Subject: About RS232
Posted by alvintseng on Mon, 26 Jun 2006 16:21:09 GMT
View Forum Message <> Reply to Message

May I use U++ to write the rs232 client program?

Subject: Re: About RS232
Posted by mirek on Mon, 26 Jun 2006 18:05:38 GMT
View Forum Message <> Reply to Message

alvintseng wrote on Mon, 26 June 2006 12:21May I use U++ to write the rs232 client program?

While there is no specific support, it is still regular C++, so everything you can do in C/Win32 API (or C/Posix/Linux API), you can do in U++ too...

Mirek

Subject: Re: About RS232
Posted by qwerty on Tue, 11 Jul 2006 17:44:48 GMT
View Forum Message <> Reply to Message

try libwxctb v0.8. it's great. I am using it in upp extensively.
here: http://www.iftools.com/ctb.en.html
I can send you a package, but it's really simple.

Subject: Re: About RS232
Posted by qwerty on Sun, 10 Sep 2006 23:25:16 GMT
View Forum Message <> Reply to Message

I've glued up together a working version(linux, win32) of 'plugin' of wxctb. if anyone interesed...

Subject: Re: About RS232
Posted by supertorresmo on Tue, 24 Apr 2007 17:34:08 GMT
View Forum Message <> Reply to Message

I'm trying to implement an RS232 GUI program, but I had no success yet.

Does anyone have an working example?

Subject: Re: About RS232

Posted by ebojd on Wed, 25 Apr 2007 18:39:38 GMT
View Forum Message <> Reply to Message

A cople of projects you might look at are Minicom <http://alioth.debian.org/projects/minicom> and Kermit <http://www.columbia.edu/kermit/>. You might run into licensing issues if you use their code, but it will give you something to compare against...

EBo --

Subject: Re: About RS232
Posted by supertorresmo on Thu, 26 Apr 2007 11:40:35 GMT
View Forum Message <> Reply to Message

I've finally got it working.

Here is what I did for anyone who might be interested:

```
MyClass::MyClass()
{
 CtrlLayout(*this, "Window title");
 Zoomable().Sizeable();
 connect <<= THISBACK(OnConnect);
 disconnect <<= THISBACK(OnDisconnect);
 send <<= THISBACK(OnSend);
 conectado = 0;
 SetTimeCallback(-1, callback(this, &MyClass::Timer));
}


GUI_APP_MAIN
{
 MyClass().Run();
}


void MyClass::OnConnect(){

 dev = new wxSerialPort();

 if(dev->Open(wxCOM1) < 0) {
    PromptOK("Cannot open COM Port");
    delete dev;
    return;
    }
```

```
    // set the baudrate
    conectado = 1;
    ((wxSerialPort*)dev)->SetBaudRate(wxBAUD_57600);
}

void MyClass::OnDisconnect(){

 conectado = 0;
 dev->Close();
        delete dev;
}

void MyClass::OnSend(){

 dev->Writev("Hellow",6,0);
}

void MyClass::Timer() {
 char ch;
        int n = 0;

 if (conectado) {
     n = dev->Read(&ch,1);
     if (n>0)
     editbox1.Insert(ch);
 }

}
```

This is a simple program that puts any incoming characters in a edit box and sens a "Hellow" string when the send button is pressed...

Subject: Re: About RS232
Posted by nixnixnix on Fri, 04 Jan 2008 16:37:30 GMT
View Forum Message <> Reply to Message

Is this still the present state of serial port communication under UPP?

Are there any examples?

I have to say that I would rather do imperfect serial port programming and stay within the confines of UPP than start incorporating 3rd party code.

I am trying to read NMEA sentences from a (bluetooth)GPS connected to (virtual) serial port (3). I don't care about data loss as the GPS streams sentences continually. So long as I catch at least

one whole sentence each time I look at the GPS port, I'm good. With this in mind, can anyone tell me why the following does not work?

```
MyClass::MyClass()
{
 CtrlLayout(*this, "Window title");
 Zoomable().Sizeable();

 SetTimeCallback(1000, callback(this, &MyClass::Timer));

}

void MyClass::Timer()
{
 char cBuf[300];

 void* ptr = (void*)0x2f8;

 memcpy(cBuf,ptr,200);

 m_s.SetText(cBuf);
}

GUI_APP_MAIN
{
 MyClass().Run();
}
```

It throws an exception whenever I try to directly access the address 0x2f8

Any ideas or insults welcome

EDIT: ah ok so I've found that direct access of these addresses is outlawed under newer versions of windows - great!

I am finding freely available serial port classes but they are either pre 1995 or written to run under MSVC and use handles etc. If I find one that can be converted I will post it here.

If anyone knows how to get around the windows access exception, I would be very happy to hear it.

Nick

Subject: Re: About RS232
Posted by Mindtraveller on Fri, 04 Jan 2008 22:48:13 GMT
View Forum Message <> Reply to Message

Nick, your code will of course throw an exception.
Mistakes you`ve made:
1) You try to read 0x2f8 address instead of port 2f8, which is usually connected with one of RS232 ports.
2) You try to read from port in Windows, where direct working with hardware ports is prohibited (if you`re not kernel-mode device driver).
3) Virtual COM-ports like bluetooth ones are not physical RS-232 ports. So there is no way to read them with any physical port, like 2f8.

Instead, you should do following:
1) Open port with CreateFile
2) Do reading/writing with ReadFile/WriteFile
3) Close the port with CloseHandle.

Things you must consider:
1) Your application must be multithreaded. One thread works with COM-port, while other thread respond to users`s GUI input.
2) You should make some kind of options or config, where to change COM-port number and it`s settings (speed, timeouts, etc) without recompiling the program

I strongly recommend you reading these topics:
(basic)
MSDN: CreateFile, ReadFile, WriteFile (COM-ports)
MSDN: Threads
U++: Config
(intermediate)
MSDN: Threads synchronization
(advanced)
MSDN: Overlapped i/o

Recently I`ve written U++ class, which makes all the hard work with COM-port. You should only write appropriate callbacks. For now, it works only for Windows, since I`m just starting learning Linux. If it`s needed in current state of development - I`ll upload it here.

Subject: Re: About RS232
Posted by nixnixnix on Sun, 06 Jan 2008 02:26:54 GMT
View Forum Message <> Reply to Message

Thanks! I'll checkout your suggestions.

I actually managed to find a freely available serial port class that works under windows but I suspect it wont work under linux.

I would be very interested in a U++ class that works under both linux and windows but I am happy to wait until you have it finished - that would be fantastic.

I managed to write a small app that reads location from a bluetooth GPS and if anyone wants it I'll happily post it here. However, I think what you propose sounds way better. I hope that it gets incorporated into the next release of U++.

Thanks for replying,

Nick

Subject: Re: About RS232
Posted by forlano on Fri, 10 Oct 2008 19:33:53 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Fri, 04 January 2008 23:48
Recently I`ve written U++ class, which makes all the hard work with COM-port. You should only write appropriate callbacks. For now, it works only for Windows, since I`m just starting learning Linux. If it`s needed in current state of development - I`ll upload it here.

Hello,

please let me ask you if your class is already included in the current 2008.1 or it is still under development.
Even in this state, if available, it could be very useful. I have got a temperature sensor connected to COM 1 and I would like to read it within U++.

Bolshoie spasiba!
Luigi

Subject: Re: About RS232
Posted by Mindtraveller on Fri, 10 Oct 2008 20:49:27 GMT
View Forum Message <> Reply to Message

For now class is working good and only for Windows. It already was refactoried and tested in released application. The main drawback is lack of *nix support - this was the cause of not-publishing it into U++ (also I didn`t discuss it`s adding at all). If you need Windows-only version I`ll prepare it and post in this thread tomorrow.

Subject: Re: About RS232
Posted by forlano on Sat, 11 Oct 2008 09:17:09 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Fri, 10 October 2008 22:49 If you need Windows-only version I`ll prepare it

and post in this thread tomorrow.

Windows is quite enough at the moment.
Thanks,
Luigi

Subject: Re: About RS232
Posted by Mindtraveller on Sat, 11 Oct 2008 20:36:00 GMT
View Forum Message <> Reply to Message

Here are these files: ConveyorThread.h & RS232Thread.h
You should use RS232Thread class.

Kran.h & main.cpp are files from sample project which uses RS232Thread class.

Currently I`m writing manual for RS232Thread. It will be ready within some days. Hope it will answer your questions.

File Attachments
1) ConveyorThread.h, downloaded 991 times
2) RS232Thread.h, downloaded 1505 times
3) main.cpp, downloaded 1127 times
4) Kran.h, downloaded 1033 times

Subject: Re: About RS232
Posted by forlano on Sat, 11 Oct 2008 21:07:49 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Sat, 11 October 2008 22:36Here are these files: ConveyorThread.h & RS232Thread.h
You should use RS232Thread class.

Kran.h & main.cpp are files from sample project which uses RS232Thread class.

Currently I`m writing manual for RS232Thread. It will be ready within some days. Hope it will answer your questions.

Hi Pavel,

Is it possible to have even the Kran.lay file in order to have a working example? The file with extension .iml I suppose is not important.

Thanks a lot,
Luigi

Subject: Re: About RS232
Posted by Mindtraveller on Sun, 12 Oct 2008 07:40:04 GMT
View Forum Message <> Reply to Message

Here is an archive with my project. Hope this helps.

File Attachments
1) 4.ZIP, downloaded 1143 times

Subject: Re: About RS232
Posted by Mindtraveller on Sun, 12 Oct 2008 20:35:20 GMT
View Forum Message <> Reply to Message

Half of manual is written. If you wait for a couple of days, you won`t have to dig into my code with comments in Russian.

Subject: Re: About RS232
Posted by forlano on Sun, 12 Oct 2008 21:03:41 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Sun, 12 October 2008 22:35Half of manual is written. If you wait for a couple of days, you won`t have to dig into my code with comments in Russian.

Soglassian!

I'll wait as many days you need
Your package compiles and run, but I have not understood how to get the data from the device connected to the serial port.
At the moment I've a working console example for linux and it uses the class "termios". It seems that the most important part is done by the line

```
/* get a line of data from the sensor */
p = serial_read(buffer, 128);
```

and then performs some parsing of the string buffer. So I need to know how to open the connection, retrieve the string, close the connection.

Luigi

Subject: Re: About RS232
Posted by Mindtraveller on Mon, 13 Oct 2008 18:08:21 GMT
View Forum Message <> Reply to Message

Here is an archive with my manual. It is almost finished - last advanced topic is to be written. In

it`s current condition Manual will likely answer the most of your questions.
I`d apreciate any comments and suggestions.

Extract this archive into your package (project) directory and you will find documentation in TheIDE help within your package node.

File Attachments

1) srcdoc.tpp.zip, downloaded 1049 times

Subject: Re: About RS232
Posted by forlano on Mon, 13 Oct 2008 21:00:27 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Mon, 13 October 2008 20:08Here is an archive with my manual. It is almost finished - last advanced topic is to be written. In it`s current condition Manual will likely answer the most of your questions.
I`d apreciate any comments and suggestions.

Extract this archive into your package (project) directory and you will find documentation in TheIDE help within your package node.

Hi,

thanks for the doc. Unfortunately the very important tutorial3 seems damaged. I can't read it after the phrase:

1. I just want to send some bytes to remote device.
ERROR: Invalid face number:) void main()]&][s

Perhaps some character is not well parsed by my theide 2008.1 (perhaps have you used another version to prepare the docs?)

Luigi

Subject: Re: About RS232
Posted by Mindtraveller on Mon, 13 Oct 2008 21:25:40 GMT
View Forum Message <> Reply to Message

Of course. Since 2008.1 there where big number of improvements (thanx to Mirek) and QTF as well as TheIDE Help system is upgraded. I recommend to use Projects > SVN synchronize ability to obtain latest sources and recompile TheIDE.
I use latest SVN sources and manual was developed under latest version.

Subject: Re: About RS232
Posted by forlano on Mon, 20 Oct 2008 17:27:45 GMT
View Forum Message <> Reply to Message

Hello,

I'm trying to simplify your application (using only methods from tutorial_1-4) to read the temperature of a sensor connected to a micro controller. Here is a picture of this stuff http://quozl.netrek.org/ts/

I have tried the program tsl-1.2.tar.gz in C given by the author (same link above) that run under windows in consolle mode and works OK.
His short code uses the termios lib available under linux.
I would like to replicate it with your library and add a GUI that should even draw a graph during the reading (other three sensors are coming and this increase the fun).

At the moment I was not able to wake up the micro controller. Please let me ask you if DTR in set ON automatically or I must add "dtr=on" in the comString.

Thanks,
Luigi

Subject: Re: About RS232
Posted by gxl117 on Wed, 11 Nov 2009 07:44:38 GMT
View Forum Message <> Reply to Message

I'm try to compile your example Kran,but compiler show a error:
main.cpp
f:\MyApps\siral\Kran\../COMMON\RS232Thread.h(103) : error C2664:
'ConveyorThread<T>::Request' : cannot conver
 t parameter 1 from 'Upp::Gate' to 'Upp::Callback1<P1>'
        with
        [
            T=RS232RequestElement
        ]
        and
        [
            P1=const RS232RequestElement &
        ]
        No constructor could take the source type, or constructor overload resolution was ambiguous
        F:\MyApps\siral\Kran\main.cpp(164) : see reference to function template instantiation 'void
RS232Thre
 ad::RequestIOSlave<Kran::Triad>(const RS232RequestElement &)' being compiled

How to fix that?

Subject: Re: About RS232
Posted by Mindtraveller on Wed, 11 Nov 2009 20:05:13 GMT
View Forum Message <> Reply to Message

Sorry, it was a long time since Kran was developed. It looks like ConveyorThread is of much newer version than RS232Thread you use. I'm afraid for now I have no time to support it, as completely new version of RS232 library is under heavy development (it is crossplatform and based on 3rd part lib).

Subject: Re: About RS232
Posted by gxl117 on Fri, 13 Nov 2009 03:40:47 GMT
View Forum Message <> Reply to Message

Look it like a VC7.1's RTTI problem.

Subject: Re: About RS232
Posted by nixnixnix on Thu, 14 Jan 2010 23:10:54 GMT
View Forum Message <> Reply to Message

Hi,

I'm wondering if there was ever any progress made on getting a Linux version to work. I have a serial port class for windows which I didn't post as it seems like it got overtaken by other posts.

Seems like there should be plenty of open-source code out there but I'm not finding it.If I find a way before this gets answered then I'll post what I find.

Nick

Subject: Re: About RS232
Posted by nlneilson on Tue, 26 Jan 2010 03:39:25 GMT
View Forum Message <> Reply to Message

I would also like something in U++ to work with serial ports.

The example for sockets is very good and I will be using that.
If something similar for serial ports could be done that would be great.

I have worked with serial ports for GPS data using:

Python with PySerial

Java with rxtx

Windows with:
#using <System.dll>
using namespace System;
using namespace System::IO::Ports;
using namespace System::Threading;
using namespace System::IO;
using namespace std;

I have tried and have the code for wxTerminal but digging through the code and trying to get something to work with U++ is something I have not tried.

Something to work with serial ports for Win and Linux in U++ would be a real plus.

Subject: Re: About RS232
Posted by Didier on Tue, 26 Jan 2010 21:54:43 GMT
View Forum Message <> Reply to Message

I have a basic serial port package for linux only (sorry) but it works well.
I'll post it as soon as possible

Subject: Re: About RS232
Posted by nixnixnix on Tue, 26 Jan 2010 22:21:23 GMT
View Forum Message <> Reply to Message

Hi Didier,

That would be great as then its just a case of preprocessor directives to choose between my windows serial code and the linux serial code.

Cheers,

Nick

Subject: Re: About RS232
Posted by Didier on Thu, 04 Feb 2010 23:19:46 GMT
View Forum Message <> Reply to Message

Hi nixnixnix,

sorry for delay, but I'm missing time.

Here is my very basic serial port package for linux.

There is only open(port, speed)/close and read/read_timed/write.

But with that you can treat 90% of the cases.

File Attachments

1) SerialPort.tgz, downloaded 746 times

Subject: Re: About RS232
Posted by nixnixnix on Mon, 08 Feb 2010 20:45:15 GMT
View Forum Message <> Reply to Message

Thanks Didier,

Will give it a shot. Am just reading NMEA sentences from GPS so it should be more than I need

Cheers,

Nick

Subject: Re: About RS232
Posted by nixnixnix on Sat, 06 Mar 2010 01:31:56 GMT
View Forum Message <> Reply to Message

Hi Didier,

Works fine although one needs to know of course that the serial ports are named

/dev/ttyS1
/dev/ttyS2

etc.

Also, its more useful to return whether the port is open or not. I will post my modification of your serial port class once I've had time to clean them up but if anyone else wants serial port code, post here and I will pull my finger out.

Nick

Subject: Re: About RS232
Posted by nlneilson on Sat, 06 Mar 2010 10:38:25 GMT
View Forum Message <> Reply to Message

It would be good to get this implemented in upp.
Maybe
If win ports = COM7, COM8, ...

If unix ports = /dev/ttyS7, /dev/ttyS8, ...

As posted on 26, January I have used serial ports extensively on Win. Getting something that is cross platform would be great.

Getting rid of the requirement for:
using namespace System::IO::Ports;
and using something that works on Win and Linux.

All the parsing, etc. is no problem.

Subject: Re: About RS232
Posted by Reini on Sat, 06 Mar 2010 17:42:33 GMT
View Forum Message <> Reply to Message

Hello NL,

Since I red a lot of it in the past that U++ Users request RS232 support I just thaught of trying the challenge.

Several years ago I programmed the RS232 with Visual Basic and had the common problem that most VB Users can code 95% of the features in 10% of the time or so

Back then the whole PC was at 100% CPU load, stuck completely and I had data loss packages if I went above 56k

It was a nightmare and in the end I couldn't solve it with VB since it does not allow threading and stuff.

Anyway because I spent back then a lot of time with RS232 and designed an RS232 Monitoring Program for filtering characters my Idea would be to publish an updated version in C++ for this.
So I could try to get all my knowledge from then into an RS232 Lib for Upp could be a challenge.

Do you have already some code ?
If you are interested I could provide also the old VB Version.

Greetings

Subject: Re: About RS232
Posted by nixnixnix on Sat, 06 Mar 2010 23:33:28 GMT
View Forum Message <> Reply to Message

Ok here is what I've got. Its clumsy but it works for both Windows and Linux. I have attempted to modify Didier's code to provide a more unified interface but it could be done a lot better.

Basically it does what I need just now and like I say it works on both platforms. My main addition is to provide a common Open function as well as an IsOpen function.

Nick

File Attachments
1) Serial.cpp, downloaded 1335 times
2) Serial.h, downloaded 1076 times

Subject: Re: About RS232
Posted by nlneilson on Sun, 07 Mar 2010 11:58:44 GMT
View Forum Message <> Reply to Message

Reini: The code I use just follows after the connection.

For the connection on Win it's System::IO::Ports
With Java it's rxtx
With Python it's PySerial

With your experience and what Nick has maybe something can
be put together that Mirek and others can check and possibly have that included in Upp.

I will look closer at Nick's code, I couldn't find termios.h, is that included in C++ ?

Neil

Subject: Re: About RS232
Posted by Didier on Sun, 07 Mar 2010 12:38:37 GMT
View Forum Message <> Reply to Message

Hi Nick and others,

I'm happy to see that that you managed to use my linux serial port package and even added the windows support.

It would be a good idea to continue the effort and make a complete management of the serial port, add support for:
 - handshake
 - parity

- ....

Also, I think it would be a good idea to have SerialConfig struct that would enable easy management of predefined configs.

We could use the port number instead of /dev/tts0 or COM1 to make it fully portable.

I can manage the linux version enhancements, can you manage the windows version ?

What do you think Nick.

Subject: Re: About RS232
Posted by Mindtraveller on Sun, 07 Mar 2010 16:25:58 GMT
View Forum Message <> Reply to Message

Why don't use wxSerial from wxWidgets?

Subject: Re: About RS232
Posted by Didier on Sun, 07 Mar 2010 18:50:43 GMT
View Forum Message <> Reply to Message

Why don't use wxSerial from wxWidgets?

If there are no links to wxWidgets inside the code that can get very simple to do.
But even if there are some links to wxWidgets, we will probably find some corner cases linked with porting issues that we would also have to resolve.

So taking a look will probably get very instructive.

I'll try to find the time to look at it.

Thank's for the tip

Subject: Re: About RS232
Posted by Didier on Sun, 07 Mar 2010 19:56:22 GMT
View Forum Message <> Reply to Message

I've just looked at wxSerial.
OK it's a serial port

But I'm not sure the wxWindows license is compatible with the BSD of UPP ?

Anyway a serial port is simple enough to do it without copying it.

Subject: Re: About RS232
Posted by mr_ped on Wed, 16 Jun 2010 14:55:18 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Sun, 07 March 2010 00:33Ok here is what I've got. Its clumsy but it works for both Windows and Linux. I have attempted to modify Didier's code to provide a more unified interface but it could be done a lot better.

Basically it does what I need just now and like I say it works on both platforms. My main addition is to provide a common Open function as well as an IsOpen function.

Nick

Did this evolve further? The attached source code is a bit unfinished, although it works. (I mean MS Win code path)

I think it has everything I need for now, but I don't like to use unfinished code with dead variables and code, so I wonder if there's something more fresh, or I can clean it a bit and upload back here.

Subject: Re: About RS232
Posted by mr_ped on Wed, 16 Jun 2010 14:57:03 GMT
View Forum Message <> Reply to Message

And I will be very likely creating some mock object for UnitTest++ to be able to test rest of classes, I don't expect anyone of you did such mock already, but in case somebody did, please share.

Subject: Re: About RS232
Posted by Didier on Wed, 16 Jun 2010 20:08:53 GMT
View Forum Message <> Reply to Message

Hi mr_ped,

I didn't have time to make it evolve.
Feel free to make it evolve if you wan't.

But I think it's better to have enums for speed/parity/... instead of a config string like in windows

version: it's easyer to use and debug.

Subject: Re: About RS232
Posted by jeremy_c on Thu, 19 Aug 2010 23:30:00 GMT
View Forum Message <> Reply to Message

Where can one find wxSerial? It seems to be hiding from me :-/

Jeremy

Subject: Re: About RS232
Posted by koldo on Fri, 20 Aug 2010 06:23:03 GMT
View Forum Message <> Reply to Message

jeremy_c wrote on Fri, 20 August 2010 01:30Where can one find wxSerial? It seems to be hiding from me :-/

Jeremy
Hello Jeremy

wxSerial refers to wxWindows serial support. It cannot be used here.

I am not sure about the status of the serial support in U++. Now it seems to be:

- nixnixnix has included in Forum (in this post) a serial class for Linux and Windows
- jerson has done a class for Windows to be posted
- Mindtraveller has done a more complete class but is part of a closed source package

Subject: Re: About RS232
Posted by jerson on Fri, 20 Aug 2010 08:00:08 GMT
View Forum Message <> Reply to Message

Yes, I have a working single threaded version of the serial port class and I have tested it with WinXP. Posted here http://www.ultimatepp.org/forum/index.php?t=msg&th=5401& amp;start=0&

Subject: Re: About RS232
Posted by jeremy_c on Fri, 20 Aug 2010 11:58:23 GMT
View Forum Message <> Reply to Message

Last night I grabbed a copy of libctb (https://iftools.com/opensource/ctb.en.php). I modified the

source a bit and made a U++ package out of it. So now I just say Add Package, Ctb and my app has serial support.

I have not tested it yet under linux but will be doing so shortly. The only problem with what I did was I had to change several headers for the package. For example, the ctb package uses:

#include "ctb-0.15/abcdef.h"

I changed these to read:

#include <Ctb/abcdef.h>

So, with each update to libctb, that would have to be done. Is there a better way? I wanted to let U++ handle the building of the package.

I started with Serial.h/.cpp that I found here and it worked fine for some devices I was controlling but not for others. I needed the ability to set 2 stop bits, to raise the DTR and RTS states, etc...

I am not sure what is the best idea here. To keep my Ctb package or to browse it's source, learn how the Data Bits, Parity, Stop Bits and Line States are set/queried and port them to the Serial.h/Serial.cpp. I kind of liked the Serial.h/.cpp's function ReadDataWaiting(). w/Ctb you can read X number of bytes with a timeout but say you want 30 bytes, Readv(buf, 30); may result in a partial read, i.e. maybe 15 were available before the timeout occurs. You'll get the 15. With Serial.h/.cpp I could enter a loop and say:

```
tries = 0;
while (ReadDataWaiting() < 30) {
  tries++;
  if (tries > 3) do_fail_code();
  Sleep(50);
}
```

Jeremy

Subject: Re: About RS232
Posted by koldo on Fri, 20 Aug 2010 12:30:19 GMT
View Forum Message <> Reply to Message

Quote:So, with each update to libctb, that would have to be done. Is there a better way? I wanted to let U++ handle the building of the package.In "Package Organizer" you can set with "Internal Includes" additional folders to look for includes.

Subject: Re: About RS232
Posted by nlneilson on Sun, 21 Nov 2010 01:17:25 GMT
View Forum Message <> Reply to Message

What is the current status of what is included in Upp for:
RS232, serial port, COM port, or whatever the designation or how it is wrapped.

I am currently using 2827 but could update.

I am interested in the #include and the few lines of code required to hook up to serial port.

Python uses PySerial
Java uses rxtx (note that Oracle/Sun has dropped support for comm)
Win uses System::IO::Ports;
I have good results with all of these.

Since I am not that experienced with Upp a simple complete package with an example (like those in the "examples" folder included with Upp), if there is one, would be appreciated. Otherwise I spend several hours on something very basic before getting something to compile/link.

There is info in the thread like:
jerson's CommPak.zip
Pavel's 4.zip
Didier' SerialPort.tgz
Nick's Serial.cpp, Serial.h

As mentioned an example that works with what is included in the Upp install is what is preferred, or with least addition of header files or ??.

luzr wrote on Mon, 26 June 2006 20:05While there is no specific support, it is still regular C++, so everything you can do in C/Win32 API (or C/Posix/Linux API), you can do in U++ too...
Mirek

I try to use:
using namespace System::IO::Ports;
from my existing code that runs in MSVC that gives errors in Upp.

Subject: Re: About RS232
Posted by nlneilson on Sun, 21 Nov 2010 12:49:29 GMT
View Forum Message <> Reply to Message

jerson's CommPak.zip gives the info I needed.

#include "Serial.h"
and Serial.h needs to be in the package.

```
 if (!CommPort.Open(6,57600))
```

```
 Exclamation("Cannot open COM6");
else
 SetTimeCallback(-100,THISBACK(CheckRxTmr), idRxTimer);
```

That is what I needed!

Getting the data from GPS receivers will be done in a separate thread.

The porting of my code from Java (it's later and more extensive than the MSVC code) should be straight forward.

Thanks jerson for the example.

Neil

Subject: Re: About RS232
Posted by jerson on Sun, 21 Nov 2010 15:21:47 GMT
View Forum Message <> Reply to Message

Thanks Neil. I'm glad my work has been useful to you.

Subject: Re: About RS232
Posted by nlneilson on Sun, 21 Nov 2010 21:06:46 GMT
View Forum Message <> Reply to Message

That was very useful!

Having an example that can be downloaded, unzipped, placed in MyApps, opened in TheIde, select execute, and have it compile/link/run is great.

For someone that is fairly new to Upp getting a few lines of code and trying to get it to run is very frustrating and takes time.

I did notice that the pull down menu in the Setup Comms box for baud rate does not include these.
115200
57600
38400
These cannot be seen/picked even though they are listed in Serial.h. I added them in main.cpp line 57:
int Port,Baud[]={1200,2400,4800,9600,19200,38400,57600,115200};

Whoever ported the code from C++ to be compatible in Upp in Serial.h and Serial.cpp did a good job.

Neil

Subject: Re: About RS232
Posted by nlneilson on Tue, 23 Nov 2010 02:50:48 GMT
View Forum Message <> Reply to Message

Looking at Serial.h and Serial.cpp they were not familiar to what I have previously used.

To help someone else this is what was done.
In my Python and Java code and the reason for doing it this way.
1. Loop through the ports from COM4 to COM20 (COM3 is the default for Win printer).
2. Loop through the ports to check the baud rate.
3. Open each port that returns the NMEA GPS GGA sentence that run concurrently, three is the most tried so far without threads, no problems.

Here is a snip of code to work with the Serial.h and Serial.cpp

```
    Thread work;

   void OpenAction() {
  if (!CommPort.Open(6,57600)) Exclamation("Cannot open COM6");
   Sleep(1000);
    work.Run(THISBACK(Work));
   }
   void Work(){
  char ch;
  char buf1[1];
  String St1;
  while (CommPort.ReadDataWaiting() ) {
   CommPort.ReadData(buf1, 1);
   ch = buf1[0];
   if(ch!='\n' || ch!='\r')St1 << ch;
   if(ch=='\n'){
    Data<<=St1;
                              ...
          Sleep(980);
```

The COM port and baud is hard coded here, the app will find and insert these parameters.

Note the Sleep(1000); before the work thread is run and Sleep(980);, otherwise the ReadDataWaiting() is empty.
This reads a char at a time but the parameter has to be a char buf, hence the necessity for char buf1[1];

This is very basic but shows how you can get data from a serial port you can work with. There is also a lot of try/catch, if(...), continue, break, parse(...), send(...), swear(), etc..

Subject: Re: About RS232
Posted by jibe on Fri, 15 Mar 2013 13:40:18 GMT
View Forum Message <> Reply to Message

Hi,

Is there something new about all that ?

There is already some good job done. If I don't forget anything, we have those works :

- Mindtraveller (Pavel) 4.ZIP Oct 12, 2008
- Didier SerialPort.tgz Feb 05, 2010
- nixnixnix (Nick) Serial.h and Serial.cpp Mar 07, 2010
- jerson CommPak.zip Aug 06, 2010

What about starting from all that to build a final package and add it to upp, at least in bazaar ?

If now PC have often no more COM ports, RS232 is still used a lot in industry. More : several USB devices have a driver allowing to use them as a COM device. Such a package would be very useful !

I could try to do it... Any help or suggestion will be appreciated, especially what is interresting or missing in every package above and how they could (or not...) be mixed together.

Subject: Re: About RS232
Posted by jibe on Fri, 15 Mar 2013 14:55:05 GMT
View Forum Message <> Reply to Message

A first study of the code confirms what I was guessing reading the thread : nixnixnix and jerson codes are just enhancements of Didier's code. So, I think that the best is just to use the most recent one, that is jerson's.

Mindtraveller's code seems to be interresting, but very different and more complicated. So, I think that it's better to start from jerson's, isn't it ?

I worry why Windows part is so different from Linux part ? When I compile jerson's CommPack, I get errors with missing members !

For me, the whole structure of the class should be the same for Windows and Linux, no ? Any reason/explanation about those different members ?

Subject: Re: About RS232
Posted by jerson on Fri, 15 Mar 2013 15:00:27 GMT

Hello Jibe

I am not sure what kind of errors you get. I use only the windows platform and the code works well on it. I never have used/tried a compile on Linux, so, possibly the errors are there; I don't know.

I'd like to collaborate if there is any movement on this and may also add the modbus RTU feature if prodded. Right now, I am juggling too many things and time sharing is hard.

Regards
Jerson

Subject: Re: About RS232
Posted by Zbych on Fri, 15 Mar 2013 15:04:15 GMT

If you want I can help you with linux part of serial port implementation.

First you should tell me what interface you want.

My proposition:

1. Vector<String> ListPorts();
returns list of available ports. On windows you can use QueryDosDeviceW to get list of devices. On linux udev does similar thing.

2. bool Open(const char * port_name, enum speed, enum flow_control, enum parity)

3. void Close()

4. int Write(const void * data, int len, int timeout_ms = 5000);
5. int Write(String data, int timeout_ms = 5000);

6. String ReadUntil(int amount, int stop_byte = -1, int timeout_ms = 5000);

7. void FlushInput();
to clean input queue

8. void FlushOutput();
to clean output queue

I would like to have timeouts in every Write and Read functions, because they often depend on the command you send to device.

Things like stop_byte are useful for communication with modems.

Maybe we should also have Callbacks (when new data arrive, when usb serial device is (un)plugged)?

Subject: Re: About RS232
Posted by jibe on Fri, 15 Mar 2013 21:45:10 GMT
View Forum Message <> Reply to Message

Thanks for the replies

jerson wrote on Fri, 15 March 2013 16:00I am not sure what kind of errors you get. I use only the windows platform and the code works well on it.
... And I'm almost sure that it can work with Linux as well : As I said, it's just that there is not the same methods ! I had just a quick try and did not study well all that, probably it's just different names...

I posted immediately before going further just to know if there is some good reason. If not, I'll just make so that the class is exactly the same for Linux or Windows, as far as possible.

jerson wrote on Fri, 15 March 2013 16:00I'd like to collaborate if there is any movement on this and may also add the modbus RTU feature if prodded. Right now, I am juggling too many things and time sharing is hard.
You do as you can according to your free time. If it's possible for you, I'll let you try and eventually debug the Windows part, as I'm working only under Linux.

Zbych wrote on Fri, 15 March 2013 16:04If you want I can help you with linux part of serial port implementation.

First you should tell me what interface you want.
Thanks

But for now, I'm just beginning to discover what has been already done. I think that it will be enough for what I have to do (just read the output of an A/D converter and display the signal like an oscilloscope).

I'll see later in more details your proposition, when I'll know better jerson's package. I think it's already good, but could be better with some enhancements or additions like a more complete Open function with flow control and parity as you propose : if we make a package to add to upp, it must be some more complete.

Subject: Re: About RS232
Posted by nlneilson on Sat, 16 Mar 2013 21:40:53 GMT

The best serial port code of all I have tried is from M$.

It is managed code and Sender Ghost was good to help on getting it to work in a U++ app.
http://www.ultimatepp.org/forum/index.php?t=msg&goto=383 03&&srch=serial#msg_38303

My use is specifically for GPS data.
If I recall it was jerson's and nixnixnix's code I tinkered with the most of all that are listed in this thread.

Subject: Re: About RS232
Posted by Mindtraveller on Sun, 17 Mar 2013 07:56:20 GMT

Let me share some thoughts on serial port communication as industrial automation developer.
Critics is welcome.
First of all, the main idea behind all this functionality is sending and receiving some actual data to/from PC. The most common scenario is to send request and receive answer (PC = master, device = slave). We also must understand highly asynchronous nature of all these tasks, because at no circiumstances serial port i/o must be done in common thread (especially when it handles GUI). The next idea is crossplatform code, because our great U++ framework is crossplatfrom too. The last requirement is the ease of tuning: most of time we need to tune some parameters of serial port "on the field", and we must me able to do it without recompiling our code.
These features put together really professional-grade serial library.
Let's recollect them, dividing by levels of abstraction:

1. Port handling and I/O (lowest level).
Crossplatform code handling system-dependent serial port code.
Done with tiny library called crossplatserial with my little modifications. It has Apache license (which is the same as BSD, AFAIK).
This level creates SerialPort objects from external file or code. So if you need to tune i.e. the baudrate, you just change external file and restart your app.
I attach crossplatserial to this message and you may use it, if it is all that you want.

2. Datagramms <-> values (protocol stuff).
Actually the main code should work with data. Not with actual bytes you send through serial port. I achieved it with my BNF library which is responsible for using protocols.
That is how it works:
You have external file defining a protocol. BNF takes plain bytes and returns Vector<Value> and back. For example: ***modbus (crc_start byte byte word_be word_be crc_end crc_modbus)
        | (crc_start byte byte byte word_be crc_end crc_modbus);
***icp_din_clear_request "$" HEX:2 "C" "\0D";
***icp_din_clear_answer (("!" arg:1 HEX:2) | ("?" arg:0 HEX:2)) "\0D";
Ok. On this level we've left protocols behind and made possible for SerialPort to work with actual Vector<Value> (not with plain bytes). I repeat that protocol is defined in external file, so if you need to tune it, you just change protocol description in file and restart your app. Extremely useful

in some cases "in the field".

3. Asynchrony (highest level).
IMO the ideal approach for asynchrony of serial i/o is using queued threads. This means each thread has it's queue and you put there this thread's callback with custom parameters. That is how threads interact, no syncronization objects needed.
This is done with bazaar/MtAlt package I wrote some time ago.
In spite of this frightening description, the actual use of this is simple. Let's look at real-world example: class SensorNB3000: public SerialPortNotify<CallbackQueue> {/*...*/}; //creates callback queue

```
void SensorNB3000::RequestMeasure()
{
 Vector<Value> args;
 args << ((int) addr.GetData()) << 6 << 50 << 1;
 sensorPort->SendReceiveSleep_(PREPARE_TIMEOUT_NB3000);
 sensorPort->SendReceive_
 (
  "modbus",
  args,
  "modbus",
  RECEIVE_TIMEOUT_NB3000,
  NULL,
  Ptr<SiloSensorNB3000>(this),
  &SensorNB3000::OnMeasure
 );
}

void SensorNB3000::OnMeasure(bool result, Vector<Value> args, void *custom)
{
    //...
}
```

This code means: SensorNB3000 class adds some timeout to sensorPort thread queue and then requests send/receive operation with modbus protocol. Send modbus args are {addr,6,50,1} and the handling routing is OnMeasure. The SensorNB3000::OnMeasure() routine is syncroneously called in the SensorNB3000 thread (no sync is needed in your code!).

Here are the most of the concepts I propose. If you find them too comlicated, you may just use attached library. If you consider using all these levels of abstraction in U++, we may discuss it further.

File Attachments

1) crossplatserial.zip, downloaded 666 times

Subject: Re: About RS232

Posted by jibe on Wed, 20 Mar 2013 14:22:30 GMT
View Forum Message <> Reply to Message

Hi,

Thanks for all those explanations, Pavel !

I globally agree with you. I'm studying all that, but as I'm doing several things in the same time, so I cannot be very fast...

I already studied more jerson's code, and as I said there is incompatibilities between Linux and Windows versions. There is surely solutions, but it seems that the approach is different (not same functions/methods...), so it will not be as easy as I thought first to make a package working as well under W$ as under Linux.

I'll try your code, Mindtraveller. I already had a look on the crossplatserial code, and finally it seems clear, well done and really cross-platforms ! I'll make as soon as possible a small program to test it or more exactly to have a better understanding of your approch that seems very interresting.

Subject: Re: About RS232
Posted by deep on Thu, 09 May 2013 06:58:48 GMT
View Forum Message <> Reply to Message

Hi Pavel,

Mindtraveller wrote on Sun, 17 March 2013 13:26
2. Datagramms <-> values (protocol stuff).
Actually the main code should work with data. Not with actual bytes you send through serial port. I achieved it with my BNF library which is responsible for using protocols.
That is how it works:
You have external file defining a protocol. BNF takes plain bytes and returns Vector<Value> and back. For example: ***modbus (crc_start byte byte word_be word_be crc_end crc_modbus)
        | (crc_start byte byte byte word_be crc_end crc_modbus);
***icp_din_clear_request "$" HEX:2 "C" "\0D";
***icp_din_clear_answer (("!" arg:1 HEX:2) | ("?" arg:0 HEX:2)) "\0D";

....

Here are the most of the concepts I propose. If you find them too comlicated, you may just use attached library. If you consider using all these levels of abstraction in U++, we may discuss it further.

I would like to use all these ideas in U++. I want to begin with modbus-rtu or modbus-ascii.

Can you give further info about BNF?

Thanks and regards.

Deepak

Subject: Re: About RS232
Posted by Mindtraveller on Fri, 10 May 2013 22:38:29 GMT
View Forum Message <> Reply to Message

Heelo, Deepak!

BNF is very well known around:
http://en.wikipedia.org/wiki/Backus%E2%80%93Naur_Form

As of Modbus RTU, you may see my version of it in example above.
