

rs232...

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2006-09/msg00630.html>

- *From:* Robby <Robby@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 22 Sep 2006 00:11:01 -0700
-

Hello Scott and everyone else who will read this post.

After many messages that I posted on the news groups about rs232, I have come up with functional code which reads in as many as 25 characters in one operation, meaning that a micro-controller can send 25 bytes to the PC and VC++ has no problem reading this. The code can be altered to read many more bytes than 25 if required, however I only tested up to 25. All you need to do is to change the first defined statement > BYTESTOREAD!

Scott, I have learnt a lot form our exchange of posts, so I figure I would like to share with you, what I have come up with, and I would appreciate your opinion & However, I know that the code can probably be made more efficient, however I think, for me, for now, its quite okay!

This is not the way I will leave the code, there will be some final touches such as better placement of #define statements and so forth!

I will put comments under every line of code, so if you disagree with the commentary feel free to correct it.

```
////////////////////////////////////
```

```
#include <windows.h>
```

```
CSerialWnd MySerial;  
//Create an object of Cserial type.
```

```
LRESULT CALLBACK WndProc_CW1 (HWND hwnd, UINT message,  
WPARAM wParam, LPARAM lParam)  
{
```

```
//RS232 *Receive mode
```

```
#define BYTESTOREAD 25 //Depicts the amount of bytes to read!  
#define OB_BYTES ((BYTESTOREAD*2)+1) //Number of bytes for official Buffer !
```

```
static DWORD dwBytesRead;  
//Declaration of variable to indicate actual bytes that are  
//read in within the time limit that windows allowed!
```

rs232...

```
static byte CatchByte_BUFFER[BYTESTOREAD+1];
//Bytes returned from the MySerial.Read() function, will be
//immediatelly assigned to this array.

static byte Accum_BUFFER[BYTESTOREAD+1];
//Since windows cannot assure a constant amount of bytes
//returned from the MySerial.Read() function(which come
// from the rs232 port), then when CatchByteBUFFER
//is full, I transfer its bytes to the Accum_BUFFER. So
//now, CatchByteBUFFER is ready for another, set of bytes
//returned from the next MySerial.Read() function.
static TCHAR Official_BUFFER[OB_BYTES];
//This array holds the final and complete string that was read
in .

int k,n,j;
//General purpose variables used to loop and index arrays

//RS232 SERIAL PORT MONITOR
if (message == CSerialWnd::mg_nDefaultComMsg)
{
const CSerialWnd::EEvent eEvent = SerialWnd::EEvent(LOWORD(wParam));
const CSerialWnd::EError eError = CSerialWnd::EError(HIWORD(wParam));

switch (eEvent)
{
case CSerialWnd::EEventRecv:
// A serial message occurred
j = 0;
//Holds Accumulative array index value

do
{
MySerial.Read(CatchByte_BUFFER,
sizeof(CatchByte_BUFFER),
&dwBytesRead);
//Read the port, and the third parameter
//will hold the amount of bytes read in!

CatchByte_BUFFER[dwBytesRead] = '\0';
//Null Terminate the CatchByte_BUFFER array!

n = dwBytesRead;
//Re-assign the amount of bytes read in to another
variable&
//The reason for this was that I would get the
following error, if I
//were to put this variable straight into the for
loop command:
//c:\_DTS_PROGRAMMING\C_PROGRAMMING\vc++\MY_APPS_LAB
```

rs232...

rs232...

```
// \XPPLC\WndProc_CW1.cpp(220) : warning C4018: '<':  
//signed/unsigned mismatch.  
//So for now, I just re-assigned it!
```

```
for(k=0;k<n;k++)
```

```
{  
Accum_BUFFER[k+j] = CatchByte_BUFFER[k];  
//Store the bytes that were just read in into an  
//accumulator array!  
}
```

```
j = j + dwBytesRead;
```

```
//Advance the index for the Accum_BUFFER array  
//so we can append the next bytes which reside in the  
//CatchByte_BUFFER array in the next iteration!  
}
```

```
while(j != (BYTESTOREAD) && j != 0);
```

```
//If all the bytes were read in, then , j will
```

```
be equal to
```

```
//BYTESTOREAD and will break out of the while  
loop, and
```

```
//also, for unnecessary subsequent port reads  
where
```

```
//dwBytesRead will equal to 0, meaning that no  
bytes were
```

```
//read, then this will cause the breaking out
```

```
//of the while loop aswell!
```

```
MultiByteToWideChar(CP_ACP,0, (LPCSTR)Accum_BUFFER,-1,Official_BUFFER,OB_BYTES);
```

```
//Finally, assign the fully accumulated bytes
```

```
//to a TCHAR string array, which are  
converted
```

```
//to unicode, if I am not mistaken!
```

```
break;
```

```
default:
```

```
break;
```

```
}
```

```
return 0;
```

```
}
```

```
switch(message)
```

```
{
```

```
case WM_CREATE:
```

```
MySerial.Open(TEXT("COM1"),hwnd,WM_NULL,lParam,0,0);
```

```
//Open the rs232 port
```

```
MySerial.Setup(CSerial::EBaud9600,CSerial::EData8,
```

```
CSerial::EParNone,CSerial::EStop1);
```

```
//Set up the rs232 port
```

rs232...

rs232...

```
return 0;

case WM_CLOSE:
MySerial.Close(); //Close Port!
break;
return 0;
}

return DefWindowProc(hwnd, message,wParam,lParam);
}
```

//

The above code has been extensively tested, many times and has never failed. After spending many hours on the subject, I thought I could shine some light on rs232 in a laymens kind of way, I know for many of you, this is probably unworthy of your time or knowledge, however, if I would of have had this type of post 3 months ago, it really would of made the rs232 implementation curve a lot simpler for me! I invite anyone that is a beginner and finds rs232 complicated just as I did, (and still do! Because I am sure there is much more to this) to use it and improve upon it!

I have one question though! What is the advantage of using the Wndserial class as opposed to using functions like CreateFile, WriteFile, ReadFile, and CloseHandle.

Would I be better off, or is it because in Windows it would be too complicated!
Just asking!

I thank Scott and all members of the news groups that have helped me on the subject!

You guys, are a great resource on the subjects of C,C++,VC++.

—
Best regards
Robert

.