S)) >

<- REJ
>RJ

2.5.63 SABM frame received by IUT while in remote busy (RB) state

The tester sends an RNR frame to force the IUT into the remote busy (RB) state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to perform some action on the IUT that would normally send an I frame to the tester. This I frame should not be transmitted since the IUT is in the RB state.The tester then sends a SABM frame and the IUT is expected to respond with a UA frame and migrate to the normal state. The IUTs link layer should flush any queued I frames and reset N(R) and N(S) = 0. The test operator is then prompted to have the IUT send an I frame. The tester verifies that N(R) and N(S) are 0 and responds with an RR frame followed by an I frame. The IUT is expected to respond with an RR frame.

TESTER IUT
RNR >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
SABM >
< UA
>NM
Attempt to send an I frame (e.g., Setup)
<-I frame (e.g., Setup)
RR >
I (good) >
< RR

2.5.64 SABM frame received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject sent state. The tester sends a SABM frame and expects the IUT to respond with a UA frame and migrate to the normal state. The IUTs link layer should then flush any queued frames and reset its N(R) and N(S) = 0. The test operator is then prompted to have the IUT send an I frame. The tester verifies that N(S) and N(R) are 0 and responds with a RR frame. The tester then verifies that the IUT is in Normal state by sending an I frame with a bad N(S). The IUT should respond with a REJ frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
I frame (bad N(S)) >
<- REJ
>RJ and RB
SABM >
<-UA
>NM
Attempt to send an I frame (e.g., Setup)
<- I frame (e.g., Setup)
RR >
I frame (bad N(S)) >

<- REJ
>RJ

2.5.65 SABM frame(P=1) received by IUT while in normal (NM) state

The IUT should respond to the tester's SABM frame (P=1) with a UA frame (F=1) and remain in the normal state. The I frame is sent to verify that the IUT has not gone into the SABM reset state.

TESTER IUT
SABM (P=1) >
< UA(F=1)
I frame (good) >
< RR

2.5.66 SABM frame(P=1) received by IUT while in reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. Another I frame with a bad N(S) is then sent to ensure the IUT is in the RJ state. The IUT should ignore this frame. A SABM frame (P=1)is then sent by the tester and the IUT is expected to respond with a UA frame(F=1) and migrate to the normal state. The tester then sends a good I frame. The IUT should respond with a RR frame. The tester then sends another I frame with a bad N(S) to verify that the IUT is in the Normal (NM) state. The IUT should respond with a REJ frame and migrate to the reject sent (RJ) state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
SABM (P=1) >
< UA(F=1)
>NM
I frame (good) >
< RR
I frame (bad N(S)) >

<- REJ
>RJ

2.5.67 SABM frame(P=1) received by IUT while in remote busy (RB) state

The tester sends an RNR frame to force the IUT into the remote busy (RB) state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to perform some action on the IUT that would normally send an I frame to the tester. This I frame should not be transmitted since the IUT is in the RB state.The tester then sends a SABM frame (P=1) and the IUT is expected to respond with a UA frame(F=1) and migrate to the normal state. The IUTs link layer should flush any queued I frames and reset N(R) and N(S) = 0. The test operator is then prompted to have the IUT send an I frame. The tester verifies that N(R) and N(S) are 0 and responds with an RR frame followed by an I frame. The IUT is expected to respond with an RR frame.

TESTER IUT
RNR >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
SABM (P=1) >
< UA(F=1)
>NM
Attempt to send an I frame (e.g., Setup)
< I frame (e.g., Setup)

RR >
I (good) >
< RR

2.5.68 SABM frame (P=1) received by IUT while in remote busy and reject sent (RB and RJ) state
The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject sent state. The tester sends a SABM (P=1) frame and expects the IUT to respond with a UA frame(F=1) and migrate to the normal state. The IUTs link layer should then flush any queued frames and reset its N(R) and N(S) = 0. The test operator is then prompted to have the IUT send an I frame. The tester verifies that N(S) and N(R) are 0 and responds with a RR frame. The tester then verifies that the IUT is in Normal state by sending an I frame with a bad N(S). The IUT should respond with a REJ frame.
TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
<- REJ
>RJ and RB
SABM(P=1) >
<UA(F=1)
>NM
A**ttempt to send an I frame (e.g., Setup)**
<- I frame (e.g., Setup)
RR >
I frame (bad N(S)) >
<- REJ
>RJ

2.5.69 UA frame received by IUT while in normal (NM) state
The tester sends a UA frame and the IUT is expected to respond with a SABM frame and migrate to the SABM reset state. The tester verifies that the IUT is in the SABM reset state by sending an I frame. The IUT is expected to ignore it. When T200 expires the IUT is expected to send a SABM frame(P=1).
TESTER IUT
UA >
< SABM
>RS, start T200
I frame (good) >
T200 expires
<- SABM(P=1)
restart T200

2.5.70 UA frame received by IUT while in reject sent (RJ) state
The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends another bad I frame to ensure the IUT is in the RJ state. The tester then sends a UA frame and the IUT is expected to send a SABM frame and migrate to the SABM reset state. An I frame is then sent by the tester to verify that the IUT has migrated to the SABM reset state. The IUT should ignore this frame and when T200 expires it should transmit another SABM frame.
TESTER IUT
I bad N(S) >

```
<       REJ
        >RJ
I bad N(S)      >
        ignore
UA      >
<       SABM
        >RS, start T200
I (good) >
        T200 expires
<       SABM
        restart T200
```

2.5.71 UA frame received by IUT while in remote busy (RB) state

The tester sends an RNR frame to force the IUT into the remote busy state. The tester then sends a UA frame and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then sends an I frame to verify that the IUT is in the SABM reset state.The IUT should ignore this frame. When T200 expires the IUT should transmit a SABM frame(P=1).

```
TESTER          IUT
RNR (P = 1)     >
        >RB
<       RR (F=1)
UA      >
<       SABM
        >RS, start T200
I (good) >
        T200 expires
<-      SABM(P=1)
        restart T200
```

2.5.72 UA frame received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends another I frame with a bad N(S) to verify the IUT is in RJ and RB state. The tester sends a UA frame and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

```
TESTER          IUT
RNR (P=0)       >
        >RB
        T200 expires
<       RR(P=1)
RNR(F=1)        >
        Attempt to send an I frame (e.g., Setup)
I frame (bad N(S))      >
<       REJ
        >RJ and RB
I frame (bad N(S))      >
 UA     >
<-      SABM
        >RS, start T200
I frame (good)  >
        ignore
        T200 expires
<-      SABM
```

restart T200

2.5.73 Good I frame received by IUT while in reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. The tester then sends another bad I frame to ensure the IUT is in the RJ state. The tester then sends a good I frame and expects the IUT to respond with a RR frame and migrate to the normal state. The tester then sends an I frame with a bad N(S) to verify that the IUT is in the normal state. The IUT should respond with a REJ frame.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
I frame (good) >
<- RR
>NM
I frame (bad N(S)) >
< REJ
>RJ

2.5.74 Good I frame received by IUT while in remote busy (RB) state

The tester sends a RNR frame to put the IUT in the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester then sends a good I frame and expects the IUT to respond with a RR frame and remain in remote busy state. The test operator should then attempt to send an I frame to the tester. This I frame should not be sent to the tester until the tester sends an RR frame to the IUT.

TESTER IUT
RNR >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
I frame (good) >
<- RR
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup.)**
RR (P=1) >
< RR (F=1)
>NM
< I frame (e.g., Setup)
start T200

2.5.75 Good I frame received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a good I frame and expects the IUT to respond with a RR frame and migrate to the RB state. The tester then verifies this state transition by waiting TWAIT seconds to ensure the IUT's queued I frame is not transmitted. When the IUT's T200 timer expires, the IUT is expected to transmit an RR(P=1). The tester then sends a RR(F=1) frame and expects the IUT to migrate to the NM state and transmit its queued I frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
I frame (bad N(S)) >
<- REJ
>RJ and RB
I frame (good) >
<- RR
>RB
T200 expires
< RR(P=1)
RR(F=1) >
>NM
< I frame (e.g., Setup)
start T200

2.5.76 I frame with bad N(S) received by IUT while in reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. The tester then sends multiple I frames with bad N(S)'s and expects the IUT to ignore them. The tester then sends a good I frame and expects the IUT to send an RR frame and migrate to the normal state.The tester then verifies that the IUT is in the normal state by sending an I frame with a bad N(S). The IUT should send a reject.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
I frame (bad N(S)) >
ignore
I frame (good) >
<- RR
>NM
I frame (bad N(S)) >
< REJ

2.5.77 I frame with bad N(S) received by IUT while in remote busy (RB) state and also in the remote busy and reject (RJ and RB) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends another I frame with a bad N(S) to make sure the tester is in the RJ and RB state. The IUT should ignore this frame. The tester sends a RR frame with the poll bit set and expects the IUT to respond with a RR frame with the final bit set and migrate to the RJ state. The IUT should now send the queued I frame. The tester then sends an I frame with a bad N(S) to ensure the IUT is in the reject state. The IUT should ignore this. The tester then sends a good I frame and expects the IUT to respond with a RR frame and migrate to the NM state.

TESTER IUT
RNR (P=0) >
>RB
T200 expires

< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
I frame (bad N(S)) >
< REJ
>RJ and RB
I frame (bad N(S)) >
ignore
T200 expires
< RR(P=1)
RNR(F=1) >

RR (P=1) >
<- RR (F=1)
> RJ
< I frame (e.g., Setup)
I frame (bad N(S)) >
ignore
I frame (good) >
<- RR
>NM

2.5.78 UI frame received by IUT while in Normal (NM) state

The tester sends a UI frame and the IUT is expected to pass the contents of the UI frame on to its layer 3 and remain in the NM state. The tester verifies this by sending an I frame. The IUT is expected to respond with an RR frame.

TESTER IUT
UI >
ignore
I frame (good) >
<- RR

2.5.79 UI frame received by IUT while in Reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. The tester then sends multiple I frames with bad N(S)'s and expects the IUT to ignore them. The tester then sends a good UI frame and expects the IUT to pass the contents of the UI frame on to its layer 3 and remain in the RJ state.The tester then verifies that the IUT is still in the RJ state by sending an I frame with a bad N(S). The IUT should ignore it.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
I frame (bad N(S)) >
ignore
UI frame >
I frame (bad N(S)) >
ignore

2.5.80 UI frame received by IUT while in Remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is prompted to perform an action on the IUT that would normally cause an I frame to be sent (e.g., Setup).The tester then sends a UI frame and the IUT should inform its layer 3 and send nothing back to the tester.

When the IUT's T200 timer expires, the IUT is expected to send an RR(P=1). The tester then sends an RR(F=1) frame. The IUT is expected to migrate to the normal state and transmit its queued I frame.

TESTER IUT
RNR (P=1) >
<- RR (F =1)
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
UI >
T200 expires
< RR(P=1)
RR(F=1) >
>NM
<- I frame (e.g., Setup)
start T200
RR >
stop T200

2.5.81 UI frame received by IUT while in Reject sent and Remote Busy (RJ and RB) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends another I frame with a bad N(S) to verify that the IUT is in RJ and RB state.The tester then sends a UI frame The IUT is expected to inform its layer 3 and remain in the RB and RJ state. When the IUT's T200 timer expires, the IUT is expected to send an RR(P=1).The tester then sends a RR (F=1) frame and expects the IUT to migrate to the RJ state and transmit its queued I frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
I frame (bad N(S)) >
T200 expires
< RR(P=1)
RNR(F=1) >
UI >
T200 expires
< RR(P=1)
RR(F=1) >
>RJ
< I frame (e.g., Setup)

2.5.82 RR frame (P=0) received by IUT while in normal (NM) state

The tester sends a RR frame with P=0 to the IUT and expects no response. The tester then sends a RR frame with the poll bit set and expects the IUT to respond with an RR frame with F = 1. The IUT should remain in the normal state.

TESTER IUT

RR (P=0) >
ignore
RR (P=1) >
< RR (F=1)

2.5.83 RR frame (P=0) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends a RR frame and the IUT is expected to ignore it and remain in the reject sent state. An I frame with a bad N(S) is then transmitted by the tester to ensure that the IUT is still in the reject sent state. The IUT is expected to ignore this frame.

TESTER IUT
I frame (bad N(S)) >
< REJ
> RJ
RR (P=0) >
ignore
I frame (bad N(S)) >
ignore

2.5.84 RR frame (P=0) received by IUT in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is prompted to perform an action on the IUT that would normally cause an I frame to be sent (e.g., Setup).The tester then sends an RR (P= 0) frame and the IUT should migrate to the normal state and transmit its queued I frame.

TESTER IUT
RNR (P=1) >
< RR (F =1)
>RB
T200 expires
< RR(P=1)
RNR(F=1) >

A**ttempt to send an I frame (e.g., Setup)**
RR (P=0) >
>NM
< I frame (e.g., Setup)
start T200
RR >
stop T200

2.5.85 RR frame(P=0) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester then sends another I frame with a bad N(S) to verify that the IUT is in the RJ and RB state. The IUT is expected to ignore this I frame. The tester sends a RR frame (P=0) and expects the IUT to migrate to the RJ state and transmit its queued I frame. The tester then sends another I frame with a bad N(S) to ensure the IUT is still in the reject sent state.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)

RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
I frame (bad N(S)) >
ignore
T200 expires
< RR(P=1)
RNR(F=1) >
RR (P=0) >
>RJ
<- I frame (e.g., Setup)
I frame (bad N(S)) >
ignore

2.5.86 RR frame (P=1) received by IUT in normal (NM) state
The tester sends an RR frame with the poll bit set. The IUT is expected to respond with an RR final = 1 frame.
TESTER IUT
RR (P=1) >
<- RR (F=1)

2.5.87 RR frame (P=1) received by IUT in reject sent (RJ) state
The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. The tester verifies this state change by sending another I frame with a bad N(S). The tester then sends a RR frame poll =1. The IUT is expected to respond with an RR frame final = 1 and remain in the RJ state. The tester verifies this by sending an I frame with a bad N(S). The IUT should ignore this frame.
TESTER IUT
I frame (bad N(S)) >
<- REJ
>RJ
I frame (bad N(S)) >
ignore
RR (P=1) >
< RR (F=1)
I frame (bad N(S)) >
ignore

2.5.88 RR frame (P=1) received by IUT in remote busy (RB) state
The tester forces the IUT into the remote busy state by sending an RNR frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). This test operator is then requested to make the IUT attempt to send an I frame (e.g., Setup). The tester then sends an RR poll = 1 frame. The IUT is expected to respond with a RR frame final = 1, migrate to the normal state and transmit its queued I frame.
TESTER IUT
RNR (P=1) >
< RR (F=1)
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
RR (P=1) >
< RR (F=1)
>NM

< I frame (e.g., Setup)
start T200
RR >
stop T200

2.5.89 RR frame(P=1) received by IUT while in remote busy and reject sent (RB and RJ) state
The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a RR frame (P=1) and expects the IUT to respond with a RR frame (F=1), migrate to the RJ state and transmit its queued I frame.
TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
I frame (bad N(S)) >
ignore
T200 expires
< RR(P=1)
RNR(F=1) >
RR (P=1) >
<- RR (F=1)
>RJ
<- I frame (e.g., Setup)
I frame (bad N(S)) >
ignore

2.5.90 RR frame (F=1) received by IUT while in normal (NM) state
The IUT sends an I frame to the tester. The tester waits T200 seconds for the IUT to send an RR frame(P=1).The tester responds with a RR frame (F=1). The IUT is expected to re-transmit the I frame.
TESTER IUT
Send an I frame **(e.g., Setup)**
< I frame (e.g., Setup)
start T200
T200 expires
< RR (P=1))
start T200
RR (F=1) >
N(R) = (N(S) of received I frame)- 1 stop T200
< I frame (e.g., Setup)
start T200

2.5.91 RR frame (F=1) received by IUT while in Reject Sent (RJ) state
The tester forces the IUT into the reject sent state. The IUT then sends an I frame. The tester waits T200 seconds for the IUT to send a RR frame (P=1). The tester responds with an RR frame (F=1). The IUT should re-transmit the I frame.
TESTER IUT
I frame (bad N(S) >

< REJ
>RJ
Send an I frame **(e.g., Setup)**
< I frame (e.g., Setup)
start T200
T200 expires
< RR (P=1))
start T200
RR (F=1) >
N(R) = (N(S) of received I frame)- 1 stop T200
< I frame (e.g., Setup)
start T200

2.5.92 RR frame (F=1) received by IUT while in Remote busy (RB) state

The IUT sends an I frame. The tester responds with an RNR frame (which does not acknowledge the I frame) forcing the IUT into the remote busy state. The tester then waits T200 seconds for the IUT to send a RR frame(P=1). The tester responds with a RR frame (F=1).This forces the IUT into Normal state. The IUT should then re-transmit the queued I frame. The tester responds with a RR frame.

TESTER IUT
Send an I frame **(e.g., Setup)**
<- I frame (e.g., Setup)
start T200
RNR (P=0) >
N(R) = (N(S) of received I frame)- 1 stop T200
T200 expires
< RR (P=1))
start T200
RR(F=1) >
>NM
< I frame (e.g., Setup)
restart T200
RR (F=0) >
stop T200

2.5.93 RR frame (F=1) received by IUT while in the Reject sent and Remote busy (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the RB and RJ state. The tester then waits T200 seconds for the IUT to send a RR frame(P=1).The tester sends a RR frame (F=1) and expects the IUT to migrate to the reject state and transmit the queued I frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
T200 expires
< RR (P=1))
start T200
RR (F=1) >

>RJ, stop T200
<- I frame (e.g., Setup)
start T200
RR >
stop T200

2.5.94 REJ frame (P=0) received by IUT in normal (NM) state

The IUT sends an I frame to the tester. The tester responds with a REJ frame. The IUT is expected to re-transmit the I frame and start T200. The tester does not respond to this I frame. T200 expires and the IUT should send an RR frame(P=1). The tester will respond with an RR frame(F=1). The IUT should then re-transmit the I frame. This time the tester responds with an RR frame.

TESTER IUT
Send an I frame **(e.g., Setup)**
< I frame (e.g., Setup)
start T200
REJ (P=0) >
N(R) = N(S) of received I framestop T200
<- I frame (e.g., Setup)
start T200
T200 expires
<- RR (P=1)
start T200
RR (F=1) >
N(R) -1 of received frame stop T200
<- I frame (e.g., Setup)
start T200
RR (F=0) >
stop T200

2.5.95 REJ frame (P=0) received by IUT in reject sent (RJ) state

The tester forces the IUT into the reject sent state. The IUT then sends an I frame and the tester rejects it. The IUT re-transmits the I frame and the tester does not respond. The IUT's T200 expires and the IUT is expected to send a RR frame(P=1). The tester responds with an RR frame(F=1). The IUT is then expected to re-transmit the I frame. The tester then sends another I frame with a bad N(S) to ensure the IUT is still in the RJ state. The tester then sends a good I frame and the IUT is expected to respond with an RR frame and migrate to the normal state. Note that the I frame sent by the tester acknowledges the outstanding I frame that the IUT sent.

TESTER IUT
I frame (bad N(S) >
< REJ
>RJ
Send an I frame **(e.g., Setup)**
<- I frame (e.g., Setup)
start T200
REJ (P=0) >
N(R)=N(S) of the received I frame stop T200
< frame (e.g., Setup)
restart T200
T200 expires
<- RR (P=1)
start T200
RR (F=1) >
stop T200
< I frame (e.g., Setup)
start T200
I frame (bad N(S) >

ignore
I frame (good) >
stop T200
<- RR
>NM

2.5.96 REJ frame (P=0) received by IUT in remote busy (RB) state

The IUT sends an I frame. The tester then forces the IUT into the remote busy state by sending an RNR frame. Note however this RNR frame does not acknowledge the I frame sent by the IUT. The tester then rejects the IUT's I frame. The IUT should re-transmit the I frame, start T200 and migrate to the normal state. The tester lets T200 expire without acknowledging the I frame. The IUT should then send an RR frame(P=1). The tester responds with a RR frame(F=1).

TESTER IUT
Send an I frame **(e.g., Setup)**
<- I frame (e.g., Setup)
start T200
RNR (P=0) >
N(R) =(N(S) of received I frame -1) >RB, restart T200
REJ (P=0) >
N(R)=N(S) of the received I frame
<- I frame (e.g., Setup)
>NM, start T200
T200 expires
<- RR (P=1)
start T200
RR (F=1) >
stop T200

2.5.97 RNR frame (P=0) received by IUT in normal (NM) state

The tester sends an RNR frame to the IUT. The IUT should migrate to the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is prompted to have the IUT send an I frame. No message should be sent to the tester. The tester then clears the busy condition by sending an RR frame to the IUT. The IUT responds with a RR frame followed by the queued I frame. The tester responds with an RR frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
RR (P=1) >
< RR(F=1)
>NM
< I frame (e.g., Setup)
start T200
RR (F=0) >
stop T200

2.5.98 RNR frame (P=0)received by IUT in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester sends a RNR (P=0) frame to the IUT. The IUT should migrate to the reject sent and remote busy state. The tester then sends an I frame with a bad N(S) to verify that the IUT is in the RJ and RB state. The test operator is then prompted to attempt to have the IUT send an I frame. The I frame should not be sent by

the IUT as its in the RB and RJ state. The tester then sends a good I frame and expects the IUT to respond with an RR frame and migrate to the RB state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
RNR (P=0) >
>RJ and RB
T200 expires
< RR(P=1)
RNR(F=1) >
I frame (bad N(S)) >
ignore
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (good) >
<- RR
>RB

2.5.99 RNR frame(P=0) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a RNR frame (P=0) and expects the IUT to ignore it. When the IUT's T200 timer expires, the IUT sends a RR frame(P=1) and the tester responds with a RR frame (F=1). This causes the IUT to migrate to the Reject sent state and transmit the queued I frame. The tester sends another I frame with a bad N(S). The IUT should ignore it.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
RNR (P=0) >
ignore
T200 expires
< RR(P=1)
RR(F=1) >
>RJ
<- I frame (e.g., Setup)
start T200
I frame (bad N(S)) >
ignore

2.5.100 RNR frame (P=1) received by IUT in normal (NM) state

The tester forces the IUT into the remote busy state by sending an RNR frame(P=1). The IUT is expected to respond with a RR frame (F=1). Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator should now attempt to have the IUT send an I

frame. The tester then sends another RNR frame with the poll bit set and the IUT is expected to respond with an RR frame (F=1). The tester now clears the busy condition by sending an RR frame to the IUT. The IUT should respond with an RR frame followed by the I frame (Setup) and migrate to the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
RNR (P=1) >
<- RR (F=1)
RR (P=1) >
<- RR (F=1)
>NM
< I frame (e.g., Setup)
start T200
RR >
stop T200

2.5.101 RNR frame (P=1) received by IUT in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends a RNR (P=1) frame to the IUT. The IUT should respond with a RR frame (F=1) and migrate to the reject sent and remote busy state. The tester verifies this by first sending an I frame with a bad N(S) to which it expects no response, and then by prompting the test operator to attempt to send an I frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester then sends a good I frame and expects the IUT to respond with an RR frame and migrate to the RB state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
RNR (P=1) >
>RJ and RB
<- RR (F=1)
I frame (bad N(S)) >
ignore
T200 expires
< RR(P=1)
RNR(F=1) >
Attempt to send an I frame (e.g., Setup)
I frame (good) >
<- RR
>RB

2.5.102 RNR frame (P=0) and RNR frame (P=1) received by IUT in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is prompted to attempt to have the IUT send an I frame (e.g., Setup) and the tester verifies there is no response.The tester then sends another RNR frame (P=0) and the IUT should ignore the message. The tester then sends a RNR frame (P=1) and the IUT should respond with an RNR frame (F=1) The tester then sends an RR frame to put the IUT back to the normal state. The IUT should respond with an RR frame followed by the queued I frame.

TESTER IUT
RNR (P=0) >

```
          >RB
          T200 expires
<         RR(P=1)
RNR(F=1)        >
          Attempt to send an I frame (e.g.,. Setup)
RNR (P=0)       >
          ignore
          T200 expires
<         RR(P=1)
RNR(F=1)        >
RNR (P=1)       >
<-        RR (F=1)
RR (P=1)        >
<-        RR (F=1)
          >NM
<         I frame (e.g., Setup)
          start T200
RR        >
          stop T200
```

2.5.103 RNR frame(P=1) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a RNR frame (P=1) and expects the IUT to respond with a RR frame (F=1). When the IUT's T200 timer expires , the IUT is expected to send an RR(P=1) . The tester responds with an RR(F=1). The IUT is expected to migrate to the RJ state and transmit the queued I frame. The tester acknowledges with an RR and then sends another I frame with a bad N(S) to ensue the IUT is still in the reject sent state. The IUT should ignore this frame.

```
TESTER          IUT
 RNR (P=0)      >
          >RB
          T200 expires
<         RR(P=1)
RNR(F=1)        >
          A
```
ttempt to send an I frame (e.g., Setup)
```
I frame (bad N(S))          >
<         REJ
          >RJ and RB
RNR (P=1)       >
<-        RR (F=1)
          T200 expires
<         RR(P=1)
RR(F=1)         >
          >RJ
<         I frame (e.g., Setup)
          start T200
RR        >
          stop T200
I frame (bad N(S))          >
          ignore
```

2.5.104 RNR frame (F=1) received by IUT while in the Normal (NM) state

The IUT sends an I frame. The tester does not respond to this I frame. T200 expires and the IUT is expected to send a RNR frame(P=1). The tester responds with an RNR frame(F=1). The IUT is expected to migrate to the RB state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is prompted to have the IUT send an I frame. This I frame should not be sent as the IUT is in the RB state. The tester then sends an RR frame(P=1). The IUT is expected to respond with an RR frame (F=1),migrate to the NM state and transmit the queued I frame.

TESTER IUT
Send an I frame **(e.g., Setup)**
<- I frame (e.g., Setup)
start T200
T200 expires
<- RR (P=1)
start T200
RNR (F=1) >
restart T200
>RB
T200 expires
< RR(P=1)
RNR(F=1) >
A**ttempt to send an I frame (e.g., Setup)**
RR (P=1) >
<- RR (F=1)
>NM
< I frame (e.g., Setup)
start T200
RR >
stop T200

2.5.105 RNR frame (F=1) received by IUT while in the Reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The test operator is then prompted to have the IUT send an I frame. The tester does not respond to this I frame. T200 expires and the IUT is expected to send a RR frame(P=1). The tester responds with a RNR frame (F=1). The IUT is expected to send nothing to the tester and migrate to the RJ and RB state. The tester verifies this by sending an I frame with a bad N(S). The IUT should ignore it.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
Send an I frame **(e.g., Setup)**
<- I frame (e.g., Setup)
start T200
T200 expires
<- RR (P=1)
start T200
RNR (F=1) >
restart T200, >RJ and RB
I frame (bad N(S)) >
ignore
T200 expires
< RR(P=1)
RR(F=1) >
>RJ

2.5.106 RNR frame (F=1) received by IUT while in the Remote busy (RB)) state

The tester forces the IUT into the remote busy state by sending an RNR frame. The test operator is prompted to attempt to have the IUT send an I frame (e.g., Setup) and the tester verifies there is no

response. When T200 expires the IUT is expected to send an RR frame(P=1). The tester will respond with an RNR frame(F=1). The IUT should restart T200. The tester waits for T200 to expire and the IUT is then expected to send an RR frame(P=1). This time the tester responds with an RR frame(F=1). The IUT should migrate to the Normal state and transmit the I frame. The tester responds with an RR frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g.,. Setup)
T200 expires
<- RR (P=1)
start T200
RNR (F=1) >
restart T200
T200 expires
<- RR (P=1)
start T200
RR (F=1) >
>NM,stop T200
<- I frame (e.g., Setup)
start T200
RR (F=0) >
stop T200

2.5.107 RNR frame (F=1) received by IUT while in the Reject sent and Remote busy (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state.Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester then waits T200 seconds for the IUT to transmit a RR frame(P=1). The tester responds with an RNR frame(F=1). The tester again waits T200 seconds for the IUT to send another RR frame(P=1). This time the tester sends an RR frame(F=1). The IUT should migrate to the Reject sent state and transmit the queued I frame. The tester responds with a RR frame followed by another I frame with a bad N(S). The IUT should ignore this frame as it is in the reject sent state.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
T200 expires
<- RR (P=1)
start T200
RNR (F=1) >
restart T200
T200 expires
<- RR (P=1)
start T200
RR (F=1) >
>RJ, stop T200
<- I frame (e.g., Setup)

restart T200
RR (F=0) >
stop T200
I frame (bad N(S)) >
ignore

2.5.108 T200 expires (RC< N200) while in normal (NM) state

The test operator is prompted to have the IUT send an I frame (e.g., Setup). This causes an I frame to be sent to the tester. The tester does not respond to this I frame. T200 expires and the IUT is expected to send a RR frame(P=1). The tester responds with a RR frame(F=1) and the IUT is expected to re-transmit the I frame. This time the tester responds with an RR frame.

TESTER IUT
Send an I frame **(e.g., Setup)**
<- I frame (good)
start T200
T200 expires
<- RR (P=1)
restart T200
RR (F=1) >
stop T200
<- I frame (good)
start T200
RR >
stop T200

2.5.109 T200 expires (RC< N200) while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S) to the IUT. The test operator invokes the IUT to send an I frame to the tester. The tester does not respond to the I frame, so after T200 seconds the IUT is expected to send a RR frame(P=1). The tester responds with an RR frame(F=1) and the IUT is expected to re-transmit the I frame. The tester then sends a good I frame to the IUT. The IUT is expected to acknowledge it with an RR frame.

TESTER IUT
I frame (bad N((S)) >
< REJ
>RJ
Send an I frame **(e.g., Setup)**
< I frame
start T200
T200 expires (RC< N200)
<- RR (P=1)
restart T200
RR (F=1) >
N(R) = (N(S) of received I frame)- 1 stop T200
< I frame
restart T200
RR >
stop T200
I frame (good) >
>NM
<- RR

2.5.110 T200 expires (RC< N200) while in Remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame(P=0). After T200 expires the IUT is expected to send an RR frame(P=1). The tester responds with an RR frame(F=1). The IUT should migrate to the Normal state. This is verified by prompting the test operator to have the IUT send an I frame.

TESTER IUT

RNR (P=0) >
N(R) 1 less then N(S) of the received I frame >RB
T200 expires
<- RR (P=1)
restart T200
RR (F=1) >
>NM,stop T200
Send an I frame**(e.g.,SETUP)**
< I frame
start T200
RR >
stop T200

2.5.111 T200 expires (RC< N200) while in Reject sent and Remote busy (RJ and RB) state
The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester does not respond to the reject so when T200 expires, the IUT sends an RR frame (P=1).
TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB, **start T200**
T200 expires
<- RR (P=1)

2.5.112 T200 expires (RC = N200) while in normal (NM) state
The test operator is prompted to have the IUT send an I frame. This causes an I frame to be transmitted to the tester. The tester does not respond, so after T200 seconds the IUT the IUT sends a RR frame(P=1). The tester still does not respond so after T200 seconds the IUT re-transmits the RR frame(P=1). This continues until RC = N200. At this point the IUT is expected to send a SABM frame (P=1) and migrate to the SABM reset state.
TESTER IUT
Send an I frame **(e.g., Setup)**
< I frame (e.g., Setup)
start T200
T200 expires (RC< N200)
< RR(P=1)
restart T200
T200 expires (T200< N200)
< RR(P=1)
restart T200
T200 expires (RC = N200)
<- SABM (P=1)
>RS start T200

2.5.113 T200 expires (RC = N200) while in reject sent (RJ) state
The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The test operator is prompted to have the IUT send an I frame. This causes an I frame to be transmitted to the tester. The

tester does not respond, so after T200 seconds the IUT re-transmits the I frame. The tester still does not respond so after T200 seconds the IUT sends a RR frame(P=1). This continues until RC = N200. At this point the IUT is expected to send a SABM frame (P=1) and migrate to the SABM reset state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
Send an I frame **(e.g., Setup)**
< I frame
start T200
T200 expires (RC< N200)
< RR(P=1)
restart T200
T 200 expires (RC= N200)
<- SABM (P=1)
>RS, start T200

2.5.114 T200 expires (RC = N200) while IUT in Remote busy (RB) state

The IUT sends an I frame to the tester. The tester sends an RNR frame (P=0) with a sequence number 1 less than the one received in the I frame. This forces the IUT into the RB state but does not acknowledge the I frame. When T200 expires the IUT is expected to send a RR frame(P=1). The tester does not respond. When T200 expires again, the IUT is expected to send another RR frame(P=1). When T200 expires N200 times the IUT is expected to send a SABM frame (P=1) and migrate to the RS state.

TESTER IUT
Send an I frame **(e.g., Setup)**
< I frame
start T200
RNR (P=0) >
N(R) 1 less then N(S) of the received I frame >RB
T200 expires
<- RR (P=1)
T200 expires
<- RR (P=1)
T200 expires
<- RR (P=1)
T200 expires (RC =N200)
<- SABM (P=1)
>RS, start T200

2.5.115 T200 expires (RC = N200) while IUT in Reject sent and Remote busy (RJ and RB) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester does not respond to the REJ frame so after T200 expires the IUT is expected to send a RR frame(P=1). The tester does not respond. When T200 expires again the IUT is expected to transmit another RR frame(P=1). This continues until T200 expires N200 times. At this point, the IUT is expected to send a SABM frame (P=1) and migrate to the SABM reset state.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**

I frame (bad N(S)) >
< REJ
>RJ and RB, **start T200**
T200 expires (RC< N200)
< RR(P=1)
restart T200
T200 expires (RC< N200)
< RR(P=1)
restart T200
T200 expires (RC = N200)
<- SABM (P=1)
>RS,start T200

2.5.116 I frame (bad N(R)) received by IUT while in normal (NM) state

The tester sends an I frame with a bad N(R) to the IUT. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. When the IUT's T200 timer expires, the IUT is expected to transmit a SABM (P=1). The tester then sends a UA frame (F=1) to put the IUT in the normal state.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
< SABM(P=1)
UA(F=1) >
>NM, stop T200

2.5.117 I frame (bad N(R)) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends another I frame with a bad N(S) to verify that the IUT is in the RJ state. The tester then sends an I frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA frame to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
I frame (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.118 I frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to attempt to force the IUT to send an I frame. The tester then sends an I frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA frame putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
Attempt to send an I frame (e.g., Setup)
I frame (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.119 I frame (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state
The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a I frame with a bad N(R) and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.
TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
Attempt to send an I frame (e.g., Setup)
I frame (bad N(S)) >
< REJ
>RJ and RB
I frame (bad N(R)) >
<- -SABM
>RS, start T200
I frame (good) >
ignore
T200 expires
<- -SABM(P=1)
restart T200

2.5.120 RR frame (bad N(R)) received by IUT while in normal (NM) state
The tester sends an RR frame with a bad N(R) to the IUT. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA frame to put the IUT in the normal state.
TESTER IUT
RR (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >

>NM, stop T200

2.5.121 RR frame (bad N(R)) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then verifies the IUT is in the RJ state by sending another I frame with a bad N(S). The tester then sends an RR frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA frame to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
RR (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.122 RR frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester then verifies the IUT is in the RB state by prompting the test operator to have the IUT send an I frame The tester then sends an RR frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA frame putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
RR (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.123 RR frame (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a RR frame with a bad N(R) and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >

```
          >RB
          T200 expires
<-        RR (P=1)
RNR (F=1)         >
          A
```
ttempt to send an I frame (e.g., Setup)
```
I frame (bad N(S))          >
<         REJ
          >RJ and RB
RR (bad N(R))     >
<         SABM
          >RS, start T200
I frame (good)    >
          ignore
          T200 expires
<         SABM
          restart T200
```

2.5.124 RNR frame (bad N(R)) received by IUT while in normal (NM) state

The tester sends an RNR frame with a bad N(R) to the IUT. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA frame to put the IUT in the normal state.

```
TESTER            IUT
RNR (bad N(R))    >
<         SABM
          >RS, start T200
 I frame (good)   >
          ignore
UA        >
          >NM, stop T200
```

2.5.125 RNR frame (bad N(R)) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S) and then verifies the state change by sending another I frame with a bad N(S). The tester then sends an RNR frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA frame to put the IUT in the normal state.

```
TESTER            IUT
 I frame (bad N(S))          >
<         REJ
          >RJ
I frame (bad N(S))          >
          ignore
RNR (bad N(R))    >
<         SABM
          >RS, start T200
I frame (good)    >
          ignore
UA        >
          >NM, stop T200
```

2.5.126 RNR frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame and then verifies the state transition by having the test operator attempt to have the IUT send an I frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester then sends an RNR frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the

SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA frame putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
RNR (bad N(R)) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.127 RNR frame (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a RNR frame with a bad N(R) and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
RNR (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
T200 expires
< SABM
restart T200

2.5.128 REJ frame (bad N(R)) received by IUT while in normal (NM) state

The tester sends an REJ frame with a bad N(R) to the IUT. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. When the IUT's T200 expires, the IUT is expected to send a SABM(P=1).The tester then sends a UA(F=1) frame to put the IUT in the normal state.

TESTER IUT
REJ (bad N(R)) >
< SABM
>RS, start T200

I frame (good) >
ignore
T200 expires
< SABM(P=1)
UA(F=1) >
>NM, stop T200

2.5.129 REJ frame (bad N(R)) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). It then verifies the state transition by sending another I frame with a bad N(S). The tester then sends an REJ frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. The tester sends a UA frame to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
REJ (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.130 REJ frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame and then verifies the state transition by prompting the test operator to attempt to force the IUT to send an I frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester then sends an REJ frame with a bad N(R). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA frame putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
REJ (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.131 REJ frame (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The

tester sends a REJ frame with a bad N(R) and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
REJ (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
T200 expires
< SABM
restart T200

2.5.132 Bad command received by IUT while in normal (NM) state

The tester sends a bad command (i.e., REJ(P=1) frame to the IUT. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA frame to put the IUT in the normal state. The tester will then repeat this test over and over until all combinations of bad commands have been tested.

TESTER IUT
bad command >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200
I frame >
<- RR

2.5.133 Bad command received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). It then verifies the state transition by sending another I frame with a bad N(S).The tester then sends an bad command frame. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA frame to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
bad command >
< SABM
>RS, start T200
I frame (good) >
ignore

UA >
>NM, stop T200

2.5.134 Bad command received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame and then verifies this transition by prompting the test operator to attempt to have the IUT send an I frame. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester then sends a bad command frame. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA frame putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
bad command >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.135 Bad command received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends a bad command and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
bad command >
< SABM
>RS, start T200
I frame (good) >
ignore
T200 expires
< SABM
restart T200

2.5.136 Bad response received by IUT while in normal (NM) state

The tester sends a bad response (FRMR as its not supported by MiLAP) frame to the IUT. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA frame to put the IUT in the normal state.

TESTER IUT
FRMR >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.137 Bad response received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then verifies this state transition by sending another I frame with a bad N(S). The IUT should ignore this frame. The tester then sends an bad response frame (FRMR as its not supported by MiLAP). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA frame to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
FRMR >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.138 Bad response received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame.Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester verifies this state change by prompting the test operator to attempt to have the IUT send an I frame. The tester then sends a bad response frame (FRMR as it's not supported by MiLAP). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA frame putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
FRMR >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.139 Unsolicited F bit received by IUT while in normal (NM) state

The tester sends an unsolicited F bit (UA (F=1))to the IUT. The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA frame to put the IUT in the normal state.

TESTER IUT
UA (F=1) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

2.5.140 Unsolicited F bit received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester verifies this state transition by sending another I frame with a bad N(S). The IUT should ignore this frame. The tester then sends an unsolicited F bit (REJ frame (F=1)). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA frame to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
REJ (F=1) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, T200

2.5.141 Unsolicited F bit received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame.Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The tester verifies this state transition by prompting the test operator to attempt to have the IUT send an I frame. The tester then sends an unsolicited F bit (REJ frame (F=1)). The IUT is expected to send a SABM frame and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA frame putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
Attempt to send an I frame (e.g., Setup)
REJ (F=1) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
> NM, stop T200

2.5.142 Unsolicited F bit received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state.Note that if the IUT's T200 timer expires the IUT will send an RR(P=1). The tester will respond with an RNR(F=1). The test operator is requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester sends an unsolicited F bit (REJ frame (F=1)) and expects the IUT to respond with a SABM frame and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
T200 expires
<- RR (P=1)
RNR (F=1) >
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
T200 expires
<- RR (P=1)
RNR (F=1) >
REJ (F=1) >
< SABM
>RS, start T200
I frame (good) >
ignore
T200 expires
< SABM(P=1)
restart T200

2.5.143 No Buffers available while IUT in Normal (NM) state

The tester rapidly sends I frames to the IUT. This forces the IUT into local busy state. The IUT is expected to send a RNR frame to the tester. When the IUT frees up a buffer it then is expected to clear the busy condition by sending a RR frame to the tester.

TESTER IUT
Send lots of I frames as quickly as possible
I frame (good) >
ignore
<- RNR
>LB
<- RR (P=1)
>NM

2.5.144 No Buffers available while IUT in Reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester verifies this state transition by sending another I frame with a bad N(S). The IUT should ignore this frame. The tester rapidly sends I frames with bad N(S)'s until the IUT sends an RNR frame and migrates to the reject sent and local busy state. When the IUT frees up a buffer, it should clear the busy condition by sending a RR frame and returns to the reject sent state. The tester verifies this by sending a I frame with a bad N(S). The IUT should ignore this frame. The tester then sends a good I frame and the IUT is expected to send an RR frame and migrate to the normal state.

TESTER IUT
I frame (bad N(S)) >
< REJ

>RJ
send lots of these
I frame (bad N(S)) >
ignore
<- RNR
>RJ and LB, start T200
<- RR
>RJ, start T200
I frame (bad N(S)) >
ignore
I frame (good) >
<- RR
>NM

2.5.145 No Buffers available while IUT in Remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. The tester verifies this state transition by prompting the test operator to attempt to have the IUT send an I frame. The tester rapidly sends I frames to the IUT. When the IUT runs out of buffers, it sends an RNR frame to the tester and migrates to the RB and LB state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
A**ttempt to send an I frame (e.g., Setup)**
I frame (good) (lots of these) >
<- RNR
>LB and RB
<- RR (P=1)
>RB
RR (P=1) >
< RR (F=1)
> NM

2.5.146 No Buffers available while IUT in Remote busy and Reject sent (RB and RJ) state

The tester sends a RNR frame to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ frame and migrate to the remote busy and reject state. The tester rapidly sends UI frames to the IUT. When the IUT runs out of buffers it is expected to transmit a RNR frame and migrate to the RB,LB and RJ state.

TESTER IUT
RNR (P=0) >
>RB
A**ttempt to send an I frame (e.g., Setup)**
I frame (bad N(S)) >
< REJ
>RJ and RB
Send lots of UI frames
UI >
<- RNR

2.5.147 Modulus 8 Verification of IUT's N(R)

The tester sends I frames with incrementing sequence numbers and expects the IUT to respond with RR frames with incrementing sequence numbers. After a sequence number of 7 the numbers should reset to 0 and begin incrementing again. Depending on the IUT, there may be I frames sent to the tester after the

SABM/UA exchange (e.g., set id). The number of I frames sent by an IUT is a PIXIT option. Thus N(R) is shown as X for the tester side in the diagram below.

```
TESTER          IUT
SABM  >
<         UA
          set V(R,S) = 0
I frame (good)    >
N(S)=0, N(R) = X
<-        RR
          N(R) = 1
I frame (good)    >
N(S)=1, N(R) = X
<-        RR
          N(R) = 2
I frame (good)    >
N(S)=2, N(R) = X
<-        RR
          N(R) = 3
I frame (good)    >
N(S)=3, N(R) = X
<-        RR
          N(R) = 4
I frame (good)    >
N(S)=4, N(R) = X
<-        RR
          N(R) = 5
I frame (good)    >
N(S)=5, N(R) = 0
<-        RR
          N(R) = 6
I frame (good)    >
N(S)=6, N(R) = X
<-        RR
          N(R) = 7
I frame (good)    >
N(S)=7, N(R) = X
<-        RR
          N(R) = 0
I frame (good)    >
          N(S)=0, N(R) = X
<-        RR
          N(R) = 1
```

2.5.148 Modulus 8 Verification of IUT's N(S)

The IUT sends I frames and the tester verifies that the sequence numbers increment and then reset to 0 after a sequence number of 7.

```
TESTER          IUT
SABM  >
<         UA
           set V(R,S) = 0
<-        I frame (good)
          N(S)=0, N(R) = 0, start T200
RR        >
N(R) = 1          stop T200
<-        I frame (good)
          N(S)=1, N(R) = 0, start T200
```

RR >
N(R) = 2 stop T200
<- I frame (good)
N(S)=2, N(R) = 0, start T200
RR >
N(R) = 3 stop T200
<- I frame (good)
N(S)=3, N(R) = 0, start T200
RR >
N(R) = 4 stop T200
<- I frame (good)
N(S)=4, N(R) = 0, start T200
RR >
N(R) = 5 stop T200
<- I frame (good)
N(S)=5, N(R) = 0, start T200
RR >
N(R) = 6 stop T200
<- I frame (good)
N(S)=6, N(R) = 0, start T200
RR >
N(R) = 7 stop T200
<- I frame (good)
N(S)=7, N(R) = 0, start T200
RR >
N(R) = 0 stop T200
<- I frame (good)
N(S)=0, N(R) = 0, start T200
RR >
N(R) = 1 stop T200

2.5.149 Transmit Window size test

The IUT sends k -I frames (within T200 seconds) before the tester sends an RR. The RR sent by the tester will acknowlegde the k-I frames sent by the IUT. The example below assumes the IUT has a transmit window size of 3.

TESTER IUT
< I frame (good)
N(S)=0, N(R) = 0, start T200
< I frame (good)
N(S)=1, N(R) = 0
< I frame (good)
N(S)=2, N(R) = 0
RR >
N(R) = 3 stop T200
< I frame (good)
N(S)=3, N(R) = 0, start T200
RR >
N(R) = 4 stop T200

2.6 Optional MiLINK Tests

2.6.1Address Negotiation (Normal procedure)

This test ensures the IUT can successfully negotiate a new address (TEI) with a MiLINK peripheral. The tester will begin sending SABM frames using address $1E. The IUT is expected to send a UA frame. The tester will then send a Report Set ID I frame to the IUT. The IUT is expected to acknowledge the I frame and send a Define Channel and MiLINK Parameters ($CF) I frame to the tester, instructing it to use address

x. The tester will acknowledge this I frame and enter the SABM reset state. This time though it will use the newly assigned address x. The IUT is expected to respond with a UA frame. The tester will again send a Report Set ID I frame. The IUT is expected to acknowledge it and send another Define Channel and MiLINK Parameters ($CF) I frame to the tester. The tester will then acknowledge the I frame.

TESTER IUT
SABM (address = 1E) >
start T200, T210
< UA (address = 1E)
I frame (report set ID, address = 1E) >
< RR (address = 1E)
< I frame $CF (change to address x)
RR (address = 1E) >
SABM (address = x) >
< UA (address = x)
I frame (report set ID, address = x) >
< RR (address = x)
< I frame $CF (keep address, change to B2)
RR (address = x) >

2.6..2 Address Negotiation (T 210 expires)

T210 expiry not currently implemented on any of the MiLINK devices.

3.0 Abstract test suite for MiLAP-S D-channel

3.1 Preamble

The following preamble will be used for this ATS. This will ensure that the IUT is in the normal (NM) state.

TESTER IUT
I frame (bad N(R)) >
< SABM
UA >
Depending on the IUT, I frames may be transmitted to the tester as shown below. This settings in the PIXIT menu will determine if the tester will expect these I frames.
< I frame (report set ID, hook switch, etc.)
RR >

3.2 Postamble

The following postamble will be used for this ATS.

TESTER IUT
I frame (bad N(R)) >
< SABM

3.3 PIXIT proforma

Set /PBX Type
Date of testing
Limitations of IUT
Tester serial number
Test software version number
Tester operator's name
Can IUT send SABM on demand?
Can the IUT be forced to send an I frame on demand?
Can the IUT be forced to send more than 1 I frame on demand?
Which I frame will IUT send for these tests?
Which I frame can the tester send to the IUT (programmable field of I frame)?
What I frames (if any) does the tester send upon receiving a UA from the tester?.

3.4 PICS proforma

Value of T200
Address of IUT
Value of k (transmit window)
Value of k (receive window)
N(201) maximum octets in an I frame.
N(200) maximum re-transmissions.
Does IUT support Go-Ahead procedures?
Does IUT support the broadcast address?

3.5 Test cases

The following test cases are written using X- Notation. The Conformance tester sends the commands shown by the right arrows and the IUT sends the messages shown by the left arrows. State transitions are shown with a > symbol. This means the IUT is expected to migrate to the state shown after the > symbol. Note that although not shown, the preamble is executed at the beginning of each test and the postamble at the end of each test. The preamble puts the IUT into the normal (NM) state. The postamble puts the IUT into SABM reset (RS) state. Even if the test case fails the postamble will be executed. As well, Go-Ahead will be sent by the tester for each request made by the IUT.
The term "ignore" means the tester will wait TWAIT seconds to be sure the IUT does not send a frame. For tests which require the IUT to send I frames (e.g., off hook), the tester will wait for the test operator to perform the necessary function on the IUT. The exact time it will wait is an implementation issue.

3.5.1 HDLC capability (housekeeping bit) test

This test verifies that the IUT will upon initialization, check the C-channel housekeeping bit (HK) in order to determine whether or not to use go-ahead procedures. If the HK bit is 0, go-ahead procedures shall be used. If the HK bit is 1, go-ahead procedures shall not be used (the IUT can transmit a frame t any time). The test operator is first requested to plug in the IUT. The tester will proceed to send idle ones or flags with the C-channel HK bit set to 0. The IUT should send an idle ones pattern and then switch this to continuous flags. The tester will respond with a go-ahead. The IUT is then expected to send a SABM and then return to sending the idle ones pattern. The tester will then prompt the test operator to unplug the IUT and then plug it back in. This time the tester will set the HK bit to 1. The IUT should send a SABM frame without first requesting go-ahead.

TESTER IUT
Plug in IUT
idle ones or flags (HK=0) >
<- Idle ones pattern
<- Continuous flags
Go-Ahead (7F, 0) >
<- SABM
>RS, start T200
<- Idle ones pattern
Unplug IUT
Plug in IUT
idle ones or flags (HK=1) >
<- SABM
>RS, start T200

3.5.2 Go-ahead procedure test

This test ensures the physical layer Go-Ahead procedure is working. The test operator plugs in the IUT. The IUT should send idle ones and then switch to continuous flags. When the tester receives the continuous flags pattern, it sends multiple Go-Aheads to the IUT. The IUT is then expected to send a SABM frame and then send an idle ones pattern. The tester responds with a UA. The IUT is then expected to send a continuous flags pattern. The tester will then send a Go-Ahead to the IUT. The IUT is expected to send a

Report Set ID I frame to the tester followed by an idle ones pattern. The tester will then respond with an RR.

TESTER IUT
plug in IUT
idle ones or flags >
<- Idle ones pattern
<- Continuous flags
Multiple Go-Aheads (7F, 0) >
<- SABM
>RS, start T200
<- Idle ones pattern
UA >
>NM, stop T200
<- Continuous flags
Go-Ahead (7F, 0) >
< I frame (report Set ID)
start T200
<- Idle ones pattern
RR >
stop T200

3.5.3 SABM received by IUT while in SABM reset (RS) state

The tester sends an I frame with a bad N(R) to force the IUT into SABM reset (RS) state. The tester then sends a SABM. The IUT is expected to send a UA and remain in SABM reset state. The tester then sends an I frame to verify that the IUT is in SABM reset state. The IUT should ignore this I frame and send another SABM frame after T200 expires.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
SABM >
< UA
I frame (good) >
ignore
< SABM
restart T200

3.5.4 SABM (P=1) received by IUT while in SABM reset (RS) state

The tester sends an I frame with a bad N(R) to force the IUT into SABM reset state. The tester then sends a SABM with poll bit set to 1. The IUT is expected to send a UA with the final bit set to 1 and the IUT should remain in the SABM reset state. This is tested by sending an I frame. The I frame should be ignored by the IUT. When T200 expires the IUT should send another SABM frame.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
SABM (P = 1) >
< UA (F = 1)
I frame (good) >
ignore
< SABM
restart T200

3.5.5 UA received by IUT while in SABM reset (RS) state

In this test the tester sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends a UA which should cause the IUT to enter the normal state (NM). The tester verifies this by sending an I frame. The IUT should then respond with an RR.

TESTER IUT
I frame (bad N(R)) >
<- SABM
>RS, start T200
UA >
>NM, stop T200
< I frame (report set id) *This I frame is optional
start T200
RR >
stop T200
I frame (good) >
< RR

3.5.6 Bad command received by IUT while in SABM reset (RS) state

The tester forces the IUT into the SABM reset state by sending it an I frame with a bad N(R). The tester then sends a bad command (DISC as it's not supported by MiLAP-s.). The IUT should ignore this message and remain in the SABM reset state. When T200 expires, the IUT should send another SABM frame.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
DISC >
ignore
< SABM
restart T200

3.5.7 Unsolicited F bit received by IUT while in SABM reset (RS) state

The tester forces the IUT into the SABM reset state by sending an I frame with a bad N(R). The tester then sends an unsolicited F bit. The IUT should ignore this and remain in SABM reset state. When T200 expires, the IUT is expected to send another SABM frame.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
UA(F=1) >
ignore
< SABM
re start T200

3.5.8 I frame received by IUT while in SABM reset (RS) state

In this test, the tester sends an I frame with a bad N(R) to force the IUT in the SABM reset state. The tester then sends an I frame. This I frame should be ignored by the IUT. When T200 expires, the IUT should send another SABM frame.

TESTER IUT
I frame (bad N(R)) >
< SABM
>RS, start T200
I frame >
ignore
< SABM
restart T200

3.5.9 RR supervisory frame received by IUT while in SABM reset (RS) state

This test verifies that the IUT ignores a supervisory frame while in the SABM reset state. The tester first sends an I frame with a bad N(R) to force the IUT into the SABM reset state. Then the tester sends an RR frame. The IUT should ignore it. When T200 expires the IUT is expected to transmit another SABM frame.

TESTER IUT
I frame (bad N(R)) >
<- SABM
>RS, start T200
RR >
ignore
<- SABM
restart T200

3.5.10 RNR received by IUT while in SABM reset (RS) state

This test verifies that the IUT ignores a supervisory frame while in the SABM reset state. The tester first sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends an RNR frame. The IUT should ignore it. When T200 expires the IUT is expected to transmit another SABM frame.

TESTER IUT
I frame (bad N(R)) >
<- SABM
>RS, start T200
RNR >
ignore
<- SABM
restart T200

3.5.11 REJ received by IUT while in SABM reset (RS) state

This test verifies that the IUT ignores a supervisory frame while in the SABM reset state. The tester first sends an I frame with a bad N(R) to force the IUT into the SABM reset state. The tester then sends an REJ frame. The IUT should ignore it. When T200 expires the IUT should transmit another SABM frame.

TESTER IUT
I frame (bad N(R)) >
<- SABM
>RS, start T200
REJ >
ignore
<- SABM
restart T200

3.5.12 Frame exceeding maximum number of bytes (N201) received by IUT while in normal (NM) state

This test sends a frame that is too long for the IUT. The IUT should respond with a SABM and migrate to the SABM reset state.

TESTER IUT
Good I-Frame (too long) >
<- SABM
>RS, start T200

3.5.13 Frame shorter than the minimum number of bytes received by IUT while in normal (NM) state

This test verifies that the IUT will ignore a frame that is too short. A frame less than 4 octets is considered too short.

TESTER IUT
Frame (2 octets) >
ignore
Frame (3 octets) >
ignore

3.5.14 I frame (P=1) received by IUT while in normal (NM) state

The tester sends an I frame with the poll bit set to 1. The IUT should send a SABM and migrate to the SABM reset state.

TESTER IUT
I frame (P = 1) >
<- SABM
>RS, start T200

3.5.15 Address test while in NM state.

This test checks that the IUT will only respond to its particular address or to a SABM or UA using the broadcast address (63). Since the broadcast address is not supported by all IUT's, it is a PIXIT option.

TESTER IUT
I frame (good) address = X >
< RR
I frame (good) address = Y >
ignore
I frame (good) address = X >
< RR

Next message sent only if PIXIT option for broadcast address supported is set to YES.

SABM (address=63) >
< UA

3.5.16 SABM received by IUT while in normal (NM) state

The IUT should respond to the tester's SABM with a UA and remain in the normal state. The I frame is sent to verify that the IUT has not gone into the SABM reset state.

TESTER IUT
SABM >
< UA
< I frame (report set id) *This I frame is optional
start T200
RR >
stop T200
I frame (good) >
<- RR

3.5.17 SABM received by IUT while in reject sent (RJ) state

This test sends an I frame with a bad N(S) to force the IUT into the reject sent state. Another I frame with a bad N(S) is then sent to ensure the IUT is in the RJ state. The IUT should ignore this frame. A SABM is then sent by the tester and the IUT is expected to respond with a UA and migrate to the normal state. The tester then sends another I frame with a bad N(S) to verify that the IUT is in the Normal (NM) state. The IUT should respond with a REJ.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ, **do not start T200**
I frame (bad N(S)) >
ignore
SABM >
< UA
>NM
< I frame (report set id) *This I frame is optional
start T200
RR >
stop T200
I frame (bad N(S)) >

<- REJ
>RJ, **do not start T200**

3.5.18 SABM received by IUT while in remote busy (RB) state

The tester sends an RNR frame to force the IUT into the remote busy (RB) state. The test operator is then requested to perform some action on the IUT that would normally send an I frame to the tester. This I frame should not be transmitted since the IUT is in the RB state.The tester then sends a SABM and the IUT is expected to respond with a UA, set N(R) and N(S) to 0, and migrate to the normal state. The IUT may then optionally transmit a report set id. I frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
SABM >
< UA
>NM
< I frame (report set id) *This I frame is optional
start T200
RR >
stop T200

3.5.19 SABM received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a SABM frame and expects the IUT to respond with a UA, set N(R) and N(S) to 0, and migrate to the normal state. The IUT may then optionally send a report set id.I frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
SABM >
< UA
>NM
< I frame (report set id) *This I frame is optional
start T200
RR >
stop T200

3.5.20 UA received by IUT while in normal (NM) state

The tester sends a UA and the IUT is expected to respond with a SABM and migrate to the SABM reset state. The tester verifies that the IUT is in the SABM reset state by sending an I frame. The IUT is expected to ignore it.

TESTER IUT
UA >
<- SABM
>RS, start T200
I frame (good) >

3.5.21 UA received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends another bad I frame to ensure the IUT is in the RJ state. The tester then sends a UA and the IUT is

expected to send a SABM and migrate to the SABM reset state. An I frame is then sent by the tester to verify that the IUT has migrated to the SABM reset state. The IUT should ignore this frame and when T200 expires it should transmit another SABM frame.

TESTER IUT
I bad N(S) >
<- REJ
>RJ, **do not start T200**
I bad N(S) >
ignore
UA >
<- SABM
>RS, start T200
I (good) >
ignore
< SABM
restart T200

3.5.22 UA received by IUT while in remote busy (RB) state

The tester sends an RNR to force the IUT into the remote busy state. The tester then sends a UA frame and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then sends an I frame to verify that the IUT is in the SABM reset state.

TESTER IUT
RNR(P = 1) >
>RB
< RR (F=1)
UA >
<- SABM
>RS, start T200
I (good) >
ignore

3.5.23 UA received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a UA frame and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
UA >
<- SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
<- SABM
restart T200

3.5.24 Good I frame received by IUT while in reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. The tester then sends another bad I frame to ensure the IUT is in the RJ state. The tester then sends a good I frame and expects the IUT to respond with a RR and migrate to the normal state. The tester then sends an I frame with a bad N(S) to verify that the IUT is in the normal state. The IUT should respond with a REJ.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ, **do not start T200**
I frame (bad N(S)) >
ignore
I frame (good) >
< RR
>NM
I frame (bad N(S)) >
< REJ
>RJ, **do not start T200**

3.5.25 Good I frame received by IUT while in remote busy (RB) state

The tester sends a RNR to put the IUT in the remote busy state. The tester then sends a good I frame and expects the IUT to respond with a RR and remain in remote busy state. The test operator should then attempt to send an I frame to the tester. This I frame should not be sent to the tester until the tester sends an RR to the IUT.

TESTER IUT
RNR (P=0) >
>RB
I frame (good) >
< RR
Attempt to send an I frame (e.g., off hook.)
RR (P=1) >
< RR (F=1)
>NM
< I frame (e.g., off hook)
start T200

3.5.26 Good I frame received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a good I frame and expects the IUT to respond with a RR and migrate to the RB state. The tester then verifies this state transition by waiting TWAIT seconds to ensure the IUT's queued I frame is not transmitted. The tester then sends a RR and expects the IUT to migrate to the NM state and transmit its queued I frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
I frame (good) >
< RR
>RB
RR >
>NM
< I frame (e.g., off hook)
start T200

3.5.27 I frame with bad N(S) received by IUT while in reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. The tester then sends multiple I frames with bad N(S)'s and expects the IUT to ignore them. The tester then sends a good I frame and expects the IUT to send an RR and migrate to the normal state.The tester then verifies that the IUT is in the normal state by sending an I frame with a bad N(S). The IUT should send a reject.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ, **do not start T200**
I frame (bad N(S)) >
ignore
I frame (bad N(S)) >
ignore

I frame (good) >
<- RR
>NM
I frame (bad N(S)) >
< REJ

3.5.28 I frame with bad N(S) received by IUT while in remote busy (RB) state and also in the remote busy and reject (RJ and RB) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends another I frame with a bad N(S) to make sure the tester is in the RJ and RB state. The IUT should ignore this frame. The tester sends a RR with the poll bit set and expects the IUT to respond with a RR with the final bit set and migrate to the RJ state. The IUT should now send the queued I frame. The tester then sends an I frame with a bad N(S) to ensure the IUT is in the reject state. The IUT should ignore this. The tester then sends a good I frame and expects the IUT to respond with a RR and migrate to the NM state.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
I frame (bad N(S)) >
ignore
RR (P=1) >
<- RR (F=1)
> RJ
< I frame (e.g., off hook)
I frame (bad N(S)) >
ignore
I frame (good) >
< RR
>NM

3.5.29 RR (P=0) received by IUT while in normal (NM) state

The tester sends a RR with P=0 to the IUT and expects no response. The tester then sends a RR with the poll bit set and expects the IUT to respond with an RR with F = 1. The IUT should remain in the normal state.

TESTER IUT

RR (P=0) >
ignore
RR (P=1) >
< RR (F=1)

3.5.30 RR (P=0) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends a RR and the IUT is expected to ignore it and remain in the reject sent state. An I frame with a bad N(S) is then transmitted by the tester to ensure that the IUT is still in the reject sent state. The IUT is expected to ignore this frame.

TESTER IUT
I frame (bad N(S)) >
<- REJ
RJ, **do not start T200**
RR (P=0) >
ignore
I frame (bad N(S)) >
ignore

3.5.31 RR (P=0) received by IUT in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. The test operator is then prompted to perform an action on the IUT that would normally cause an I frame to be sent (e.g., off hook).The tester then sends an RR poll = 0 frame and the IUT should migrate to the normal state and transmit its queued I frame.

TESTER IUT
RNR (P=1) >
< RR (F =1)
>RB
Attempt to send an I frame (e.g., off hook)
RR (P=0) >
>NM
< I frame (e.g., off hook)
start T200
RR >
stop T200

3.5.32 RR(P=0) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a RR (P=0) and expects the IUT to migrate to the RJ state and transmit its queued I frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
RR (P=0) >
>RJ
< I frame (e.g., off hook)
I frame (bad N(S)) >
ignore

3.5.33 RR (P=1) received by IUT in normal (NM) state

The tester sends an RR with the poll bit set. The IUT is expected to respond with an RR final = 1 frame.

TESTER IUT
RR (P=1) >
< RR (F=1)

3.5.34 RR (P=1) received by IUT in reject sent (RJ) state

The tester sends an I frame with a bad N(S) to force the IUT into the reject sent state. The tester verifies this state change by sending another I frame with a bad N(S). The tester then sends a RR poll =1. The IUT is expected to respond with an RR final = 1.

TESTER IUT
I frame (bad N(S)) >
<- REJ
>RJ , **do not start T200**
I frame (bad N(S)) >
ignore
RR (P=1) >
< RR (F=1)

3.5.35 RR (P=1) received by IUT in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR frame. This is verified by having the IUT attempt to send an I frame (e.g., off hook). The tester then sends an RR poll = 1 frame. The IUT is expected to respond with a RR final = 1, migrate to the normal state and transmit its queued I frame.

TESTER IUT
RNR (P=1) >
< RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
RR (P=1) >
< RR (F=1)
>NM
< I frame (e.g., off hook)
start T200
RR >
stop T200

3.5.36 RR(P=1) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a RR (P=1) and expects the IUT to respond with a RR (F=1), migrate to the RJ state and transmit its queued I frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
RR (P=1) >
< RR (F=1)
>RJ
< I frame (e.g., off hook)

3.5.37 REJ (P=0) received by IUT in normal (NM) state

The IUT sends an I frame to the tester. The tester responds with a REJ. The IUT is expected to repeat the I frame and start T200. The tester does not respond to this I frame. T200 expires and the IUT should repeat the I frame. This time the tester responds with an RR.

TESTER IUT
Send an I frame (e.g., off hook)
< I frame (e.g., off hook)
start T200
REJ (P=0) >
N(R) = N(S) of received I frame stop T200
< I frame (e.g., off hook)
start T200
T200 expires
< I frame (e.g., off hook)
start T200
RR (F=0) >
stop T200

3.5.38 REJ (P=0) received by IUT in reject sent (RJ) state

The tester forces the IUT into the reject sent state. The IUT then sends an I frame and the tester rejects it. The IUT repeats the I frame and the tester does not respond. The IUT's T200 expires and the IUT repeats the I frame. The tester then sends another I frame with a bad N(S) to ensure the IUT is still in the RJ state. The tester then sends a good I frame and the IUT is expected to respond with an RR and migrate to the normal state. Note that the I frame sent by the tester acknowledges the outstanding I frame that the IUT sent.

TESTER IUT
I frame (bad N(S)) >
<- REJ
>RJ, **do not start T200**
Send an I frame (e.g., off hook)
< I frame (e.g.,off hook)
start T200
REJ (P=0) >
N(R)=N(S) of the received I frame stop T200
< I frame (e.g.,off hook)
restart T200
T200 expires
< I frame (e.g.,off hook)
start T200
I frame (bad N(S)) >
ignore
I frame (good) >
stop T200
<- RR
>NM

3.5.39 REJ (P=0) received by IUT in remote busy (RB) state

The IUT sends an I frame. The tester then forces the IUT into the remote busy state. The tester then sends a REJ to the IUT's previous I frame. The IUT should repeat the I frame, start T200 and migrate to the normal state. The tester lets T200 expire without acknowledging the I frame. The IUT will then repeat the I frame on T200 expiry. The tester then acknowledges the I frame with an RR.

TESTER IUT
Send an I frame (e.g., off hook)
< I frame (e.g., off hook)
start T200
RNR (P=0) >
>RB, stop T200

Wait TWAIT seconds to ensure IUT stopped its T200
REJ (P=0) >
N(R)=N(S) of the received I frame
< I frame (e.g., off hook)
>NM, start T200
T200 expires
< I frame (e.g., off hook)
restart T200
RR (F=0) >
stop T200

3.5.40 RNR (P=0) received by IUT in normal (NM) state

The tester sends an RNR to the IUT. The IUT should migrate to the remote busy state. The test operator is then prompted to have the IUT send an I frame. No message should be sent to the tester. The tester then clears the busy condition by sending an RR to the IUT. The IUT responds with a RR followed by the queued off hook I frame. The tester responds with an RR.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
RR (P=1) >
< RR(F=1)
>NM
< I frame (e.g., off hook)
start T200
RR (F=0) >
stop T200

3.5.41 RNR (P=0)received by IUT in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends a RNR (P=0) frame to the IUT. The IUT should migrate to the reject sent and remote busy state. The tester then sends an I frame with a bad N(S) to verify that the IUT is in the RJ and RB state. The test operator is then prompted to attempt to have the IUT send an I frame. The I frame should not be sent by the IUT as its in the RB and RJ state. The tester then sends a good I frame and expects the IUT to respond with an RR and migrate to the RB state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ, **do not start T200**
RNR (P=0) >
>RJ and RB
I frame (bad N(S)) >
ignore
Attempt to send an I frame (e.g., off hook)
I frame (good) >
<- RR
>RB

3.5.42 RNR(P=0) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a RNR (P=0) and expects the IUT to ignore it.

TESTER IUT
RNR (P=0) >

>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
RNR (P=0) >
ignore

3.5.43 RNR (P=1) received by IUT in normal (NM) state

The tester forces the IUT into the remote busy state by sending an RNR(P=1). The IUT is expected to respond with a RR (F=1). The test operator should now attempt to have the IUT send an I frame. The tester then sends another RNR with the poll bit set and the IUT is expected to respond with an RR (F=1). The tester now clears the busy condition by sending an RR to the IUT. The IUT should respond with an RR followed by the I frame (off hook) and migrate to the normal state.

TESTER IUT
RNR (P=1) >
< RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
RNR (P=1) >
<- RR (F=1)
RR (P=1) >
<- RR (F=1)
>NM
<- I frame (e.g., off hook)
start T200
RR >
stop T200

3.5.44 RNR (P=1) received by IUT in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends a RNR (P=1) frame to the IUT. The IUT should respond with a RR (F=1) and migrate to the reject sent and remote busy state. The tester verifies this by first sending an I frame with a bad N(S) to which it expects no response, and then by prompting the test operator to attempt to send an I frame. The tester then sends a good I frame and expects the IUT to respond with an RR and migrate to the RB state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ, **do not start T200**
RNR (P=1) >
>RJ and RB
<- RR (F=1)
I frame (bad N(S)) >
ignore
Attempt to send an I frame (e.g., off hook)
I frame (good) >
<- RR
>RB

3.5.45 RNR (P=0) and RNR (P=1) received by IUT in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR. The test operator is prompted to attempt to have the IUT send an I frame (e.g., take the IUT off hook) and the tester verifies there is no response.The tester then sends another RNR (P=0) and the IUT should ignore the message. The tester then sends a RNR (P=1) and the IUT should respond with an RNR (F=1) The tester then sends an RR to put the IUT back to the normal state. The IUT should respond with an RR followed by the queued I frame.

TESTER IUT

RNR (P=0) >
>RB
Attempt to send an I frame (e.g.,. off hook)
RNR (P=0) >
ignore
RNR (P=1) >
< RR (F=1)
RR (P=1) >
< RR (F=1)
>NM
<- I frame (e.g., off hook)
start T200
RR >
stop T200

3.5.46 RNR(P=1) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a RNR (P=1) and expects the IUT to respond with a RNR (F=1).

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
RNR (P=1) >
< RR (F=1)

3.5.47 T200 expires (T200<N200) while in normal state

The test operator is prompted to have the IUT send an I frame (e.g., off hook). This causes an I frame to be sent to the tester. The tester does not respond to this I frame. T200 expires and the IUT is expected to repeat the I frame. This time the tester responds with an RR.

TESTER IUT
Send an I frame (e.g., off hook)
<- I frame (good)
start T200
T200 expires
<- I frame (good)
restart T200
RR >
stop T200

3.5.48 T200 expires (T200<N200) while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S) to the IUT. The test operator then takes the IUT off hook and an I frame should be sent to the tester. The tester does not respond to the I frame, so after T200 seconds the IUT is expected to repeat the I frame. The tester then sends a good I frame to the IUT. The IUT is expected to acknowledge it with an RR.

TESTER IUT
I frame (bad N((S)) >
<- REJ
>RJ
Send an I frame (e.g., off hook)
< I frame

start T200
T200 expires (T200< N200)
<- I frame
restart T200
RR >
stop T200
I frame (good) >
>NM
<- RR

3.5.49 T200 expires N200 times while in normal (NM) state

The test operator is prompted to have the IUT send an I frame. This causes an I frame to be transmitted to the tester. The tester does not respond, so after T200 seconds the IUT the IUT repeats the I frame. The tester still does not respond so after T200 seconds the IUT repeats the I frame. This continues until T200 = N200. At this point the IUT is expected to send a SABM and migrate to the SABM reset state.

TESTER IUT
Send an I frame (e.g., off hook)
< I frame (e.g., off hook)
start T200
T200 expires(T200< N200)
< I frame (e.g., off hook)
restart T200
T200 expires (T200< N200)
< I frame (e.g., off hook)
restart T200
T200 expires (T200 = N200)
< SABM
>RS start T200

3.5.50 T200 expires N200 times while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The test operator is prompted to have the IUT send an I frame. This causes an I frame to be transmitted to the tester. The tester does not respond, so after T200 seconds the IUT repeats the I frame. The tester still does not respond so after T200 seconds the IUT repeats the I frame. This continues until T200 = N200. At this point the IUT is expected to send a SABM and migrate to the SABM reset state.

TESTER IUT
I frame(bad N(S)) >
< REJ
>RJ
Send an I frame (e.g., off hook)
<- I frame
start T200
T200 expires(T200< N200)
<- I frame
restart T200
T 200 expires (T200 = N200)
<- SABM
>RS, start T200

3.5.51 N200 variable not incremented when T200 expires while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending it an RNR. The test operator then prompted to have the IUT send an I frame. No message should be sent to the tester. T200 is allowed to expire several times before the tester sends an RR. The IUT should, upon receiving this RR, send the queued I frame. Note that the frame counter (counts up to N200) does not get incremented in the remote busy state.

TESTER IUT
RNR (P=1) >

< RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
T200 expires(T200< N200)
T200 expiresagain (T200< N200)
RR not sent until after T200 expires more than N200 times
RR >
< I frame
>NM, start T200

3.5.52 I frame (bad N(R)) received by IUT while in normal (NM) state

The tester sends an I frame with a bad N(R) to the IUT. The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.

TESTER IUT
I frame (bad N(R)) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.53 I frame (bad N(R)) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then sends another I frame with a bad N(S) to verify that the IUT is in the RJ state. The tester then sends an I frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
<- REJ
>RJ
I frame (bad N(S)) >
ignore
I frame (bad N(R)) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.54 I frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR. The test operator is then requested to attempt to force the IUT to send an I frame. The tester then sends an I frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(R)) >
< SABM

>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.55 I frame (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state
The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a I frame with a bad N(R) and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.
TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
I frame (bad N(R)) >
< -SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
< -SABM
restart T200

3.5.56 RR frame (bad N(R)) received by IUT while in normal (NM) state
The tester sends an RR frame with a bad N(R) to the IUT. The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.
TESTER IUT
RR (bad N(R)) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.57 RR frame (bad N(R)) received by IUT while in reject sent (RJ) state
The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then verifies the IUT is in the RJ state by sending another I frame with a bad N(S). The tester then sends an RR frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA to put the IUT in the normal state.
TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
RR (bad N(R)) >

< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.58 RR frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR. The tester then verifies the IUT is in the RB state by prompting the test operator to have the IUT send an I frame The tester then sends an RR frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
< RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
RR (bad N(R)) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.59 RR (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a RR with a bad N(R) and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
RR(bad N(R)) >
< SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
< SABM
restart T200

3.5.60 RNR frame (bad N(R)) received by IUT while in normal (NM) state

The tester sends an RNR frame with a bad N(R) to the IUT. The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.

TESTER IUT
RNR (bad N(R)) >

```
<       SABM
        >RS, start T200
I frame (good)  >
        ignore
UA      >
        >NM, stop T200
```

3.5.61 RNR frame (bad N(R)) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S) and then verifies the state change by sending another I frame with a bad N(S). The tester then sends an RNR frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA to put the IUT in the normal state.

```
TESTER          IUT
I frame (bad N(S))      >
<       REJ
        >RJ
I frame (bad N(S))      >
        ignore
RNR (bad N(R))  >
<       SABM
        >RS, start T200
I frame (good)  >
        ignore
UA      >
        >NM, stop T200
```

3.5.62 RNR frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR and then verifies the state transition by having the test operator attempt to have the IUT send an I frame. The tester then sends an RNR frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

```
TESTER          IUT
RNR (P=1)       >
<       RR (F=1)
        >RB
        Attempt to send an I frame (e.g., off hook)
RNR (bad N(R))  >
<-      SABM
        >RS, start T200
I frame (good)  >
        ignore
UA      >
        >NM, stop T200
```

3.5.63 RNR (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a RNR with a bad N(R) and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

```
TESTER          IUT
RNR (P=0)       >
        >RB
```

Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
RNR(bad N(R)) >
< SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
< SABM
restart T200

3.5.64 REJ frame (bad N(R)) received by IUT while in normal (NM) state

The tester sends an REJ frame with a bad N(R) to the IUT. The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.

TESTER IUT
REJ (bad N(R)) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.65 REJ frame (bad N(R)) received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). It then verifies the state transition by sending another I frame with a bad N(S). The tester then sends an REJ frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. The tester sends a UA to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
<- REJ
>RJ
I frame (bad N(S)) >
ignore
REJ (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.66 REJ frame (bad N(R)) received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR and then verifies the state transition by prompting the test operator to attempt to force the IUT to send an I frame. The tester then sends an REJ frame with a bad N(R). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
< RR (F=1)

>RB
Attempt to send an I frame (e.g., off hook)
REJ (bad N(R)) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.67 REJ (bad N(R)) received by IUT while in remote busy and reject sent (RB and RJ) state
The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a REJ with a bad N(R) and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.
TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
REJ (bad N(R)) >
< SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
< SABM
restart T200

3.5.68 Bad command received by IUT while in normal (NM) state
The tester sends a bad command frame (DISC as it's not supported in MiLAP-s) to the IUT. The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.
TESTER IUT
DISC >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.69 Bad command received by IUT while in reject sent (RJ) state
The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). It then verifies the state transition by sending another I frame with a bad N(S).The tester then sends an bad command frame (UI as it's not supported by MiLAP-s). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA to put the IUT in the normal state.
TESTER IUT
I frame (bad N(S)) >
< REJ

>RJ
I frame (bad N(S)) >
ignore
UI >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.70 Bad command received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR and then verifies this transition by prompting the test operator to attempt to have the IUT send an I frame. The tester then sends a bad command frame (DISC as it's not supported by MiLAP-s). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
< RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
DISC >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.71 Bad command received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a bad command (UI as its not supported by MiLAP-S) and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
UI >
< SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
< SABM
restart T200

3.5.72 Bad response received by IUT while in normal (NM) state

The tester sends a bad response frame (DM as it's not supported in MiLAP-s) to the IUT. The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.

TESTER IUT
DM >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.73 Bad response received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester then verifies this state transition by sending another I frame with a bad N(S). The IUT should ignore this frame. The tester then sends an bad response frame (DM as it's not supported by MiLAP-s). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
<- REJ
>RJ
I frame (bad N(S)) >
ignore
DM >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.74 Bad response received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR. The tester verifies this state change by prompting the test operator to attempt to have the IUT send an I frame. The tester then sends a bad response frame (FRMR as it's not supported by MiLAP-s). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
< RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
FRMR >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.75 Bad response received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is

expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends a bad response (DM as its not supported by MiLAP-s) and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
DM >
< SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
< SABM
restart T200

3.5.76 Unsolicited F bit received by IUT while in normal (NM) state

The tester sends an unsolicited F bit (UA (F=1))to the IUT. The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.

TESTER IUT
UA (F=1) >
< SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.77 Unsolicited F bit received by IUT while in reject sent (RJ) state

The tester forces the IUT into the reject sent state by sending an I frame with a bad N(S). The tester verifies this state transition by sending another I frame with a bad N(S). The IUT should ignore this frame. The tester then sends an unsolicited F bit (REJ (F=1)). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame. The IUT is expected to ignore it. Then tester sends a UA to put the IUT in the normal state.

TESTER IUT
I frame (bad N(S)) >
< REJ
>RJ
I frame (bad N(S)) >
ignore
REJ (F=1) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, T200

3.5.78 Unsolicited F bit received by IUT while in remote busy (RB) state

The tester forces the IUT into the remote busy state by sending an RNR. The tester verifies this state transition by prompting the test operator to attempt to have the IUT send an I frame. The tester then sends an unsolicited F bit (REJ (F=1)). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

TESTER IUT
RNR (P=1) >
<- RR (F=1)
>RB
Attempt to send an I frame (e.g., off hook)
REJ (F=1) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
> NM, stop T200

3.5.79 Unsolicited F bit received by IUT while in remote busy and reject sent (RB and RJ) state

The tester sends a RNR to force the IUT into the remote busy state. The test operator is then requested to invoke an action on the IUT that would normally cause an I frame to be sent. The I frame should not be sent to the tester since the IUT is in RB state. The tester then sends an I frame with a bad N(S). The IUT is expected to transmit a REJ and migrate to the remote busy and reject state. The tester sends an unsolicited F bit (REJ (F=1)) and expects the IUT to respond with a SABM and migrate to the SABM reset state. The tester then verifies this state transition by sending a good I frame. The IUT should ignore this frame.

TESTER IUT
RNR (P=0) >
>RB
Attempt to send an I frame (e.g., off hook)
I frame (bad N(S)) >
<- REJ
>RJ and RB, **do not start T200**
REJ (F=1) >
< SABM
>RS, start T200
I frame(good) >
ignore
T200 expires
< SABM
restart T200

3.5.80 Unsolicited F bit received by IUT while in normal (NM) state

The IUT sends an I frame and the tester responds with an unsolicited F bit (REJ (F=1)). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester then sends a good I frame to the IUT to test that the IUT is in the SABM reset state. The IUT is expected to ignore this I frame. The tester then sends a UA to put the IUT in the normal state.

TESTER IUT
Send an I frame (e.g., off hook)
<- I frame (e.g., off hook)
start T200
REJ (F=1) >
stop T200
< SABM
>RS, start T200
I frame (good) >
ignore

UA >
>NM, stop T200

3.5.81 REJ (P=1) received in the normal state

The tester sends an REJ (P=1)). The IUT is expected to send a SABM and migrate to the SABM reset state. The tester verifies this by sending a good I frame which the IUT should ignore. The tester then sends a UA putting the IUT into the normal state.

TESTER IUT
REJ (P=1) >
<- SABM
>RS, start T200
I frame (good) >
ignore
UA >
>NM, stop T200

3.5.82 T200 expires in SABM reset (RS) state

The tester forces the IUT into SABM reset state and expects the IUT to send SABM frames every T200 seconds. This should repeat indefinitely.

TESTER IUT
RNR (bad N(R)) >
< SABM
>RS, start T200
T200 expires
<- SABM
restart T200
T200 expires
< SABM
restart T200

3.5.83 Modulus 8 verification of IUT's N(R)

The tester sends I frames with incrementing sequence numbers and expects the IUT to respond with RR's with incrementing sequence numbers. After a sequence number of 7 the numbers should reset to 0 and begin incrementing again. Depending on the IUT, there may be I frames sent to the tester after the SABM/UA exchange (e.g., set id). The number of I frames sent by an IUT is a PIXIT option. Thus N(R) is shown as X for the tester side in the diagram below.

TESTER IUT
SABM >
< UA
set V(R,S) = 0
I frame (good) >
N(S)=0, N(R) = X
<- RR
N(R) = 1
I frame (good) >
N(S)=1, N(R) = X
<- RR
N(R) = 2
I frame (good) >
N(S)=2, N(R) = X
<- RR
N(R) = 3
I frame (good) >
N(S)=3, N(R) = X
<- RR
N(R) = 4

I frame (good) >
N(S)=4, N(R) = X
<- RR
N(R) = 5
I frame (good) >
N(S)=5, N(R) = 0
<- RR
N(R) = 6
I frame (good) >
N(S)=6, N(R) = X
<- RR
N(R) = 7
I frame (good) >
N(S)=7, N(R) = X
<- RR
N(R) = 0
I frame (good) >
N(S)=0, N(R) = X
<- RR
N(R) = 1

3.5.84 Modulus 8 verification of IUT's N(S)

The IUT sends I frames and the tester verifies that the sequence numbers increment and then reset to 0 after a sequence number of 7.

TESTER IUT
SABM >
< UA
set V(R,S) = 0
<- I frame (good)
N(S)=0, N(R) = 0, start T200
RR >
N(R) = 1 stop T200
<- I frame (good)
N(S)=1, N(R) = 0, start T200
RR >
N(R) = 2 stop T200
<- I frame (good)
N(S)=2, N(R) = 0, start T200
RR >
N(R) = 3 stop T200
<- I frame (good)
N(S)=3, N(R) = 0, start T200
RR >
N(R) = 4 stop T200
<- I frame (good)
N(S)=4, N(R) = 0, start T200
RR >
N(R) = 5 stop T200
<- I frame (good)
N(S)=5, N(R) = 0, start T200
RR >
N(R) = 6 stop T200
<- I frame (good)
N(S)=6, N(R) = 0, start T200
RR >
N(R) = 7 stop T200

<- I frame (good)
N(S)=7, N(R) = 0, start T200
RR >
N(R) = 0 stop T200
<- I frame (good)
N(S)=0, N(R) = 0, start T200
RR >
N(R) = 1 stop T200

3.6 Optional MiLINK tests

3.6.1 Address negotiation (normal procedure)

This test ensures the MiLINK IUT can successfully negotiate a new address (TEI). The tester operator will plug the IUT into the MiLINK bus. The IUT will begin sending SABMs using address $1E. The tester will send a UA. The IUT is expected to migrate to the normal state and then send a Report Set ID I frame to the tester. The tester will acknowledge the I frame and send a Define Channel and MiLINK Parameters ($CF) I frame to the IUT, instructing it to use address $03. The IUT is expected to acknowledge this I frame and enter the SABM reset state. This time though it will use the newly assigned address $03. The tester will then respond with a UA. The IUT will again send a Report Set ID I frame. The tester will acknowledge it and send another Define Channel and MiLINK Parameters ($CF) I frame to the IUT, instructing it to use the B2 channel. The IUT is expected to then acknowledge the I frame.

TESTER IUT
plug in MiLINK device
<- SABM (address = 1E)
start T200, T210
UA(address = 1E) >
>NM, stop T200
< I frame(report set ID , address = 1E)
start T200
RR (address = 1E) >
stop T200
I frame$CF (change to address 3) >
<- RR (address = 1E)
>RS, stop T210
start T210
< SABM (address = 3)
start T200, stop T210
UA(address = 3) >
>NM, stop T200
< I frame(report set ID , address = 3)
start T200,T210
RR (address = 3) >
stop T200,T210
I frame$CF (keep address , change to B2) >
start T210
<- RR (address = 3)
stop T210

3.6.2 Address Negotiation (T210 expires)

This test verifies that the MiLINK IUT responds correctly when T210 expires before address negotiation has been completed. The tester operator will plug the IUT into the MiLINK bus. The IUT will begin sending SABMs using address $1E. The tester will send a UA. The IUT is expected to migrate to the normal state and then send a Report Set ID I frame to the tester. The tester will acknowledge the I frame. The tester will not send a new address, so T210 will expire.The IUT is expected to migrate to the SABM reset state.

TESTER IUT

plug in MiLINK device
<- SABM (address = 1E)
start T200, T210
UA(address = 1E) >
>NM, stop T200
< I frame(report set ID , address = 1E)
start T200
RR (address = 1E) >
stop T200
T210 expires, release bus
<- SABM
Note - T210 expiry not currently implemented on any of the MiLINK devices.

3.6.3 Address negotiation (system denies allocation)

This test verifies that if the IUT is assigned the "dead address" it will no longer send any frames. The test operator plugs the IUT into the MiLINK bus. The IUT should begin sending SABMs using address $1E. The tester responds with a UA. The IUT is expected to send a Report Set ID I frame to the tester. The tester acknowledges it and sends a Define Channel and MiLINK Parameters ($CF) I frame to the IUT telling it to change to the "dead address" $1F. The IUT is expected to acknowledge this frame and then go into an unusable state. The IUT should not send any frames to the tester. The tester then prompts the test operator to attempt to have the IUT send an I frame. The IUT should not transmit anything. The tester then sends an I frame with an address of 0. The IUT should ignore this. The tester then sends an I frame using the dead address 1F. The IUT should reset itself and go into the SABM reset state.

TESTER IUT
plug in MiLINK device
<- SABM (address = 1E)
start T200, T210
UA(address = 1E) >
>NM , stop T200
< I frame(report set ID , address = 1E)
start T200
RR (address = 1E) >
stop T200
I frame$CF (change to address 1F) >
<- RR (address = 1E)
stop T210
Attempt to send an I frame (e.g., off hook)
I frame(address = 0) >
ignore
I frame(address = 1F) >
< SABM (address = 1E)
start T200, T210