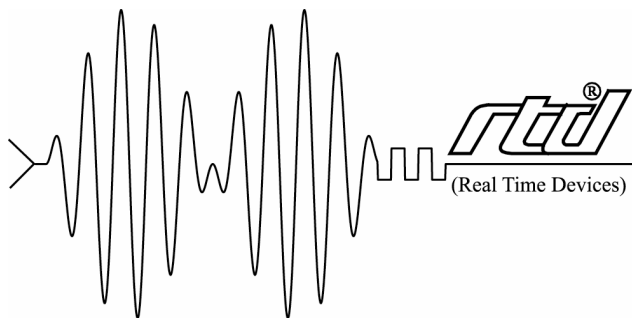


APPLICATION NOTE

Programming Serial Ports in RS422/RS485 Mode



RTD Embedded Technologies, Inc.

"Accessing the Analog World"®

SWM-64000012
rev. B

APPLICATION NOTE

Programming Serial Ports in RS422/RS485 Mode



RTD Embedded Technologies, INC.

103 Innovation Blvd.
State College, PA 16803-0906

Phone: +1-814-234-8087

FAX: +1-814-234-5218

E-mail

sales@rtd.com

techsupport@rtd.com

web site

<http://www.rtd.com>

Revision History

03/26/2004	Revision A issued. New manual naming method.
06/11/2004	Revision B issued. Removed references to ANC115. Cleaned up formatting. Cleaned up copyright and trademarks. Added section for Linux

Published by:

RTD Embedded Technologies, Inc.
103 Innovation Blvd.
State College, PA 16803-0906

Copyright 2004 by RTD Embedded Technologies, Inc.
All rights reserved
Printed in U.S.A.

The RTD Logo is a registered trademark of RTD Embedded Technologies. cpuModule and utilityModule are trademarks of RTD Embedded Technologies. MS-DOS, Windows, Windows 95, Windows 98, Windows NT, and Windows XP are trademarks of Microsoft Corp. All other trademarks appearing in this document are the property of their respective owners.

Introduction

All cpuModules and utilityModules CM310, CM312, CM313 designed by Real Time Devices have serial ports which support RS232 and RS422/485. RS-232 is a well known interface used to connect a computer to serial mice, modems and other devices. RS422/485 is less popular but it has some advantages such as cable length up to 4000 feet and the option to connect up to 32 computers/devices in a network. Due to the fact that there are no standard RS422/485 devices (like mice or modems) users need to develop their own protocol and software for serial communication with RS422/485. For more details on RS422/485 see a book: Jan Axelson "Serial Port Complete" ISBN 0965081923

Programming

In software, using RS422/485 is very similar to using RS232. One difference is that when using interface RS422/485 several computers can be connected together. When connecting more than 2 nodes some form of arbitration is needed. The user must develop a protocol to make sure that no two devices send data at the same time. Usually, communication is initiated by a specified master computer. The other computer/device transmits data in reply to the master's request. All other computers/devices would stay in receiving mode with disabled sending (line break). To control sending, the RTS signal is used. On all RTD boards, a low RTS signal enables sending and high RTS disables it. Hardware flow control is not used for RS422/485 communications because RTS is in use.

DOS Example Code

```
/******  
File Name:          DRV485.C  
Operating System:   ROM-DOS  
Compiler:           Borland C++ 3.1  
Version:            1.0
```

Sending a string of data. The first byte to be sent is the address of the recipient computer/device. All computers must check the address and ignore the message if it is different from its own.

PORT – IO address the serial port.

TERMINATION – end of transmitting byte = 0xC

Copyright (c) 2003, Real Time Devices USA, Inc.

```
*****/
```

```
//send a string to the other computer  
//address - address of the other computer/device  
// commandString - pointer to a string to send  
// returns TRUE if string sent,  
// FALSE if error occur
```

```
BOOL sendString(char address, char* commandString) {  
    char byteToSend=0;  
    //flag that sets when TX buffer is ready to get more bytes  
    char checkbuffer = 0x20;  
    /* Turn RTS off to enable transmission*/  
    outportb(portVar.PORT + 4 , 0x09);  
    outportb(portVar.PORT, address);  
    while (1) {  
        // wait for buffer ready or empty (depending on buffercheck)  
        // wait until buffer is ready  
        while((inportb(portVar.PORT+5) & checkbuffer) == 0x00);  
        if (TERMINATION==byteToSend) break;
```

```

        if (*commandString) {
            byteToSend = *commandString;
            commandString++;
        }
        else {
            byteToSend = TERMINATION;
            checkbuffer = 0x40;        //flag that sets when TX buffer is empty
        }
        outputb(portVar.PORT, byteToSend);
    }
    /* Turn RTS on to disable transmission */
    outputb(portVar.PORT + 4 , 0x0B);
    return TRUE;
}

```

Windows Example Code

This code is a modified Visual C++ 6.0 example from the Microsoft Development Network (MSDN). The original example program can be downloaded from the following URL:

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vcsample98/html/vcsmpserialsampleforcommunicationsdemonstration.asp>

```

/*****
File Name      : TTY.C
Operating System : Windows 95, 98, ME, NT, 2000, XP
Compiler       : Visual C++
Version        : 6.0

*   This is a part of the Microsoft Source Code Samples.
*   Copyright (C) 1993-1997 Microsoft Corporation.
*   All rights reserved.
*****/

```

To make this program work with interface RS422/485 need to modify the following code of the original program:

```

1) In function
BOOL NEAR SetupConnection( HWND hWnd )
Replace code:
    if (bSet)
        dcb.fDtrControl = DTR_CONTROL_HANDSHAKE ;
    else
        dcb.fDtrControl = DTR_CONTROL_ENABLE ;

```

with the following code:

```

    dcb.rRtsControl = RTS_CONTROL_DISABLE; //disable hardware flow control

```

```

2) In function
BOOL NEAR WriteCommBlock( HWND hWnd, LPSTR lpByte , DWORD dwBytesToWrite)

```

After code:

```

    if (NULL == (npTTYInfo = GETNPTTYINFO( hWnd )))
        return ( FALSE ) ;

```

need to insert the following code:

```
EscapeCommFunction( COMDEV(npTTYINFO), CLRRTS); // Clear RTS right before
                                                    //starting transmitting
```

3) In function
DWORD FAR PASCAL CommWatchProc(LPSTR lpData)

Need to replace the following code:
if (!SetCommMask(COMDEV(npTTYInfo), EV_RXCHAR))
 return (FALSE) ;

With the following:
if (!SetCommMask(COMDEV(npTTYInfo), EV_RXCHAR|EV_TXEMPTY))
 return (FALSE) ; // Add the event to monitor: empty transmitting buffer

After the following code:
WaitCommEvent(COMDEV(npTTYInfo), &dwEvtMask, NULL);

Need to add the following code:
if ((dwEvtMask & EV_TXEMPTY) == EV_TXEMPTY) {
 EscapeCommFunction(COMDEV(npTTYInfo), SETRTS) ; // Set RTS
 //immediately after transmitting is done
}

Linux Example Code

For an example of how to use RS422/485 serial port mode under Linux, please see the Software Product SWP-700020032 "RS422/485 Serial Port Mode Example Program for Linux" available from the RTD web site.