Borst Automation Embedded Solutions

Engineering Consultancy for new
advanced Information Technologies

Voice: +49 4721 6985 100
Fax: +49 4721 6985 102
Email:info@borst-automation.de
Home: borst-automation.de

Borst Automation, Kapitaen-Alexander-Strasse 39, DE-27472 Cuxhaven, GERMANY

Quotation: Real Time Serial Port SCM

Product name:	Real Time Serial Port SCM
Contained item:	Test client for Windows (executable)
Delivery material:	Zip-File with source code and documentation. Delivery is by electronic means only (e-mail or transfer method of your choice).
Delivery date:	10 working days after reception of the purchase order
Payment:	Account transfer or cheque. VAT: DE 170387245 IBAN: DE48 5105 0015 0962 0286 24 SWIFT-BIC: NASSDE55XXX
Price:	499,- Euro
Updates:	For one year after the delivery of the source code updates to new revisions are free of charge.
Warranty	Borst Automation (Walter Borst) insures that the software will perform substantially in accordance with the documentation (see Technical Details). However, this source code is supplied with NO WARRANTIES because it is subject to be adapted/changed by the user. Walter Borst expressly disclaims any warranty for the software package. This software package and related documents are provided "AS IS"; without warranty of any kind, expressed or implied. This includes implied warranties of fitness for a particular purpose. All risk arising out of use of this package remains with the user. By using this software package, the user agrees that no event shall Borst Automation or Walter Borst make responsible or liable for damages whatsoever. This includes, without limitation, damages for loss of business profits, loss due to business interruption, loss of business information, or any other pecuniary loss, arising out of the use of or the inability to use this software package.
Validity	The quotation is valid for three months since the download of this file.
Ordering address:	Borst Automation Kapitaen-Alexander-Strasse 39 DE-27472 Cuxhaven GERMANY Fon: +49 (0)4721 6985 100 Fax: +49 (0)4721 6985 102 Email: info@borst-automation.de Home: borst-automation.de

Notes:

1) The General Terms and Conditions of Borst Automation[1], December 18, 2013, shall apply.
2) The Software License Agreement of Borst Automation[2], December 20, 2013, shall apply.
3) Prices do not include tax.
4) Please provide an e-mail address for the delivery

Cuxhaven, March 9, 2023

(Walter Borst)

[1] GSTC-BorstAutomation-18.12.2013.pdf

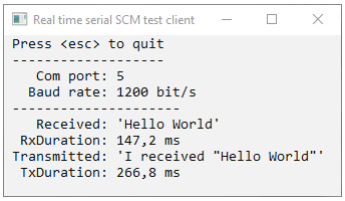
[2] SLA-BorstAutomation-20.12.2013.pdf

Voice: +49 4721 6985 100
Fax: +49 4721 6985 102
Email:info@borst-automation.de
Home: borst-automation.de
Borst Automation, Kapitaen-Alexander-Strasse 39, DE-27472 Cuxhaven, GERMANY

Appendix: Technical Details

Test-Environment

- Simple console application as test client to insure that C++ is the only required programming language and easy to understand.



Technical Details:

- Windows 7, 8, 10, 11, Server 2016, Server 2022
- Com port 1..255
- DTR and RTS support for protocols like HART.
- Baudrates 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
- Timing accuracy ~+ 5 ms (5/1000 seconds)
- Used WINAPI functions: CreateFile(), SetCommMask(), SetupComm(), PurgeComm(), SetCommTimeouts(), GetCommState(), SetCommState(), ReadFile(), WriteFile(), ClearCommError(), EscapeCommFunction(), timeGetTime(), timeBeginPeriod(), timeEndPeriod(), CreateThread(), SetThreadPriority(), RaiseException(), memcpy(), memset(), memcmp(), Sleep().
- Programming language: C++
- IDE: Visual Studio 2019
- Implemented: Serial communication, thread handling, system time handling

Background Information

As is well known, half-duplex interfaces (e.g. two-wire transmissions) require a protocol that regulates data traffic. HART is one such protocol, for example.

On Windows, however, this can lead to a problem. As long as the com ports were physically built into the computers as UARTs, software could be informed by the event system when the last character was sent. However, after USB com ports were used more and more often, it turned out that the event TX_END is set as soon as the last character has been transmitted to the next driver layer. However, this does not mean that the last character has already been output at the physical interface. The temporal reference to when the transmission ended was lost.

When implementing a serial interface, I decided at this point to use a different method. This procedure is actually very simple.

1. It is calculated exactly how long the transmission has to take until the last byte has been sent.

2. The state machine for sending waits until this time has elapsed.

In the source code it looks like this.

```
switch (m_Status)
{
  case CStatus::START_TX:
    mpcl_Timer->Start(CTimer::GetTxDuration(m_len, mpcl_channel->get_baudrate()));
    m_status = CStatus::WAIT_TX_END;
    return CProtocol::CToDo::WAIT_TX_END;
    break;
  case CStatus::WAIT_TX_END:
    if (mpcl_timer->is_expired() == TRUE )
    {
      m_status = CStatus::IDLE;
      return CProtocol::CToDo::END_TRANSMIT;
    }
    break;
}
```

This state machine is called approximately every 2 ms.

Furthermore, there are the following details to consider.

- Nonoverlapped I/O should be used. Only then can it be ensured when writing that sending the bytes has actually started.
- The event system should not be used because the reference to the time axis is relatively imprecise and unreliable.
- In the thread that monitors the com port, it should be ensured with the functions timeBeginPeriod(1) and timeEndPeriod(1) that execution takes place in milliseconds steps.

Technical Data Sheet

The technical data sheet is also part of this quotation.