WINMOR Protocol Specification (Preliminary)

Revision: 0.9 Sept 14, 2009

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. This document describes the 200 Hz, 500Hz, 1000 Hz and 2000 Hz bandwidth modes.

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 and error correction schemes to adapt to changing channel conditions. There is currently 4 operating bandwidths of 200, 500, 1000, and 2000Hz (@ 26 db below peak power output:

- 200 Hz BW 1 carrier 31.25 Baud 4FSK or 62.5 baud PSK using TCM 4PSK, 8PSK or 16PSK
- 500 Hz BW 3 carriers 31.25 Baud 4FSK or 62.5 baud PSK using TCM 4PSK, 8PSK or 16PSK
- 1000 Hz BW 7 carriers 31.25 Baud 4FSK or 62.5 baud PSK using TCM 4PSK, 8PSK or 16PSK
- 2000 Hz BW 15 carriers 31 .25 Baud 4FSK or 62.5 baud PSK using TCM 4PSK, 8PSK or 16PSK

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

4.1 Error Correcting Mechanisms

WINMOR employs a number of powerful error detecting and correcting mechanisms which are specifically adapted to the types of errors found in HF communications using popular FSK and PSK modulation schemes. This section outlines the error correction approach used.

Normally error correction is done in terms of “layers” for improved effectiveness and efficiency. The following briefly describes these layers and how they are used in WINMOR.

4.1.1. Outer Sumcheck Layer

This layer applies a standard CRC sumcheck calculation on the “corrected” data. It insures to a very high probability that the corrections are indeed correct and the data matches that which was transmitted. For Connect request frames and all data frames a 16 bit CRC Polynomial of `x^16 + x^12 +x^5 + 1` is used. For short coded control and ACK frames an 8 bit CRC polynomial of `x^8 + x^7 +x^3 + x^2 + 1` is used.

4.1.2 Reed Solomon Layer

Reed Solomon (R-S) FEC appends parity blocks (characters) to an uncoded message which are used to detect and correct errors. The total message size (with parity) must be equal to $2^n – 1$ where n is the character size in bits. In WINMOR two character sizes are used. Short coded control and ACK frames use 4 bit characters (max message size of 15 four bit characters) and all other frames use 8 bits characters (max message size of 255 eight bit characters). Shortened R-S codes (where sender and receiver agree a priori to the message size and not transmit the “fill” part of the message) are used as is typical in many R-S applications.

For data frames the Reed Solomon layer is actually implemented in two formats. The first format is what is called weak R-S where a relatively few parity characters are appended to the message. This weak R-S format is usually sufficient to correct the frame under most conditions. If a data frame must be repeated (reciever did not ACK) then the second strong R-S format is used which transmits *just* the parity charactes of a more robust R-S code. These parity characters are then used along with the prior sent data (ignoring the weak R-S Parity bytes) to attempt to correct the original message part of the first weak R-S format. These two formats are alternated with each repeat of a data frame.

4.1.3 Viterbi Encoded TCM Layer

The next layer uses what is called Viterbi Encoded Pragmatic Trellis Coded Modulation. (See appendix B) This is used on all PSK modes but is not used on FSK. This scheme reaches to within about .2 db of the theoretical coding gain of the best similar length Trellis codes but uses a standard Viterbi encoder/decoder (NASA Voyager R=1/2, K=7). In TCM a single bit is added to each PSK symbol doubling the number of phases. The gain provided by the code exceeds that lost by the tighter phase constellation by typically about 3 dB giving essentially a 3dB power improvement with no change in payload throughput or bandwidth. The layered use of the Viterbi inner encoding and R-S outer encoding is common in many advanced error correcting schemes.

4.1.4 Memory ARQ

The final layer used is what is commonly called memory ARQ (Automatic Retry reQuest). If decoding on the received data using the above layers is not sucessful Memory ARQ averages the received demodulated (soft) symbol values (frequency or phase/magnitude) on a symbol by symbol basis and attempts a decode (using the above layers) on the averaged values. This can be effective in very weak signal conditions. Memory ARQ is done only for data frames and is applied to both the weak R-S and strong R-S data formats.

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 Upper sideband 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 < .2Hz/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) On Radios with built in sound card interfaces (e.g. Icom 7200) it is possible to use the radios built in sound card to send and receive SSB audio.
7) The sound card or DSP processor *MUST* be able to support a real or interpolated sampling frequency of 8000Hz +/- .1% (+/-1000 parts per million)
8) 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) It may be possible to reduce the PC requirement in the future at the expense of session throughput.

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 either 4 or 5 modulation modes supported depending on the desired session bandwidth:

1) 200Hz BW: 1 Carrier at 1500.0 Hz
   Modes: 4FSK, 4PSK TCM, 8PSK TCM, 16PSK TCM
2) 500 Hz BW: 3 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
   Modes 1 Carrier 4FSK, 3 Carrier 4FSK, 3 Carrier 4PSK TCM, 3 Carrier 8PSK TCM, 3 carrier 16 PSK TCM.
3) 1000 Hz BW: Carriers nominally start at an audio frequency of 1125.0 H Hz and are spaced equally at 125 +/- .1% Hz. Carrier 1 is 1125 Hz and carrier 7 is 1875.0 Hz, The pilot carrier is carrier 4 @ 1500.0 Hz
   Modes 3 Carrier 4FSK, 7 Carrier 4FSK, 7 Carrier 4PSK TCM, 7 Carrier 8PSK TCM, 7 Carrier 16 PSK TCM.
4) 2000 Hz BW: Carriers nominally start at an audio frequency of 625.0 H Hz and are spaced equally at 125 +/- .1% Hz. Carrier 1 is 625 Hz and carrier 15 is 2375.0 Hz, The pilot carrier is carrier 8 @ 1500.0 Hz
   Modes 7 Carrier 4FSK, 15 Carrier 4FSK, 15 Carrier 4PSK TCM, 15 Carrier 8PSK TCM, 15 Carrier 16 PSK TCM.

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 (256 ms) 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} := 4 adjacent 4FSK symbols. These 4 symbols encode the 4 bit frame type with an extended 8,4 hamming code.

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 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 PSK modulated with a root 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 100% of the weight of the pilot carrier.

3) 3 simultaneous carriers PSK modulated with a root raised cosine envelope. Each carrier is weighted at 40% of the maximum PEP value.

4) 3 simultaneous carriers each 4FSK (one of 4 tones). Each carrier weighted at 1/3 the maximum PEP value.

5) 7 simultaneous carriers PSK modulated with a root raised cosine envelope. Each carrier is weighted at 20% of the maximum PEP value.

6) 7 simultaneous carriers each 4FSK (one of 4 tones). Each carrier weighted at 1/7 the maximum PEP value

7) 15 simultaneous carriers PSK modulated with a root raised cosine envelope. Each carrier is weighted at 10% of the maximum PEP value.

8) 15 simultaneous carriers each 4FSK (one of 4 tones). Each carrier weighted at 1/15 the maximum PEP value.

(Note: these carrier weightings use limited clipping to reduce the crest factor.

For PSK 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. There is no reference symbol for 4FSK modes.

Byte := the number of contiguous symbols to make one byte. 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 Viterbi encode Trellis Coded Modulation (Pragmatic TCM) PSK (differential phase shift keying) 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 rogue signal. 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. For Detailed frame parameters see the spread sheet in Appendix A.

6.2.1 Control frames:

$\mathbf{F_{crq}}$ Connect ReQuest frame: 1 Car Viterbi 4PSK = type 0

Sent by the station initiating the connection (Client). Contains call signs of calling and target stations and 2 byte sumcheck. The session Bandwidth is set by the answering (Server) station.

F_{crq} Encoding:

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

B_{data} := Calling call sign & Remote call sign. Callsigns are packed to 12 byte array of 6 bit characters and must be A-Z, 0-9 with an optional –ssid of 0 – 15. Function FormatCallsToByte is used to pack the 12 byte array.

Note: B_{mode} has been eliminated and server sets session bandwidth with the type of Idle frame used to acknowledge.

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 4 check parity bytes from a shortened RS (255,251) code correcting up to 2 bytes

F_{crq} Total payload 18 bytes including CRC16 and RS correction check bytes and is always sent using single carrier Viterbi encoded 4PSK modulation (1 user bit per symbol)

Total symbols = 171 + leader extension = 2.736 sec + leader extension

F_{ccf} Coded Control Frame (1 Car 4FSK 31.25 baud = type 1)

Handles the following sub types by 1 byte code in the control frame:

F_{drq} Disconnect Request (code HFF)

Note: The following change allows Server to set session Bandwidth. This replaces the previous idle with a bandwidth specific idle sent by the server setting the bandwidth for the session. From that point all idles would be the same for the entire session (no bandwidth changes during a session).

F_{idl0} Idle0 200Hz (Sent when no data in output buffer) Code H00

F_{idl1} Idle1 500Hz (Sent when no data in output buffer) Code H01

F_{idl2} Idle2 1000Hz (Sent when no data in output buffer) Code H02

F_{idl4} Idle4 2000Hz (Sent when no data in output buffer) Code H04

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 & 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 3 check parity bytes from a shortened RS (15,9)

Total symbols = 33 + .5 * leader extension (equivalent 31.25 baud symbols) = 1.056 seconds plus leader extension

6.2.2 ACK Frames:

F_{ack} := Ack (1 Car 4FSK FEC = type 2)

Handles ACK for all carrier modes

F_{ack}:= Pilot & 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 information bits and each 4 bit value is sent as a 4 bit character (six 4-bit characters)

B_{RS} is a 16 bit field consisting of four 4 bit Characters from a shortened RS (15,11) code of 4 bit characters. This code corrects up to two 4 bit characters in the 10 character total ACK.

Total symbols = 33 + .5 * leader extension (equivalent 31.25 baud symbols) = 1.056 seconds plus leader extension

6.2.3 Data frames:
Data frames consists of four 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 R-S encoding) if the data is not decoded correctly it is sent again as a type 2 (strong R-S Parity only). This strong R-S parity is appended to the *data* portion of the *previous* Data + Weak R-S Encoding (the Weak R-S parity bytes are discarded) and a new more robust R-S decode is attempted. Data frames alternate between Type 1 and Type 2 until there is a successful decode. Data type 2 is distinguished from type 1 by using the ones compliment of the Session ID. Some form of data summation (analog memory ARQ) *MAY* be used to average repeated Data + Weak R-S or Strong R-S Parity only to improve decoding performance.

F_{d16TCM} 16PSK Pragmatic TCM

Encoding for type 1 (Data + weak Reed-Solomon error correction):

F_{d16TCM} := Pilot & S_r & B_{sid} & B_{psn} & B_{bc} & B_{data} & B_{pad} & B_{sch} & B_{scl} & B_{RS} & B_{00}

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 96 bytes)

B_{pad} is remaining B_{00} if required to fill B_{data} frame if < 96 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 20 byte RS weak parity using a shortened RS code of 235,255 (10 error correcting)

B_{00} is a one byte fill necessary to complete the 21 bytes due to the 3 bits/symbol

Encoding for type 2 (Extended RS Parity):

F_{d16TCM} := Pilot & S_r & B_{sid} & B_{RSX} & B_{00}

Where:

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

B_{RSX} are the 120 extended Reed Solomon Parity

B_{00} is a one byte fill necessary to complete the 121 bytes due to the 3 bits/symbol

Data Frames using this modulation mode:

One carrier 16PSK, 200 Hz BW
Three carrier 16PSK, 500 Hz BW

Seven carrier 16PSK, 1000 Hz BW
Fifteen carrier 16PSK, 2000 Hz BW

F_{d8TCM} 8PSK Pragmatic TCM

Encoding for type 1 (Data + weak Reed-Solomon error correction):

F_{d8TCM} := 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_{d8TCM}** := 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).

Data Frames using this modulation mode:

One carrier 8PSK, 200 Hz BW
Three carrier 8PSK, 500 Hz BW
Seven carrier 8PSK, 1000 Hz BW
Fifteen carrier 8PSK, 2000 Hz BW

F_{d4TCM} 4PSK Pragmatic TCM

Encoding for type 1 (Data + weak Reed-Solomon error correction):

F_{d4TCM} := 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 24 bytes)

B_{pad} is remaining B_{00} if required to fill B_{data} frame if < 24 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 10 byte RS weak parity using a shortened RS code of 245,255 (5 error correcting)

Encoding for type 2 (Extended RS Parity): $\mathbf{F_{d4TCM}}$:= Pilot & S_r & B_{sid} & B_{RSX}
Where:

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

B_{RSX} are the 38 extended Reed Solomon Parity bytes *only* of a strong RS code 217,255 (19 error correcting).

Data Frames using this modulation mode:

- One carrier 4PSK, 200 Hz BW
- Three carrier 4PSK, 500 Hz BW
- Seven carrier 4PSK, 1000 Hz BW
- Fifteen carrier 4PSK, 2000 Hz BW

$\mathbf{F_{d4FSK}}$ 4FSK modulation @ 31.25 baud

Encoding for type 1 (Data + weak Reed-Solomon error correction):
$\mathbf{F_{d4FSK}}$:= 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/carrier)
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)

$\mathbf{F_{d4FSK}}$ Encoding for type 2 (Extended RS Parity):
$\mathbf{F_{d4FSK}}$:= 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).

For 4FSK each carrier group of 4 tones is separated by 4 x 31.25 or 125 Hz.
Data Frames using this modulation mode:

- One carrier 4FSK, 200 Hz BW
- Three carrier 4FSK, 500 Hz BWF
- Seven carrier 4FSK, 1000 Hz BW
- Fifteen carrier 4FSK, 2000 Hz BW

6.3 Protocol Details

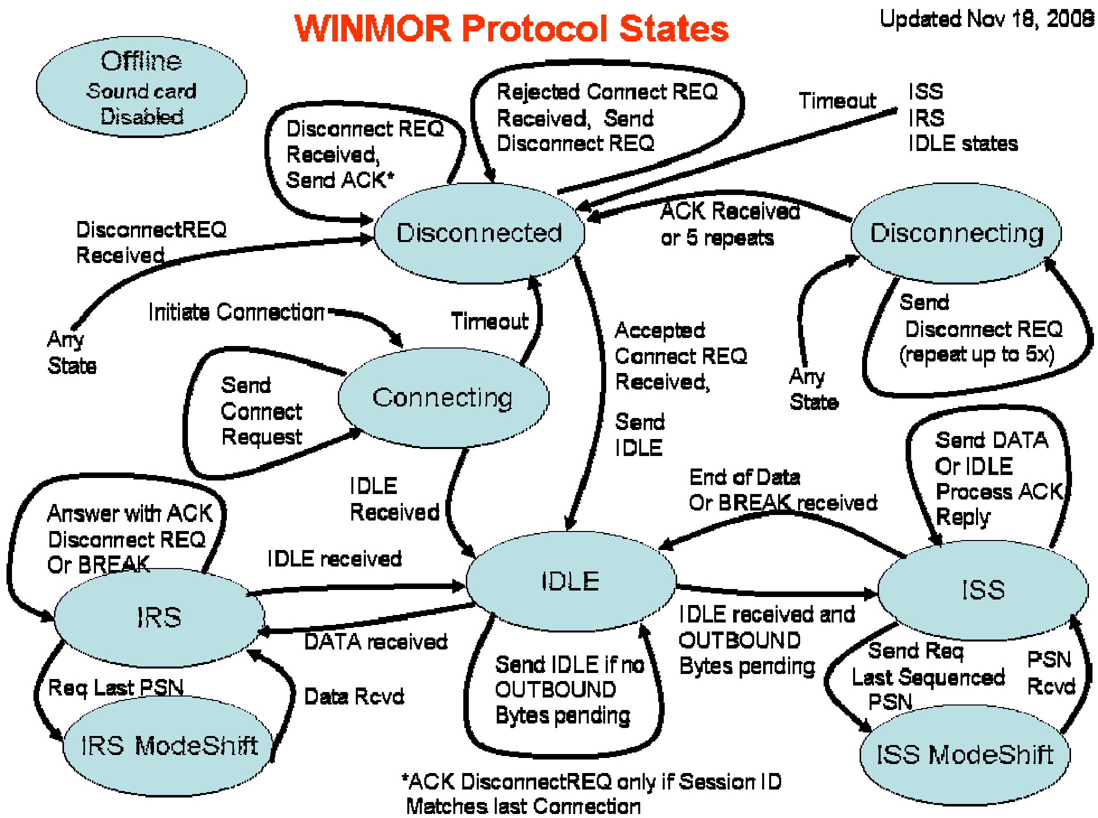

Fig 6 – 1 Simplified Protocol State diagram:

6.3.1 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) All other states, events, actions and state sequencing details are shown in the Protocol rules of Appendix C.

7.0 Example Forwarding Scenarios:

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

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(0)
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 (repeat)		
		DISCONNECTED	IDLE
CONNECTING	CONREQ (missed IDLE reply)		
		IDLE	IDLE
IDLE	IDLE		
		ISS	DATA
IDLE	nothing (missed DATA)		
		ISS	DATA(repeat)
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		
		IDLE	(nothing..no decode)
DISCONNECTING	DISCONNECT REQ (repeat)		
		DISCONNECTED	ACK(0)
DISCONNECTING	DISCONNECT REQ (missed ACK, repeat)		
		DISCONNECTED	ACK(0)
DISCONNECTED			

Appendix A: WINMOR Mode Rate Worksheet

(details of frame construction for all modes, all bandwidths)

'WINMOR Mode Rate Worksheet (200, 500, 1000, 2000Hz B **Revised: 9/14/2009** **Rick Muething, KN6KB**

Mode Description	Info b/Sym	Samp/ Sym	Baud	# of car	~BW (Hz)	Raw bps/H:	Leader (Sym)	OH/Car (Sym)	PL/Car (sym)	RS-FEC/ Car (sym)	Frame Length (sym)	Frame Length (sec)	ACK (sec)	Rx+Tx +O (sec)	Tot Cycle Len (sec)	Net max Throughput (bits/sec)	(byt/min)
15 Car 16PSK Prag TCM + RS	3	128	62.50	15	2000	1.41	27	16	256	56	355	5.680	1.056	0.3	7.036	1637.3	12280
15 Car 8PSK Prag TCM + RS	2	128	62.50	15	2000	0.94	27	24	256	48	355	5.680	1.056	0.3	7.036	1091.5	8186
15 Car 4PSK Prag TCM + RS	1	128	62.50	15	2000	0.47	27	48	192	80	347	5.552	1.056	0.3	6.908	416.9	3127
15 Car 4FSK +RS	2	256	31.25	15	2000	0.47	13	24	64	64	165	5.280	1.056	0.3	6.636	289.3	2170
7 Car 16PSK Prag TCM + RS	3	128	62.50	7	1000	1.31	27	16	256	56	355	5.680	1.056	0.3	7.036	764.1	5731
7 Car 8PSK Prag TCM + RS	2	128	62.50	7	1000	0.88	27	24	256	48	355	5.680	1.056	0.3	7.036	509.4	3820
7 Car 4PSK Prag TCM + RS	1	128	62.50	7	1000	0.44	27	48	192	80	347	5.552	1.056	0.3	6.908	194.6	1459
7 Car 4FSK +RS	2	256	31.25	7	1000	0.44	13	24	64	64	165	5.280	1.056	0.3	6.636	135.0	1013
3 Car 16PSK Prag TCM + RS	3	128	62.50	3	500	1.13	27	16	256	56	355	5.680	1.056	0.3	7.036	327.5	2456
3 Car 8PSK Prag TCM + RS	2	128	62.50	3	500	0.75	27	24	256	48	355	5.680	1.056	0.3	7.036	218.3	1637
3 Car 4PSK Prag TCM + RS	1	128	62.50	3	500	0.38	27	48	192	80	347	5.552	1.056	0.3	6.908	83.4	625
3 Car 4FSK +RS	2	256	31.25	3	500	0.38	13	24	64	64	165	5.280	1.056	0.3	6.636	57.9	434
1 Car 16PSK Prag TCM + RS	3	128	62.50	1	200	0.94	27	16	256	56	355	5.680	1.056	0.3	7.036	109.2	819
1 Car 8PSK Prag TCM + RS	2	128	62.50	1	200	0.63	27	24	256	48	355	5.680	1.056	0.3	7.036	72.8	546
1 Car 4PSK Prag TCM + RS	1	128	62.50	1	200	0.31	27	48	192	80	347	5.552	1.056	0.3	6.908	27.8	208
1 Car 4FSK +RS	2	256	31.25	1	200	0.31	13	24	64	64	165	5.280	1.056	0.3	6.636	19.3	145
1 Car Connect Request (V4PSK) +RS	1	128	62.50	1	200	0.31	27	0	104	32	163	2.608					
1 Car Coded Control 4FSK + RS	2	256	31.25	1	200	0.31	13	0	8	8	29	0.928					
1 Car ACK 4FSK +RS	2	256	31.25	1	200	0.31	13	0	12	8	33	1.056					

Leader extension (symbols 0-16)	**0**	See Note 2
Calculated Leader extension (ms)	0	

Notes: 1) The 4FSK mode was modified for lower user data and higher RS FEC and now runs slower than the 4PSK mode.

2) Leader extension up to 16 symbols (256 ms) may be used for slow switchover Trancievers or VOX operated PTT

3) The above modes yield the following speed ranges depending on session bandwidth:
note these selections may change based on testing!
2000 Hz BW Sessions: 15x16PSK, 15x8PSK, 15x4PSK, 15x4FSK, 7x4FSK ~ 12:1 speed range
1000 Hz BW Sessions: 7x16PSK, 7x8PSK, 7x4PSK, 7x4FSK, 3x4FSK ~ 13:1 speed range
500 Hz BW Sessions: 3x16PSK, 3x8PSK, 3x4PSK, 3x4FSK,1x4FSK ~ 18:1 speed range
200 Hz BW Sessions: 1x16PSK, 1x8PSK, 1x4PSK , 1x4FSK ~ 6:1 speed range

4) Session BW is set by Server (answering) station using one of 4 coded Idle frames (200, 500, 1000, or 2000 Hz)

5) All PSK modes use pragmatic Trellis Code Modulation (one redundancy bit/symbol) and use the standard R=1/2, K=7 (NASA Voyager) Viterbi Encoder/Decoder based on Phil Karns Code.

6) Rx + Tx + O refers to the receive to transmit, transmit to receive and software overhead delays and is typical for modern hardware. The protocol acutally measures the latency due to RX>TX switchover, sound card and CPU processing latency.

Appendix B: Pragmatic Trellis Code Modulation (PTCM)

Trellis code modulation is a combination of FEC encoding with PSK modulation used to improve the Bit error rate of uncoded PSK modulation. Pragmatic means using standard available encoders/decoders (e.g. Viterbi) in place of the slightly more optimized Ungerbroeck Trellis coded modulation encoder/decoders. Pragmatic TCM is within about .2 dB of the optimized Ungerbroeck code of the same constraint length over the typical bit error rates encountered.

Figure B-1 shows the block diagram of the PTCM encoder as employed in the WINMOR midrange speed mode (Trellis 8PSK). Similar schemes are used for the 4PSK and 16PSK modulation modes. In all cases the TCM adds one bit to the user symbol doubling the number of PSK phases per symbol.

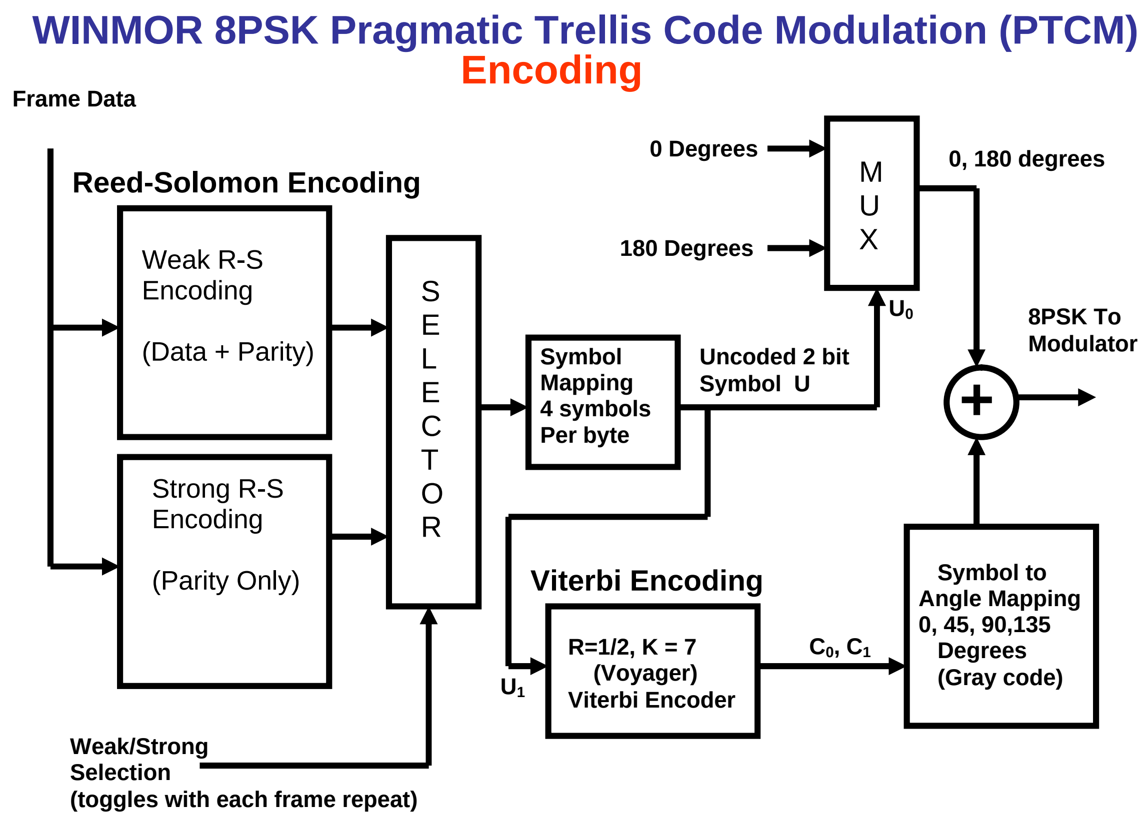

Figure B-1 WINMOR 8PSK PTCM Encoding

The encoding is summarized as follows:
A frame consists of 64 Payload (user data) bytes + 6 bytes of overhead. This frame is encoded using a weak R-S (Reed-Solomon) code (243,255) shortened to 82 bytes. This code will correct up to 6 byte errors in the 82 transmitted bytes. The resulting 82

bytes are mapped into 328 symbols of 2 bits each. The most significant symbol Bit U_0 is not FEC coded and selects an angle of 0 degrees ($U_0 = 0$) or 180 degrees ($U_0 = 1$) The least significant bit U_1 is fed into a standard (NASA Voyager) R= ½ , K=7 Viterbi Encoder which produces 2 FEC coded output bits C_0 and C_1 for each input bit U_1.
C_0 and C_1 are mapped to one of 4 phase values 0, 45, 90 or 135 using a gray code mapping. This phase value is added to the output of the multiplexer (0 or 180 degrees) to obtain the final 8PSK modulation angle (0 to 315 degrees in 45 degree steps)

Figure B-2 is a diagram of the PTCM decoder used by WINMOR's Trellis 8PSK mode.

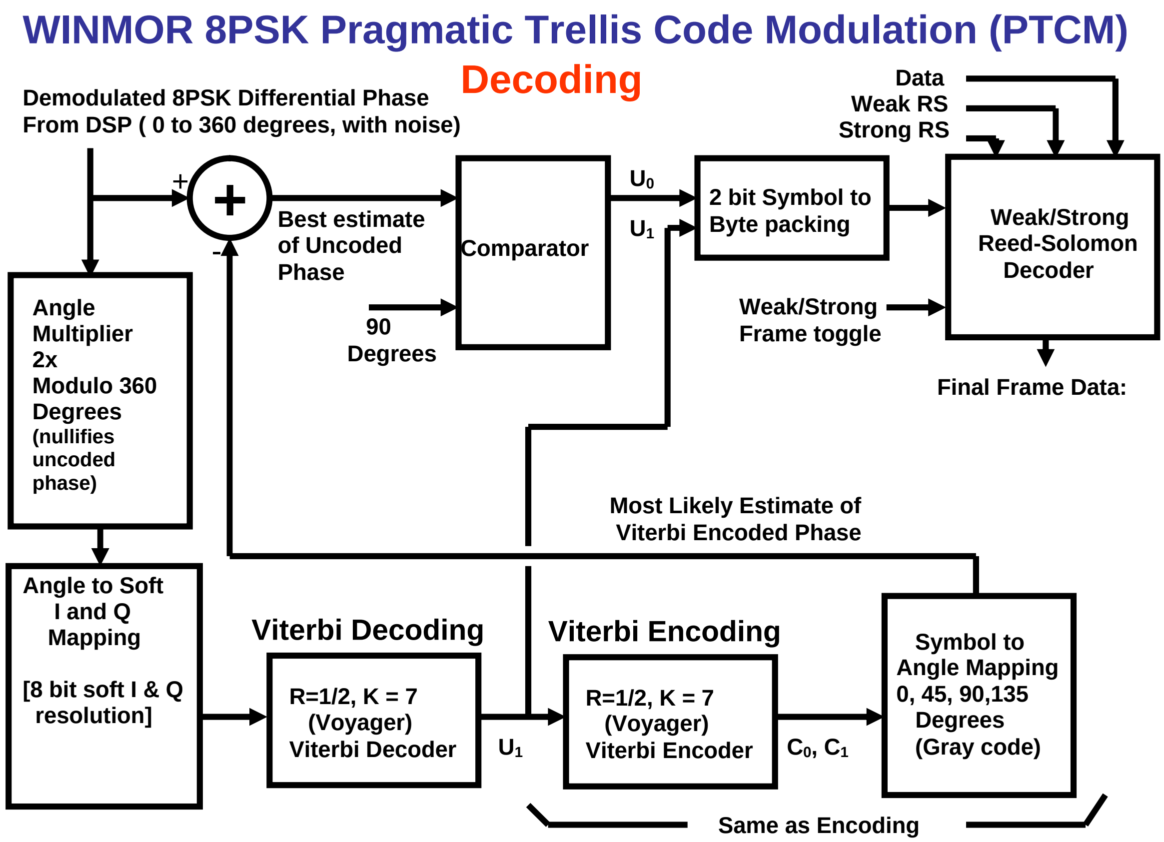

Figure B-2 WINMOR 8PSK PTCM Decoding

The decoding operation is somewhat more complicated and summarized as follows: The demodulated differential phase angle from the DSP (0 to +360 degrees, with noise) is the input to the decoder (one angle per symbol time). The angle is doubled, modulo 360 which nullifies the 0 or 180 degree component of the uncoded bit. The resulting phase value 0 to 360 degrees in nominal (now 90) degree steps) is recoded to soft I and Q values which will be the soft I and Q inputs for two binary values input to the Viterbi encoder. Figure B-3 shows the mechanism for the angle to soft I

& Q mapping. The Viterbi decoder generates the best estimate for the original binary bit for each pair of soft I and Q inputs. This bit becomes the decoded symbol least significant bit U_1. U_1 is also fed into a Viterbi Encoder and Symbol to Angle mapper (exactly the same as is used in the encoder in Fig B-1) to yield the most likely estimate of the original Viterbi encoded phase angle value. This most likely estimate (0 to 135 degrees in 45 degree steps) is then subtracted from the original differential phase angle from the DSP to yield a best estimate of the uncoded bit phase (nominally 0 or 180 degrees). The resultant best estimate is compared to 90 degrees to generate U_0 the most significant bit of the symbol. The symbols composed of U_0 and U_1 are then packed 4 symbols/byte for Reed-Solomon decoding. If this is the initial transmission of the frame the weak R-S code of 243,255 (6) shortened to 82 bytes is used. This weak R-S code will correct up to 6 byte errors in the total 82 bytes transmitted. If the frame is a repeat (requested after a decode failure of initial attempt) then the 82 bytes of data are interpreted as the ID + parity only component of a strong R-S code of 175,255 code which can correct up to 40 errors of the shortened 150 code consisting of the 70 original payload + overhead bytes concatenated with the 80 strong R-S parity bytes. Thus the strong R-S code can correct up to 40 errors of the 150 bytes of data plus strong parity (transmitted over two frames).

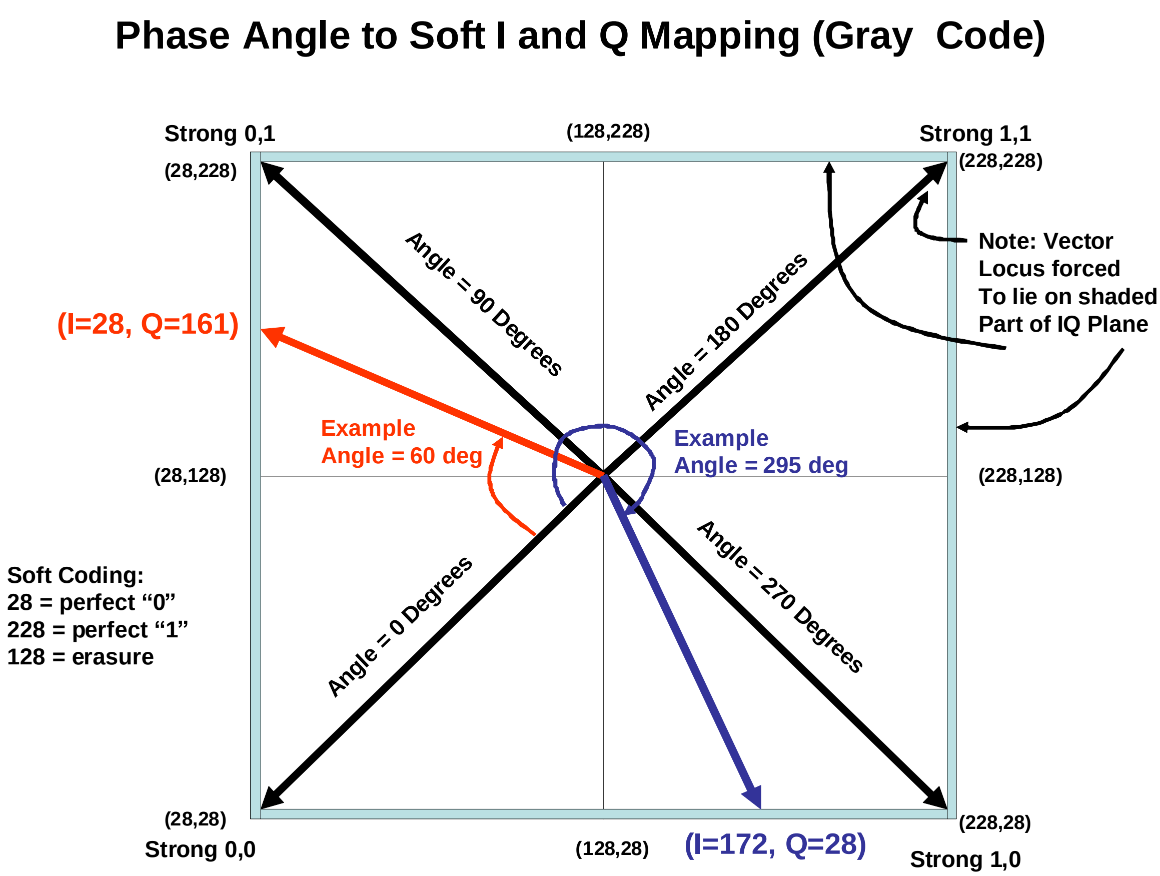

Figure B-3 Phase Angle to Soft I and Q Mapping with examples.

A similar approach to PTCM encoding and decoding is done on both the 4PSK mode (no uncoded bits, 2 Viterbi bits) and the 16PSK mode (2 uncoded bits, 2 Viterbi bits).

References:

1) A Pragmatic Approach to Trellis-Coded Modulation. A. Viterbi, J. Wolf, E. Zejavo, R. Padovani IEEE Communications Magazine July 1989, pp11-19
http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=31452

2) Data Recovery in Differentially Encoded Quadrature Phase Shift Keying. J. Bard, M. Nezami, and M. Diaz Mnemonics, Inc Melbourne, FL.
http://whitepapers.silicon.com/0,39024759,60446221p,00.htm

3) Error Control Coding, Second Edition. Shu Lin and Daniel Costello
Pearson Prentice Hall 2004 ISBN 0-13-042672-5

4) Trellis Coded Modulation Tutorial, Charan Langton, 2004.
http://www.complextoreal.com/chapters/tcm.pdf

Appendix C: Detailed State Change Rules

These rules provide the detail that implements the WINMOR Protocol State Diagram shown in Fig 6-1.

1 Rules for IDLE State

1.1 Event = Connect request
Conditions: Successful Decode to Target station
Action: Send Bandwidth specific Idle, no repeat
Next State: IDLE

1.2 Event = IDLE Received
Conditions: No Outbound Pending
Action: Send Idle, No repeat
Next State = IDLE

1.3 Event: IDLE Received
Conditions: Outbound Pending
Action: Send Data at current data mode
Next State: ISS

1.4 Event Disconnect Request to Target station
Conditions: none
Action: Send ACK 0, Schedule optional CWID
Next State: DISCONNECTED

1.5 Event: Inactivity Timeout
Conditions: none
Action: Send Disconnect Request with Repeat
Next State: DISCONNECTING

2 Rules for DISCONNECTED State

2.1 Event = Connect request Frame detect (before Decode)
Conditions: Decode Frame type Connect Request
Action: none
Next State: CONNECT PENDING

2.2 Event: Disconnect Request
Conditions: Session ID matches last session
Action: Send ACK(0) with last session ID
Next State: DISCONNECTED

3 Rules for CONNECTING State

3.1 Event = Bandwidth specific IDLE received
Conditions: Matches current session ID
Action: Send IDLE, set Repeat ON
Next State: IDLE

3.2 Event: Connect Request Timeout
Conditions: none
Actions: none
Next State: DISCONNECTED

4 Rules for CONNECT PENDING State

4.1 Event = Successful Decode to Target Call sign
Conditions: Target Call sign matches local call sign
Action: Send bandwidth specific IDLE, set Repeat OFF
Next State: IDLE

4.2 Event: Decode Failure
Conditions: Sumcheck fail or Target <> Local call sign
Actions: none
Next State: DISCONNECTED

5 Rules for DISCONNECTING State

5.1 Event = Control Frame Timeout
Conditions: Disconnect Repeat count < 5
Action: Send Disconnect Request, set Repeat ON
Next State: DISCONNECTING

5.2 Event: Control Frame Timeout
Conditions: Disconnect Repeat count > = 5
Actions: none
Next State: DISCONNECTED

6 Rules for IRS State

6.1 Event = Data Received, Good match to ID bits
Conditions: Session ID match
Action: Send ACK for each carrier correct, no repeats
Next State: IRS

6.2 Event: Data Received, Poor match to ID
Conditions: Session ID mismatch
Actions: none
Next State: IRS

6.3 Event: Control Received, Request Last PSN
Conditions: Session ID match, Sumcheck OK
Actions: Send ACK with MSBit set and LSByte containing Last PSN
Next State: IRS MODESHIFT

7 Rules for IRS MODE SHIFT State

7.1 Event = Data Received
Conditions: Session ID Match
Action: Send ACK for each carrier no repeats
Next State: IRS

7.2 Event: Control Frame Idle received
Conditions: none
Actions: Send Idle, No repeate
Next State: IDLE

8 Rules for ISS State

8.1 Event = ACK received
Conditions: OB bytes pending after ACK processed, no speed shift required
Action: Send next OB Packet with Repeat
Next State: ISS

8.2 Event: ACK received
Conditions: OB bytes pending after ACK processed, speed shift
Actions: Send Control request last PSN with repeat
Next State: ISS MODESHIFT

8.3 Event: ACK received
Conditions: no OB bytes pending after ACK processed
Actions: Send Control Idle with repeat?
Next State: IDLE

9 Rules for ISS MODE SHIFT State

7.1 Event = PSN Received
Conditions: Session ID Match, Sumcheck OK, OB Packets Remaining
Action: Send next Data packet
Next State: ISS

7.2 Event = PSN Received
Conditions: Session ID Match, Sumcheck OK, no OB Packets Remaining
Action: Send Idle, set repeat
Next State: IDLE

7.3 Event: Control Frame Idle received ,Sumcheck Fail
Conditions: Sumcheck Fail
Actions: Send Request Last PSN, repeat
Next State: ISS MODE SHIFT

7.4 Event: Inactivity Timeout
Conditions: none
Actions: Send Disconnect Request, repeat
Next State: DISCONNECTING