MultiCom/MV 1.0

Software Guide for
LabVIEW® for Windows™

Viewpoint Systems, Inc. does not warrant that the Program will meet Customer's requirements or will operate in the combinations which may be selected by the Customer or that the operation of the Program will be uninterrupted or error free or that all Program defects will be corrected.

VIEWPOINT SYSTEMS, INC. DOES NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS THAT MAY BE OBTAINED BY USING THIS SOFTWARE. ACCORDINGLY, THE SOFTWARE AND ITS DOCUMENTATION ARE SOLD "AS IS" WITHOUT WARRANTY AS TO THEIR PERFORMANCE, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE PROGRAM IS ASSUMED BY YOU.

NEITHER VIEWPOINT SYSTEMS, INC. NOR ANYONE ELSE WHO HAS BEEN INVOLVED IN THE CREATION, PRODUCTION, OR DELIVERY OF THIS SOFTWARE SHALL BE LIABLE FOR ANY DIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, SUCH AS, BUT NOT LIMITED TO, LOSS OF ANTICIPATED PROFITS OR BENEFITS, RESULTING FROM THE USE OF THE PROGRAM OR ARISING OUT OF ANY BREACH OF ANY WARRANTY. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF DIRECT INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR THE ABOVE MAY NOT APPLY TO YOU.

LabVIEW® is a trademark of National Instruments Inc.

All other brand and product names are trademarks or registered trademarks of their respective companies.

Viewpoint Systems, Inc.
800 West Metro Parkway
Rochester, NY 14623-2640

Printed in the U.S.A.
August 2000
Part Number: VS-MCMV001

Table Of Contents

Introduction

MultiCom/MV is a set of LabVIEW VIs that allow communication with serial interface boards from multiple vendors. Depending upon the hardware that was purchased, the MultiCom/MV system supports either 8 or 16 RS-232 or RS-422 communications ports. Each port is configurable to support a wide range of baud rates, handshake modes and other communications parameters.

For specific installation instructions, refer to the MultiCom hardware guide that is included with the system.

MultiCom Files

MCOM32.DLL
This file is the Windows driver file the software uses to interface to the MultiCom hardware. This file is placed in the \WINDOWS\SYSTEM directory for Windows 95/98 installations and in \WINDOWS\SYSTEM32 for Windows NT and Windows 2000 installations.

README.TXT
This file contains any last minute information not included in either the MultiCom/MV software or hardware guides.

SERIAL.LLB

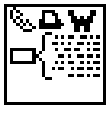
Use this set of VIs if you intend to convert existing VIs that use the computer's COM ports to use the MultiCom ports. This file is designed to look exactly like the SERIAL.LLB file that comes from National Instruments as a part of LabVIEW. The MultiCom SERIAL.LLB contains five VIs:

- Bytes at Serial Port.vi
- Serial Port Break.vi
- Serial Port Init.vi
- Serial Port Read.vi
- Serial Port Write.vi

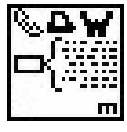
Note that the names of the VIs are exactly the same as the names of the VIs in the original SERIAL.LLB. This was done so that an existing LabVIEW application can be switched over to use the MultiCom software to access either PC or MultiCom COM ports by simply replacing the original SERIAL.LLB with the MultiCom SERIAL.LLB. Once the MultiCom SERIAL.LLB is in place, LabVIEW will automatically use the new VIs instead of the old ones. The MultiCom VIs are wired the same as the LabVIEW serial VIs.

When replacing the original SERIAL.LLB be careful not to overwrite or delete it. It is a good idea to rename the original SERIAL.LLB file before copying the MultiCom version onto your hard disk.

Once you have replaced the serial VI library, the only noticeable difference that you will see is on the lower right corner of the icons where there will be a small "m" to signify "MultiCom" as shown below.



Original icon



MultiCom icon

SERIALMC.LLB

This library contains 14 VIs. The first five are renamed copies of the VIs in the SERIAL.LLB.

- MultiCom Bytes at Serial Port.vi
- MultiCom Serial Port Break.vi
- MultiCom Serial Port Init.vi
- MultiCom Serial Port Read.vi
- MultiCom Serial Port Write.vi

Three VIs are permutations of the original five VIs listed above. MultiCom Serial Port Init+.vi allows different input and output buffer sizes to be specified. MultiCom Serial Port Read+.vi includes EOS (End-Of-String) detection and timeout. MultiCom Serial Port Write+.vi includes a timeout value.

- MultiCom Serial Port Init+.vi
- MultiCom Serial Port Read+.vi
- MultiCom Serial Port Write+.vi

The remaining VIs were included to increase the power and ease-of-use of the MultiCom/MV software.

- MultiCom Bytes at Output Serial Port.vi
- MultiCom Comm Input Status.vi
- MultiCom CTS/DSR Status.vi
- MultiCom EOS Status.vi
- MultiCom Output Buf Clear.vi
- MultiCom RTS/DTR Control.vi

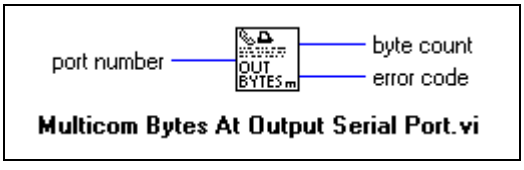
MultiCom VIs

The VIs described in this section are the MultiCom VIs that go beyond the functionality of the serial VIs provided in LabVIEW. For information on the following VIs:

- MultiCom Bytes at Serial Port.vi
- MultiCom Serial Port Break.vi
- MultiCom Serial Port Init.vi
- MultiCom Serial Port Read.vi
- MultiCom Serial Port Write.vi

refer to the documentation on the equivalent VI in the LabVIEW "Function and VI Reference Manual."

MultiCom Bytes at Output Serial Port.vi

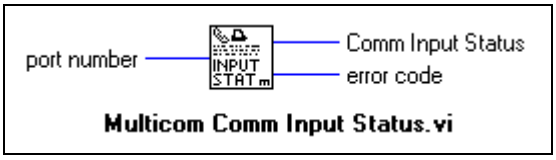


This VI returns the number of bytes waiting to be transmitted in the output buffer. Typically the only time the byte count will be a non-zero value is if some sort of output handshaking is enabled and the receiver is holding off the data transmission.

I32 ***port number*** is the zero-based MultiCom port number.

U16 ***byte count*** is the number of bytes in the output buffer that have not been sent.

MultiCom Comm Input Status.vi



I32 ***port number*** is the zero-based MultiCom port number.

U16 ***Comm Input Status*** returns a bit field with the following data:

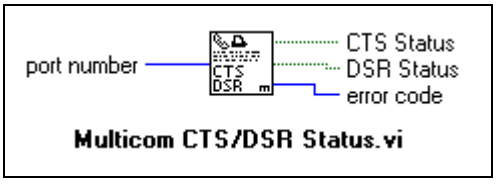
Bit 0: Input data overrun error - updated after a read; is set when both the input buffer and the FIFO (12 bytes) are full.

Bit 1: Parity error in input stream

Bit 2: Framing error in input stream

Bit 3: Break character received

MultiCom CTS/DSR Status.vi

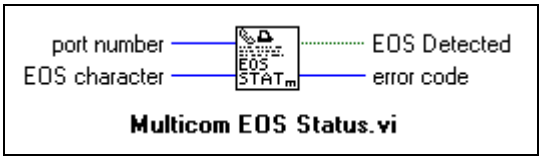


I32 ***port number*** is the zero-based MultiCom port number.

TF ***CTS Status*** returns the status of the "Clear to Send" line. A true indicates that CTS is asserted.

TF ***DSR Status*** returns the status of the "Data Set Ready" line. A true indicates that DSR is asserted.

MultiCom EOS Status.vi



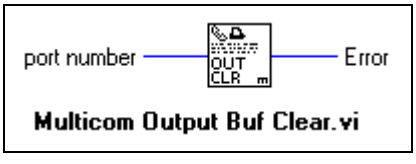
This VI checks the input buffer for the specified "end of string" character. No data is removed or transferred out of the input buffer. Use this VI to check for an EOS character before the MultiCom Serial Port Read+.vi is called to eliminate the need for the read operation to wait for the reception of the EOS character.

I32 ***port number*** is the zero-based MultiCom port number.

U8 ***EOS character*** is the ASCII value of the character to be checked for.

TF ***EOS Detected*** returns a true if the *EOS character* is detected in the input buffer.

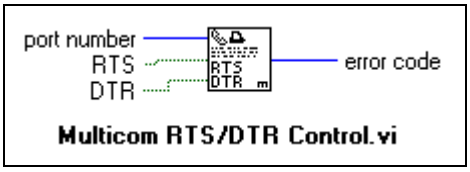
MultiCom Output Buf Clear.vi



This VI clears (empties) the output buffer of the specified port. Use this VI if the buffer needs to be cleared because the receiving device is holding off data transfer with either the CTS line or an XOff handshake.

I32 ***port number*** is the zero-based MultiCom port number.

MultiCom RTS/DTR Control.vi



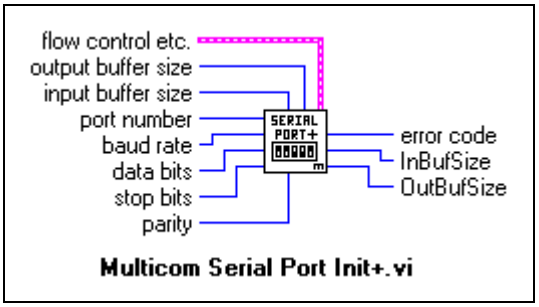
This VI controls the RTS and DTR lines. Note that this VI will set the state of these lines but it does not guarantee that the lines will remain at that state. If input handshaking is enabled, the RTS and DTR lines may change due to input buffer changes.

I32 ***port number*** is the zero-based MultiCom port number.

TF ***RTS*** sets the status of the "Request to Send" line. A true value asserts the RTS line.

TF ***DTR*** sets the status of the "Data Transfer Ready" line. A true value asserts the DTR line.

MultiCom Serial Port Init+.vi



This VI performs the same function as the MultiCom Serial Port Init.vi with several additional inputs to increase functionality. The primary new feature of this VI is the ability to set the input buffer and output buffer to different sizes. Only the additional inputs are documented here – the functionality of the other inputs is unchanged.

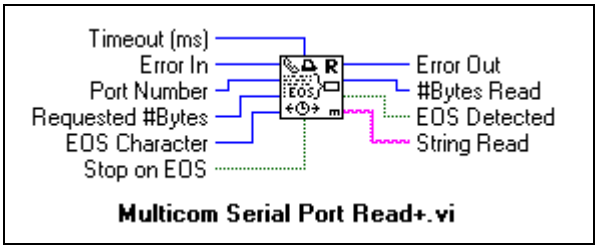
U16 ***output buffer size*** sets the size of the output buffer. A value of zero leaves the buffer size unchanged.

U16 ***input buffer size*** sets the size of the input buffer. A value of zero leaves the buffer size unchanged.

U16 ***OutBufSize*** returns the size that the output buffer is actually set to. If the requested size is too large, the VI will return a –4 error code and the buffer sizes will be unchanged. See "Buffer Sizes" in the accompanying MultiCom hardware guide for more information.

U16 ***InBufSize*** returns the size that the input buffer is actually set to. If the requested size is too large, the VI will return a –4 error code and the buffer sizes will be unchanged. See "Buffer Sizes" in the accompanying MultiCom hardware guide for more information.

MultiCom Serial Port Read+.vi



This VI performs the same function as the MultiCom Serial Port Read VI but includes options for EOS (End-Of-String) detection and timeout. Also, this VI includes an Error In terminal for easy sequencing of multiple reads.

U32 ***Timeout*** is the maximum time in milliseconds that the MultiCom software will wait for a character to be received. If the *Timeout* period expires, the MultiCom Serial Port Read+ will return with *Error Out* equal to -2. To turn off the timeout feature, set *Timeout* to 0 (zero).

I32 ***Error In*** controls whether the VI will execute. If *Error In* is 0 (zero), the VI will execute normally. If *Error In* is non-zero, the VI will simply pass the *Error In* value through to *Error Out* without executing a read.

Error In can be used to control the sequencing of multiple read+ VIs without using a LabVIEW sequence frame.

I32 ***Port Number*** is the MultiCom port number to be read. Port numbering is zero based.

U32 ***Requested # Bytes*** is the maximum number of characters that will be read from the MultiCom port regardless of whether an EOS is found.

EOS Character is the ASCII code for the character which will cause the serial read to stop (if Stop on EOS is turned on). For example, if a serial device returns a string of the form,

+#.####<cr><lf>

setting the EOS Character to 10 (a linefeed character) will cause the read to stop after it has received the linefeed character.

Stop on EOS determines whether the read will search for an EOS character. The read will be terminated on EOS if *Stop on EOS* is true.

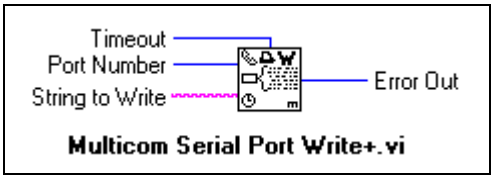
Error Out is the error value returned by the read function. If *Error Out* is a negative number, a board level error occurred.

#Bytes Read is the length of the *String Read.*

EOS Detected shows whether the read stopped because of an EOS detection.

String Read is the data received from the MultiCom port.

MultiCom Serial Port Write+.vi



This VI performs the same function as the MultiCom Serial Port Write VI but includes an option for timeout.

U32 ***Timeout*** is the maximum time in milliseconds that the MultiCom software will wait to come back from a write operation. If the *Timeout* period expires, the MultiCom Serial Port Write+ will return with *Error Out* equal to -2. To turn off the timeout feature, set *Timeout* to 0 (zero).

I32 ***Port Number*** is the MultiCom port number to be written to. Port numbering is zero based.

abc ***String to Write*** is the data that is written to the MultiCom port.

I32 ***Error Out*** is the error value returned by the write function. If *Error Out* is a negative number, a board level error occurred.

Reads/Writes May Hang LabVIEW

The MultiCom software is not interrupt driven and therefore once LabVIEW enters the MultiCom code, no Windows activity (LabVIEW or otherwise) can occur. Consequently, if you request more characters to be read than have been transmitted, the computer will wait indefinitely until the requested number of characters are read (unless a timeout is specified or an EOS character is detected). Likewise, if a write is held off due to a handshake situation, the computer will wait indefinitely until the hold off condition is released. To avoid this problem it is important to follow these guidelines:

Reads

- Use the *MultiCom Bytes at Serial Port* VI to determine the number of bytes that can be read. The proper use of this VI will also increase the execution efficiency of your code since no time will be spent in the MultiCom software waiting to data to be received.
- Use the *MultiCom Serial Port Read+* VI with timeout enabled to prevent LabVIEW from hanging if no data is available.

Writes

- Use the *MultiCom Bytes at Output Port* VI to determine if there is room in the output buffer for the new output string. The proper use of this VI will also increase the execution efficiency of your code since no time will be spent in the MultiCom software waiting for the output buffer to empty.
- Use the *MultiCom Serial Port Write+* VI with timeout enabled to prevent LabVIEW from hanging due to handshaking.

If LabVIEW hangs in one of these wait loops, press <Ctrl><Alt><Del> followed by <Enter> to shut down LabVIEW and return to Windows.

Error Codes

Listed below are the error codes returned by the MultiCom VIs. The VIs are listed under the VI names in SERIALMC.LLB. The VIs in SERIAL.LLB return the same error codes as their equivalent functions in SERIALMC.LLB.

All MultiCom VIs return the following error codes

0 – no error

-1 – port not initialized

-2 – timeout exceeded

-3 – port does not exist

-4 – insufficient resources to perform operation

-5 – invalid parameter specified

-999 – Windows system error

How to Reach Us

Viewpoint Systems, Inc.
800 West Metro Parkway
Rochester, NY 14623-2640

phone: (716) 475-9555
fax: (716) 475-9645
e-mail: support@ViewpointUSA.com

Technical support is available by e-mail, fax or voice. Phone support is available business days, 9:00 a.m. to 5:00 p.m. Eastern time.

Before calling for technical support please double check your work. When calling, it is important to have all relevant information on hand.