Rust 'serialport-rs' Transmit-Receive Timing and Functional Test Utility

Serial port basic test application 'receive_timing_info'

This document describes the **receive_timing_info** utility application, which is a timing characterization program for testing and supporting the serialport-rs crate. It builds with and uses the serialport-rs crate, and was created in response to the serialport-rs Github repo issue #106. While using this new utility I've identified a need for a functional patch for the serialport-rs crate's Windows platform set_timeout() method. I've now generated and tested this patch, again using this new utility to verify the new patch functions as planned. The patch eliminates an issue under Windows in which a zero (0) read timeout setting causes the serial port's read() method to block indefinitely (an infinite timeout) when all requested data isn't yet received. In comparison, the serialport-rs Linux read() operation in this same scenario returns immediately with whatever data is available on entry to read() – even if no data is available. Note that without this patch this particular read() scenario exhibits entirely different (let's call them opposite) behaviors on these two major rust run-time platforms – not a desirable situation.

I hope this patch, as detailed in the final section of this document, can be integrated into an upcoming serialport-rs crate release.

In my examination to date I've not found online serialport-rs crate documentation that describes the set_timeout()'s influence on the read() method's behavior with Windows and Linux, short of analyzing and determining the run-time semantics directly from each platform's internal source code. If any existing rust client code purposely utilizes this read() with 0 timeout scenario on Windows, they are relying on 'undocumented' behavior. If such client code is in-fact being used, and this code migrates to a newer serialport-rs with the herein suggested patch, then owners of such client code should revise their code to change from the 0 timeout setpoint to instead use a very large timeout setting. A large timeout setting will effectively replicate/emulate the current Windows read() behavior with the current timeout setting of 0. And this would certainly be the preferable way to accomplish such an indefinitely blocking read(). To be clear, note that set_timeout()'s duration parameter's maximum possible value will result in a read() blockage that last for 137+ years (not likely to ever be realized).

The *receive_timing_info* utility application source is packaged in a single file named receive_timing_info.rs, which can be built using cargo by copying it into the serialport-rs project's '*examples*' sub-folder. The application has been tested on both Windows 10 and Ubuntu 22.04 LTS (both x64 Intel hardware). In theory it should run on any platform supported by the standard rust compiler, as well as the **serialport-rs** crate itself. Currently this test application uses three utility crates beyond those that serialport-rs crate itself presently uses – which are identified in the following paragraph.

To build receive_timing_info, follow these steps:

a. Using 'git', clone the crate project source for serialport-rs from the Github repo at https://github.com/serialport/serialport-rs.
b. Copy the 'receive_timing_info.rs' source file into the cloned crate project's '*examples*' sub-folder.

c. This new test application currently requires a few additional dependencies beyond those which serialport-rs crate presently requires. The additional dependency crates are '**log**', '**fast-log**', and '**spin-sleep**'. One may add these additional dependencies to serialport-rs crates's 'Cargo.toml' file by submitting the following three commands from the serialport-rs crate project's top-most folder while in a terminal command window =>

   i. '**cargo add log**'
   ii. '**cargo add fast_log**'
   iii. '**cargo add spin_sleep**'

d. Then build the receive_timing_info test application by entering the following commands, also while in a terminal command window with your working directory set to the top-most folder of the serialport-rs project.

   '**cargo clean**'
   '**cargo build –example receive_timing_info**' or '**cargo build –release –example receive_timing_info**'

e. The resulting executable file will be located in the project's '**target\debug\examples\**' or '**target\release\examples\**' sub-folder respectively.

Next is an example MS-Windows terminal window command line that launches the new test application =>

```
>receive_timing_info.exe --txport=COM5 --rxport=COM6 --baud=115200 --log=D:\filename.log --rxtmo=20 --posttxdelayms=0 –xfrstalledtmo=12000
                              --txlen=10 --rxlen=20 --repeat=10 --fulldbg=Y
```

The following table shows the available command line arguments that the receive_timing_info utility supports as command line arguments at launch. Note that the test currently defaults to using the crate's '**None**' flow control setting and therefore doesn't test any platform or hardware specific serial port's flow control functionality. If desired this could be added.

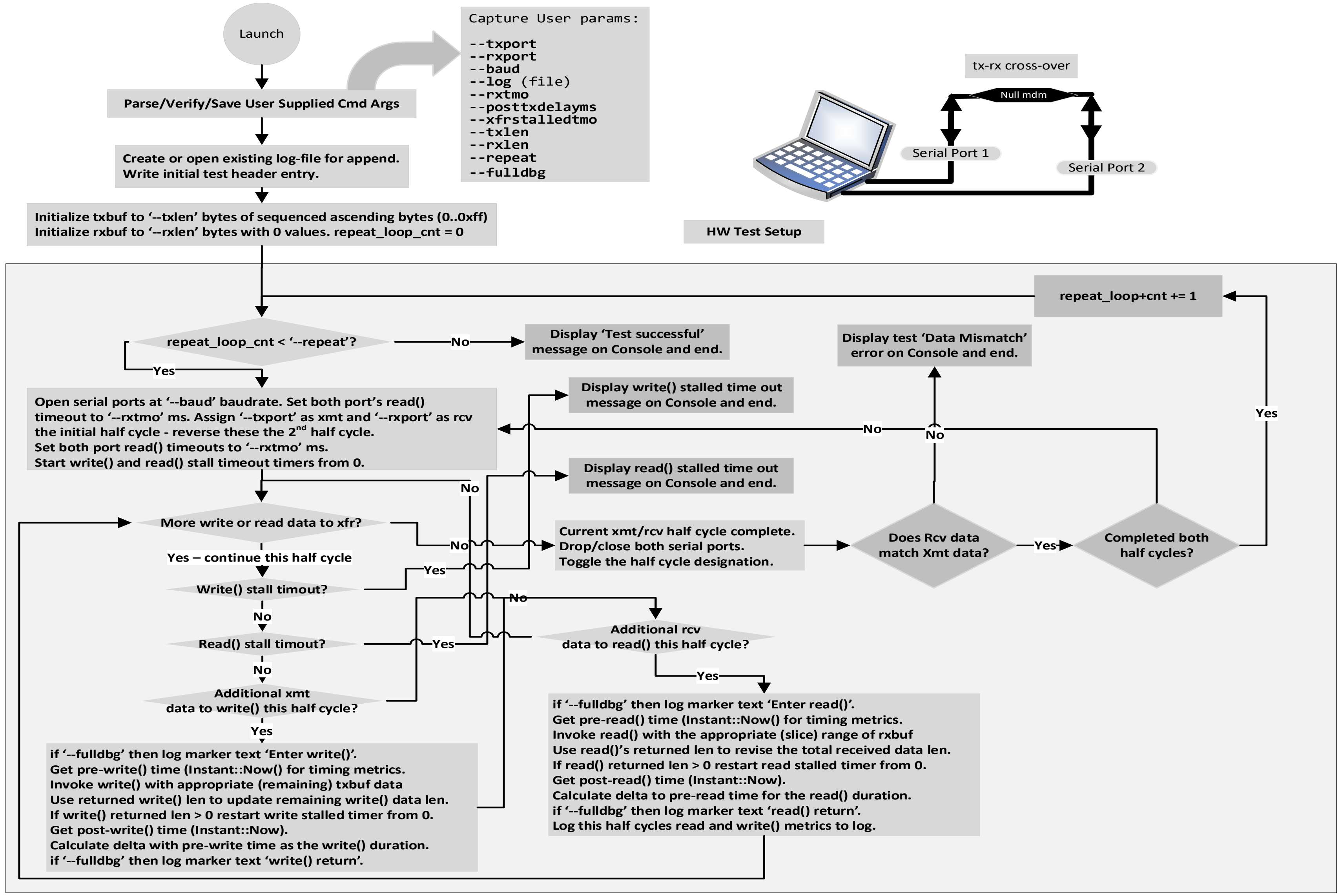
Syntax	Switch Purpose	Required or Optional
`--txport=`*`port-name`*	Platform specific port name (i.e. *COM4*)	Required
`--rxport=`*`port-name`*	Platform specific port name (i.e. *COM5*)	Required
`--log=`*`file-path`*	Platform specific log-file path	Required
`--baud=`*`bbbb`*	Baud rate (integer) – typical values are *9600*, *115200*, *200000* etc…	Required
`--rxtmo=`*`tttt`*	**read()** timeout in **ms** (integer)	Required
`--posttxdelayms=`*`tttt`*	The delay in **ms** to wait after **write()** before invoking the corresponding **read()**	Optional (default=0)
`--xfrstalledtmo=tttt`	The time-period before aborting when an in-progress test transfer sequence of consecutive **write()** or **read()** invocations time-out repeatedly, when additional transfer data is expected. Separate timers are internally maintained for **write()** and **read()**. The respective	

	transfer stalled timer is reset when any **write()** or **read()** in the sequence returns success and a positive 'len' value. This parameter's used value will automatically be set to the greater of 10000 (the default when not specified), or '**rxtmo** x **4**' when '**rxtmo** x **4**' is greater than both the default (10000) and the specified value. Note that the value actually used is indicated in the launch-time initial displayed informational text line.	
--txlen=*nnnn*	Specifies the total number of bytes to **write()** each half cycle. This byte sequence is an auto-generated [u8] array buffer with an ascending order byte pattern sequencing from **0** .. **0xff**, repeating until exactly '**txlen**' bytes are generated. This buffer is then supplied to the **write()** invocation.	Required
--rxlen=*nnnn*	Specifies the number of bytes requested when initially invoking **read()** following its associated **write()** operation in a given half cycle. While outstanding transmitted data has not yet been received, **read()** will be invoked repeated until the lesser of '**rxlen**' and '**txlen**' total bytes of accumulated data have been read, or until an '--**xfrstalledtmo**' timeout occurs. Subsequent **read()** invocations (following the initial **read**) in the same half-cycle use a request length that is reduced in magnitude downward from '**rxlen**' by the count of bytes already received. '**txlen**' and '**rxlen**' are commonly specified with the same positive value when primarily testing for transfer throughput, data transfer integrity and timing metrics. Note that these two parameters may be specified with different values, which has benefits for certain testing purposes. To test the **read()** timeout behavior and its platform specific timing characteristics, specifying values for which '**rxlen**' > '**txlen**' will (or should!) result in **read()** timeout errors, before an eventual '**xfrstalledtmo**' timeout eventually halts the test. One caveat for the current Window's crate is that this test scenario blocks the test application indefinitely when the read timeout parameter is set to 0 ('**--rxtmo=0**'. Specifying with '**rxlen**' < '**txlen**' parameter values should return '**rxlen**' bytes successfully, while leaving '**txlen**' – '**rxlen**' bytes unread in the read serial port's input buffer (which is not a problem). Finally note that specifying a '**txlen**' value larger than ~7500 on Windows will likely result in lost or corrupted transfer data. This is due to data buffer overruns in the platform OS layer, and is a by-product of the current test application's simple implementation/design. Similar data overruns may also happen on Linux, but at a larger '**txlen**' value: I haven't yet characterized this under Linux. To elaborate, currently the test's blocking **write()** invocation executes prior to the initial invocation of the blocking **read(),** with both occurring inline in the test application's main() thread**.** This current simple design does not support separately	Required

	threaded and overlapped **write()** and **read()** invocations. This can be added in a subsequent version, or possibly in a separate new test program.	
`--repeat=`*`nnnn`*	Total number of read/write full cycles. Note that each full cycle consists of two (2) half-cycles, where each half-cycle writes '**txlen**' bytes to one of the two serial ports and (attempts to) read '**rxlen**' bytes from the other, while the second half-cycle reverses the direction of transmission with the two serial ports. Note that both ports are closed and reopened at each half-cycle transition. The closing and reopening of the ports at each cycle (or half cycle) may be eliminated or altered in a future release, or in a new test app.	Optional (default=1)
`--fulldbg=Y (or N)`	'Y' enables additional debug log messages to be generated during the test. This flag should normally be 'N' (disabled), since enabling it results in undesirably lengthening certain timing metrics. 'N' is its default setting. Enabling the flag is useful when a specific test execution hangs indefinitely (which is abnormal), and with it enabled the run-time test logic inserts additional log-file 'entering' and 'returning' log marker text lines before and after each run-time invocation of '**write()**' and **'read()'**. Then one can examine the resulting log-file after the hang up occurs, and its clear which specific invocation is the cause of the hang-up. This is mainly useful in theorizing whether the hangup is due to faulty hardware, a bug in target specific crate code, electrical or cabling issues, or a previously undiscovered bug in the test software. ***Note that 'read()' or 'write()' blocking indefinitely could be detected with a separate monitoring thread, and an error message displayed for the test operator at execution time. This requires re-implementing the test with a multithreaded design (or in a separate new test application).***	Optional (default=N)

<This space left intentionally blank>

'receive_timing_info' Test Logic Flow-Chart



Launch

Parse/Verify/Save User Supplied Cmd Args

```
Capture User params:

--txport
--rxport
--baud
--log (file)
--rxtmo
--posttxdelayms
--xfrstalledtmo
--txlen
--rxlen
--repeat
--fulldbg
```

Create or open existing log-file for append.
Write initial test header entry.

Initialize txbuf to '--txlen' bytes of sequenced ascending bytes (0..0xff)
Initialize rxbuf to '--rxlen' bytes with 0 values. repeat_loop_cnt = 0

HW Test Setup

repeat_loop+cnt += 1

repeat_loop_cnt < '--repeat'?

No → Display 'Test successful' message on Console and end.

Yes

Open serial ports at '--baud' baudrate. Set both port's read() timeout to '--rxtmo' ms. Assign '--txport' as xmt and '--rxport' as rcv the initial half cycle - reverse these the 2nd half cycle.
Set both port read() timeouts to '--rxtmo' ms.
Start write() and read() stall timeout timers from 0.

Display write() stalled time out message on Console and end.

Display read() stalled time out message on Console and end.

Display test 'Data Mismatch' error on Console and end.

More write or read data to xfr?

No → Current xmt/rcv half cycle complete. Drop/close both serial ports. Toggle the half cycle designation.

Yes – continue this half cycle

Write() stall timout?

Yes

No

Read() stall timout?

Yes

No

Additional xmt data to write() this half cycle?

No

Yes

if '--fulldbg' then log marker text 'Enter write()'.
Get pre-write() time (Instant::Now() for timing metrics.
Invoke write() with appropriate (remaining) txbuf data
Use returned write() len to update remaining write() data len.
If write() returned len > 0 restart write stalled timer from 0.
Get post-write() time (Instant::Now).
Calculate delta with pre-write time as the write() duration.
if '--fulldbg' then log marker text 'write() return'.

Additional rcv data to read() this half cycle?

No

Yes

if '--fulldbg' then log marker text 'Enter read()'.
Get pre-read() time (Instant::Now() for timing metrics.
Invoke read() with the appropriate (slice) range of rxbuf
Use read()'s returned len to revise the total received data len.
If read() returned len > 0 restart read stalled timer from 0.
Get post-read() time (Instant::Now).
Calculate delta to pre-read time for the read() duration.
if '--fulldbg' then log marker text 'read() return'.
Log this half cycles read and write() metrics to log.

Does Rcv data match Xmt data?

No

Yes

Completed both half cycles?

No

Yes

Description of v4.2.1 serialport-rs behavior for 'read()' and 'set_timeout()' methods with Linux and MS-Windows platforms

As previously mentioned I couldn't find online documentation which details the serialport-rs crate's read() and write() methods expected behavior for various calling scenarios of baud rate settings, the actual time of arrival of arriving serial line data in relationship to the point in time at which the read() method is invoked, and the active setting of the read() timeout value for the port. Therefore, I ran experiments after examining the crate's (platform specific) source code from the v4.2.1 serialport-rs Github repo master branch at https://github.com/serialport/serialport-rs. Next I describe the behaviors I identified for both Windows 10 and (Ubuntu) Linux 22.04 LTS. My assumption at this time, although I haven't confirmed it, is that Windows 11 should produce the same behavior (with possibly slightly different timing metrics) as Windows 10 – which I used during my testing.

Linux specific read() behavior =>

For *Linux*, a positive (> 0) read timeout value is the period of time that read() blocks when no incoming data has arrived. If at least one (1) byte of data is received and available in the read's internal buffer, whether before initial entry to read() or any time thereafter prior to the timeout period expiring, read() returns immediately (at that instance) with the available data. If no received data is available at the conclusion of the timeout period, then read() returns at that time with a timeout error (and no data). The Linux read() returns the available data as soon as its available (and at initial entry when received data's already available), and its returned data len is only limited in size by the read() invocation's input buffer parameter length. With a read timeout setting of 0, the Linux read() returns with whatever data (if any) is available on initial entry, or immediately with no data, but always return immediately with or without data. When read() returns with no data (because none is available), it returns a timeout error indication. Please note also that the Linux read() non-zero timeout setting does not result in a read() internally extending its blocked time period waiting for *additional* data to arrive in order to more fully satisfy the requested data length, but rather only extends its wait when no data at all has been received.

Windows specific read() behavior =>

The Window's read() for v4.2.1 behaves differently in various respects to the Linux implementation.

On Windows when the timeout value is non-zero (> 0), then the read() behaves only roughly similar to Linux, and varies in a couple of measurable aspects.

a. The Windows read() waits for the entire timeout period while the entire requested read byte count (buffer size) is not yet available, and returns only at the conclusion of the read() timeout period with the lesser data amount that is available at conclusion of the timeout period - but no timeout occurs if at least one byte of data is returned. The Window's read() returns sooner than the timeout period *only if* the read's entire requested data count (the supplied buffer's size) is received prior to the conclusion of the read timeout period.
b. The second difference from the Linux read() is more consequential to applications. When the Window's read timeout set-point value is 0, the Windows read() blocks until ALL requested bytes (per the supplied buffer's length) become available. Essentially a 0 timeout set-point in

Windows operates as an infinite timeout period in order to satisfy the read()'s full requested length. To be clear, in this scenario a 0 timeout setpoint and having less than the read buffer's length of data arriving, the read() invocation blocks (forever) and never times-out.

As a practitioner's note regarding the Windows read() operation, with smaller read timeout settings in the range from 1 and 15 ms, a read() invocation can be randomly extended in time by up to ~15 to 20 ms prior to returning with already received data, or a timeout indication if no data has arrived within the timeout period. While this is certainly not ideal, by all accounts is something Microsoft is aware of and offers no solution to. Their documentation indicates this relates to the Windows OS API specification that Windows sleep timers have a published resolution of no less than ~16 milliseconds. This random delay clearly occurs with its native API Readfile() method, which is internally used by the crate's read() trait method, but is only a significant timing hindrance if one is attempting to write applications which must deterministically (hard real-time) respond to incoming serial data with repeatable timing accuracies in the sub (<) 100 ms range. These types of application timing requirements are generally not suitable for Windows as a target computer, for this and several other reasons. It is well known that Windows is *not* a real-time operating system!

Nevertheless, there are (many) successful and reliable industrial control applications which are quite effective using RS-232 serial communication connections while running on modern Windows high-performance multi-core CPU based computer system. They are effective and demonstrate high percentage repeatability with timer accuracies in the ~100 ms (and above) time region. The Linux OS can of course offer performance which is somewhat better than Windows, using its **-rt** (kernel option) and its **pthread** real-time priority threads - but I'm unaware if Rust threads on Linux can 'out of the box' utilize this Linux platform specific threading feature.

'receive_timing_info' test example data log snippets for three (3) command argument scenarios with read timeouts.

Next are three (3) generated Windows test logs with parameters which intentionally induce read timeouts. Note that each log file's second text line entry indicates the run-time supplied command line argument values which were provided.

Note that only the initial (first) half cycle of each of these test runs was undertaken, since the induced read timeout (induced by specifying an –rxlen and –txlen parameter pair where rxlen > txlen) resulted in a 'transfer stalled' timeout error during its first half cycle.

For this first Windows test run, its first half-cycle's write() method invocation transmits a total of one (1) byte, per the –txlen parameter, but the corresponding initial read() requests two (2) bytes, per the –rxlen parameter. Note that the read timeout is set to 50 ms, per the –rxtmo value. Also the initial read() is invoked precisely 100 ms after its matching half-cycle write() completes, per the –posttxdelayms parameter. This sets up the situation where the initial read() should find already received data available upon entry – precisely one byte. Since the –rxtmo value wasn't 0, this run doesn't result in the problematic indefinite read() blocking scenario. Note the port baud rates are set to 200,000 baud. But this initial half-cycle will not successfully fulfill its expected total read count of two(2), and therefore will eventually trigger a transfer stalled timeout after one second, per its –xfrstalledtmo parameter of 1000 ms. Please note that each subsequent read() invocation while the half-cycle is active will reduce its read() buffer request size by the accumulatively received data (to this point) in the active half-cycle.

Note that in all the logs shown herein which use the existing crate version, the initial half-cycles will all error or block indefinitely, but this was arranged intentionally (via the supplied execution parameters) for the purposes of this explanatory document.

```
2023-07-16 19:11:59.2347468 INFO receive_timing_info - 'receive_timing_info' cross platform dual RS-232 port null modem cable connected rcv+xmt+timeout test and characterization tool: v1.0
2023-07-16 19:11:59.2351164 INFO receive_timing_info - Test setup: Platform='windows', Baud=200000, rxtmo=50 ms, posttxdelayms=100 ms, xfrstalledtmo=1000 ms, txlen=1, rxlen=2, repeat=1, fulldbg=false
2023-07-16 19:11:59.2378638 INFO receive_timing_info - Test Logfile Name: 'D:\Users\ricej\windows_receive_timing_info.txt'
2023-07-16 19:11:59.29698   INFO receive_timing_info -
2023-07-16 19:11:59.2969872 INFO receive_timing_info - ** Start of cycle 1. **
2023-07-16 19:11:59.2969881 INFO receive_timing_info - Cycle 1 first phase -> Rx port = 'COM7', Tx port = 'COM6' .
2023-07-16 19:11:59.65374   INFO receive_timing_info - txport.write() sent 1 bytes while blocked for 548 us. Read() invoked 100412 us after write(), rxport.read(2) returned 1 bytes while blocked for 50255 us.
2023-07-16 19:11:59.7129727 INFO receive_timing_info -                          Read() invoked 150719 us after write(), rxport.read(1) returned 0 bytes while blocked for 59176 us. Rcv timeout.
2023-07-16 19:11:59.7629789 INFO receive_timing_info -                          Read() invoked 209949 us after write(), rxport.read(1) returned 0 bytes while blocked for 49959 us. Rcv timeout.
2023-07-16 19:11:59.8131091 INFO receive_timing_info -                          Read() invoked 259953 us after write(), rxport.read(1) returned 0 bytes while blocked for 50050 us. Rcv timeout.
2023-07-16 19:11:59.8729912 INFO receive_timing_info -                          Read() invoked 310088 us after write(), rxport.read(1) returned 0 bytes while blocked for 59832 us. Rcv timeout.
2023-07-16 19:11:59.923074  INFO receive_timing_info -                          Read() invoked 369984 us after write(), rxport.read(1) returned 0 bytes while blocked for 50024 us. Rcv timeout.
2023-07-16 19:11:59.9730056 INFO receive_timing_info -                          Read() invoked 420051 us after write(), rxport.read(1) returned 0 bytes while blocked for 49882 us. Rcv timeout.
2023-07-16 19:12:00.0301661 INFO receive_timing_info -                          Read() invoked 469980 us after write(), rxport.read(1) returned 0 bytes while blocked for 57127 us. Rcv timeout.
2023-07-16 19:12:00.0802684 INFO receive_timing_info -                          Read() invoked 527131 us after write(), rxport.read(1) returned 0 bytes while blocked for 50065 us. Rcv timeout.
2023-07-16 19:12:00.1401687 INFO receive_timing_info -                          Read() invoked 577248 us after write(), rxport.read(1) returned 0 bytes while blocked for 59835 us. Rcv timeout.
2023-07-16 19:12:00.190347  INFO receive_timing_info -                          Read() invoked 637145 us after write(), rxport.read(1) returned 0 bytes while blocked for 50135 us. Rcv timeout.
2023-07-16 19:12:00.2403654 INFO receive_timing_info -                          Read() invoked 687319 us after write(), rxport.read(1) returned 0 bytes while blocked for 49979 us. Rcv timeout.
2023-07-16 19:12:00.2992358 INFO receive_timing_info -                          Read() invoked 737337 us after write(), rxport.read(1) returned 0 bytes while blocked for 58831 us. Rcv timeout.
2023-07-16 19:12:00.3492649 INFO receive_timing_info -                          Read() invoked 796206 us after write(), rxport.read(1) returned 0 bytes while blocked for 49992 us. Rcv timeout.
2023-07-16 19:12:00.3992507 INFO receive_timing_info -                          Read() invoked 846241 us after write(), rxport.read(1) returned 0 bytes while blocked for 49941 us. Rcv timeout.
2023-07-16 19:12:00.458996  INFO receive_timing_info -                          Read() invoked 896227 us after write(), rxport.read(1) returned 0 bytes while blocked for 59714 us. Rcv timeout.
2023-07-16 19:12:00.5091288 INFO receive_timing_info -                          Read() invoked 955960 us after write(), rxport.read(1) returned 0 bytes while blocked for 50082 us. Rcv timeout.
2023-07-16 19:12:00.5690609 INFO receive_timing_info -                          Read() invoked 1006104 us after write(), rxport.read(1) returned 0 bytes while blocked for 59874 us. Rcv timeout.
```

```
2023-07-16 19:12:00.6191622 INFO receive_timing_info -                    Read() invoked 1066021 us after write(), rxport.read(1) returned 0 bytes while blocked for 50077 us. Rcv timeout.
2023-07-16 19:12:00.6696142 INFO receive_timing_info -                    Read() invoked 1116135 us after write(), rxport.read(1) returned 0 bytes while blocked for 50413 us. Rcv timeout.
2023-07-16 19:12:00.6696732 INFO receive_timing_info -
TRANSFER STALLED TIMEOUT ERROR: 'rxport::read()' repeatedly timed-out without receiving its requested incoming data. If not induced, inspect+verify the serial connections. Aborting.
```

Next is the second Windows log example.

For this (second) Windows test run, its first half-cycle's write() invocation again transmits a total of one (1) byte, per the –txlen parameter, and the corresponding read() again requests two (2) bytes, per its –rxlen parameter. Note that the read timeout is set to 1 ms, per its –rxtmo value. Also the initial read() is invoked immediately following its matching write() completes, per the –posttxdelayms parameter of 0. This sets up the situation where our initial read() will likely not find already received data available upon entry. Since the –rxtmo value wasn't 0 for this run, it doesn't result in the problematic indefinite read() blocking scenario. Rather, since the total accumulated read will not never fulfill the total requested read() count of 2 (per –rxlen), the half cycle will eventually error with a transfer stalled timeout after 1000 ms (1 second). Finally, note the port baud rates are set to 200,000 baud.

```
2023-07-16 19:13:30.8416232 INFO receive_timing_info - 'receive_timing_info' cross platform dual RS-232 port null modem cable connected rcv+xmt+timeout test and characterization tool: v1.0
2023-07-16 19:13:30.84176   INFO receive_timing_info - Test setup: Platform='windows', Baud=200000, rxtmo=1 ms, posttxdelayms=0 ms, xfrstalledtmo=1000 ms, txlen=1, rxlen=2, repeat=1, fulldbg=false
2023-07-16 19:13:30.8458403 INFO receive_timing_info - Test Logfile Name: 'D:\Users\ricej\windows_receive_timing_info.txt'
2023-07-16 19:13:30.8690723 INFO receive_timing_info -
2023-07-16 19:13:30.8690797 INFO receive_timing_info - ** Start of cycle 1. **
2023-07-16 19:13:30.8690811 INFO receive_timing_info - Cycle 1 first phase -> Rx port = 'COM7', Tx port = 'COM6' .
2023-07-16 19:13:31.0837107 INFO receive_timing_info - txport.write() sent 1 bytes while blocked for 427 us. Read() invoked 25 us after write(), rxport.read(2) returned 0 bytes while blocked for 9449 us. Rcv timeout.
2023-07-16 19:13:31.0935963 INFO receive_timing_info -                    Read() invoked 9520 us after write(), rxport.read(2) returned 1 bytes while blocked for 9827 us.
2023-07-16 19:13:31.1037244 INFO receive_timing_info -                    Read() invoked 19400 us after write(), rxport.read(1) returned 0 bytes while blocked for 10087 us. Rcv timeout.
2023-07-16 19:13:31.1137371 INFO receive_timing_info -                    Read() invoked 29555 us after write(), rxport.read(1) returned 0 bytes while blocked for 9945 us. Rcv timeout.
2023-07-16 19:13:31.1237386 INFO receive_timing_info -                    Read() invoked 39547 us after write(), rxport.read(1) returned 0 bytes while blocked for 9953 us. Rcv timeout.
2023-07-16 19:13:31.1337384 INFO receive_timing_info -                    Read() invoked 49549 us after write(), rxport.read(1) returned 0 bytes while blocked for 9951 us. Rcv timeout.
2023-07-16 19:13:31.1435955 INFO receive_timing_info -                    Read() invoked 59552 us after write(), rxport.read(1) returned 0 bytes while blocked for 9806 us. Rcv timeout.
2023-07-16 19:13:31.1537175 INFO receive_timing_info -                    Read() invoked 69407 us after write(), rxport.read(1) returned 0 bytes while blocked for 10074 us. Rcv timeout.
2023-07-16 19:13:31.1637297 INFO receive_timing_info -                    Read() invoked 79529 us after write(), rxport.read(1) returned 0 bytes while blocked for 9966 us. Rcv timeout.
2023-07-16 19:13:31.1737385 INFO receive_timing_info -                    Read() invoked 89542 us after write(), rxport.read(1) returned 0 bytes while blocked for 9963 us. Rcv timeout.
2023-07-16 19:13:31.1837062 INFO receive_timing_info -                    Read() invoked 99549 us after write(), rxport.read(1) returned 0 bytes while blocked for 9922 us. Rcv timeout.
2023-07-16 19:13:31.1936502 INFO receive_timing_info -                    Read() invoked 109518 us after write(), rxport.read(1) returned 0 bytes while blocked for 9898 us. Rcv timeout.
2023-07-16 19:13:31.2037372 INFO receive_timing_info -                    Read() invoked 119456 us after write(), rxport.read(1) returned 0 bytes while blocked for 10045 us. Rcv timeout.
2023-07-16 19:13:31.2137275 INFO receive_timing_info -                    Read() invoked 129550 us after write(), rxport.read(1) returned 0 bytes while blocked for 9944 us. Rcv timeout.
2023-07-16 19:13:31.223706  INFO receive_timing_info -                    Read() invoked 139539 us after write(), rxport.read(1) returned 0 bytes while blocked for 9932 us. Rcv timeout.
2023-07-16 19:13:31.2337312 INFO receive_timing_info -                    Read() invoked 149519 us after write(), rxport.read(1) returned 0 bytes while blocked for 9978 us. Rcv timeout.
2023-07-16 19:13:31.2436519 INFO receive_timing_info -                    Read() invoked 159544 us after write(), rxport.read(1) returned 0 bytes while blocked for 9870 us. Rcv timeout.
2023-07-16 19:13:31.2537485 INFO receive_timing_info -                    Read() invoked 169468 us after write(), rxport.read(1) returned 0 bytes while blocked for 10046 us. Rcv timeout.
2023-07-16 19:13:31.263729  INFO receive_timing_info -                    Read() invoked 179559 us after write(), rxport.read(1) returned 0 bytes while blocked for 9937 us. Rcv timeout.
2023-07-16 19:13:31.2760295 INFO receive_timing_info -                    Read() invoked 189541 us after write(), rxport.read(1) returned 0 bytes while blocked for 12259 us. Rcv timeout.
2023-07-16 19:13:31.2860841 INFO receive_timing_info -                    Read() invoked 201835 us after write(), rxport.read(1) returned 0 bytes while blocked for 10016 us. Rcv timeout.
2023-07-16 19:13:31.2961383 INFO receive_timing_info -                    Read() invoked 211896 us after write(), rxport.read(1) returned 0 bytes while blocked for 10006 us. Rcv timeout.
2023-07-16 19:13:31.3143309 INFO receive_timing_info -                    Read() invoked 221949 us after write(), rxport.read(1) returned 0 bytes while blocked for 18144 us. Rcv timeout.
2023-07-16 19:13:31.3243362 INFO receive_timing_info -                    Read() invoked 240141 us after write(), rxport.read(1) returned 0 bytes while blocked for 9960 us. Rcv timeout.
2023-07-16 19:13:31.3343353 INFO receive_timing_info -                    Read() invoked 250148 us after write(), rxport.read(1) returned 0 bytes while blocked for 9948 us. Rcv timeout.
2023-07-16 19:13:31.3442805 INFO receive_timing_info -                    Read() invoked 260143 us after write(), rxport.read(1) returned 0 bytes while blocked for 9903 us. Rcv timeout.
```

```
2023-07-16 19:13:31.3543142 INFO receive_timing_info -          Read() invoked 270112 us after write(), rxport.read(1) returned 0 bytes while blocked for 9969 us. Rcv timeout.
2023-07-16 19:13:31.3643089 INFO receive_timing_info -          Read() invoked 280127 us after write(), rxport.read(1) returned 0 bytes while blocked for 9949 us. Rcv timeout.
2023-07-16 19:13:31.3743395 INFO receive_timing_info -          Read() invoked 290121 us after write(), rxport.read(1) returned 0 bytes while blocked for 9984 us. Rcv timeout.
2023-07-16 19:13:31.3842602 INFO receive_timing_info -          Read() invoked 300151 us after write(), rxport.read(1) returned 0 bytes while blocked for 9871 us. Rcv timeout.
2023-07-16 19:13:31.3959133 INFO receive_timing_info -          Read() invoked 310072 us after write(), rxport.read(1) returned 0 bytes while blocked for 11609 us. Rcv timeout.
2023-07-16 19:13:31.4146801 INFO receive_timing_info -          Read() invoked 321722 us after write(), rxport.read(1) returned 0 bytes while blocked for 18720 us. Rcv timeout.
2023-07-16 19:13:31.4245156 INFO receive_timing_info -          Read() invoked 340484 us after write(), rxport.read(1) returned 0 bytes while blocked for 9796 us. Rcv timeout.
2023-07-16 19:13:31.4346106 INFO receive_timing_info -          Read() invoked 350328 us after write(), rxport.read(1) returned 0 bytes while blocked for 10048 us. Rcv timeout.
2023-07-16 19:13:31.4446121 INFO receive_timing_info -          Read() invoked 360415 us after write(), rxport.read(1) returned 0 bytes while blocked for 9962 us. Rcv timeout.
2023-07-16 19:13:31.4546299 INFO receive_timing_info -          Read() invoked 370427 us after write(), rxport.read(1) returned 0 bytes while blocked for 9969 us. Rcv timeout.
2023-07-16 19:13:31.464612  INFO receive_timing_info -          Read() invoked 380441 us after write(), rxport.read(1) returned 0 bytes while blocked for 9937 us. Rcv timeout.
2023-07-16 19:13:31.4746493 INFO receive_timing_info -          Read() invoked 390424 us after write(), rxport.read(1) returned 0 bytes while blocked for 9989 us. Rcv timeout.
2023-07-16 19:13:31.4846173 INFO receive_timing_info -          Read() invoked 400462 us after write(), rxport.read(1) returned 0 bytes while blocked for 9920 us. Rcv timeout.
2023-07-16 19:13:31.49463   INFO receive_timing_info -          Read() invoked 410434 us after write(), rxport.read(1) returned 0 bytes while blocked for 9962 us. Rcv timeout.
2023-07-16 19:13:31.5045199 INFO receive_timing_info -          Read() invoked 420442 us after write(), rxport.read(1) returned 0 bytes while blocked for 9842 us. Rcv timeout.
2023-07-16 19:13:31.5146423 INFO receive_timing_info -          Read() invoked 430334 us after write(), rxport.read(1) returned 0 bytes while blocked for 10075 us. Rcv timeout.
2023-07-16 19:13:31.524661  INFO receive_timing_info -          Read() invoked 440453 us after write(), rxport.read(1) returned 0 bytes while blocked for 9974 us. Rcv timeout.
2023-07-16 19:13:31.5346001 INFO receive_timing_info -          Read() invoked 450494 us after write(), rxport.read(1) returned 0 bytes while blocked for 9869 us. Rcv timeout.
2023-07-16 19:13:31.5445119 INFO receive_timing_info -          Read() invoked 460414 us after write(), rxport.read(1) returned 0 bytes while blocked for 9864 us. Rcv timeout.
2023-07-16 19:13:31.5546218 INFO receive_timing_info -          Read() invoked 470323 us after write(), rxport.read(1) returned 0 bytes while blocked for 10064 us. Rcv timeout.
2023-07-16 19:13:31.5646134 INFO receive_timing_info -          Read() invoked 480432 us after write(), rxport.read(1) returned 0 bytes while blocked for 9945 us. Rcv timeout.
2023-07-16 19:13:31.5746458 INFO receive_timing_info -          Read() invoked 490426 us after write(), rxport.read(1) returned 0 bytes while blocked for 9987 us. Rcv timeout.
2023-07-16 19:13:31.5845786 INFO receive_timing_info -          Read() invoked 500456 us after write(), rxport.read(1) returned 0 bytes while blocked for 9869 us. Rcv timeout.
2023-07-16 19:13:31.5946506 INFO receive_timing_info -          Read() invoked 510385 us after write(), rxport.read(1) returned 0 bytes while blocked for 10027 us. Rcv timeout.
2023-07-16 19:13:31.6046229 INFO receive_timing_info -          Read() invoked 520465 us after write(), rxport.read(1) returned 0 bytes while blocked for 9922 us. Rcv timeout.
2023-07-16 19:13:31.6146285 INFO receive_timing_info -          Read() invoked 530438 us after write(), rxport.read(1) returned 0 bytes while blocked for 9959 us. Rcv timeout.
2023-07-16 19:13:31.6246623 INFO receive_timing_info -          Read() invoked 540438 us after write(), rxport.read(1) returned 0 bytes while blocked for 9964 us. Rcv timeout.
2023-07-16 19:13:31.6345675 INFO receive_timing_info -          Read() invoked 550476 us after write(), rxport.read(1) returned 0 bytes while blocked for 9853 us. Rcv timeout.
2023-07-16 19:13:31.6446325 INFO receive_timing_info -          Read() invoked 560379 us after write(), rxport.read(1) returned 0 bytes while blocked for 10019 us. Rcv timeout.
2023-07-16 19:13:31.6546681 INFO receive_timing_info -          Read() invoked 570443 us after write(), rxport.read(1) returned 0 bytes while blocked for 9987 us. Rcv timeout.
2023-07-16 19:13:31.6645206 INFO receive_timing_info -          Read() invoked 580486 us after write(), rxport.read(1) returned 0 bytes while blocked for 9799 us. Rcv timeout.
2023-07-16 19:13:31.6746493 INFO receive_timing_info -          Read() invoked 590332 us after write(), rxport.read(1) returned 0 bytes while blocked for 10083 us. Rcv timeout.
2023-07-16 19:13:31.6845833 INFO receive_timing_info -          Read() invoked 600462 us after write(), rxport.read(1) returned 0 bytes while blocked for 9889 us. Rcv timeout.
2023-07-16 19:13:31.6946451 INFO receive_timing_info -          Read() invoked 610393 us after write(), rxport.read(1) returned 0 bytes while blocked for 10019 us. Rcv timeout.
2023-07-16 19:13:31.7044569 INFO receive_timing_info -          Read() invoked 620456 us after write(), rxport.read(1) returned 0 bytes while blocked for 9755 us. Rcv timeout.
2023-07-16 19:13:31.7146347 INFO receive_timing_info -          Read() invoked 630267 us after write(), rxport.read(1) returned 0 bytes while blocked for 10135 us. Rcv timeout.
2023-07-16 19:13:31.7246317 INFO receive_timing_info -          Read() invoked 640446 us after write(), rxport.read(1) returned 0 bytes while blocked for 9952 us. Rcv timeout.
2023-07-16 19:13:31.7347149 INFO receive_timing_info -          Read() invoked 650443 us after write(), rxport.read(1) returned 0 bytes while blocked for 10037 us. Rcv timeout.
2023-07-16 19:13:31.74461   INFO receive_timing_info -          Read() invoked 660524 us after write(), rxport.read(1) returned 0 bytes while blocked for 9847 us. Rcv timeout.
2023-07-16 19:13:31.7545724 INFO receive_timing_info -          Read() invoked 670422 us after write(), rxport.read(1) returned 0 bytes while blocked for 9911 us. Rcv timeout.
2023-07-16 19:13:31.7646463 INFO receive_timing_info -          Read() invoked 680383 us after write(), rxport.read(1) returned 0 bytes while blocked for 10028 us. Rcv timeout.
2023-07-16 19:13:31.7746377 INFO receive_timing_info -          Read() invoked 690457 us after write(), rxport.read(1) returned 0 bytes while blocked for 9948 us. Rcv timeout.
2023-07-16 19:13:31.7844717 INFO receive_timing_info -          Read() invoked 700448 us after write(), rxport.read(1) returned 0 bytes while blocked for 9784 us. Rcv timeout.
2023-07-16 19:13:31.7946455 INFO receive_timing_info -          Read() invoked 710282 us after write(), rxport.read(1) returned 0 bytes while blocked for 10129 us. Rcv timeout.
2023-07-16 19:13:31.8046542 INFO receive_timing_info -          Read() invoked 720458 us after write(), rxport.read(1) returned 0 bytes while blocked for 9957 us. Rcv timeout.
2023-07-16 19:13:31.8145803 INFO receive_timing_info -          Read() invoked 730471 us after write(), rxport.read(1) returned 0 bytes while blocked for 9870 us. Rcv timeout.
2023-07-16 19:13:31.8246227 INFO receive_timing_info -          Read() invoked 740391 us after write(), rxport.read(1) returned 0 bytes while blocked for 9998 us. Rcv timeout.
2023-07-16 19:13:31.8345487 INFO receive_timing_info -          Read() invoked 750434 us after write(), rxport.read(1) returned 0 bytes while blocked for 9876 us. Rcv timeout.
2023-07-16 19:13:31.8445683 INFO receive_timing_info -          Read() invoked 760363 us after write(), rxport.read(1) returned 0 bytes while blocked for 9988 us. Rcv timeout.
2023-07-16 19:13:31.8562955 INFO receive_timing_info -          Read() invoked 770367 us after write(), rxport.read(1) returned 0 bytes while blocked for 11700 us. Rcv timeout.
2023-07-16 19:13:31.8664263 INFO receive_timing_info -          Read() invoked 782097 us after write(), rxport.read(1) returned 0 bytes while blocked for 10095 us. Rcv timeout.
```

```
2023-07-16 19:13:31.8763461 INFO receive_timing_info -                         Read() invoked 792237 us after write(), rxport.read(1) returned 0 bytes while blocked for 9871 us. Rcv timeout.
2023-07-16 19:13:31.8864335 INFO receive_timing_info -                         Read() invoked 802153 us after write(), rxport.read(1) returned 0 bytes while blocked for 10048 us. Rcv timeout.
2023-07-16 19:13:31.8962505 INFO receive_timing_info -                         Read() invoked 812242 us after write(), rxport.read(1) returned 0 bytes while blocked for 9779 us. Rcv timeout.
2023-07-16 19:13:31.9150854 INFO receive_timing_info -                         Read() invoked 822056 us after write(), rxport.read(1) returned 0 bytes while blocked for 18794 us. Rcv timeout.
2023-07-16 19:13:31.9250768 INFO receive_timing_info -                         Read() invoked 840901 us after write(), rxport.read(1) returned 0 bytes while blocked for 9943 us. Rcv timeout.
2023-07-16 19:13:31.9349086 INFO receive_timing_info -                         Read() invoked 850888 us after write(), rxport.read(1) returned 0 bytes while blocked for 9783 us. Rcv timeout.
2023-07-16 19:13:31.9450669 INFO receive_timing_info -                         Read() invoked 860720 us after write(), rxport.read(1) returned 0 bytes while blocked for 10108 us. Rcv timeout.
2023-07-16 19:13:31.9550776 INFO receive_timing_info -                         Read() invoked 870873 us after write(), rxport.read(1) returned 0 bytes while blocked for 9967 us. Rcv timeout.
2023-07-16 19:13:31.9650426 INFO receive_timing_info -                         Read() invoked 880889 us after write(), rxport.read(1) returned 0 bytes while blocked for 9920 us. Rcv timeout.
2023-07-16 19:13:31.9750956 INFO receive_timing_info -                         Read() invoked 890853 us after write(), rxport.read(1) returned 0 bytes while blocked for 9978 us. Rcv timeout.
2023-07-16 19:13:31.9849724 INFO receive_timing_info -                         Read() invoked 900902 us after write(), rxport.read(1) returned 0 bytes while blocked for 9835 us. Rcv timeout.
2023-07-16 19:13:31.9949931 INFO receive_timing_info -                         Read() invoked 910784 us after write(), rxport.read(1) returned 0 bytes while blocked for 9976 us. Rcv timeout.
2023-07-16 19:13:32.0050653 INFO receive_timing_info -                         Read() invoked 920798 us after write(), rxport.read(1) returned 0 bytes while blocked for 10033 us. Rcv timeout.
2023-07-16 19:13:32.0149805 INFO receive_timing_info -                         Read() invoked 930876 us after write(), rxport.read(1) returned 0 bytes while blocked for 9870 us. Rcv timeout.
2023-07-16 19:13:32.0250305 INFO receive_timing_info -                         Read() invoked 940788 us after write(), rxport.read(1) returned 0 bytes while blocked for 10008 us. Rcv timeout.
2023-07-16 19:13:32.0350116 INFO receive_timing_info -                         Read() invoked 950839 us after write(), rxport.read(1) returned 0 bytes while blocked for 9940 us. Rcv timeout.
2023-07-16 19:13:32.0450704 INFO receive_timing_info -                         Read() invoked 960822 us after write(), rxport.read(1) returned 0 bytes while blocked for 9988 us. Rcv timeout.
2023-07-16 19:13:32.0549696 INFO receive_timing_info -                         Read() invoked 970876 us after write(), rxport.read(1) returned 0 bytes while blocked for 9862 us. Rcv timeout.
2023-07-16 19:13:32.0649347 INFO receive_timing_info -                         Read() invoked 980779 us after write(), rxport.read(1) returned 0 bytes while blocked for 9928 us. Rcv timeout.
2023-07-16 19:13:32.074916  INFO receive_timing_info -                        Read() invoked 990727 us after write(), rxport.read(1) returned 0 bytes while blocked for 9973 us. Rcv timeout.
2023-07-16 19:13:32.0849049 INFO receive_timing_info -                         Read() invoked 1000709 us after write(), rxport.read(1) returned 0 bytes while blocked for 9977 us. Rcv timeout.
2023-07-16 19:13:32.0949135 INFO receive_timing_info -                         Read() invoked 1010697 us after write(), rxport.read(1) returned 0 bytes while blocked for 10000 us. Rcv timeout.
2023-07-16 19:13:32.0949308 INFO receive_timing_info -
TRANSFER STALLED TIMEOUT ERROR: 'rxport::read()' repeatedly timed-out without receiving its requested incoming data. If not induced, inspect+verify the serial connections. Aborting.
```

Next is the third Windows example log, which demonstrates the indefinite blocking issue with read(), where the read timeout setpoint is 0 and all requested receive data doesn't arrive. It also uses the –fulldbg flag enable setting.

Summarizing this run in more detail, the initial half-cycle's write() invocation again transmits a total of (only) 1 byte, but the corresponding read() requests two (2) bytes (per –rxlen). In this run the read timeout is 0 ms, per the specified –rxtmo value. The initial read() is invoked immediately after its matching write() completes, per its –posttxdelayms parameter value of 0. This sets up the situation where our initial read() will likely find no received data available upon initial entry, but this is irrelevant as more significantly it never receives the full two (2) bytes requested by the initial read() call. Consequently, since the –rxtmo value is 0, this results in the problematic infinite blocking read() situation. Baud rates are set to 200,000 baud. Note the –fulldbg flag has resulted in explicit marker text preceding and following each write() and read() invocation, which as shown from the log clearly shows the initial read() never returns – it blocks indefinitely.

```
2023-07-16 19:17:03.334684  INFO receive_timing_info - 'receive_timing_info' cross platform dual RS-232 port null modem cable connected rcv+xmt+timeout test and characterization tool: v1.0
2023-07-16 19:17:03.3348747 INFO receive_timing_info - Test setup: Platform='windows', Baud=200000, rxtmo=0 ms, posttxdelayms=0 ms, xfrstalledtmo=1000 ms, txlen=1, rxlen=2, repeat=1, fulldbg=true
2023-07-16 19:17:03.3377227 INFO receive_timing_info - Test Logfile Name: 'D:\Users\ricej\windows_receive_timing_info.txt'
2023-07-16 19:17:03.4205881 INFO receive_timing_info -
2023-07-16 19:17:03.4205962 INFO receive_timing_info - ** Start of cycle 1. **
2023-07-16 19:17:03.4205977 INFO receive_timing_info - Cycle 1 first phase -> Rx port = 'COM7', Tx port = 'COM6' .
2023-07-16 19:17:03.62881   INFO receive_timing_info - Enter write(txbuf[0..1]).
2023-07-16 19:17:03.6291673 INFO receive_timing_info - Return from write().
2023-07-16 19:17:03.6292021 INFO receive_timing_info - Enter read(rxbuf[0..2]).
```

The above very short log-file content resulted from the test application once it blocked/suspended for nearly two (2) minutes prior to the test operator (myself), eventually terminating the obviously blocked test application from the keyboard.

Next are the equivalent Linux logs for the same three (3) test invocation argument setups, generated on Ubuntu Linux executing in a VMWare guest on the same laptop computer. The first log =>

```
2023-07-16 16:43:14.272409174 INFO receive_timing_info - 'receive_timing_info' cross platform dual RS-232 port null modem cable connected rcv+xmt+timeout test and characterization tool: v1.0
2023-07-16 16:43:14.272426885 INFO receive_timing_info - Test setup: Platform='linux', Baud=200000, rxtmo=50 ms, posttxdelayms=100 ms, xfrstalledtmo=1000 ms, txlen=1, rxlen=2, repeat=1, fulldbg=false
2023-07-16 16:43:14.272434667 INFO receive_timing_info - Test Logfile Name: '/home/ricej/ubuntu_receive_timing_info.txt'
2023-07-16 16:43:14.895900214 INFO receive_timing_info - 
2023-07-16 16:43:14.895949482 INFO receive_timing_info - ** Start of cycle 1. **
2023-07-16 16:43:14.89595447 INFO receive_timing_info - Cycle 1 first phase -> Rx port = '/dev/ttyUSB1', Tx port = '/dev/ttyUSB0' .
2023-07-16 16:43:15.198883731 INFO receive_timing_info - txport.write() sent 1 bytes while blocked for 2425 us. Read() invoked 100269 us after write(), rxport.read(2) returned 1 bytes while blocked for 17 us.
2023-07-16 16:43:15.249000315 INFO receive_timing_info -                              Read() invoked 100344 us after write(), rxport.read(1) returned 0 bytes while blocked for 50064 us. Rcv timeout.
2023-07-16 16:43:15.299208892 INFO receive_timing_info -                              Read() invoked 150429 us after write(), rxport.read(1) returned 0 bytes while blocked for 50186 us. Rcv timeout.
2023-07-16 16:43:15.350425949 INFO receive_timing_info -                              Read() invoked 200662 us after write(), rxport.read(1) returned 0 bytes while blocked for 51169 us. Rcv timeout.
2023-07-16 16:43:15.400619098 INFO receive_timing_info -                              Read() invoked 251881 us after write(), rxport.read(1) returned 0 bytes while blocked for 50143 us. Rcv timeout.
2023-07-16 16:43:15.451641702 INFO receive_timing_info -                              Read() invoked 302069 us after write(), rxport.read(1) returned 0 bytes while blocked for 50977 us. Rcv timeout.
2023-07-16 16:43:15.502401107 INFO receive_timing_info -                              Read() invoked 353102 us after write(), rxport.read(1) returned 0 bytes while blocked for 50699 us. Rcv timeout.
2023-07-16 16:43:15.553152011 INFO receive_timing_info -                              Read() invoked 403870 us after write(), rxport.read(1) returned 0 bytes while blocked for 50684 us. Rcv timeout.
2023-07-16 16:43:15.603731655 INFO receive_timing_info -                              Read() invoked 454621 us after write(), rxport.read(1) returned 0 bytes while blocked for 50516 us. Rcv timeout.
2023-07-16 16:43:15.654415799 INFO receive_timing_info -                              Read() invoked 505188 us after write(), rxport.read(1) returned 0 bytes while blocked for 50633 us. Rcv timeout.
2023-07-16 16:43:15.705176217 INFO receive_timing_info -                              Read() invoked 555874 us after write(), rxport.read(1) returned 0 bytes while blocked for 50707 us. Rcv timeout.
2023-07-16 16:43:15.756190409 INFO receive_timing_info -                              Read() invoked 606634 us after write(), rxport.read(1) returned 0 bytes while blocked for 50962 us. Rcv timeout.
2023-07-16 16:43:15.806492464 INFO receive_timing_info -                              Read() invoked 657645 us after write(), rxport.read(1) returned 0 bytes while blocked for 50252 us. Rcv timeout.
2023-07-16 16:43:15.857302959 INFO receive_timing_info -                              Read() invoked 707978 us after write(), rxport.read(1) returned 0 bytes while blocked for 50730 us. Rcv timeout.
2023-07-16 16:43:15.908021805 INFO receive_timing_info -                              Read() invoked 758762 us after write(), rxport.read(1) returned 0 bytes while blocked for 50666 us. Rcv timeout.
2023-07-16 16:43:15.958705464 INFO receive_timing_info -                              Read() invoked 809474 us after write(), rxport.read(1) returned 0 bytes while blocked for 50637 us. Rcv timeout.
2023-07-16 16:43:16.009378698 INFO receive_timing_info -                              Read() invoked 860161 us after write(), rxport.read(1) returned 0 bytes while blocked for 50621 us. Rcv timeout.
2023-07-16 16:43:16.059899261 INFO receive_timing_info -                              Read() invoked 910848 us after write(), rxport.read(1) returned 0 bytes while blocked for 50456 us. Rcv timeout.
2023-07-16 16:43:16.110283419 INFO receive_timing_info -                              Read() invoked 961356 us after write(), rxport.read(1) returned 0 bytes while blocked for 50332 us. Rcv timeout.
2023-07-16 16:43:16.160563977 INFO receive_timing_info -                              Read() invoked 1011715 us after write(), rxport.read(1) returned 0 bytes while blocked for 50254 us. Rcv timeout.
2023-07-16 16:43:16.211238623 INFO receive_timing_info -                              Read() invoked 1062019 us after write(), rxport.read(1) returned 0 bytes while blocked for 50625 us. Rcv timeout.
2023-07-16 16:43:16.211282556 INFO receive_timing_info - 
TRANSFER STALLED TIMEOUT ERROR: 'rxport::read()' repeatedly timed-out without receiving its requested incoming data. If not induced, inspect+verify the serial connections. Aborting.
```

Here's the second Linux log example. This full log is considerably lengthier than the equivalent Windows log, due to Linux more responsive (i.e. faster) read() timeout returns. For brevity I've excluded many uninteresting intermediate read() log entries soon after the initial byte is read, and resume with the final reads leading up to the terminating transfer stalled timeout error - around 1 second after the initial read() following the earlier matching write().

```
2023-07-16 16:36:48.733217257 INFO receive_timing_info - 'receive_timing_info' cross platform dual RS-232 port null modem cable connected rcv+xmt+timeout test and characterization tool: v1.0
2023-07-16 16:36:48.733246792 INFO receive_timing_info - Test setup: Platform='linux', Baud=200000, rxtmo=1 ms, posttxdelayms=50 ms, xfrstalledtmo=1000 ms, txlen=1, rxlen=2, repeat=1, fulldbg=false
2023-07-16 16:36:48.733254245 INFO receive_timing_info - Test Logfile Name: '/home/ricej/ubuntu_receive_timing_info.txt'
2023-07-16 16:36:49.716092489 INFO receive_timing_info - 
2023-07-16 16:36:49.716142723 INFO receive_timing_info - ** Start of cycle 1. **
2023-07-16 16:36:49.716146655 INFO receive_timing_info - Cycle 1 first phase -> Rx port = '/dev/ttyUSB1', Tx port = '/dev/ttyUSB0' .
2023-07-16 16:36:49.969758513 INFO receive_timing_info - txport.write() sent 1 bytes while blocked for 2773 us. Read() invoked 50299 us after write(), rxport.read(2) returned 1 bytes while blocked for 23 us.
2023-07-16 16:36:49.971226136 INFO receive_timing_info -                              Read() invoked 50380 us after write(), rxport.read(1) returned 0 bytes while blocked for 1420 us. Rcv timeout.
2023-07-16 16:36:49.972910507 INFO receive_timing_info -                              Read() invoked 51818 us after write(), rxport.read(1) returned 0 bytes while blocked for 1662 us. Rcv timeout.
```

```
2023-07-16 16:36:49.974913961 INFO receive_timing_info -                    Read() invoked 53501 us after write(), rxport.read(1) returned 0 bytes while blocked for 1989 us. Rcv timeout.
2023-07-16 16:36:49.976142741 INFO receive_timing_info -                    Read() invoked 55495 us after write(), rxport.read(1) returned 0 bytes while blocked for 1223 us. Rcv timeout.
2023-07-16 16:36:49.977314849 INFO receive_timing_info -                    Read() invoked 56736 us after write(), rxport.read(1) returned 0 bytes while blocked for 1155 us. Rcv timeout.
2023-07-16 16:36:49.97928537 INFO receive_timing_info -                    Read() invoked 57904 us after write(), rxport.read(1) returned 0 bytes while blocked for 1957 us. Rcv timeout.
2023-07-16 16:36:49.98031803 INFO receive_timing_info -                    Read() invoked 59874 us after write(), rxport.read(1) returned 0 bytes while blocked for 1021 us. Rcv timeout.
2023-07-16 16:36:49.982303652 INFO receive_timing_info -                    Read() invoked 60906 us after write(), rxport.read(1) returned 0 bytes while blocked for 1974 us. Rcv timeout.
2023-07-16 16:36:49.984328611 INFO receive_timing_info -                    Read() invoked 62892 us after write(), rxport.read(1) returned 0 bytes while blocked for 2013 us. Rcv timeout.
2023-07-16 16:36:49.986306959 INFO receive_timing_info -                    Read() invoked 64909 us after write(), rxport.read(1) returned 0 bytes while blocked for 1975 us. Rcv timeout.
2023-07-16 16:36:49.988299764 INFO receive_timing_info -                    Read() invoked 66895 us after write(), rxport.read(1) returned 0 bytes while blocked for 1981 us. Rcv timeout.
2023-07-16 16:36:49.990303784 INFO receive_timing_info -                    Read() invoked 68887 us after write(), rxport.read(1) returned 0 bytes while blocked for 1993 us. Rcv timeout.
2023-07-16 16:36:49.992307476 INFO receive_timing_info -                    Read() invoked 70892 us after write(), rxport.read(1) returned 0 bytes while blocked for 1992 us. Rcv timeout.
2023-07-16 16:36:49.99349152 INFO receive_timing_info -                    Read() invoked 72895 us after write(), rxport.read(1) returned 0 bytes while blocked for 1172 us. Rcv timeout.
2023-07-16 16:36:49.995454175 INFO receive_timing_info -                    Read() invoked 74083 us after write(), rxport.read(1) returned 0 bytes while blocked for 1947 us. Rcv timeout.
2023-07-16 16:36:49.996552099 INFO receive_timing_info -                    Read() invoked 76043 us after write(), rxport.read(1) returned 0 bytes while blocked for 1086 us. Rcv timeout.
```

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

```
2023-07-16 16:36:50.94697694 INFO receive_timing_info -                    Read() invoked 1026199 us after write(), rxport.read(1) returned 0 bytes while blocked for 1348 us. Rcv timeout.
2023-07-16 16:36:50.948476102 INFO receive_timing_info -                    Read() invoked 1027599 us after write(), rxport.read(1) returned 0 bytes while blocked for 1448 us. Rcv timeout.
2023-07-16 16:36:50.949956468 INFO receive_timing_info -                    Read() invoked 1029098 us after write(), rxport.read(1) returned 0 bytes while blocked for 1430 us. Rcv timeout.
2023-07-16 16:36:50.951655698 INFO receive_timing_info -                    Read() invoked 1030575 us after write(), rxport.read(1) returned 0 bytes while blocked for 1652 us. Rcv timeout.
2023-07-16 16:36:50.953217239 INFO receive_timing_info -                    Read() invoked 1032274 us after write(), rxport.read(1) returned 0 bytes while blocked for 1515 us. Rcv timeout.
2023-07-16 16:36:50.954952595 INFO receive_timing_info -                    Read() invoked 1033836 us after write(), rxport.read(1) returned 0 bytes while blocked for 1689 us. Rcv timeout.
2023-07-16 16:36:50.956611695 INFO receive_timing_info -                    Read() invoked 1035569 us after write(), rxport.read(1) returned 0 bytes while blocked for 1615 us. Rcv timeout.
2023-07-16 16:36:50.958619843 INFO receive_timing_info -                    Read() invoked 1037231 us after write(), rxport.read(1) returned 0 bytes while blocked for 1961 us. Rcv timeout.
2023-07-16 16:36:50.960044618 INFO receive_timing_info -                    Read() invoked 1039235 us after write(), rxport.read(1) returned 0 bytes while blocked for 1382 us. Rcv timeout.
2023-07-16 16:36:50.962187024 INFO receive_timing_info -                    Read() invoked 1041244 us after write(), rxport.read(1) returned 0 bytes while blocked for 1515 us. Rcv timeout.
2023-07-16 16:36:50.963525653 INFO receive_timing_info -                    Read() invoked 1042804 us after write(), rxport.read(1) returned 0 bytes while blocked for 1294 us. Rcv timeout.
2023-07-16 16:36:50.965264495 INFO receive_timing_info -                    Read() invoked 1044147 us after write(), rxport.read(1) returned 0 bytes while blocked for 1690 us. Rcv timeout.
2023-07-16 16:36:50.966984191 INFO receive_timing_info -                    Read() invoked 1045881 us after write(), rxport.read(1) returned 0 bytes while blocked for 1676 us. Rcv timeout.
2023-07-16 16:36:50.968903071 INFO receive_timing_info -                    Read() invoked 1047599 us after write(), rxport.read(1) returned 0 bytes while blocked for 1876 us. Rcv timeout.
2023-07-16 16:36:50.970892558 INFO receive_timing_info -                    Read() invoked 1049520 us after write(), rxport.read(1) returned 0 bytes while blocked for 1945 us. Rcv timeout.
2023-07-16 16:36:50.970934005 INFO receive_timing_info -
TRANSFER STALLED TIMEOUT ERROR: 'rxport::read()' repeatedly timed-out without receiving its requested incoming data. If not induced, inspect+verify the serial connections. Aborting.
```

Here's the third Linux example log. Again for brevity this shows the leading section of the full log, followed by the section where the first byte is successfully read(), and third the final section where the transfer timeout error occurs. It that it obviously doesn't incur the indefinite blocking which Windows experiences with a read timeout setpoint of 0 and all requested receive data not arriving. Note also in this Linux test run (as compared to its corresponding Windows run) I didn't enable the –fulldbg flag, since it merely increases the log size and skews certain timing data and makes it more difficult to interpret the results. To be clear, I've indeed executed this same test argument configuration on Linux with the --fulldbg flag enabled, and it runs as above, but with many more log entries and skewed timing numbers. You may do so as well, if you're curious about this scenario.

```
2023-07-16 17:33:51.182958831 INFO receive_timing_info - 'receive_timing_info' cross platform dual RS-232 port null modem cable connected rcv+xmt+timeout test and characterization tool: v1.0
2023-07-16 17:33:51.182994242 INFO receive_timing_info - Test setup: Platform='linux', Baud=200000, rxtmo=0 ms, posttxdelayms=0 ms, xfrstalledtmo=1000 ms, txlen=1, rxlen=2, repeat=1, fulldbg=false
2023-07-16 17:33:51.183009314 INFO receive_timing_info - Test Logfile Name: '/home/ricej/ubuntu_receive_timing_info.txt'
2023-07-16 17:33:51.343628307 INFO receive_timing_info -
2023-07-16 17:33:51.343678069 INFO receive_timing_info - ** Start of cycle 1. **
2023-07-16 17:33:51.343682827 INFO receive_timing_info - Cycle 1 first phase -> Rx port = '/dev/ttyUSB1', Tx port = '/dev/ttyUSB0' .
```

```
2023-07-16 17:33:51.54727904 INFO receive_timing_info - txport.write() sent 1 bytes while blocked for 3010 us. Read() invoked 9 us after write(), rxport.read(2) returned 0 bytes while blocked for 12 us. Rcv timeout.
2023-07-16 17:33:51.547330689 INFO receive_timing_info -                          Read() invoked 68 us after write(), rxport.read(2) returned 0 bytes while blocked for 6 us. Rcv timeout.
2023-07-16 17:33:51.547337022 INFO receive_timing_info -                          Read() invoked 80 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.547342551 INFO receive_timing_info -                          Read() invoked 86 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.547348054 INFO receive_timing_info -                          Read() invoked 92 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.547354325 INFO receive_timing_info -                          Read() invoked 97 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.547360614 INFO receive_timing_info -                          Read() invoked 103 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.547366008 INFO receive_timing_info -                          Read() invoked 110 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.547372565 INFO receive_timing_info -                          Read() invoked 115 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.547378652 INFO receive_timing_info -                          Read() invoked 122 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
```

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

```
2023-07-16 17:33:51.561370021 INFO receive_timing_info -                          Read() invoked 14112 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.561376082 INFO receive_timing_info -                          Read() invoked 14119 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.561382249 INFO receive_timing_info -                          Read() invoked 14125 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.561388814 INFO receive_timing_info -                          Read() invoked 14131 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.561394964 INFO receive_timing_info -                          Read() invoked 14138 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.561664472 INFO receive_timing_info -                          Read() invoked 14144 us after write(), rxport.read(2) returned 1 bytes while blocked for 264 us.
2023-07-16 17:33:51.561686314 INFO receive_timing_info -                          Read() invoked 14430 us after write(), rxport.read(1) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 17:33:51.561687918 INFO receive_timing_info -                          Read() invoked 14435 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:51.561689288 INFO receive_timing_info -                          Read() invoked 14437 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 17:33:51.561690846 INFO receive_timing_info -                          Read() invoked 14438 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:51.561692203 INFO receive_timing_info -                          Read() invoked 14440 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 17:33:51.561694084 INFO receive_timing_info -                          Read() invoked 14441 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:51.561695649 INFO receive_timing_info -                          Read() invoked 14443 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
```

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

… … … … … *<intentionally omitted uninteresting **read()** log entries **…** >*

```
2023-07-16 17:33:52.561620759 INFO receive_timing_info -                          Read() invoked 1014367 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.561622825 INFO receive_timing_info -                          Read() invoked 1014370 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.561624658 INFO receive_timing_info -                          Read() invoked 1014372 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.56162675 INFO receive_timing_info -                          Read() invoked 1014373 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.561629146 INFO receive_timing_info -                          Read() invoked 1014376 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.561631284 INFO receive_timing_info -                          Read() invoked 1014378 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.561633332 INFO receive_timing_info -                          Read() invoked 1014380 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.561635306 INFO receive_timing_info -                          Read() invoked 1014382 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.561776316 INFO receive_timing_info -                          Read() invoked 1014384 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 17:33:52.56177811 INFO receive_timing_info -
TRANSFER STALLED TIMEOUT ERROR: 'rxport::read()' repeatedly timed-out without receiving its requested incoming data. If not induced, inspect+verify the serial connections. Aborting.
```

Proposed patch to v4.2.1 'serialport-rs' crate's 'set_timeout()' method for the next crate release. A simple patch- which improves the Windows platform read() behavior and eliminates its 0 timeout read() 'indefinite blocking' problem

This section discusses a Windows platform 'set_timeout()' trait method patch which eliminates the read() indefinite blocking issue when not all requested data is received. The patch conditionally alters (at run-time) the internal serial port timeout constants within the set_timeout() method body, located in the **'serialport-rs\src\windows\com.rs**' source file. The patch revises the Windows read() timeout behavior for this scenario to return immediately from read() with any available received data at initial entry, or immediately with a timeout error when no data is available.

Below I show the current patch to the set_timeout() method, along with corresponding test run logs of the Windows read() behavior with a set_timeout() setting of 0, and where not all data is received for satisfying the posted read()**'s** buffer length. This log shows the read()'s indefinite blocking issue no longer occurs.

To be perfectly clear, the logs above were generated using the existing serialport-rs crate release v4.2.1, while the log below was generated running my patched version of the crate (per the indicated patch).

In the serialport-rs crate's currently released Windows platform specific **'com.rs'** file, at source line 242, is the set_timeout() method trait source =>

```
  fn set_timeout(&mut self, timeout: Duration) -> Result<()> {
   let milliseconds = timeout.as_secs() * 1000 + timeout.subsec_nanos() as u64 / 1_000_000;

   let mut timeouts = COMMTIMEOUTS {
     ReadIntervalTimeout: 0,
     ReadTotalTimeoutMultiplier: 0,
     ReadTotalTimeoutConstant: milliseconds as DWORD,
     WriteTotalTimeoutMultiplier: 0,
     WriteTotalTimeoutConstant: 0,
   };

   if unsafe { SetCommTimeouts(self.handle, &mut timeouts) } == 0 {
     return Err(super::error::last_os_error());
   }

   self.timeout = timeout;
   Ok(())
 }
```

Below is my patched 'com.rs' source file replacement text - with a few explanatory comment lines =>

```
 fn set_timeout(&mut self, timeout: Duration) -> Result<()> {
   let milliseconds = timeout.as_secs() * 1000 + timeout.subsec_nanos() as u64 / 1_000_000;
   let mut read_interval_timeout : u32 = 0u32; // Different internal setting value used if supplied time-out setting is 0 vs a non-zero input arg value.

   if milliseconds == 0 {
      read_interval_timeout = 0xFFFFFFFF; // The 'ReadIntervalTimeout' value setting of MAX_DWORD (0xFFFFFFFF), combined with a 0 'timeout duration parameter,
                                          //   results in an immediate return from read() with whatever bytes are available at entry.  If no data is available, read() immediately
                                          //   returns a Timeout error (and no data). The 'ReadIntervalTimeout' value setting of 0, combined with a positive (> 0) 'timeout duration
                                          //   parameter, waits for up to the indicated timeout period for the read()'s requested bytes to be available, returning
                                          //   whatever is available at the end of the timeout duration or sooner if all requested data is received earlier.
                                          //   If no data is available at the end of the timeout duration, then a timeout error is reported (with no data returned).
    }
    let mut timeouts = COMMTIMEOUTS {
      ReadIntervalTimeout: read_interval_timeout, // 0 or 0xFFFFFFFF,
      ReadTotalTimeoutMultiplier: 0 as DWORD, // milliseconds as DWORD,
      ReadTotalTimeoutConstant: milliseconds as DWORD,
      WriteTotalTimeoutMultiplier: 0,
      WriteTotalTimeoutConstant: 0,
    };

    if unsafe { SetCommTimeouts(self.handle, &mut timeouts) } == 0 {
      return Err(super::error::last_os_error());
    }
    self.timeout = timeout;
    Ok(())
  }
```

While this current patch eliminates the 0 timeout infinite blocking issue, unfortunately it doesn't improve performance for the Windows read() method's non-zero read timeout return responsiveness for non-full buffers. In the non-zero timeout cases where some but not all requested data arrives prior to the time-out period expiring, this simple patch still blocks for the full timeout period. Only if the full request is satisfied earlier than the timeout period, does it return early with the full buffer. Otherwise it returns with the lesser available data buffer at the conclusion of the timeout period - although with no timeout error. I have a concept/idea for a more sophisticated patch to the Windows read() trait method itself, which potentially allows the non-zero timeout scenario read() to return as soon as any data arrives (see next paragraph).

Future work: Based on Microsoft's documentation concerning its native Readfile() system library call, it seems that a re-implemented read() trait method's that internally utilizes Microsoft's proprietary ReadFile() Overlapped IO capability might allow the read() to return almost immediately after receiving any read data and before the full timeout period transpires – as the Linux read() behaves now. I haven't yet had an opportunity to (attempt to) develop and verify this more sophisticated Windows patch, but hope to if there's interest.

Finally, here is the initial section of the Windows log produced by executing the test application built with the above patch, for the (previously) problematic third test run with the scenario of a set_timeout value of 0 and a read() whose requested buffer size worth of data doesn't fully arrive. I've truncated the listing for brevity following receipt of the one and only byte, since with the patched crate our one second of logging (prior to its eventual 'transfer stalled' timeout occurring) becomes quite large. Nevertheless, it's clear that the read() infinite blocking problem doesn't occur with the above patch in effect =>

```
2023-07-16 19:26:01.4057889 INFO receive_timing_info - 'receive_timing_info' cross platform dual RS-232 port null modem cable connected rcv+xmt+timeout test and characterization tool: v1.0
2023-07-16 19:26:01.4061976 INFO receive_timing_info - Test setup: Platform='windows', Baud=200000, rxtmo=0 ms, posttxdelayms=0 ms, xfrstalledtmo=1000 ms, txlen=1, rxlen=2, repeat=1, fulldbg=false
2023-07-16 19:26:01.4090968 INFO receive_timing_info - Test Logfile Name: 'D:\Users\ricej\windows_receive_timing_info.txt'
2023-07-16 19:26:01.456284  INFO receive_timing_info -
2023-07-16 19:26:01.4562918 INFO receive_timing_info - ** Start of cycle 1. **
```

```
2023-07-16 19:26:01.4562933 INFO receive_timing_info - Cycle 1 first phase -> Rx port = 'COM7', Tx port = 'COM6' .
2023-07-16 19:26:01.6590273 INFO receive_timing_info - txport.write() sent 1 bytes while blocked for 277 us. Read() invoked 9 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 19:26:01.6590364 INFO receive_timing_info -                        Read() invoked 23 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6590396 INFO receive_timing_info -                        Read() invoked 26 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590417 INFO receive_timing_info -                        Read() invoked 29 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590457 INFO receive_timing_info -                        Read() invoked 31 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590478 INFO receive_timing_info -                        Read() invoked 35 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590516 INFO receive_timing_info -                        Read() invoked 37 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590534 INFO receive_timing_info -                        Read() invoked 41 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590552 INFO receive_timing_info -                        Read() invoked 43 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590572 INFO receive_timing_info -                        Read() invoked 45 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590595 INFO receive_timing_info -                        Read() invoked 47 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6590612 INFO receive_timing_info -                        Read() invoked 49 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590631 INFO receive_timing_info -                        Read() invoked 51 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590648 INFO receive_timing_info -                        Read() invoked 53 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590665 INFO receive_timing_info -                        Read() invoked 54 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590681 INFO receive_timing_info -                        Read() invoked 56 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590767 INFO receive_timing_info -                        Read() invoked 58 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590786 INFO receive_timing_info -                        Read() invoked 66 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590803 INFO receive_timing_info -                        Read() invoked 68 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659083  INFO receive_timing_info -                        Read() invoked 70 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590901 INFO receive_timing_info -                        Read() invoked 73 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590922 INFO receive_timing_info -                        Read() invoked 80 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590941 INFO receive_timing_info -                        Read() invoked 82 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590957 INFO receive_timing_info -                        Read() invoked 84 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6590984 INFO receive_timing_info -                        Read() invoked 85 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591056 INFO receive_timing_info -                        Read() invoked 88 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591083 INFO receive_timing_info -                        Read() invoked 95 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591101 INFO receive_timing_info -                        Read() invoked 98 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591122 INFO receive_timing_info -                        Read() invoked 100 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591149 INFO receive_timing_info -                        Read() invoked 102 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659117  INFO receive_timing_info -                        Read() invoked 105 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591186 INFO receive_timing_info -                        Read() invoked 107 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591202 INFO receive_timing_info -                        Read() invoked 108 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591239 INFO receive_timing_info -                        Read() invoked 110 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591291 INFO receive_timing_info -                        Read() invoked 114 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591313 INFO receive_timing_info -                        Read() invoked 119 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591333 INFO receive_timing_info -                        Read() invoked 121 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591416 INFO receive_timing_info -                        Read() invoked 123 us after write(), rxport.read(2) returned 0 bytes while blocked for 3 us. Rcv timeout.
2023-07-16 19:26:01.6591485 INFO receive_timing_info -                        Read() invoked 131 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591502 INFO receive_timing_info -                        Read() invoked 138 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591519 INFO receive_timing_info -                        Read() invoked 140 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591537 INFO receive_timing_info -                        Read() invoked 141 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591554 INFO receive_timing_info -                        Read() invoked 143 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591571 INFO receive_timing_info -                        Read() invoked 145 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591589 INFO receive_timing_info -                        Read() invoked 147 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591608 INFO receive_timing_info -                        Read() invoked 149 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659165  INFO receive_timing_info -                        Read() invoked 150 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591666 INFO receive_timing_info -                        Read() invoked 154 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591684 INFO receive_timing_info -                        Read() invoked 156 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591701 INFO receive_timing_info -                        Read() invoked 158 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591718 INFO receive_timing_info -                        Read() invoked 160 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591735 INFO receive_timing_info -                        Read() invoked 161 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591752 INFO receive_timing_info -                        Read() invoked 163 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591772 INFO receive_timing_info -                        Read() invoked 165 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591791 INFO receive_timing_info -                        Read() invoked 167 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591809 INFO receive_timing_info -                        Read() invoked 169 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591827 INFO receive_timing_info -                        Read() invoked 171 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
```

```
2023-07-16 19:26:01.6591843 INFO receive_timing_info -          Read() invoked 172 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659186  INFO receive_timing_info -         Read() invoked 174 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591877 INFO receive_timing_info -          Read() invoked 176 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591894 INFO receive_timing_info -          Read() invoked 177 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591917 INFO receive_timing_info -          Read() invoked 179 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6591935 INFO receive_timing_info -          Read() invoked 181 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591959 INFO receive_timing_info -          Read() invoked 183 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6591975 INFO receive_timing_info -          Read() invoked 185 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659201  INFO receive_timing_info -         Read() invoked 187 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592029 INFO receive_timing_info -          Read() invoked 191 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592066 INFO receive_timing_info -          Read() invoked 192 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6592083 INFO receive_timing_info -          Read() invoked 196 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.65921   INFO receive_timing_info -        Read() invoked 198 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592117 INFO receive_timing_info -          Read() invoked 200 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592139 INFO receive_timing_info -          Read() invoked 201 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6592156 INFO receive_timing_info -          Read() invoked 203 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592174 INFO receive_timing_info -          Read() invoked 205 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6592192 INFO receive_timing_info -          Read() invoked 207 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659221  INFO receive_timing_info -         Read() invoked 209 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592227 INFO receive_timing_info -          Read() invoked 210 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592245 INFO receive_timing_info -          Read() invoked 212 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592262 INFO receive_timing_info -          Read() invoked 214 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659228  INFO receive_timing_info -         Read() invoked 216 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592353 INFO receive_timing_info -          Read() invoked 218 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592371 INFO receive_timing_info -          Read() invoked 225 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592388 INFO receive_timing_info -          Read() invoked 227 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592405 INFO receive_timing_info -          Read() invoked 228 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592437 INFO receive_timing_info -          Read() invoked 230 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592454 INFO receive_timing_info -          Read() invoked 233 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592472 INFO receive_timing_info -          Read() invoked 235 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592488 INFO receive_timing_info -          Read() invoked 237 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592504 INFO receive_timing_info -          Read() invoked 238 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592521 INFO receive_timing_info -          Read() invoked 240 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592538 INFO receive_timing_info -          Read() invoked 242 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592555 INFO receive_timing_info -          Read() invoked 243 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592572 INFO receive_timing_info -          Read() invoked 245 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592591 INFO receive_timing_info -          Read() invoked 247 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592612 INFO receive_timing_info -          Read() invoked 249 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659263  INFO receive_timing_info -         Read() invoked 251 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592653 INFO receive_timing_info -          Read() invoked 253 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592678 INFO receive_timing_info -          Read() invoked 255 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592696 INFO receive_timing_info -          Read() invoked 257 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592712 INFO receive_timing_info -          Read() invoked 259 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592729 INFO receive_timing_info -          Read() invoked 261 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592788 INFO receive_timing_info -          Read() invoked 262 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6592806 INFO receive_timing_info -          Read() invoked 268 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592837 INFO receive_timing_info -          Read() invoked 270 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592854 INFO receive_timing_info -          Read() invoked 273 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592871 INFO receive_timing_info -          Read() invoked 275 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592904 INFO receive_timing_info -          Read() invoked 277 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659292  INFO receive_timing_info -         Read() invoked 280 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592938 INFO receive_timing_info -          Read() invoked 282 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592955 INFO receive_timing_info -          Read() invoked 283 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592974 INFO receive_timing_info -          Read() invoked 285 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6592994 INFO receive_timing_info -          Read() invoked 287 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593012 INFO receive_timing_info -          Read() invoked 289 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593034 INFO receive_timing_info -          Read() invoked 291 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6593059 INFO receive_timing_info -          Read() invoked 293 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
```

```
2023-07-16 19:26:01.6593076 INFO receive_timing_info -          Read() invoked 295 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593095 INFO receive_timing_info -          Read() invoked 297 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593112 INFO receive_timing_info -          Read() invoked 299 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659313  INFO receive_timing_info -          Read() invoked 301 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.659315  INFO receive_timing_info -          Read() invoked 303 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593168 INFO receive_timing_info -          Read() invoked 305 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593203 INFO receive_timing_info -          Read() invoked 306 us after write(), rxport.read(2) returned 0 bytes while blocked for 2 us. Rcv timeout.
2023-07-16 19:26:01.6593224 INFO receive_timing_info -          Read() invoked 310 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593243 INFO receive_timing_info -          Read() invoked 312 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593263 INFO receive_timing_info -          Read() invoked 314 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593281 INFO receive_timing_info -          Read() invoked 316 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593299 INFO receive_timing_info -          Read() invoked 318 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593318 INFO receive_timing_info -          Read() invoked 319 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593335 INFO receive_timing_info -          Read() invoked 321 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593353 INFO receive_timing_info -          Read() invoked 323 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593373 INFO receive_timing_info -          Read() invoked 325 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593399 INFO receive_timing_info -          Read() invoked 327 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593417 INFO receive_timing_info -          Read() invoked 329 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593436 INFO receive_timing_info -          Read() invoked 331 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593455 INFO receive_timing_info -          Read() invoked 333 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593494 INFO receive_timing_info -          Read() invoked 335 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6593514 INFO receive_timing_info -          Read() invoked 339 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593531 INFO receive_timing_info -          Read() invoked 341 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659355  INFO receive_timing_info -          Read() invoked 343 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593568 INFO receive_timing_info -          Read() invoked 344 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593588 INFO receive_timing_info -          Read() invoked 346 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593606 INFO receive_timing_info -          Read() invoked 348 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593624 INFO receive_timing_info -          Read() invoked 350 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593641 INFO receive_timing_info -          Read() invoked 352 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659366  INFO receive_timing_info -          Read() invoked 354 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593678 INFO receive_timing_info -          Read() invoked 355 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593698 INFO receive_timing_info -          Read() invoked 357 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593715 INFO receive_timing_info -          Read() invoked 359 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593755 INFO receive_timing_info -          Read() invoked 361 us after write(), rxport.read(2) returned 0 bytes while blocked for 2 us. Rcv timeout.
2023-07-16 19:26:01.6593779 INFO receive_timing_info -          Read() invoked 365 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6593817 INFO receive_timing_info -          Read() invoked 367 us after write(), rxport.read(2) returned 0 bytes while blocked for 2 us. Rcv timeout.
2023-07-16 19:26:01.6593836 INFO receive_timing_info -          Read() invoked 371 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593853 INFO receive_timing_info -          Read() invoked 373 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659387  INFO receive_timing_info -          Read() invoked 375 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593887 INFO receive_timing_info -          Read() invoked 377 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593924 INFO receive_timing_info -          Read() invoked 378 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593941 INFO receive_timing_info -          Read() invoked 382 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593974 INFO receive_timing_info -          Read() invoked 384 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6593991 INFO receive_timing_info -          Read() invoked 387 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659401  INFO receive_timing_info -          Read() invoked 389 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594028 INFO receive_timing_info -          Read() invoked 391 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594045 INFO receive_timing_info -          Read() invoked 392 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594063 INFO receive_timing_info -          Read() invoked 394 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594081 INFO receive_timing_info -          Read() invoked 396 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594116 INFO receive_timing_info -          Read() invoked 398 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6594139 INFO receive_timing_info -          Read() invoked 401 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594158 INFO receive_timing_info -          Read() invoked 403 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594174 INFO receive_timing_info -          Read() invoked 405 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594192 INFO receive_timing_info -          Read() invoked 407 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594209 INFO receive_timing_info -          Read() invoked 409 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594227 INFO receive_timing_info -          Read() invoked 410 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594244 INFO receive_timing_info -          Read() invoked 412 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594262 INFO receive_timing_info -          Read() invoked 414 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
```

```
2023-07-16 19:26:01.6594279 INFO receive_timing_info -    Read() invoked 416 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594297 INFO receive_timing_info -    Read() invoked 417 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594314 INFO receive_timing_info -    Read() invoked 419 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594333 INFO receive_timing_info -    Read() invoked 421 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594352 INFO receive_timing_info -    Read() invoked 423 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594374 INFO receive_timing_info -    Read() invoked 425 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6594412 INFO receive_timing_info -    Read() invoked 427 us after write(), rxport.read(2) returned 0 bytes while blocked for 2 us. Rcv timeout.
2023-07-16 19:26:01.659443  INFO receive_timing_info -   Read() invoked 431 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594448 INFO receive_timing_info -    Read() invoked 433 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594473 INFO receive_timing_info -    Read() invoked 434 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594504 INFO receive_timing_info -    Read() invoked 437 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594522 INFO receive_timing_info -    Read() invoked 440 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594577 INFO receive_timing_info -    Read() invoked 442 us after write(), rxport.read(2) returned 0 bytes while blocked for 4 us. Rcv timeout.
2023-07-16 19:26:01.6594595 INFO receive_timing_info -    Read() invoked 447 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594612 INFO receive_timing_info -    Read() invoked 449 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594631 INFO receive_timing_info -    Read() invoked 451 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659469  INFO receive_timing_info -   Read() invoked 457 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594708 INFO receive_timing_info -    Read() invoked 459 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594745 INFO receive_timing_info -    Read() invoked 460 us after write(), rxport.read(2) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6594782 INFO receive_timing_info -    Read() invoked 464 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.659533  INFO receive_timing_info -   Read() invoked 468 us after write(), rxport.read(2) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6595498 INFO receive_timing_info -    Read() invoked 534 us after write(), rxport.read(2) returned 1 bytes while blocked for 3 us.
2023-07-16 19:26:01.6595519 INFO receive_timing_info -    Read() invoked 539 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6595539 INFO receive_timing_info -    Read() invoked 541 us after write(), rxport.read(1) returned 0 bytes while blocked for 1 us. Rcv timeout.
2023-07-16 19:26:01.6595556 INFO receive_timing_info -    Read() invoked 543 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6595572 INFO receive_timing_info -    Read() invoked 545 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6595598 INFO receive_timing_info -    Read() invoked 547 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6595617 INFO receive_timing_info -    Read() invoked 549 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.6595633 INFO receive_timing_info -    Read() invoked 551 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
2023-07-16 19:26:01.659565  INFO receive_timing_info -   Read() invoked 553 us after write(), rxport.read(1) returned 0 bytes while blocked for 0 us. Rcv timeout.
```

<This space left intentionally blank>