WINMOR Protocol Specification (Preliminary)

Revision: 0.2 Nov 24, 2008
Rick Muething, KN6KB, AAA9WK

1.0 Scope:
This document describes the preliminary WINMOR sound card protocol at the physical and data link levels. It is the complete specification of the WINMOR protocol. It does not address higher level protocol layers. The WINMOR protocol is not proprietary and is released to the public domain.

2.0 Purpose:
The intent of this document is two fold:
a) To serve as a working document during protocol development and testing
b) To serve as a template to allow others familiar with the art to build compatible drivers that support the data link protocol layer.

3.0 Definitions and Syntax:
Several specific terms and syntax are used in this document:
Definitions: A term or item is defined using the := symbol. This symbol can be read as "is defined as"
Implementation directives: These are key words that indicate how an item is to be implemented or recommend a method of implementation. They are always indicated by capitalized italic words. These are:
MUST := this must be followed to implement the protocol
MUST NOT := this must not be done to implement the protocol
SHOULD := this is the recommended way to implement the protocol
MAY := this is alternative way to implement the protocol.
The syntax above is always used to distinguish between the common use of the same words.
& is used to indicate catenation. E.g. Frame := Pilot & Data

4.0 Overview of the Protocol:
The WINMOR protocol is intended to be used for sending messages and data error free over a HF radio link. It is a Selective Repeat Automatic Retry reQuest (SRARQ) protocol where the Information Receiving Station (IRS) acknowledges receipt of the data to the Information Sending Station (ISS). Normally during a connection session the IRS and ISS exchange roles multiple times. The protocol is designed to handle the type conditions normally encountered in amateur radio transmission. Specifically:
Generally low S/N levels
Non "channelized" frequencies with interference
Poor to moderate propagation conditions including poor multipath environment.

Frequency offset (between send and receiver) and drift
Sound card sampling rate error and drift

The WINMOR protocol uses basic OFDM (Orthogonal Frequency Division Multiplexing) modulation and a number of modulation modes to adapt to changing channel conditions. There are 3 basic operating bandwidths that operate in 200Hz, 500Hz and 2000Hz bandwidths (@ 26 db below peak power output) :

1) Single carrier 62.5 baud PSK/QAM or 4FSK 31.25 baud for 200Hz bandwidth
2) 3 carrier 62.5 baud PSK/QAM, or 4FSK 31.25 baud for 500 Hz bandwidth
3) 15 carrier 62.5 baud PSK/QAM or 4FSK 31.25 baud for 2000Hz bandwidth

WINMOR is not optimized for keyboarding or "chat" mode applications though this may be possible with the appropriate user client.

5.0 Physical Layer Protocol Description:

The protocol requires the following hardware:

1) Radio connection. This *SHOULD* be a single sideband (SSB) transceiver capable of transmitting low distortion audio in the range of 500-2500Hz. When SSB transmission is used it *MUST* always done using Upper Sideband (USB). Other modulation schemes (e.g. NBFM) *MAY* be used in some applications.
2) Radio Frequency accuracy: If SSB modulation is used the radio *MUST* be able to be set to within +/- 100 Hz of a specific (published) frequency.
3) Frequency Drift: If SSB modulation is used the radio frequency *MUST* have a short term drift of < .1Hz/Second over any 5 second period.
4) The transceiver *MUST* have a Receive to transmit switching time of < 100 ms and a Transmit to Receive switching time of < 100 ms
5) The audio for the protocol *MAY* be generated using a standard PC sound card and appropriate software.
6) The sound card or DSP processor *MUST* be able to support a real or interpolated sampling frequency of 8000Hz +/- .1% (+/-1000 parts per million)
7) The processor or PC used to implement the protocol *MUST* be able to complete the decoding of any frame and respond with the appropriate response in 300 ms or less. (this is currently estimated to equate to a Pentium/Celeron class processor of 500 MHz or above)

6.0 Data Link Layer Protocol Description:

6.1 Definitions:

Information Sending Station (ISS) := the station currently sending data to the other station. The ISS *MUST* only send data or control frames.

Information Receiving Station (IRS) := the station currently receiving data or commands from the other station. The IRS *MUST* only send Ack or control frames.

Carrier := one of the modulation carriers. There are 3 carrier modes supported:

1) Single carrier (~200Hz BW) Nominal carrier at 1500.0 Hz
2) Three carriers (~500 Hz BW) Carriers nominally start at an audio frequency of 1375.0 H Hz and are spaced equally at 125 +/- .1% Hz. Carrier 1 is 1375 Hz and carrier 3 is 1625.0 Hz, The pilot carrier is carrier 2 @ 1500.0 Hz
3) Fifteen carriers (~2000Hz BW) Carriers nominally start at an audio frequency of 625.0 Hz and are spaced equally at 125 +/- .1% Hz. Carrier 1 is 625.0 Hz and carrier 15 is 2375.0 Hz. The pilot carrier is carrier 8 @ 1500.0 Hz

Pilot := Leader of the Frame. The Pilot is used to enable rapid identification of a transmission, to DSP tune the receiving station accurately, to establish symbol and frame sync and to indicate the frame type. The single carrier is sent at full modulation strength (Maximum PEP value) to maximize S/N during the Pilot interval

Pilot := P_{tun} & P_{fsync} & P_{fty}

P_{tun} is the tuning pilot. P_{fsync} is the frame sync identifier. P_{fty} is the frame type identifier. P_{tun} & P_{fsync} are always sent using single carrier DBPSK modulation with a root raised cosine envelope encoding for robustness.

P_{tun} := 15 adjacent symbols of the pilot carrier (1500.0 Hz) alternating phase on each symbol. The tuning signal *MAY* be extended up to 16 symbols for transceivers with slow R>T switching or slow VOX PTT response if using VOX.
P_{fsync} := Frame sync symbol consisting of one symbol of the same phase as the immediately preceding P_{tun} symbol. The P_{fsync} symbol serves as the reference symbol for the following P_{tfy} symbols.
P_{fty} := 10 adjacent DQPSK symbols. These 10 symbols encode five 4 bit characters. The first character is the 4 bit frame type (one of 16 frame types) the remaining 4 symbols form the Parity for a RS 5,1 code of 4 bit characters correcting up to 2 errors in the 5 P_{tfy} symbols. Detailed encoding example TBD. The Pilot total length is nominally 26 symbols or 416 ms and may be extended up to 42 symbols or 672ms.

Frame := a packet of information. A frame is composed of a Pilot & Data. Frames are identified by the syntax F_{xyz} where xyz is the frame descriptor.

Symbol := A symbol is one modulation burst of data. The symbol rate is 62.500 symbols per second (baud) +/- .1% for PSK and QAM modes. For 4FSK modes the symbol rate is 31.25 symbols per second (baud) +/- .1%/ Pilot symbols consist of a single carrier with a root raised cosine envelope weighted at the maximum PEP value. Data symbols consist of:

1) 1 carrier QPSK, or 16QAM modulated with a raised cosine envelope The Carrier is weighted at the same weight as the pilot carrier

2) 1 Carrier 4FSK (one of 4 tones). The carrier is weighted at 80% of the weight of the pilot carrier to reduce duty cycle.

3) 3 simultaneous carriers QPSK, or 16QAM modulated with a raised cosine envelope. Each carrier is weighted at 1/3 the maximum PEP value.

4) 3 simultaneous carriers each 4FSK (one of 4 tones). Each carrier weighted at

1/3 the maximum PEP value.

4) 15 simultaneous carriers QPSK, or 16QAM modulated with a raised cosine envelope. Each carrier is weighted at 1/15 the maximum PEP value.

5) 15 simultaneous carriers each 4FSK (one of 4 tones). Each carrier weighted at 1/15 the maximum PEP value.

(Note: some change in the carrier weighting combined with limited clipping *MAY* be used to reduce crest factor)

For PSK and QAM modes the initial symbol following the Pilot is the reference symbol S_r. This establishes the reference for the next Differential symbol. The S_r symbol carries no information but establishes the reference phase for each carrier. The reference phase for each carrier need not be the same as a mechanism of reducing the crest factor. For 16QAM modes the reference symbol also establishes the initial max carrier amplitude reference. There is no reference symbol for 4FSK modes.

Byte := the number of contiguous symbols to make one byte. For DQPSK it is 4 symbols. For 16QAM it is 2 symbols. For 4FSK this is 4 symbols. After the Frame type data all frames send an integral number of bytes with a total length determined by the frame type.

Symbol Modulation: With the exception of the pilot described above all data symbols and all carriers *MUST* use the same modulation scheme. The supported schemes *MUST* include , DQPSK (differential quadrature phase shift keying) Circular 16QAM and 4FSK.

SessionID := a 2 byte integer B_{sid} defined as CRC16 (Calling sign & Target call sign) The session ID dramatically reduces the chances of a session contamination by a remote non connected but audible connection. The Session ID is used in the computation of the sum check but is only sent specifically on data frames. Specific encoding example TBD.

6.2 Frame Types:

The following frame types *MUST* be supported.

6.2.1 Control frames:

$\mathbf{F_{crq}}$ Connect ReQuest frame: 1 Car QPSK = type 0

Sent by the station initiating the connection. Contains session MODE, Call signs of calling and target stations and 2 byte sumcheck.

F_{crq} Encoding:

F_{crq} := Pilot & S_r & B_{data} & B_{sch} & B_{scl} & B_{RS}

B_{data} := B_{mode} & Calling call sign (ASCII upper case encoding with space pad right to 10 char) & Target call sign (ASCII upper case encoding with space pad right to 10 char) (21 bytes total)

B_{mode} is an 8 bit value indicating the session bandwidth:

0 = 200 Hz bandwidth modes

1 = 500 Hz bandwidth modes

2 = 2000 Hz bandwidth modes
Other Encoding details of B_{mode} TBD.
B_{sch} is the high byte of the CRC16 sum check of B_{data}
B_{scl} is the low byte of the CRC16 sum check of B_{data}
B_{RS} is the 8 check parity bytes from a shortened RS (255,247) code correcting up to 4 bytes

F_{crq} Total payload 31 bytes including CRC16 and RS correction check bytes and is always sent using single carrier QPSK modulation.
Total symbols = pilot + 1 +4 + 80 + 4 + 4 +32 = pilot + 93 (2.384 seconds)

F_{ccf} Coded Control Frame (1 Car QPSK = type 1)
Handles the following sub types by 1 byte code in the control frame:
F_{drq} Disconnect Request (code HFF)
F_{idl} Idle (Sent when no data in output buffer) Code H00
F_{brk} Break (sent by the IRS to stop the ISS from sending data and go to the idle state) Code(HAA)

F_{ccf} := Pilot & S_r & B_{cod} & B_{sc8} & B_{RS}
B_{cod} is the 8 bit code value 00 - FF
B_{sc8} is the 8 bit sum check of B_{sid} & B_{cod}
B_{RS} is the 8 check parity bytes from a shortened RS (255,247)

Total symbols = pilot + 1 + 4 + 4 + 16 = pilot + 25 (.784 seconds)

6.2.2 ACK Frames:

F_{ack} := Ack (1 Car QPSK = type 2)
Handles ACK for all carrier modes
F_{ack}:= Pilot & S_r & B_{ack} & B_{sc8} & B_{RS}
B_{ack} is a 16 bit field. MSbit is 0. The remaining 15 bits correspond to the ACK for each carrier. The LSbit represents the highest carrier frequency (2375.0 Hz).
B_{sc8} is the 8 bit sum check of B_{sid} & B_{ack}
There are a total of 24 bits and each 4 bit value is sent as a 4 bit character
B_{RS} is a 32 bit field consisting of 8 4 bit Characters from a RS (14,6) code of 4 bit characters. This code corrects up to four 4 bit characters in the 14 character total ACK.
Total symbols = pilot + 1 + 12 + 16 = pilot + 28 (1.040 seconds)

6.2.3 Data frames:

Data frames consists of three modulation schemes each supporting two data types:

1: Data + weak Reed-Solomon FEC

2: Extended Reed-Solomon FEC
(the extended RS code is used to correct additional errors)

Data is first sent as a type 1 data frame (Data + Weak RS encoding) if the data is not decoded correctly it is sent again as a type 2 (strong RS Parity only). This strong RS parity is appended to the data portion of the previous Data + Weak RS Encoding (the Weak RS Encoding is discarded) and a new more robust RS decode is attempted. Data frames alternate between Type 1 and Type 2 until there is a successful decode. Some form of data summation (analog memory ARQ) *MAY* be used to average repeated Data + Weak RS or Strong RS Parity Only and improve decoding performance.

$\mathbf{F_{dq1}}$ One carrier DQPSK modulation (frame code 5)

F_{dq1} Encoding for type 1 (Data + weak Reed-Solomon error correction):

F_{dq1} := Pilot & S_r & B_{sid} & B_{psn} & B_{bc} & B_{data} & B_{pad} & B_{sch} & B_{scl} & B_{RS}

Where:

B_{sid} is the 16 bit Session ID.

B_{psn} is the Packet Sequence Number (1 to 255 mod 256. PSN 0 is reserved)

B_{bc} is the byte count (the number of bytes in B_{data} only)

B_{data} is the data bytes (up to 64 bytes)

B_{pad} is remaining B_{00} if required to fill B_{data} frame if < 64 bytes are used

B_{sch} is the high byte of the CRC16 sum check per carrier

B_{scl} is the low byte of the CRC16 sum check per carrier

B_{RS} is the Reed Solomon 12 byte RS weak parity using a shortened RS code of 243,255 (6 error correcting)

Encoding for type 2 (Extended RS Parity):

F_{dq1} := Pilot & S_r & B_{sid} & B_{RSX} Where:

B_{sid} is the ones compliment of the 16 bit Session ID.

B_{RSX} are the 80 extended Reed Solomon Parity bytes *only* of a strong RS code 175,255 (40 error correcting).

Total symbols (for both type 1 and type 2) = pilot + 329 (5.680 seconds)

$\mathbf{F_{dq3}}$ 3 carrier DQPSK modulation (frame code 8)

Coding similar to above, each carrier independently coded and sends a different PSN.

$\mathbf{F_{dq15}}$ 15 carrier DQPSK modulation (frame code 11)

Coding similar to above, each carrier independently coded and sends a different PSN.

F_{dqam1} One carrier D16QAM modulation (frame code 6)

F_{dqam1} Encoding for type 1 (Data + weak Reed-Solomon error correction):

F_{dqam1} := Pilot & S_r & B_{sid} & B_{psn} & B_{bc} & B_{data} & B_{pad} & B_{sch} & B_{scl} & B_{RS}

Where:

- B_{sid} is the 16 bit Session ID.
- B_{psn} is the Packet Sequence Number (0 to 255 mod 256)
- B_{bc} is the byte count (the number of bytes in B_{data} only)
- B_{data} is the data bytes (up to 110 bytes)
- B_{pad} is remaining B_{00} if required to fill B_{data} frame if < 110 bytes are used
- B_{sch} is the high byte of the CRC16 sum check per carrier
- B_{scl} is the low byte of the CRC16 sum check per carrier
- B_{RS} is the weak Reed Solomon 24 byte check sum using a shortened RS code of 231,255 (12 error correcting)

Encoding for type 2 (Extended RS Parity):

F_{dq1} := Pilot & S_r & B_{sid} & B_{RSX} Where:

- B_{sid} is the ones compliment of the 16 bit Session ID.
- B_{RSX} are the 138 extended Reed Solomon Parity bytes *only* of a strong RS code 117,255 (69 error correcting).

Total symbols = pilot + 281 (4.912 seconds)

F_{dqam3} 3 carrier D16QAM modulation (frame code 9)

Coding similar to above, each carrier independently coded and sends a different PSN.

F_{dqam15} 15 carrier D16QAM modulation (frame code 12) (currently not used)

Coding similar to above, each carrier independently coded and sends a different PSN.

F_{d4FSK1} One carrier 4FSK modulation @ 31.25 baud (frame code 4)

One of 4 tones is sent from the following set:

1468.75 Hz (code value 0)
1500.00 Hz (code value 1)
1531.25 Hz (code value 2)
1562.50 Hz (code value 3)

F_{d4FSK1} Encoding for type 1 (Data + weak Reed-Solomon error correction):

F_{d4FSK1} := Pilot & B_{sid} & B_{psn} & B_{bc} & B_{data} & B_{pad} & B_{sch} & B_{scl} & B_{RS} Where:

B_{sid} is the 16 bit Session ID.

B_{psn} is the Packet Sequence Number (0 to 255 mod 256)

B_{bc} is the byte count (the number of bytes in B_{data} only)

B_{data} is the data bytes (up to 32 bytes)

B_{pad} is remaining B_{00} if required to fill B_{data} frame if < 32 bytes are used

B_{sch} is the high byte of the CRC16 sum check per carrier

B_{scl} is the low byte of the CRC16 sum check per carrier

B_{RS} is the weak Reed Solomon 6 byte check sum using a shortened RS code of 249,255 (3 error correcting)

F_{d4FSK1} Encoding for type 2 (Extended RS Parity):

F_{d4FSK1} := Pilot & B_{sid} & B_{RSX} Where:

B_{sid} is the ones compliment of the 16 bit Session ID.

B_{RSX} are the 42 extended Reed Solomon Parity bytes *only* of a strong RS code 213,255 (21 error correcting).

Total symbols = pilot + 176 (31.25 baud) Symbols or (6.048 seconds)

F_{d4FSK3} 3 carrier 4FSK modulation @ 31.25 baud (frame code 7)

Coding similar to above, each carrier independently coded and sends a different PSN. Each carrier group of 4 tones is separated by 4 x 31.25 or 125 Hz. The lowest frequency tone is 1343.75 Hz the highest tone is 1687.50 Hz The number of active carriers is always = 3.

$F_{d4FSK15}$ 15 carrier 4FSK modulation @ 31.25 baud (frame code 10)

Coding similar to above, each carrier independently coded and sends a different PSN. Each carrier group of 4 tones is separated by 4 x 31.25 or 125 Hz. The lowest frequency tone is 593.75 Hz the highest tone is 2437.50 Hz. The number of active carriers is always = 15.

(Frame codes 3, 13 and 15 are not currently assigned)

6.3 Protocol Details

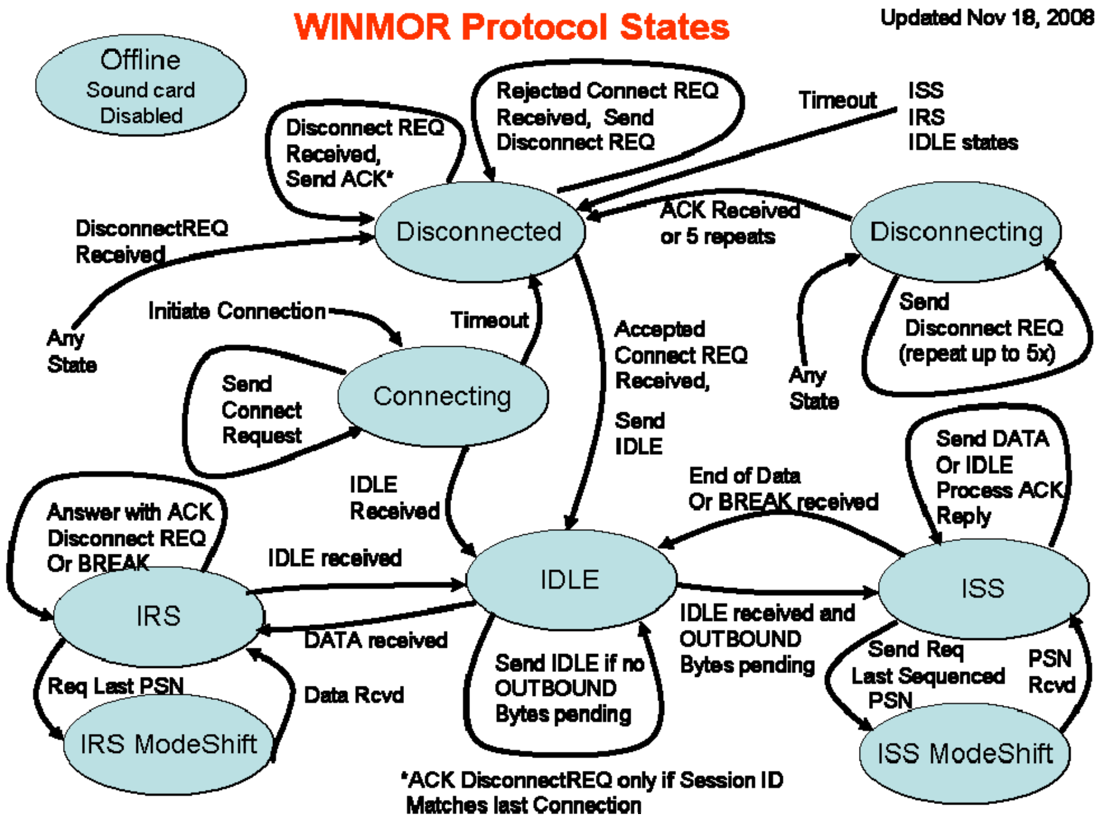

Fig 6 – 1 Protocol State diagram:

6.3.1 Basic Protocol Rules: (refer to state diagram Fig 6-1)

1) Offline.
    a. When WINMOR is in the Offline State it may send no data, receive no data and the sound card is deactivated and sound card resources released.
2) Making a connection: Connecting State
    a. The client (station initiating the connection) enters the Connecting state and periodically (~ ever 3 seconds) sends a Connect REQ frame.
    b. If the client does not receive an IDLE reply within a timeout (typically 15 -30 seconds) it enters the Disconnected state and the link attempt fails.
    c. If the Server (station being connected to) will accept a Connection REQ from the client it responds with an IDLE packet and enters the IDLE state.
    d. If the Server will not accept a Connection REQ from Client it responds with a Disconnect REQ and stays in the Disconnected state.
    e. The Server will ONLY reply to a connect request when it is in the Disconnected state or the IDLE state.
    f. If the connecting station receives a Disconnect REQ it enters the Disconnected state and the link attempt fails.

3) Ending a Connection: Disconnecting State
   a. A forwarding session may be ended by either the Client or Server by entering the Disconnecting state.
   b. In the Disconnecting state the disconnecting station replies to any Data or Control frame with a Disconnect REQ.
   c. If an ACK is received while in the Disconnecting state the station enters the Disconnected state and the link ends.
   d. If No ACK is received after 5 Disconnect REQ frames are sent the station in the disconnecting state goes to the Disconnected state (timeout) and the link ends.
   e. If a station is an any state and receives a Disconnect REQ it replies with an ACK and goes to the Disconnected state and the link ends.
   f. If a station is in the Disconnected state and receives a valid (the session ID matches the previous session) Disconnect REQ it issues an ACK otherwise it *MUST* ignore the Disconnect REQ .
   g. If a station is in any state and has not received an ACK, DATA, BREAK, or IDLE frame in 60 seconds it *MUST* enter the disconnected state and the link ends. (timeout)
   h.
4) IDLE State. A station in the IDLE state:
   a. Replies to an IDLE frame with an IDLE frame if it has no data to send.
   b. If it has data to send (Outbound bytes pending) upon receipt of an IDLE it sends DATA and enters the ISS state.
   c. If In the IDLE state and receives a DATA frame it answers the frame with an ACK and enters the IRS state.
   d. If in the IDLE state and receives a Connect REQ replies with an IDLE frame.
   e. If in the IDLE state and receives a Disconnect REQ replies with an ACK frame and goes to the disconnected state.

5) ISS (Information Sending Station) State. A station in the ISS state:
   a. Sends Data frames and responds :
      i. With the next sequential Data when an ACK is received
      ii. With a repeat of the Data Frame no ACK is received or if no Data was acknowledged (all 0 ack value).
   b. When all outbound data is exhausted and confirmed received by the IRS the station in the ISS state issues an IDLE frame and enters the IDLE state.
   c. If a BREAK is received the station in the ISS state issues an IDLE frame and enters the IDLE state.
6) IRS (Information Receiving Station) State. A station in the IRS state:
   a. Answers a DATA frame with an ACK frame.
   b. Enters the IDLE state and replies with and IDLE frame if an IDLE command is received.

7) The ISS Mode Shift State. This state is used by the ISS to shift (up or down) to another modulation mode. It requests the last *sequenced* PSN from the IRS (the highest PSN received in sequence). The request for PSN signals the IRS to clear any queued out-of-sequence PSNs and prepare for the next PSN in sequence with a new modulation mode and new packet size. The ISS Mode Shift state is exited once the last sequenced PSN is received from the IRS.
8) The IRS Mode Shift State. This state is used by the IRS in to prepare to shift to a new modulation mode. Upon receipt of a request for the last sequenced PSN from the ISS the IRS enters the IRS Mode Shift State and sends the PSN of the highest received PSN *in sequence* and purges any out-of-sequence packets. The IRS exits from the IRS Mode Shift State upon the first receipt of Data from the ISS. This mechanism streamlines the mechanism of shifting data modes which changes the size of received packets.

7.0 Example Forwarding Scenarios:

7.1 A typical Forwarding Session: (no errors or delays)

	CLIENT	**SERVER**	
State	**Frame Sent**	**State**	**Frame Sent**
Connecting	CONREQ		
		Disconnected	IDLE
IDLE	IDLE		
		IDLE	DATA
IRS	ACK		
		ISS	DATA
IRS	ACK		
	…		
		ISS	DATA
IRS	ACK		
		IDLE	IDLE
IDLE	DATA		
		IRS	ACK
ISS	DATA		
		IRS	ACK
	…		
ISS	DATA		
		IRS	ACK
IDLE	IDLE		
		IDLE	IDLE
Disconnecting	Disconnect REQ		
		Disconnected	ACK
Disconnected			

7.2 A session with Errors, delays and aborted connection:

	CLIENT	**SERVER**	
State	**Frame Sent**	**State**	**Frame Sent**
Connecting	CONREQ		
		Disconnected	(nothing..no decode)
Connecting	CONREQ		
		Disconnected	IDLE
Connecting	CONREQ (missed reply)		
		IDLE	IDLE
IDLE	IDLE		
		ISS	DATA
IDLE	nothing (missed DATA)		
		ISS	DATA
IRS	ACK		
	…		
		ISS	DATA
IRS	ACK		
		IDLE	IDLE
IDLE	DATA		
		IRS	ACK
ISS	DATA		
		IRS	ACK
	…		
ISS	DATA		
		IRS	ACK
IDLE	IDLE		
		IDLE	IDLE
Disconnecting	Disconnect REQ		
		Disconnected	ACK