Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

# Hart Slave Stack C++ 7.6

### **Technical Data Sheet**



C++ Source Code for an Embedded Firmware Module with the following Properties

- No external dynamic memory management. The amount of reserved RAM remains constant.
- The number of objects is determined at compile time and startup.
- No operating system is required to integrate the software. Timer and serial interrupts are enough.
- Simple asynchronous user interface to encapsulate the time-critical part.

The implementation is based on the Hart Documents in: HART Communication Protocol Specification, HCF\_SPEC-13, FCG TS20013 Revision 7.09, Release Date: 06 January 2023

Details for the Hart Protocol are provided via the following link: <u>https://www.fieldcommgroup.org/technologies/hart</u>.

Hart Slave Stack C++ 7.6	1
Introduction	2
Implemented Commands	2
Architecture	3
Hart Slave C++ Code	4
User Interface	4
Public Functions	4
Data Interface	6
Coding Considerations	6
Hardware Abstraction	7
Embedded System Requirements	7
Coding Conventions	7
Visual Studio 2022	8
Test Environment	8
Prerequisites	8
Development Directory Structure	8
Getting Started	8
Test Interface	9
Appendix	11
Internet Links	11
Download Location	
Legal Issues	
Conformity	
Copyright	
No Warranty	

Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

Technical Data Sheet

# Introduction

## **Implemented Commands**

#	Description	Remarks
Unive	rsal Commands	
0	Read Unique Identifier	See also Command Summary Specification
1	Read Primary Variable	
2	Read Loop Current And Percent Of Range	
3	Read Dynamic Variables And Loop Current	
6	Write Polling Address	
7	Read Loop Configuration	
8	Read Dynamic Variable Classifications	Currently only the standard device
9	Read Device Variables with Status	247, 248, 249, 0, 1, 2, 3.
11	Read Unique Identifier Associated With Tag	
12	Read Message	
13	Read Tag, Descriptor, Date	
14	Read Primary Variable Transducer Information	
15	Read Device Information	
16	Read Final Assembly Number	
17	Write Message	
18	Write Tag, Descriptor, Date	
19	Write Final Assembly Number	
20	Read Long Tag	
21	Read Unique Identifier Associated With Long Tag	
22	Write Long Tag	
38	Reset Configuration Change Flag	
48	Read Additional Device Status	The slave module saves a copy of the last additional device status sent for each master and compares it with the bitstream provided by the user application.
Comn	non Practice Commands	
33	Read Device Variables	
34	Write Primary Variable Damping Value	
35	Write Primary Variable Range Values	
49	Write Primary Variable Transducer Serial Number	
54	Read Device Variable Information	Currently only the standard device variables are supported: 244, 245, 246, 247, 248, 249, 0, 1, 2, 3.
108	Write Burst Mode Command Number	Commands 1, 2, 3 and 9 are currently
109	Burst Mode Control	accepted. Burst messages are not (yet) supported.
512	Read Country Code	
513	Write Country Code	

I consider the now implemented set of commands to be the minimum that must be available in a Hart slave. However, I also recommend making all important functions of a slave accessible via universal and common practice commands and not using user-specific commands. In this case it is not necessary to provide a device description. This saves development time and development costs.

Technical Data Sheet

## Architecture



The package Portable Hart Slave includes all sources needed to represent the slave part of the Hart protocol. The package is written in standard C++ and does not use any direct connection to a system environment. Data link layer, application layer (command interpreter) and network management of the Hart protocol are implemented. The connection to the outside occurs via three interfaces: The User Interface, a Time Trigger and the HAL to the Uart interface.

I used the C# environment to debug the Hart slave code during development. In fact, it is not(!) a simulation that is used here. The firmware is simply embedded in a Windows environment that allows the code to run in real time(!). In this way, all functions of the implementation can be analyzed in detail. The analysis of the temporal processes takes place in the range of milliseconds.

The C# software (White Box Test) was developed to create a transparent user interface for visualizing the data and communication processes. Visual Studio 2022 and .NET 6.0 were used to keep the programming effort within limits.

The command interpreter is triggered from the C# environment, but this happens within a 'real' thread and not within a worker thread from .NET:

```
CommandInterpreter = new Thread(ExecuteCommandInterpreter);
CommandInterpreter.Priority = ThreadPriority.Highest;
CommandInterpreter.Start();
and in endless loop of the thread:
    result = (EN_Bool)HartSlaveDLL.BAHASL_WasCommandReceived();
    if (result == EN_Bool.TRUE8)
    {
        // Simulate typical application
        Thread.Sleep(20);
        command = HartSlaveDLL.BAHASL_ExecuteCommandInterpreter();
```

Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

# Hart Slave C++ Code

## **User Interface**

#### **Public Functions**

The following functions are realized in the module HartS\_UartIface.cpp in the class CUartSlave. In the DLL interface for the test client the function names are preceeded by BAHASL\_.

Declaration	Description	
Operation		
<pre>EN_Bool OpenChannel( TY_Word port_number_, EN_CommType type_);</pre>	The function allocates the selected com port if possible and starts its own working thread for accessing the Hart services. The port_number_ is limited to the range of 1 254. The selected communication type (type_) should be UART in this version of the paket. The function returns TRUE8 if successful. In the present implementation only a single channel is possible. Thus no channel handle is required.	
<pre>void CloseChannel();</pre>	It is required to call this function at least when the application is terminating.	
Data Interface		
<pre>void GetConstDataHart(     TY_ConstDataHart* const_data_);</pre>	Copies constant data from the Hart slave area to the test application area.	
<pre>void SetConstDataHart(     TY_ConstDataHart* const_data_);</pre>	Copies constant data from the application area to the Hart slave area.	
<pre>void GetDynDataHart(     TY_DynDataHart* dyn_data_);</pre>	Copies dynamic data from the Hart slave area to the test application area.	
<pre>void SetDynDataHart(     TY_DynDataHart* dyn_data_);</pre>	Copies dynamic data from the application area to the Hart slave area.	
<pre>void GetStatDataHart(     TY_StatDataHart* stat_data_);</pre>	Copies static data from the Hart slave area to the test application area.	
<pre>void SetStatDataHart (     TY_StatDataHart* stat_data_);</pre>	Copies static data from the application area to the Hart slave area.	
Command Interpreter		
<pre>EN_Bool WasCommandReceived();</pre>	The function returns FB_Bool::TRUE8 if the Hart protocol has recently (a few milliseconds ago) received a command.	
<pre>TY_Word ExecuteCommandInterpreter();</pre>	This function calls the command interpreter in the slave to process any new data. If the command was recognized and executed, the function returns the number of the command. If this was not the case, the value 0xffff is returned.	
Encoding		
<pre>void PutInt8(     TY_Byte data_,     TY_Byte offset_,     TY_Byte* data_ref_);</pre>	Insert an integer 8 into the byte array buffer pointed to by data_ref_ starting at the position offset	
<pre>void PutInt16(     TY_Word data_,     TY_Byte offset_,     TY_Byte* data_ref_,     EN_Endian endian_);</pre>	Insert an integer 16 into the byte array buffer pointed to by data_ref_ starting at the position offset Start with the most significant byte if endian is MSB_FIRST(0), which is the Hart standard.	

Bortembedded Solutions

Walter Borst Kapitaen-Alexander-Strasse 39 27472 Cuxhaven, GERMANY

Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

**Technical Data Sheet** 

usid DutTat24/	In such an interest 24 into the base summer buffers as into the base and state and states at the	
Vola Putint24(	Insert an integer 24 into the byte array buffer pointed to by data_ref_starting at the	
TY_Dword data_,	position offset Start with the most significant byte if endian is MSB_FIRS1(0),	
IY_Byte offset_,	which is the Hart standard.	
IY_Byter data_ref_,		
EN_Endian endian_);		
<pre>void PutInt32(</pre>	Insert an integer 32 into the byte array buffer pointed to by data_ref_ starting at the	
TY DWord data	position offset . Start with the most significant byte if endian is MSB FIRST(0),	
TY Byte offset	which is the Hart standard.	
TY Byte* data ref		
EN Endian endian )		
Endian charan_/;		
void PutInt64(	Insert an integer 64 into the byte array buffer pointed to by data_ref_ starting at the	
TY_DWord data_,	position offset Start with the most significant byte if endian is MSB_FIRST(0),	
TY_Byte offset_,	which is the Hart standard.	
TY_Byte* data_ref_,		
<pre>EN_Endian endian_);</pre>		
void PutFloat(	Insert a single precision IEEE 754 float value into the byte array buffer pointed to	
TV Float data	by data ref_starting at the position offset. Start with the most significant byte if	
TV Byte offset	endian is MSB_FIRST(0), which is the Hart standard	
TV Byte* data nof	chulan is wisb_1 ites 1(0), which is the flatt standard.	
If_byte uata_rel_,		
EN_ENUIAN enuian_);		
void PutDFloat(	Insert a double precision IEEE 754 float value into the byte array buffer pointed to	
TY_DFloat data_,	by dataRef starting at the position offset. Start with the most significant byte if	
TY_Byte offset_,	endian is MSB_FIRST(0), which is the Hart standard.	
TY Byte* data_ref_,		
EN Endian endian );		
woid ButBackedASCII(	Insert a string (as a string ref.) of the length of as a string lan in peaked ASCII	
Volu PulpackeuAscii(	Insert a string (asc_string_rel_) of the length of asc_string_len_ in packed ASCII	
TY_Byte asc_string_ret_,	format into the byte array buffer pointed to by data_ref_ starting at the position	
IY_Byte asc_string_len_,	offset It is recommented that asc_string_len_ is an ordinary multiple of 4.	
TY_Byte offset_,		
TY_Byte* data_ret_);		
<pre>void PutOctets(</pre>	Copy a number of stream_len_ bytes into the byte array buffer pointed to by	
TY Byte* stream ref ,	data_ref_ starting at the position offset	
TY Byte stream len.		
TY Byte offset		
TY Byte* data ref ):		
Vola Putstring(	Copy a string from string_ref_ to data_ref The actual number of characters	
IY_Byte* string_ret_,	stored cannot be greater than string_max_len If the string contains a null, the last	
IY_Byte string_max_len_,	character saved is a null character if this does not exceed the string_max_len_	
TY_Byte offset_,	limit.	
TY_Byte* data_ref_);		
Decoding		
IY_Byte PickInt8(	Return the value of the byte in the byte array buffer pointed to by data_ref_ from	
TY_Byte offset_,	the position offset	
TY_Byte* data_ref_);		
TY Word PickInt16(	Return the value of the integer 16 from the byte array buffer pointed to by	
TY Byte offset	data ref from the position offset. Assume that the most significant byte is the	
TY Byte* data ref	first if endian is MSB_FIRST(0), which is the Hart standard.	
EN Endian endian ):		
TV Putter a SS in the second state of the seco	Return the value of the integer 24 from the byte array buffer pointed to by dtaRef	
IY_BYTE OTTSET_,	at the position offset. Assume that the most significant byte is the first if endian is	
IY_Byte* data_ret_,	MSB_FIRST(0), which is the Hart standard.	
<pre>EN_Endian endian_);</pre>		
TY DWord PickInt32(	Return the value of the integer 32 from the byte array buffer pointed to by	
TY Byte offset	data ref from the position offset. Assume that the most significant byte is the	
TY Byte* data ref	first if endian is MSB_FIRST(0) which is the Hart standard	
EN Endian endian ):		
IY_UINT64 PICKINt64(	Return the value of the integer 64 from the byte array buffer pointed to by	
IY_Byte offset_,	data_ret_ trom the position offset Assume that the most significant byte is the	
TY_Byte* data_ref_,	first if endian is MSB_FIRST(0), which is the Hart standard.	
<pre>EN_Endian endian_);</pre>		
TY Float PickFloat(	Return the value of the single precision IEEE754 number from the byte array	
TY Byte offset	buffer pointed to by data ref from the position offset. Assume that the most	
TY Byte* data ref	significant byte is the first if endian is MSB_FIRST(0) which is the Hart standard	
EN Endian endian V		

Ro	e ek	Automation	W
		Embedded Solutions	2

**Technical Data Sheet** 

TY_DFloat PickDFloat( TY_Byte offset_, TV_Byte*_data_pof	Return the value of the double precision IEEE754 number from the byte array buffer pointed to by data_ref_ from the position offset Assume that the most significant but is the first if and an is MSR_EIRST(0), which is the Hart standard
EN_Endian endian_);	significant byte is the first if childran is hisb_1 its (0), which is the first standard.
<pre>void PickPackedASCII(     TY_Byte* string_ref_,     TY_Byte string_len_,     TY_Byte offset_,     TY_Byte* data_ref_);</pre>	Generate a string and copy it to the buffer pointed to by sb. The final string should have the length string_len. The packedASCII source is a set of bytes in the byte array buffer pointed to by data_ref_, starting at index offset Note: The string length has to by a multiple of 4 while the number of packedASCII bytes is a multiple of 3.
<pre>void PickOctets( TY_Byte* stream_ref_, TY_Byte stream_len_, TY_Byte offset_, TY_Byte* data_ref_);</pre>	Copy a number (numOctets) of bytes from the byte array buffer pointed to by dataSource to the user buffer pointed to by dataDestination.
<pre>void PickString( TY_Byte* string_ref_, TY_Byte string_max_len_, TY_Byte offset_, TY_Byte* data_ref_);</pre>	The function reads a string from a buffer (data_ref_) starting at index offset_ and stores the characters in string_ref The string buffer is read from until a null character appears or string_max_len_ is reached. If possible, the null character is also saved.
Internal	
<pre>void FastCyclicHandler(TY_Word time_ms_);</pre>	Although this function is not accessible to the test client, it is required for the operation of the Hart protocol. The function must be called by a separate task approximately every millisecond to enable timing in the communication. The time_ms parameter indicates how many milliseconds have passed since the last call. Usually this should be a value of 1 in most cases.

#### **Data Interface**

The data interface provides three different types of data that can be written or read by the user. A structure is provided for each data type, which can be found in the file WbHartS\_Structures.h.

**Constant data** does not change. In most systems it is stored in flash memory and cannot be written.

**Dynamic data** is data that can always change. This includes measured values and status information.

**Static data** is used to configure a device. It is usually changed by external access. Whenever static data is changed, the configuration change flag must be set in Hart and the configuration change counter in Hart must be incremented.

## **Coding Considerations**

Microcontrollers which are used today for HART devices are at least 16 Bit microcontrollers. Otherwise the complexity of the measurement and number of parameters could not be managed.

Low amount of memory.
 The amount of memory is always critical because software kind of behaves like an ideal gas. It uses to fill the given space. Nevertheless, the coding of the Hart Slave was done as carefully as possible regarding the amount of flash memory and RAM.
 The user needs source code.
 The Hart Protocol requires a strict timing specially for burst mode support and the primary and secondary master time slots. To provide the optimum transparency to the user to allow all kinds of debugging and to give the opportunity to optimize code in critical sections, the Hart Slave Firmware is not realized as a

library but delivered as source code.

Technical Data Sheet

Walter Borst Kapitaen-Alexander-Strasse 39 27472 Cuxhaven, GERMANY

### **Hardware Abstraction**

OSAL is including the HAL. A Hardware Abstraction Layer is needed to design the interface of a software component independent from the hardware platform. In this very small interface of the Hart master a distinction of HAL and OSAL was not made. Therefore only an Operating System Abstraction Layer is defined which is covering all the needs of an appropriate HAL.

## **Embedded System Requirements**

It is difficult to estimate the system requirements for targets based on different micro controllers and different development environments. The following is therefore giving a very rough scenario for the target system estimated resources.

Item	Requirement/Size	Comment	
RAM	32k	Depends very much on the addressing structure of the	
ROM (Flash)	100k	contonet and the used compiler and mixer.	
Timing	1-2 ms Timer interrupt	2 ms is the minimum requirement, 1 ms would be much better.	
	50 ms cyclic call from task level	This is needed to run the command interpreter.	
I/O	UART and Hart MODEM Rx and Tx functions	Carrier detection would be helpful but is not required.	
System	<pre>Simple math +-*/ memcpy() memset() memcmp() 1 ms timing</pre>	Only a few standard library functions are required. There is no special need for multi tasking, messaging or semaphores.	
	resolution		

**Table 1: Embedded System Requirements** 

## **Coding Conventions**

Regarding this issue, I have only defined some formats that makes the scope of a label clearer. It's just to make the code easier to read. This simple type of coding convention can be used in both C++ and C#.

Pascal case			
local_variable	function_param_	m_member_var	<pre>mo_member_object</pre>
Variable with local scope	A function parameter has a tailing underscore	Basic type private member variable	Complex object member
s_member_var	<pre>so_member_object</pre>		
Basic type static private member variable	Complex static object member		
Camel case			
PublicVariable	PublicObject	AnyMethod	
Variable with public or internal scope	Object with public or internal scope	No difference between public and private	

Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

# Visual Studio 2022

## **Test Environment**

## Solution Explorer J J J J J J

**Technical Data Sheet** 

- Search Solution Explorer (Ctrl+ü)
- Solution 'CppHartSlave-7.6' (1 of 1 project)
- Solution Items
- ♦ Intervention HartSlaveDLL

There are only one project in this solution. The C++ Hart Slave is encapsulated in the HartSalveDLL project. The solution is directly in the path on which you copied the package to.

### **Prerequisites**

Microsoft Visual Studio Community 2022 (64-bit) Version 17.9.6 © 2022 Microsoft Corporation. All rights reserved. Microsoft .NET Framework Version 4.8.09032 © 2022 Microsoft Corporation. All rights reserved. The solution must be opened with VS 2022. However, the community version is sufficient. There are no further requirements.

### **Development Directory Structure**

vs	N	ame	
📒 01-Docu	01-Shell		
D2-Device		02-OSAL	
📒 01-Generic		03-Build	
02-Specific	3	HartSlaveDLL.vcxproj	
== 01-WinDLL		HartSlaveDLL.vcxproj.filters	
Contemporaries 101-Hart		] HartSlaveDLL.vcxproj.user	
📒 01-Slave			
11-Nrf52832			
🚞 01-Docu		BaHartSlave-7.6.dll	
🚞 02-Device		🖓 BaHartSlave-7.6.exp	
🚞 01-Generic		BaHartSlave-7.6.lib	
02-Specific	1	BaHartSlave-7.6.pdb	
📒 03-Test		Microsoft.AspNetCore.SystemV	
📒 01-Windows		W ReadMe.txt	
📒 01-Docu		System.Runtime.Caching.dii	
📒 02-Apps		TestCppSlave.dll	
📒 01-Hart		TestCppSlave.dll.config	
📒 03-DebugBench		🍄 TestCppSlave.exe	
2			

The project for the Hart Slave in C++ can be found in the directory: .\02-Device\02-Specific\01-WinDLL\01-Hart\ 01-Slave. However, most of the <u>C++ sources</u> used are located in the directory <u>.\02-Device\01-Generic\</u> and its subdirectories.

The test software is only be found as executable in the path <u>03-DebugBech</u>. The executable file <u>TestCppSlave.exe</u> and the simulation DLL <u>BaHartSlave-7.6.dll</u> are both located here. When you start debugging the executable ist started and loading the dll which is respresenting the slave device.

## **Getting Started**

- 1. Unzip the file hart-slave-source-code-7.6.1.zip into a directory of your choice.
- Open the solution CppHartSlave-7.6.sln with Visual Studio 2022. It has to be 2022. Other versions are not supported yet. Unless you have 2022 not installed on your computer. You can download it from microsoft: <u>https://visualstudio.microsoft.com/de/downloads/</u>.
- 3. The community version is sufficient enough and free of charge.
- 4. Perform a 'Build All'.
- 5. Start debugging and investigate the source code

Bortembedded Solutions

Walter Borst Kapitaen-Alexander-Strasse 39 27472 Cuxhaven, GERMANY Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

**Technical Data Sheet** 

## **Test Interface**

😫 lint Hart C++ Sinve, V.7.8.1			- 0 ×
rtaface Identifier Transducer	Device Device Variables	Update Save Data Clear Additional Status Options	Doplay Record Off Abou
Interface Settings	Hart Framing	Hart Communication View	
Con Data - Contra	Request Preambles	A X Departure C E	unie Normani 🗖 Arthurs
COMPARE COMS		a residues	Anter anter a subset
Poll Address: A, 90	* Response Preambles:	5 V Decodesi Deta C T	ming 💦 🚺 Status Detxils
Baudkate 1200 -	🗌 Burst Mode 🔳 Het	Enabled D Write Protected Function D More Status Avail His	art Slave 7.6.1
	02 (Cad 0) 0)	1821	
127-SECEP-17 17 17 17 17 10-04	00 (Cast 0124) 00 29 03 04 05(Cast 1) 01	G(GOLLODGO: 264/ManO/Dev2(4)/6 ( MinFFrep: S/HacKkaS Manu17: 040000/Labb Cold Start, Hore Status (70)	Re/Hart7/Tx1/Sed/Had4/FLO Fe:0/CfgChCnt:0/ExtDevStat: LetTD:00E1/Frifile:0(B6)
100+ 10-1ACKP-ITE TE ITE ITE TE-04	00 29 03 04 05(Cmd 1) 7)	0100010000197 1:7.8 meter:76/	
153) 1342-14TER (17 17 17 17 17 17 18)	01 T5 02 04 00(Cmd 3) 0	Hore Status (FA)	
	85 FF 03 04 05(Cmd 3124)	0(00010000)Curr:16 mA	27 J: 1250 mbar 4: 20 *C1261
TTO-TALKAILL LL AL AL AL ALION	00 25 03 04 051Cmd 31 71	127: 244,248,0,1,4,	4,61201
30-12CR010 07 07 07 07 00-00100	63 89 03 04 09 (Ced 3) 69	01000100001Extended Device Be DV 144 / Clean   DV 145 / Clean   DV 146 / Clean   DV 0 / Clean   DV 1 / Clean   DV 1 / Clean   DV 6 / Cl	Attas: 00005000 1 / 75 % / 4 / 15 mA / 4 / 12 mA / 5 / 1250 metas / 6 / 1250 metas / 6 / 324 not used / 6 / 32 met301
1041		Hole Stelus	
COM5   Minitaring active   Setub	recard off to daip monitoring.		M : 0000006

When the executable file is started, the container DLL for the slave is automatically loaded.

The work surface is divided into two halves.

Settings are made in the tab area, while the lower area is reserved for a monitor that shows the communication process.

While the following tabs mostly deal with the slave data, the inputs in the interface have a fairly direct effect on the running software. For example, it is possible to activate burst mode without having to use the Hart command 109.

Screenshot 1	L: The	Tab '	Interface'
--------------	--------	-------	------------

#### Data Exchange

	Update Slave Data		
ariables	Additional Status	Optio	
Message:	MESSAGE		
Message:	MOIN MOIN MOI	N	

The following tabs deal with the transmitter data. If this data is edited, this is indicated by a yellow color. The menu button also turns yellow and must be clicked for the change to take effect in the slave.

If a parameter is changed by a master connected to the slave, this change appears in the display and the parameter in question is colored red

The Part C++ Steen, V 7,61 Interface   Identifier   Transducer   Device   Device Variables   A	- C × Update Save Data Clear Display Record Off About dditional Status   Options	The tab 'Identifier' mainly deals with data related to command 0.
Device Type (Table 1): 0x00F9 Signating (Table 10): 0 Device Revision Level 1 Plags (Table 11): 0 Software Revision Level 2 Device Unique ID (Nex): 0x0304 Hardware Revision Level 3 Last Dev Variable Code: 0	Config Change Counter         65335         Profile (Table 37):         000           Extended Stetus (Table 17):         000         Details           05         Manufacturer (Table):         0x0000         Table 3           Untrituder (Table):         0x0000         Table 3         Table 3           Untrituder (Table):         0x0000         Table 3         Table 3	
	<pre>(52) 00110000 (1544/Hand/Dweid40/15 FAs/Hart7/Th1/fm2/Hid4/TL0) HanBarg:1-MasHusTVe:0/CfgChChe:0/TatDevEcat HanD2:Ne0500/LabDistD:00El/Torfile:0(B4) Cold Start, Hore Status</pre>	

Screenshot 2: The Tab 'Identifier'

Bortembedded Solutions

Walter Borst Kapitaen-Alexander-Strasse 39 27472 Cuxhaven, GERMANY Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

#### Technical Data Sheet



The tab 'Transducer' mainly deals with data related to the commands 14 and 15.

#### Screenshot 3: The Tab 'Transducer'

📽 limt Hart C++ S	inve, 9 7.6.1	- 2000			- 0	. *
Interface Identify	er Transducer Devic	Device Variables	Update Slove Data Additional Status	Clear Display Options	Record Off	About
Long Tag: 32 C	Thanactiona (see Lation-1	Message	E 32 CAPITAL LETTER	CHARACTERS		
Short Tag: OCH	IR TAG	Final Assembly Numbe	e 777777			_
Description: 16.0	H DESCRIPTOR	Country Code	e de 🖂 Si Units o	nly :		
Day: 1 Mont	th: 7 Veet 2024					
1884 199 (1863) (Carl) 8674 1993 (1865) (Carl) 1994 275 (1865) (Carl) 1994 1894 1894 1894 1894 1894 1994 1994	31361         0 <td>CARTERI LETTER CO Hore Blatton In TAU/LE CM TERRER MINARCERLYNORDER ( My Tag 32 Character</td> <th>120) 19708/Date(1.7.30) 197073(12) 197373(12) 18 Teo Latin-L</th> <th>(4) 60 ( (87)</th> <th></th> <td></td>	CARTERI LETTER CO Hore Blatton In TAU/LE CM TERRER MINARCERLYNORDER ( My Tag 32 Character	120) 19708/Date(1.7.30) 197073(12) 197373(12) 18 Teo Latin-L	(4) 60 ( (87)		
COM 5   Monitor	ing active   Switch record a	ff ta stop manituring.	1		@ 0000008	-4

The tab 'Device' mainly deals with data related to the commands 12, 13, 15, 16 and 20.

#### Screenshot 4: The Tab 'Device'

	i	(I	Update Slave Data	Clear Display	Record Off Abov
iterface identifier	Transducer Devici	Bevice Variables	Vegational Statist 0	pnons	
Percent (244)	Commt ( 245)	PV1 (0, 246)	PV2 (1, 347)	PV3 (2, 248)	PV4 (3, 249)
Class: 0	Clens: 84	Class: 69	Class: 55	Class: 73	Class: 04
Value: 75.0	Value 16.0	Value: 7.5	Value: 1250.0	Value: 1.0	Welkie: 20.0
*	mA	Unit: meter ~	Unit: entrer ~	Unit kg/l	Unit: "C 🔍
Good 🔘 Bad	O Good () Bad	O Good () Bad	O Good () Bed	O Good O Bad	O Good 🔘 Batl
S13+ B0+LACEP(Cad B	( 7) (DV) (68) 0100010000)Erro DV DV	244,245,246,247,2 ended Device State 244 / Elass: 0 / 245 / Class: 94 /	48,248,61801 4: 00105000 75 4 16 mA	/ Pratus:11000 / Status:11000	000
	DV DV DV DV DV T1.m	244 / Class: 45 / 247 / Class: 45 / 248 / Class: 73 / 242 / Class: 64 / 6 / Class: 64 / e Starp: 4047504 1	7.5 meter 1250 mbar 1 kg/1 28 °C 968 not uas /52 ms(04)	/ Status:11000 / Status:11000 / Status:11000 / Status:11000 ef / Status:00110	1000 1000 1000

The 'Device Variables' tab provides access to the data needed to implement device variables. Currently, only device variable codes in the range 244-249 and 0..3 are accepted. These are the only required device variables.

Of course, further variables for the user are possible at any time.

Screenshot 5: The Tab 'Device Variables'



Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

This is about command 48. As already mentioned elsewhere, the slave manages the responses to the two masters separately and stores which response it has sent to a master. If something changes in the

with the copies.

additional status, the software knows which master it affects because it can compare it

**Technical Data Sheet** 

				-		- 0	1.2
		Update	Slove Data	ClearC	Risplay	Record Off	Abor
iterface Identifier Tran	istucer 0	Device De	vice Variables	Additio	nal Status	Options	
Device Specific Status 0-5:	00011000	10000001	00111100	01000100	10101010	11111111	1
Standardized Status 8-3	00010000	000001	00100	10000			
Analog Channel Saturatast	0100	1	Analog Cl	hannal Fisas	E 0010	1	
Specific Status 14-24:	Set to 0,.		Table 29	11 Jak	12 Telle 2	1	
		1921			_		
18-ABCKDICHA 01241 0	100030000	288 (March/	Develation of	an (Name T)	Twi / But / the	14/11/00000	010/7
28-88CKF(Cmd. 0124) 0	100030000	1114/Hend/ HanRRapp:	Devr245/5 D 5/HardfordDV	As/Herst/ a:d/CfgCb	Tal/Sw2/Hw Cat:0/Eat0	04/FL00000	010/1
28-SACKFICed 0(24) 0	100010000	104/Men0/ MinRArep: MenuID:0x	Dev 245/5 D 5/HardfordV 0000/Leb04	As/Hert/ s:0/CfgCb stD:00E1	Tal/Swl/NW Cas:0/Eat0 /Spofile:0	04/ FL00000 wrStas: 000 (96)	00000
28-68CKFICed 0(24) 0	Hure Day	1234/Men3/ ManRArap : Menu1D: 0x	Dev245/5 D 5/HactionEV 00ED/Leb04	As/MertT/ s:d/C5gCb stID:D0E1	Tal/Sw2/fb Cat:0/Eat0 /Spofile:0	04/ FL00000 1961	0000/1
29-58CKP1Ced.0(24) 0 113- 550-1ATEP(Ced48) 0)	Hurs Day	1254/Man3/ MinRArap: ManuID:0x ManuID:0x	Dev245/5 D S/HarthurDV 00ED/Leb04	As/HertT/ s:0/CSgCb stID:00E1	Tal/Sw3/Hw Cnt:O/Eato /Brofile:0	196) 196)	00000
28-88CKB1Cmd 0(24) 0 813- 850-LATEPICmd481 81 189- 122-18-8810-m4881271 0	Hore Dear	104/Han0/ HanRkrap: Hanv1D:0x	Dev245/5 D 5/HardhurDV 0000/Leb04	As/Herti/ s:d/CfgCb stID:DOEI	Tal/Sw2/Hw Cat:O/EatO /Spofile:0	4/ FL00000 - Stat: 000 (96)	1000/1
22-48-0000000000000000000000000000000000	Hore Deal	1284/Man3/ MinSkrap: Many1D:0x 1031 (101:00011 141:10101	Dew245/5 D 5/HardbarDV 0000/Leb04 000 111:10	As/Nert7/ s:d/C5gC5 stID:D0E1 000001 11 111111 14	Tel/8+2/8+ Det:0/Eet0 /Spofile:0 1:00111100	14/ FL00000 wrStaft:000 (96) (51:01000 (71:00000	100
200-6500001Ced 01241 0 213- 250-LETEPICed451 01 200- 27-140001Ced421271 0	100010000	1084/Man0/ MinRArap: Manu1D:0x Manu1D:0x Min Min Min Min Min Min Min Min Min Min	Dew245/5 P 5/HardborDV 000D/Leb04 000 E11:10 010 E51:11 000 E51:00	As/Hert7/ s:d/CfgCb stID:D0E1 000001 11 111111 14 000001 10	Tel/8x2/8x Det:0/Ect0 /Spofile:0 1:00111100 1:00050000 3:00000100	14/ FL00000 wrStaft:000 (96) 151:01000 11:00000 11:00000	100
122- 35-68CM216ded 0(34) 0 313- 550-12002[Ged481 0) 157- 37-12000[Ged48127] 0	100010000	1254/Men3/ MinR&rep: MenalD:0x 1051 101:00011 101:00010 123:00010	Dev245/5 P 5/HasthurDV 0000/Leb04 000 E1F:L0 010 E51:00 000 E51:00	As/Mert1/ s:d/CdgCb stD:b0E1 000001 11 111111 10 000001 10 000001 14	Tal/8x2/HW Cat:0/Eat2 /Brofile:0 1:001L1100 1:00000100 1:00000100	04/FL00000 mr5tmt:000 (56) (36) (31:01000 (11:00000 (11:00000 (11:00000	000/2 00000 100 100 000
122- 38-88CK0(Cmd. 0124) 0 138- 138-18TEP(Cmd481 0) 129- 27-18CK0(Cmd48127) 0	Hore Pres	104/Men3/ MinRasp: MenuID:0x 103 103 141:10101 101:00010 103:00010 103:00010 103:00010	Dev:245/5 D 5/HasthamDV 0000/Lab0d 010 E11:E0 010 E51:11 000 E51:00 000 E51:00	As/HertT/ s:d/CfgCb stID:D0E1 111111 14 000001 10 000001 10 000000 13	Tal/301/10 Cnt 0/Est0 /Stofile:0 1:0000000 1:0000000 1:0000000 1:0000000	04/ FL00000 mr Shut : 000 [96] [51 : 01000 [11 : 00000 [51 : 00000 [51 : 00000	100 100 100 100 000 000
122- 32-38/10/04 0124 0 113- 350-12722(Ced48,1 0) 103- 27-12/022(Ced48,1 0) 27-12/022(Ced48,1 2) 0	100010000	1334/Ham3// finiEsep: Ham12:0x 103:00011 141:10101 101:00010 123:00010 123:00010 101:00000 101:00000	Dev235/5 D 5/HasthonDV 0000/Lab03 010 [11:10 010 [51:10 010 [51:00 010 [51:00 010 [51:00 010 [11:00	As/HertT/ s:d/CfgCb stID:D0E1 111111 14 000001 10 000001 10 000000 13	Tal/5x2/10 Cnt: 0/Ext2 /Postile:0 1:00111100 1:0000000 1:0000000 1:0000000 1:0000000	44/ TL00000 wr5tait (000 (36) (51:00000 (51:00000 (51:00000 (51:00000 (51:00000	100 000/J 00000 100 000 000 000
235-68/2009 (Ged. 0124) 0 335- 335- 335- 359-12722 (Ged481 1) 159- 37-12/2009 (Ged481 27) 0	Hure Day	134/Mard/ finiEsep: Baru12:0x 103:00011 141:10101 101:00010 103:00010 103:00010 103:00000 101:00000 101:00000 101:00000	Dev245/5 D 5/HasthonDV 0000/Lab03 010 E1:10 010 E5:10 010 E5:00 010 E5:00 010 E1:00 010 E1:00 010 E1:00	As/Mart/ s:d/CfgCb stD:D0E1 111111 19 000001 10 000000 10 000000 13	Tal/5x2/10 Cnt :0/Ext2 /Postile:0 1:00111100 1:0000000 1:0000000 1:0000000 1:0000000	44) FL00000 (56) (56) (51:01000 (51:00000 (51:00000 (51:00000 (51:00000 (51:00000	1000/I 000000 100 000 000 000 000
23*88/CTEICed 01241 0 23*88/CTEICed481 0 23* 23*LACTEICed481 0 23* 27*LACTEICed481271 0	100320000 Hure Pre-	1354/Hard// filmEnep: Haru1D:00 (05) (01:000L1 (41:1010) (31:000L0 (41:000L0 (41:000L0 (41:00000 (41:00000	Dev235/5 0 5/HardhamDV 0000/Le003 010 151:10 000 151:00 000 151:00 000 171:00 000 171:00 000 161:1	As/Hert/ s:d/CfgCb stD:D0E1 111111 14 000001 10 000001 14 000000 13 000000 13	Tal/5x2/10 Dat/0/Ext2 /Podile:0 1:00111100 1:0000000 1:0000000 1:0000000 1:0000000 1:0000000	G4/ FL00000 W Shan (000 196) (51:00000 C11:00000 C11:00000 C11:00000 C11:00000 C11:00000	1000/J 00000 100 000 000 000 000
23* 88/200 (Ged. 0124) 0 113* 800 (LST22)(Ged481 0) 105* 27*(LACE0)(Ged481(27) 0 37%4	100320000 Hure Pre-	1384/Herd/ HinBRing: HeralD:00 103 103 (00011 141,10101 101 (00010 103 (00010 103 (00010 103 (00010 103 (00010 101 (00000 101 (00000	Dev235/5 0 5/HacfbacDV 0000/Le003 010 [51:10 010 [51:00 000 [51:00 000 [11:00 000 [11:00 000 [11:00	As/Mach1/ s:d/CdgCb stD:D0E1 111111 14 000001 10 000010 14 000000 13 000000 13	Tal/8+2/19 Cat 0/Eart2 /#Dodile 10 1:00111100 1:0000000 1:0000000 1:00000000 1:00000000 1:00000000	54/ FL00000 w 51 w 1000 1961 C51:00000 C11:00000 C51:00000 C51:00000 C51:00000 C51:00000	1000/I 000000 100 000 000 000 000

Screenshot 6: The Tab 'Additional Status'

# Appendix

## **Internet Links**

**Specification Documents** HART Specifications FieldComm Group MODEMs RS 232 Modem Microflex USB Modem Endress + Hauser Viator USB Modem Pepperl+Fuchs **Ethernet-APL** Advanced Physical Layer Ethernet - To the Field FieldComm Group Ethernet APL Organisation HART-IP Developer Kit FieldComm Group

## **Download Location**

The software package described in this document can be downloaded via the following link:

https://www.borst-automation.com/downloads/hart-slave-source-code-7.6.1.zip

Borgenbedded Solutions Wa

Technical Data Sheet

Walter Borst Kapitaen-Alexander-Strasse 39 27472 Cuxhaven, GERMANY Fon:+49 (0) 4721 6985100 E-Mail: <u>walter.borst@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

## Legal Issues

### Conformity

This software package was developed to the best of my knowledge and my belief. The basis is the specifications of the Hart Communication Foundation in version 7.9.

However, it cannot be guaranteed that the software included in this package meets the HCF specifications in all required respects.

It is only possible to prove the conformity of this software after the user has integrated the software into his device and commissions HCF or a certified company to carry out this test. Under no circumstances am I, Walter Borst, responsible for carrying out such tests. Nor am I responsible for correcting any deficiencies resulting from such a test.

#### Copyright

Copyright, Walter Borst, 2006-2024

Kapitaen-Alexander-Strasse 39, 27472 Cuxhaven, GERMANY Fon: +49 (0)4721 6985100, Fax: +49 (0)4721 6985102 E-Mail: <u>info@borst-automation.de</u> Home: <u>https://www.borst-automation.de/</u>

#### **No Warranty**

Walter Borst expressly disclaims any warranty for the software package. This software package and related documents are provided "As Is".

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.