3GPP2 C.S0028-B

Version 1.0

May 2011

3RD GENERATION PARTNERSHIP PROJECT 2 "3GPP2"

TTY/TDD Minimum Performance Specification

Revision History

Revision		**Date**
C.S0028-0 v1.0	CDMA TTY/TDD Minimum Performance Specification	May 2001
C.S0028-0 v2.0	CDMA TTY/TDD Minimum Performance Specification	Dec. 2001
C.S0028-A v1.0	TTY/TDD Minimum Performance Specification	Jan. 2004
C.S0028-B v1.0	TTY/TDD Minimum Performance Specification	May 2011

Table of Contents

List of Tables

FOREWORD

(This foreword is not part of this specification.)

This document specifies the minimum performance requirements and a test procedure for verifying solutions for transporting the Baudot code over cdma2000 wireless networks using 3GPP2 13K Vocoder, EVRC, SMV, EVRC-B, EVRC-WB, and EVRC-NW, corresponding to Service Options 17, 3, 56, 68, 70, and 73, respectively

1 INTRODUCTION

1.1 Scope

Persons with hearing impairments or speech impediments are able to communicate over the telephone using TTY/TDD (teletype/text telephone devices for the deaf), hereafter referred to as TTY devices. These devices typically have a keyboard and a 2-line display, allowing a TTY user to type messages over the phone lines to another TTY user. TTY devices transmit the characters at either 45.45 bps or 50 bps using the Baudot code, a binary frequency shift keyed scheme that is carrierless, half duplex, and without error protection.

In June 1996, the FCC adopted Section 20.18(c) of the Commission's Rules. It requires that, as of October 1, 1997, all covered wireless carriers must be capable of transmitting 911 calls from individuals with speech or hearing disabilities through means other than mobile radio handsets, e.g., through the use of TTY devices. The pre-existing digital wireless networks have rendered TTY conversations unintelligible. The frame and bit errors that occur over the air, as well as the variable rate speech coding, conspired to create unacceptably high character error rates.

The purpose of this standard is to establish minimum performance requirements and a test procedure for verifying solutions for transporting the Baudot code over cdma2000 [7] wireless networks using 3GPP2 13K Vocoder [2], EVRC [1], SMV [8], EVRC-B, EVRC-WB, and EVRC-NW [1], corresponding to Service Options 17, 3, 56, 68, 70, and 73, respectively. An electronic package containing tools and test vectors for the purpose of performing these minimum performance tests accompanies this standard.

For the remainder of this document, the term test vocoder refers to the speech coder with TTY extension for which compliance with this document is being verified. The term reference vocoder refers to the C simulation of the vocoder with the most recent TTY extension, which is provided in the accompanying electronic package.

1.2 Requirements Language

This specification uses the following verbal forms: "Shall" and "shall not" identify requirements to be followed strictly to conform to the standard and from which no deviation is permitted. "Should" and "should not" indicate that one of several possibilities is recommended as particularly suitable, without mentioning or excluding others; that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is discouraged but not prohibited. "May" and "need not" indicate a course of action permissible within the limits of the standard. "Can" and "cannot" are used for statements of possibility and capability, whether material, physical, or causal.

1.3 References

1.3.1 Normative References

[1] C.S0014, *Enhanced Variable Rate Codec, Speech Service Options 3, 68, 70, and 73 for Wideband Spread Spectrum Digital Systems*.

[2] C.S0020, *High Rate Speech Service Option 17 for Wide Band Spread Spectrum Communication Systems*.

[3] ITU-T Recommendation V.18, Operational and Interworking Requirements for DCEs Operating in the Text Telephone Mode, November 2000.

[4] Reserved

[5] TTY Over Cellular Laboratory and Field Test Procedure, Lober & Walsh Engineering, Inc., September 2, 1998.

[6] Reserved.

[7] C.S0005, *Upper Layer (Layer 3) Signaling Standard for cdma2000 Spread Spectrum Systems*.

[8] C.S0030, *Selectable Mode Vocoder (SMV) Service Option for Wideband Spread Spectrum Communication Systems*.

1.3.2 Informative References

[9] TIA/EIA/IS-840-A, Minimum Performance Standards for Text Telephone Signal Detector and Text Telephone Signal Regenerator, September 2001.

2 THE BAUDOT CODE

TTY devices transmit characters using a 45.45 bps or 50 bps Baudot code. Each bit has a nominal duration of 22 ms for 45.45 bps and 20 ms for 50 bps. A character consists of 1 start bit, 5 data bits, and 1-2 stop bits. The code uses a carrierless, binary FSK signaling scheme. A mark, or "1", is transmitted with a 1400 Hz tone. The space, or "0", is transmitted with an 1800 Hz tone.

In TTY devices, the first tone of a character may be the space tone of the start bit, or a mark tone preceding the start bit. Furthermore, there is a mark hold tone, which extends the length of time the stop bit is transmitted following the last character from 150 ms to 300 ms. The mark hold tone is not transmitted if the character is immediately followed by another character. The mark hold tone prevents the transmitting TTY device from receiving its echo and mistaking it for an incoming character. This mechanism is effective for wireline calls but may not be sufficient to mitigate echo in wireless scenarios because of the longer delays. Refer to the TTY specification for a solution to mitigating echo in wireless networks.

With 5 data bits, the Baudot code is only capable of 32 different characters, which is not enough to represent the alphabet, numbers, and punctuation. This problem is remedied by introducing "shift" characters that change the way the receiving TTY device interprets the 5 data bits. There exists the letters shift character (31) and the figures shift character (27). Following the letters shift character, the receiving TTY device interprets the 5-bit code using the letters library, which consists of letters of the alphabet. The figures library contains numbers, punctuation, and special characters. The TTY device automatically sends the shift character when the user switches from characters in one library to the other. The consequence of this scheme is that if the shift character is received incorrectly, every character thereafter will be misinterpreted with the wrong library until another shift character is received correctly, causing a string of character errors.

3 CHARACTER ERROR RATE

Because of the shift character, it is possible to receive the 5-bit Baudot code correctly, but print the wrong character if the shift key was received in error. Therefore, there exist two different measures of character error rates; Printable Character Error Rate (PCER) and Total Character Error Rate (TCER) [5]. PCER compares the actual text sent and received without any consideration to the underlying method of transfer, which involved conversion to and from Baudot with the insertion of shift state characters. TCER recognizes the Baudot character set and the insertion of shift characters by comparing the 5-bit Baudot code directly.

For both PCER and TCER, the character error rate is computed by counting the number of missed and changed characters. The number of characters in the reference file is considered the total number of characters. *PCER = (missed + changed)/total* for the printable characters and *TCER = (missed + changed)/total* for the 5-bit Baudot code character set. The number of added characters is not considered in the character error rate.

The TTY Forum has set a goal for the character error rate to be less than 1%, however, there was no agreement as to whether that would be TCER or PCER. The TTY solution adopted for CDMA is capable of far exceeding this goal. For the purposes of this minimum performance requirement, only the TCER shall be used because it does not depend on the content of the TTY sequences. Furthermore, the TCER shall be calculated using the tools provided with this specification in conjunction with the UNIX™ diff command. See Section 5 for a description of the tools.

4 TTY/TDD TEST CASES

This section describes the test cases that are exercised in this minimum performance specification. Compliance with these requirements shall be verified by processing the accompanying test vectors and using the accompanying tools, as described in Section 6.

4.1 Interoperability with Reference TTY Vocoder

The test vocoder shall interoperate with the simulation of its corresponding reference vocoder. Packets from the test encoder shall interoperate with the reference decoder, and vice versa.

4.2 Interoperability with Reference Non-TTY Capable Vocoder

The test vocoder shall interoperate with legacy, non-TTY capable vocoders. Packets from the test encoder shall interoperate with the reference decoder, and vice versa, for TTY and non-TTY inputs. Performance with the test vocoder and non-TTY vocoders shall be comparable to the performance between two non-TTY capable phones.

4.3 Input Level

The test vocoder shall meet the performance requirements for detection and regeneration when the input signal level ranges from –45 dBm to –5 dBm.

4.4 Bit Duration

The test vocoder shall meet the performance requirements for detection and regeneration when the input bit durations are in the range 22 ms ± 0.4 ms for 45.45 bps and 20 ms ± 0.4 ms for 50 bps.

4.5 Stop Bit

The test vocoder shall meet the performance requirements for detection and regeneration with stop bits ranging in duration from 1 to 2 bits, where the bits may be in the range 22 ms ± 0.4 ms for 45.45 bps and 20 ms ± 0.4 ms for 50 bps.

4.6 Phase Changes

The performance of the test vocoder shall be independent of the input’s phase changes between TTY bits. These phase changes include:

Zero phase

Random phase

4.7 Frequency Deviations

The test vocoder shall meet the performance requirements for detection and regeneration when the mark tone (1400 Hz) and the space tone (1800 Hz) deviate from their nominal values by ± 4.0%.

4.8 Typing Mode

For most applications, text telephone conversations are conducted live, with manual typists. The TTY call is characterized by bursts of typing followed by pauses. This is referred to as manual mode. Some TTY devices are capable of transmitting stored conversations, or files. In this case, a long, uninterrupted stream of TTY characters characterizes the TTY call. This is referred to as streaming mode. The test vocoder shall be capable of supporting both typing modes.

4.9 Preamble

In TTY devices, the first tone of a character may be the space tone of the start bit, or a mark tone preceding the start bit, which is referred to as the preamble. The test vocoder shall meet the performance requirements for detection and regeneration whether or not the preamble is present.

4.10 Mark Hold Tone

The mark hold time defines an additional period of time during which the TTY transmits a mark hold tone (1400 Hz) following the last character transmitted. The mark hold tone is not transmitted between each character if the character is followed immediately by another character. The test vocoder shall regenerate a mark hold tone for 300 ms or until the next character is regenerated, whichever is less. The test vocoder shall meet the performance requirements for detection and regeneration:

When characters are followed by the mark hold tone.

When characters are followed by silence.

4.11 Voice Carryover/ Hearing Carryover (VCO/HCO)

When the TTY algorithm is enabled, the test vocoder shall be capable of supporting VCO/HCO, without any further interaction from the network or the end-users. That is to say that the TTY algorithm shall be capable of switching between TTY and non-TTY inputs freely within a call without degrading its TTY performance.

4.12 Tandem

The test vocoder shall be capable of tandeming with itself and with the reference vocoder for TTY calls.

4.13 Variable Rates

The test vocoder shall be capable of decoding TTY information from the compressed speech packets for all rates that support TTY. That is to say, that EVRC, 3GPP2 13K vocoders shall decode TTY information from both full rate and half rate frames. SMV, EVRC-B, EVRC-WB, and EVRC-NW shall decode TTY information from half rate data packets only. The rates used in future vocoders are to be determined.

4.14 Reduced Rate Modes

There are certain scenarios where the vocoder is requested to reduce its rate artificially in order to satisfy network requirements. An example of this is dim-and-burst signaling or blank-and-burst signaling. During a TTY call, the network’s request for reduced-rate modes shall take precedence and the TTY information shall be provided where possible. For example, if the test vocoder is requested to supply half rate packets during a TTY transmission, the test vocoder shall produce half rate packets as requested with the TTY information embedded.

TTY transmission is not expected in cases where the speech path is muted, as is the case in blank-and-burst signaling. In these cases, the TTY information shall be recovered as best as possible, with the understanding that character errors may occur. As a direct consequence, to achieve the required performance goals of this specification, all other tests in this document do not employ or require blank-and-burst signaling as part of a test vector set.

4.15 False Alarms

The test vocoder shall be robust to false alarms. False alarms can occur in two places, the TTY encoder and the TTY decoder. The TTY encoder may falsely label speech or tones as TTY and send TTY_SILENCE messages or TTY characters to the decoder. Likewise, the TTY decoder may misinterpret non-TTY packets as TTY information and either mute its output or replace its output with regenerated tones. For the purposes of this document, a TTY false alarm is defined as an action by the TTY algorithm, either the TTY encoder or the TTY decoder, that causes muting or tones being mistakenly regenerated at the decoder's output.

In order to make the reference vocoder robust to false alarms, more stringent rules are applied to the first character of a call. The rules are relaxed for all subsequent characters within the same call. As a result of the more stringent rules, there may be a character error at the beginning of the call as a result. The test vocoder may also exhibit similar performance.

4.16 Slew

Certain TTY devices exhibit a decaying DC bias, or slew, when generating a character after a period of silence. The test vocoder shall meet or exceed the performance requirements for detection and regeneration in the presence of slew.

4.17 Impaired Channels

The test vocoder shall not degrade its TTY performance when operating in simulated impaired channel environments with frame error rates from 0% to 6%.

4.18 Worst Case Deviations

The test vocoder shall not degrade its TTY performance when operating in a combination of extreme deviations for bit duration, stop bit length, frequency, slew, and phase changes and regardless of the typing mode, preamble, or mark hold tone.

5 TEST TOOLS

5.1 ttygen

The tool ttygen generates random characters according to [5]. The parameters for generating the Baudot tones can be varied from the command line so that the range of parameters specified in this document can be tested. Calling ttygen without parameters prints its help message. The parameters are specified as follows:

```
Usage: ttygen [options]
-a <TTY audio output file>
-b <Baudot code output file>
-c <random character output file>
-l <level in dBm,-45 - -5, (default = -10.0 dBm)>
-s <stopbits,1.0, 1.5 or 2.0 (default = 2.0)>
-f <frequency tolerance as a %,-4 to +4>
(or "+-" for random within range)>
-t <timing tolerance in ms, -0.4 to +0.4>
(or "+-" for random within range)
-p <phase mode>
0: continuous (default),
1: zero phase,
2: random phase,
3: max. discontinuous>
-g <1 for random silent gaps between chars, default = 0 for none>
-h <mark hold duration (default = 300 ms)>
-N <number of characters to generate (default = 4164)>
-r <seed for random number generator>
-x <baud rate, 45 or 50>
```

5.2 ttyrcv

The tool ttyrcv receives a PCM stream, and detects TTY/TDD Baudot characters. The 5-bit decimal value corresponding to the detected character is printed to the output file, one character per line. When the <-r reference_file> option is used, the reference symbol file is compared to the test symbol file and a TCER is calculated.

```
Usage: ttyrcv [options]
-i infile : input PCM tone filename (required)
-o symbolfile   : output file with 5 bit decimal value of Baudot code
    (default=stdout)
-r ref_file  : reference symbol file.
```

6 TTY/TDD TEST PROCEDURE

Test vectors and tools are provided as part of this standard in order to conduct the minimum performance test. The following describes the test vectors and the conditions they test, as well as the procedure for processing the files.

There are 3 different types of tests:

TTY Performance Tests/Tandem Tests: These tests are designed to verify that the test vocoder can correctly detect and regenerate the Baudot characters under a variety of conditions. Input PCM test vectors are to be processed by the test vocoder, and its output is processed to verify the total character error rate.

Interoperability Tests: These tests are designed to verify interoperability between the test vocoder and the reference vocoder, for both a single coding and tandem coding.

False Alarm Tests: A test vector with non-TTY audio is provided to verify the test vocoder's robustness to false alarms. The test vector is to be processed by the test vocoder and its output is to be examined for muting and tones that are generated as a result of a false alarm.

The tests shall be conducted with the tools that are provided with this standard. Scripts using these tools are also provided in order to facilitate testing.

6.1 Description of Test Vectors

All of the file test vectors were processed on a PC and stored as 16-bit words in PC format (LSB, MSB).

6.1.1 File Naming Convention

The following file naming convention is used for the test vectors associated with this standard:

tvecs/*_ref: Test vectors from the corresponding reference vocoder, where *_ref is either qc13_ref, evrc_ref, smv_ref, evrcb_ref, evrcwb_ref, or evrcnw_ref.

***.pcm:** Input PCM file

***.enc:** Encoded packet file

***.dec:** Decoded speech files

***.xmt:** Baudot code reference "truth" file, i.e. file containing the "transmitted" characters, represented as 5-bit Baudot code.

***.rcv** Baudot code test file, i.e., file containing the 5-bit Baudot code as detected by the tool *ttyrcv.exe*.

ttyseqNN.*: Files designed to exercise the test vocoder's robustness to variations to the Baudot code's parameters.

tandemN.*: Files designed to verify the test vocoder's ability to tandem with another TTY capable vocoder.

fer.*: Files designed to verify the test vocoder's ability to correctly regenerate the Baudot code in the presence of frame errors. The file *tvecs/*_ref/fer.enc* contains TTY vocoder packets that were injected with frame errors. The reference Baudot code file is *tvecs/pcm/fer.xmt*.

ttyhandoff.*: Files designed to verify the test vocoder's ability to recover from a hard handoff in the middle of a TTY call and to exercise the test vocoder's ability to reject TTY information that is not preceded with the TTY_SILENCE message.

false.*: Files designed to verify the test vocoder's robustness to false alarms.

6.2 Baudot Parameter Variation Test

Below is a description of the test vectors designed to test for robustness to Baudot parameter variability. Each of the test vectors uses a different seed so that each file has a different character sequence of 200 random printable characters. The exception to this is ttyseq97.pcm and ttyseq98.pcm, which were recorded from commercially available TTY devices. Because of the shift character, the total number of characters in each file is different. The parameters are:

- **Phase:** The phase between bit boundaries is either continuous (C) or discontinuous (D).
- **Stop Bit Length**: The duration of the stop bit is tested at 1.0 bits and 2.0 bits.
- **Typing Mode:** The typing modes tested are streaming mode (S) and manual (M) mode.
- **Mark Hold Time: The** mark hold time, the amount of time the stop bit is extended between characters, is between zero and 300 ms.
- **Bit Length**: The duration of the characters' bit length is varied from its nominal value of 22 ms by –0.4 ms and +0.4 ms.
- **Input Level:** The input level of the Baudot tones is tested at –5 dBm and -45 dBm.
- **Frequency Deviation:** All of the test vectors have a randomly varying frequency deviation ±4.0% from nominal.

The files shall be processed as follows:

```
test_encoder.exe –i tvecs/pcm/ttyseqNN.pcm –o tvecs/test_vocoder/ttyseqNN.enc <test
    encoder specific options>

test_decoder.exe -i tvecs/test_vocoder/ttyseqNN.enc
    -o tvecs/test_vocoder/ttyseqNN.dec <test decoder specific options>

ttyrcv.exe –i tvecs/test_vocoder/ttyseqNN.dec –o tvecs/test_vocoder/ttyseqNN.rcv
    -r tvecs/pcm/ttyseqNN.xmt
```

1 **Table 6-1: Description of TTY PCM Test Vectors**

File	Baud Rate	Phase	Stop Bit Length (bits)	Typing Mode	Mark Hold (ms)	Bit Length Deviation (ms)	Input Level (dBm)
ttyseq01.pcm	45	C	1.0	S	0	-0.4	-45
ttyseq02.pcm	45	C	1.0	S	0	-0.4	-5
ttyseq03.pcm	45	C	1.0	S	0	+0.4	-45
ttyseq04.pcm	45	C	1.0	S	0	+0.4	-5
ttyseq05.pcm	45	C	1.0	M	0	-0.4	-45
ttyseq06.pcm	45	C	1.0	M	0	-0.4	-5
ttyseq07.pcm	45	C	1.0	M	0	+0.4	-45
ttyseq08.pcm	45	C	1.0	M	0	+0.4	-5
ttyseq09.pcm	45	C	1.0	M	300	-0.4	-45
ttyseq10.pcm	45	C	1.0	M	300	-0.4	-5
ttyseq11.pcm	45	C	1.0	M	300	+0.4	-45
ttyseq12.pcm	45	C	1.0	M	300	+0.4	-5
ttyseq13.pcm	45	C	2.0	S	0	-0.4	-45
ttyseq14.pcm	45	C	2.0	S	0	-0.4	-5
ttyseq15.pcm	45	C	2.0	S	0	+0.4	-45
ttyseq16.pcm	45	C	2.0	S	0	+0.4	-5
ttyseq17.pcm	45	C	2.0	M	0	-0.4	-45
ttyseq18.pcm	45	C	2.0	M	0	-0.4	-5
ttyseq19.pcm	45	C	2.0	M	0	+0.4	-45
ttyseq20.pcm	45	C	2.0	M	0	+0.4	-5
ttyseq21.pcm	45	C	2.0	M	300	-0.4	-45
ttyseq22.pcm	45	C	2.0	M	300	-0.4	-5
ttyseq23.pcm	45	C	2.0	M	300	+0.4	-45
ttyseq24.pcm	45	C	2.0	M	300	+0.4	-5
ttyseq25.pcm	45	D	1.0	S	0	-0.4	-45
ttyseq26.pcm	45	D	1.0	S	0	-0.4	-5
ttyseq27.pcm	45	D	1.0	S	0	+0.4	-45
ttyseq28.pcm	45	D	1.0	S	0	+0.4	-5
ttyseq29.pcm	45	D	1.0	M	0	-0.4	-45
ttyseq30.pcm	45	D	1.0	M	0	-0.4	-5
ttyseq31.pcm	45	D	1.0	M	0	+0.4	-45
ttyseq32.pcm	45	D	1.0	M	0	+0.4	-5

File	Baud Rate	Phase	Stop Bit Length (bits)	Typing Mode	Mark Hold (ms)	Bit Length Deviation (ms)	Input Level (dBm)
ttyseq33.pcm	45	D	1.0	M	300	-0.4	-45
ttyseq34.pcm	45	D	1.0	M	300	-0.4	-5
ttyseq35.pcm	45	D	1.0	M	300	+0.4	-45
ttyseq36.pcm	45	D	1.0	M	300	+0.4	-5
ttyseq37.pcm	45	D	2.0	S	0	-0.4	-45
ttyseq38.pcm	45	D	2.0	S	0	-0.4	-5
ttyseq39.pcm	45	D	2.0	S	0	+0.4	-45
ttyseq40.pcm	45	D	2.0	S	0	+0.4	-5
ttyseq41.pcm	45	D	2.0	M	0	-0.4	-45
ttyseq42.pcm	45	D	2.0	M	0	-0.4	-5
ttyseq43.pcm	45	D	2.0	M	0	+0.4	-45
ttyseq44.pcm	45	D	2.0	M	0	+0.4	-5
ttyseq45.pcm	45	D	2.0	M	300	-0.4	-45
ttyseq46.pcm	45	D	2.0	M	300	-0.4	-5
ttyseq47.pcm	45	D	2.0	M	300	+0.4	-45
ttyseq48.pcm	45	D	2.0	M	300	+0.4	-5
ttyseq49.pcm	50	C	1.0	S	0	-0.4	-45
ttyseq50.pcm	50	C	1.0	S	0	-0.4	-5
ttyseq51.pcm	50	C	1.0	S	0	+0.4	-45
ttyseq52.pcm	50	C	1.0	S	0	+0.4	-5
ttyseq53.pcm	50	C	1.0	M	0	-0.4	-45
ttyseq54.pcm	50	C	1.0	M	0	-0.4	-5
ttyseq55.pcm	50	C	1.0	M	0	+0.4	-45
ttyseq56.pcm	50	C	1.0	M	0	+0.4	-5
ttyseq57.pcm	50	C	1.0	M	300	-0.4	-45
ttyseq58.pcm	50	C	1.0	M	300	-0.4	-5
ttyseq59.pcm	50	C	1.0	M	300	+0.4	-45
ttyseq60.pcm	50	C	1.0	M	300	+0.4	-5
ttyseq61.pcm	50	C	2.0	S	0	-0.4	-45
ttyseq62.pcm	50	C	2.0	S	0	-0.4	-5
ttyseq63.pcm	50	C	2.0	S	0	+0.4	-45
ttyseq64.pcm	50	C	2.0	S	0	+0.4	-5
ttyseq65.pcm	50	C	2.0	M	0	-0.4	-45

File	Baud Rate	Phase	Stop Bit Length (bits)	Typing Mode	Mark Hold (ms)	Bit Length Deviation (ms)	Input Level (dBm)
ttyseq66.pcm	50	C	2.0	M	0	-0.4	-5
ttyseq67.pcm	50	C	2.0	M	0	+0.4	-45
ttyseq68.pcm	50	C	2.0	M	0	+0.4	-5
ttyseq69.pcm	50	C	2.0	M	300	-0.4	-45
ttyseq70.pcm	50	C	2.0	M	300	-0.4	-5
ttyseq71.pcm	50	C	2.0	M	300	+0.4	-45
ttyseq72.pcm	50	C	2.0	M	300	+0.4	-5
ttyseq73.pcm	50	D	1.0	S	0	-0.4	-45
ttyseq74.pcm	50	D	1.0	S	0	-0.4	-5
ttyseq75.pcm	50	D	1.0	S	0	+0.4	-45
ttyseq76.pcm	50	D	1.0	S	0	+0.4	-5
ttyseq77.pcm	50	D	1.0	M	0	-0.4	-45
ttyseq78.pcm	50	D	1.0	M	0	-0.4	-5
ttyseq79.pcm	50	D	1.0	M	0	+0.4	-45
ttyseq80.pcm	50	D	1.0	M	0	+0.4	-5
ttyseq81.pcm	50	D	1.0	M	300	-0.4	-45
ttyseq82.pcm	50	D	1.0	M	300	-0.4	-5
ttyseq83.pcm	50	D	1.0	M	300	+0.4	-45
ttyseq84.pcm	50	D	1.0	M	300	+0.4	-5
ttyseq85.pcm	50	D	2.0	S	0	-0.4	-45
ttyseq86.pcm	50	D	2.0	S	0	-0.4	-5
ttyseq87.pcm	50	D	2.0	S	0	+0.4	-45
ttyseq88.pcm	50	D	2.0	S	0	+0.4	-5
ttyseq89.pcm	50	D	2.0	M	0	-0.4	-45
ttyseq90.pcm	50	D	2.0	M	0	-0.4	-5
ttyseq91.pcm	50	D	2.0	M	0	+0.4	-45
ttyseq92.pcm	50	D	2.0	M	0	+0.4	-5
ttyseq93.pcm	50	D	2.0	M	300	-0.4	-45
ttyseq94.pcm	50	D	2.0	M	300	-0.4	-5
ttyseq95.pcm	50	D	2.0	M	300	+0.4	-45
ttyseq96.pcm	50	D	2.0	M	300	+0.4	-5
ttyseq97.pcm	Real TTY device mixed with speech to test VCO/HCO.						
ttyseq98.pcm	Real TTY device in streaming mode.						

6.3 Interoperability Test

The files *tvecs/*_ref/ttyseqNN.pkt* are packet files that were processed by each of the reference vocoders with TTY turned ON. The test decoder shall process the appropriate reference packet files. In addition, the test encoder shall generate its own series of packet files, which shall be processed by the appropriate reference decoder.

The reference packet files shall be processed as follows:

```
test_decoder.exe –i tvecs/*_ref/ttyseqNN.enc –o tvecs/test_vocoder/iop_rtNN.dec
    <test decoder specific options>
```

```
ttyrcv.exe –i tvecs/test_vocoder/iop_rtNN.dec –o tvecs/test_vocoder/iop_rtNN.rcv
    -r tvecs/pcm/ttyseqNN.xmt
```

Likewise, interoperability between the test encoder and reference decoder shall be verified by processing the files as follows:

```
test_encoder.exe –i tvecs/*_ref/ttyseqNN.pcm –o tvecs/test_vocoder/iop_trNN.dec
    <test encoder specific options>
```

```
ref_decoder.exe –i tvecs/test_vocoder/ttyseqNN.enc –o
    tvecs/test_vocoder/iop_trNN.dec <test decoder specific options>
```

```
ttyrcv.exe –i tvecs/test_vocoder/iop_trNN.dec –o tvecs/test_vocoder/iop_trNN.rcv
    -r tvecs/pcm/ttyseqNN.xmt
```

6.4 Frame Error Handling Test

The files *tvecs/*_ref/fer.enc* are packet files with frame errors to test the decoder's TTY algorithm in a frame error environment. The model used was intended to simulate 6% FER, but the files have an actual FER of approximately 6.3%.

```
test_decoder.exe –i tvecs/*_ref/fer.enc –o tvecs/test_vocoder/fer.dec <test decoder
    specific options>
```

```
ttyrcv.exe –i tvecs/test_vocoder/fer.dec –o tvecs/test_vocoder/fer.rcv
    -r tvecs/pcm/fer.xmt
```

6.5 Hard Handoff Test

The files *ttyhandoff.** are designed to verify the test vocoder's ability to recover from a hard handoff in the middle of a TTY call. These files also exercise the test vocoder's ability to reject TTY information that is not preceded with the TTY_SILENCE message. The file *tvecs/*_ref/ttyhandoff.enc* contains TTY vocoder packets that encode two bursts of TTY characters. The first 20 packets of the file have been removed to remove the TTY_SILENCE messages that precede the first burst of characters. The test vocoder shall not regenerate characters from the first burst of characters but shall regenerate characters from the second burst. The files are processed as follows:

```
test_decoder.exe –i tvecs/*_ref/ttyhandoff.enc –o tvecs/test_vocoder/ttyhandoff.dec
    <test decoder specific options>
```

```
ttyrcv.exe –i tvecs/test_vocoder/ttyhandoff.dec –
    o tvecs/test_vocoder/ttyhandoff.rcv -r tvecs/pcm/ ttyhandoff.xmt
```

This test case does not apply to SMV, EVRC-B, EVRC-WB, and EVRC-NW.

6.6 False Alarm Test

The files *false.** are designed to verify the test vocoder's robustness to false alarms. The false alarm file is processed as follows:

```
test_encoder.exe –i tvecs/pcm/false.pcm –o tvecs/test_vocoder/false.enc <test
    encoder specific options>

test_decoder.exe -i tvecs/test_vocoder/false.enc -o tvecs/test_vocoder/false.dec
    <test decoder options>

ttyrcv.exe –i tvecs/test_vocoder/false.dec –o tvecs/test_vocoder/false.rcv
    -r tvecs/pcm/ttyseqNN.xmt
```

The decoded output file, false.dec, shall be inspected to verify that the output does not contain regenerated Baudot tones or muting caused by the TTY algorithm.

6.7 Tandem Test

Two files are designed to test TTY tandem calls, *tandem1.pcm* and *tandem2.pcm*. These files represent the first hop of a tandem. They have already been encoded and decoded by reference vocoders that contain the TTY solution. The procedure for processing them is the same as the procedure for processing the *ttyseqNN.pcm* files:

```
test_encoder.exe –i tvecs/pcm/tandemN.pcm –o tvecs/test_vocoder/tandemN.enc

test_decoder.exe -i tvecs/test_vocoder/tandemN.enc
    -o tvecs/test_vocoder/tandemN.dec

ttyrcv.exe –i tvecs/test_vocoder/tandemN.dec –o tvecs/test_vocoder/tandemN.rcv
    -r tvecs/pcm/tandemN.xmt
```

6.8 Half-Rate Max Test

In order to verify the test vocoder's ability to send TTY information in half rate packets, the test vocoder shall process files *tvecs/ttyseq48.pcm* (to test 45.45 baud) and *tvecs/ttyseq96.pcm* (to test 50 baud), saving the generated files in halfrate45.* and halfrate50.*, as follows:

```
test_encoder.exe–i tvecs/pcm/ttyseq48.pcm –o tvecs/test_vocoder/halfrate45.enc
    <test encoder half-rate max specific options>

test_decoder.exe -i tvecs/test_vocoder/halfrate45.enc –o
    tvecs/test_vocoder/halfrate45.dec <test decoder specific options>

ttyrcv.exe –i tvecs/test_vocoder/halfrate45.dec –o
    tvecs/test_vocoder/halfrate45.rcv -r tvecs/pcm/ttyseq48.xmt
```

Likewise, for 50 baud:

```
test_encoder.exe –i tvecs/pcm/ttyseq96.pcm –o tvecs/test_vocoder/halfrate50.enc
    <test encoder half-rate max specific options>

test_decoder.exe -i tvecs/test_vocoder/halfrate50.enc -o
    tvecs/test_vocoder/halfrate50.dec <test decoder specific options>

ttyrcv.exe –i tvecs/test_vocoder/halfrate50.dec –o
    tvecs/test_vocoder/halfrate50.rcv -r tvecs/pcm/ttyseq96.xmt
```

In order to verify that the test vocoder interoperates with the reference vocoder for half-rate max operation, the test vocoder shall process the half-rate max packet generated by the reference vocoder, and vice versa. The procedure is similar to the procedure described in Section 6.3, but for files halfrate45.enc and halfrate50.enc. Specifically, the reference packet files shall be processed as follows:

```
test_decoder.exe –i tvecs/*_ref/halfrate45.enc –o
    tvecs/test_vocoder/iop_rt_hr45.dec <test decoder specific options>

ttyrcv.exe –i tvecs/test_vocoder/iop_rt_hr45.dec –o
    tvecs/test_vocoder/iop_rt_hr45.rcv -r tvecs/pcm/ttyseq48.xmt
```

Likewise, interoperability between the test encoder and reference decoder shall be verified by processing the files as follows:

```
ref_decoder.exe –i tvecs/test_vocoder/halfrate45.enc –o
    tvecs/test_vocoder/iop_tr_hr45.dec <test decoder specific options>

ttyrcv.exe –i tvecs/test_vocoder/iop_tr_hr45.dec –o
    tvecs/test_vocoder/iop_tr_hr45.rcv -r tvecs/pcm/ttyseq96.xmt
```

The procedure for verifying the half-rate 50 baud file is similar, processing files halfrate50.enc and generating files iop_rt_hr50.dec, iop_rt_hr50.rcv by using the reference encoder and test decoder. Likewise, files iop_tr_hr50.dec, and iop_tr_hr50.rcv are generated using the test encoder and reference decoder.

7 REQUIREMENTS FOR MINIMUM PERFORMANCE COMPLIANCE

This section describes the requirements for complying with this specification for the TTY/TDD extension. The requirements of this specification are the same for vocoders whose speech path is bit-exact or non bit-exact. There is no separate bit-exact compliance option for TTY/TDD. The TTY minimum performance compliance test shall be performed with the tools included with this specification.

The following sections detail the minimum performance requirements for each of the test cases. The test cases and the requirements are the same, regardless of the vocoder being tested, unless stated otherwise.

Character errors refers to Total Character Errors, as described in Section 3, i.e., changes or missed Baudot codes, not Printable Character Errors.

7.1 Baudot Parameter Variation Test Requirements

The test vocoder shall process the files as described in Section 6.2 with no more than a total of 6 character errors for all of the files processed for this test case. Furthermore, there shall not be more than 2 character errors from any one file.

7.2 Interoperability Test Requirements

The test vocoder shall process the files with its respective reference as described in Section 6.3 with no more than a total of 6 character errors for all of the files processed for this test case. Furthermore, there shall not be more than 2 character errors from any one file.

7.3 Frame Error Handling Test Requirements

The test vocoder shall process the files as described in Section 6.4 with 0.0% TCER.

7.4 Hard Handoff Test Requirements

The test vocoder shall process the file as described in Section 6.5. The test vocoder shall not regenerate characters from the first burst of characters, passing the tones through the speech path instead. The test vocoder shall regenerate characters from the second burst, however, with 0.0% TCER. The number of character errors from the first burst of characters does not affect compliance with this test case; however, verifying that the first burst of characters is not regenerated is a requirement for compliance.

Since the test tools cannot distinguish characters from the two bursts, manual inspection of the decoded output is required for verifying compliance with this test case.

This test case does not apply to the SMV, EVRC-B, EVRC-WB, and EVRC-NW vocoders.

7.5 False Alarm Test Requirements

The test vocoder shall process the files as described in Section 6.6. The decoded output file shall be inspected to verify that the output does not contain regenerated Baudot tones or muting caused by the TTY algorithm.

7.6 Tandem Test Requirements

The test vocoder shall process the files as described in Section 6.7 with 0.0% TCER.

7.7 Half-Rate Max Test and Additional Requirements for EVRC-B, EVRC-WB, and EVRC-NW

The test vocoder shall process the files as described in Section 6.8 with 0.0% TCER.

Additionally, the following Capacity Operating Points shall be tested for the following vocoders:

EVRC-B: COPs 0 and 6

EVRC-WB: COPs 0 and 4

EVRC-NW: COPs 0 and 6