ABB	**Technical Information** (Technische Information)	IIM-TI01/2011-P
Template No.: **TI01.00.01**		Page: **1/45**

Product line: P (Produktbereich)	Year: 2011 (Jahr)	Archive no.: 01 (Archivierungsnummer)

Title: (Titel)	HART Specification Changes from Rev. 5.9 to Rev. 7.2
Product: (Produkt)	Common Component - HART 7 Subsystem (BETA version)
Part no. with index (Sachnummer mit Index)	n.a.
Related documents: (zugehörige Dokumnete)	HART Communication Protocol Specification Documents 7.2

Remark
(Bemerkung)
This technical information refers to the HART 7 subsystem beta version only and is not fully completed. In a first step requirements from the EDP300 development are realized.

Minden, 2011-01-07	Editor: Bernhard Tohermes, Stefan Tabelander (Bearbeiter) (Signature, Unterschrift)
Signatures(Unterschrift))	Dr. Wolfgang Scholz (R&D management)

Distribution:

- ☒ R&D Archive
- ☐ SAP (location:)
- ☒ Dr. W. Scholz (R&D management)
- ☐ W. Lasarzik (R&D design)
- ☒ A. Stelter (R&D Software)
- ☐ H. -J. Beuger (R&D Electronics)
- ☐ J. Dingerdissen (Supply management)
- ☐
- ☐ M. Klüppel (Quality management)
- ☐ B. Kruse (Quality management)
- ☐ K. Schwarzer (Quality management)
- ☐ U. Rumke (IE)
- ☐ K.-H. Christ (IE)
- ☐ F. Krietemeyer (IE)
- ☐
- ☐

Contents

1 Introduction

In this document the changes of principle HART functionality and HART command content changes between HART revision 5.9 and 7.2 are described. Important parts of the specification documents are copied into the corresponding chapters to reduce preparation work.

2 Technical Information

First, new HART commands and changes to the existing HART commands with appropriate implementation information are described. Second, all objects that are new to commands are listed to summarize additional information. Third, additional HART functionality and HART command functionality is documented with links to parts of the specifications.

2.1 Universal Commands

Commands 38 and 48 until now belonging to the Common Practice Commands are defined new as Universal Commands.

2.1.1 Command 0: Read Unique Identifier

2.1.1.1 Requirements

Page 12 in Spec 127

Additional Response Info:

Byte	Description	Implementation
12	Minimum number of preambles	Constant: must be defined for specific device
13	Maximum Number of Device Variables	Constant: must be defined for specific device
14-15	Configuration Change Counter	Variable: must be defined; increment function needed
16	Extended Field Device Status	Variable: Code of Common Table 17
17-18	Manufacturer Identification Code	Constant: Code of Common Table 8
19-20	Private Label Distributor Code	Constant: Code of Common Table 8
21	Device Profile	Constant: Code of Common Table 57

Changes in Response Code:

Code 1 – 127 Undefined

In Rev. 5 more response codes were possible.

2.1.1.2 Implementation

Using the Entry Tool the expansion of command 0 with the new needed objects could be done (see also corresponding chapters in 2.3 for new objects and in 2.6.1 for HART revision specific workflow of command 0)

2.1.2 Command 6: Write Polling Address

2.1.2.1 Requirements

Page 16 in Spec 127

Additional Request Info:

Byte	Description	Implementation
1	Loop Current Mode	Variable: Code of Common Table 16

Additional Response Info:

Byte	Description	Implementation
1	Loop Current Mode	Variable

Additional Response Code:

Code	Description	Implementation
12	Invalid Mode Selection	The requested mode is not valid
32	Busy	

Backward Compatibility Requirements
Request of single Byte -> Master with HART Revision 5:
If polling address == 0 -> enable current signaling
If polling address <> 0 -> disable current signaling

In case of single byte request answer to request without RC 5 "Too Few Data Bytes Received". Response both polling address and Loop Current Mode byte

2.1.2.2 Implementation

For command 6 in Entry Tool both objects "pollAddr" and "loopCurrentMode" are configured with type REQUEST and with type RESPONSE to get a "HART_SPECIAL_WRITE" command.

HART revision specific functionality for command 6 is documented in 2.6.1. Additional information regarding Loop Current Mode is given in 2.3.9

2.1.3 Command 7: Read Loop Configuration

2.1.3.1 Requirements

In HART 6 new command: must be implemented

Page 21 in Spec 127

2.1.3.2 Implementation

Command definition can be done with Entry Tool.

Additional information regarding Loop Current Mode is given in 2.3.9

2.1.4 Command 8: Read Dynamic Variable Classifications

2.1.4.1 Requirements

In HART 6 new command: must be implemented

Page 22 in Spec 127

The up to 4 Dynamic Variables of a device are dedicated to device specific Device Variables. So a fix corresponding Classification Code could have been set for each Dynamic Variable.

Response Data Bytes

Byte	Description	Implementation
0	Primary Variable Classification	Constant: must be defined for specific device
1	Secondary Variable Classification	Constant: must be defined for specific device
2	Tertiary Variable Classification	Constant: must be defined for specific device
3	Quaternary Variable Classification	Constant: must be defined for specific device

If Dynamic variables are not supporting classification then return “0” in response

2.1.4.2 Implementation

Command definition can be done with Entry Tool.

For detailed information about the Dynamic Variables see chapter 2.3.12

2.1.5 Command 9: Read Device Variables with Status

2.1.5.1 Requirements

In HART 6 new command: must be implemented

Page 23 in Spec 127

Request Data Bytes

Byte	Description	Implementation
0	Slot 0: Device Variable Code	
. . .		
7	Slot 7: Device Variable Code	

Up to 8 different device variables could be read. The length of response contains corresponding bytes

Response Data Bytes

Byte	Description	Implementation
0	Extended Device Status	Variable: must be defined; status actualization function needed Code of Common Table 17
1	Slot 0: Device Variable Code	Constant: Code of Common Table 34
2	Slot 0: Device Variable Classification	Constant: Code of Common Table 21
3	Slot 0: Units Code	Constant: Code of Common Table 2
4 - 7	Slot 0: Device Variable Value	Variable
8	Slot 0: Device Variable Status	Variable: must be defined; status actualization function needed
. .	Slot i: … .	< Slot 1 to Slot 7 specific variable set >

. . .	. . .	
9 - 12 . . . 65 - 68	Time	*Variable: must be defined;* *Function to increment 24h time value needed*

- Specific response data must be returned if variable is not supported (see Page 23 Spec 127)
- Command is capable of Burst Mode Operation.

Response Code: see Page 26 Spec 127

When a Device Variable requested (or dynamic variable) is not supported in the Field Device, then the following must be reported for the listed information:
Value must be set to "0x7F, 0xA0, 0x00, 0x00"
Status must be set to 0x30, (i.e., Status = "Bad" and Limit = "Constant")
Units Code must be set to "250" Not Used
Device Variable Classification set to "0", Not Yet Classified.

2.1.5.2 Implementation

Precondition for realization of command 9 is the implementation of indexed commands (see Page 32 in Spec 99).

In hart_objects.c function “HARTSpecialReadWriteObjects” the ability to handle 8 slots as indices is implemented.

Command definition can be done with Entry Tool. For command 9 in Entry Tool objects are configured for request data each with type REQUEST and for response data each with type RESPONSE to get a “HART_SPECIAL_WRITE” command.

With the mechanism for indexed commands in Entry Tool for every possible device variable addressed by its variable code as slot code and configured with type SLOT the specific response objects are defined. The data of these objects is returned at the places of the objects in response data (response data object with type RESPONSE).

HART revision specific functionality for command 9 is documented in 2.6.1. For detailed information about the Device Variables see chapter 2.3.11 and 2.3.13

2.1.6 Command 11: Read Unique Identifier Associated With Tag

2.1.6.1 Requirements

Page 27 in Spec 127

Response Data Bytes of Command 0 is returned. So the changes of Command 0 must also be implemented here.

2.1.6.2 Implementation

See chapter 2.1.1.2

2.1.7 Command 15: Read Device Information

Page 31 in Spec 127

Changed and additional Response Data Bytes:

Byte	Description	Implementation
16	Reserved	Must be set to “250”, Not Used
17	PV Analog Channel Flags	Constant: Code of Common Table 26

Private Label Distributor Code is no longer added In Byte 16.

2.1.7.1 Implementation

Command definitions can be added with Entry Tool.

HART revision specific functionality for command 15 is documented in 2.6.1. For information about new object see 2.3.17

2.1.8 Command 20: Read Long Tag

2.1.8.1 Requirements

In HART 6 new command: must be implemented

Page 37 in Spec 127

Response Data Bytes

Byte	Description	Implementation
0 - 31	Long Tag	Variable: must be defined

2.1.8.2 Implementation

Command definitions can be added with Entry Tool.

For information about new object see 2.3.17

2.1.9 Command 21: Read Unique Identifier Associated With Long Tag

2.1.9.1 Requirements

In HART 6 new command: must be implemented

Page 38 in Spec 127

Request Data Bytes

Byte	Description	Implementation
0 - 31	Long Tag	Variable: must be defined

A function is needed which compares the Long Tag of the Request with Long Tag of the device addressed via long frame or broadcast address. No response is made unless the Long Tag matches that of the device.

Response Bytes of Command 0 are returned

Code change in layer2_receive_HW.c:
Check of Long Tag length and compare of Long Tag is to implement.

2.1.9.2 Implementation

Command definitions can be added with Entry Tool (see 2.1.1.2).

For information about new object see 2.3.17

2.1.10 Command 22: Write Long Tag

2.1.10.1 Requirements

In HART 6 new command: must be implemented

Page 37 in Spec 127

Request Data Bytes

Byte	Description	Implementation
0 - 31	Long Tag	Variable: must be defined

A function is needed which copies the Long Tag out of the request into variable of device

Response Data Bytes

Byte	Description	Implementation
0 - 31	Long Tag	Variable: must be defined

2.1.10.2 Implementation

Command definitions can be added with Entry Tool.

For information about new object see 2.3.17

2.1.11 Command 38: Reset Configuration Changed Flag

2.1.11.1 Requirements

Earlier Common Practice Command in HART 7 allocated to Universal Commands.

Page 40 in Spec 127

Changed Request Data Bytes

Byte	Description	Implementation
0 - 1	Configuration Change Counter	

A function is needed which compares the master specific counter with the corresponding in the device. Only if the specific counter is equal the master specific Flag is reset. If not equal RC #9 is returned.

Changed Response Data Bytes

Byte	Description	Implementation
0 - 1	Configuration Change Counter	

Response Code see Subsection 6.23 on Page 1 Spec 127 Addendum

Backwards Compatibility Requirements
Request without data (means earlier than HART 7) resets the Master specific flag.

2.1.11.2 Implementation

For command 38 definitions in Entry Tool two objects “confChangeCounterValue” must be used with type “REQUEST” and with type “Response” to get a “HART_SPECIAL_WRITE” command. A command 38 specific part is to add in layer7.c HARTLayer7CommandInterpreter in case of “HART_SPECIAL_WRITE”. Here the function “HARTPutAttribute” with ACTION object “resetConfigFlag” as parameter is called.

In ACTION function “resetConfigFlag_HART” in “Hart_action_fct.c” the algorithm to reset the master specific Configuration Change Flag is implemented.

HART revision specific algorithm is documented in 2.6.1.2. For detailed information about Configuration Change Counter and Configuration Change Flags see 2.3.3 and 2.3.4.

2.1.12 Command 48: Read Additional device Status

2.1.12.1 Requirements

Earlier Common Practice Command in HART 7 allocated to Universal Commands.

Page 41 in Spec 127

Request Data Bytes

Byte	Description	Implementation
0 - 6	Device-Specific Status	Variable: device specific Code
6	Extended Device Status	Variable: Code of Common Table 17
7	Device Operating Mode	Variable: Code of Common Table 14
8	Standardized Status 0	Variable: Code of Common Table 29
9	Standardized Status 1	Variable: Code of Common Table 30
10	Analog Channel Saturated	Variable: Code of Common Table 27
11	Standardized Status 2	Variable: Code of Common Table 31
12	Standardized Status 3	Variable: Code of Common Table 32
13	Analog Channel Fixed	Variable: Code of Common Table 28
14 -24	Device-Specific Status	Variable: device specific Code

The request data is used for comparison with actual status values of device. If there is an exact match then the Master specific bit “More Status Available” is reset. Mismatch or too few bytes doesn’t change the bit.

Response Data Bytes

Byte	Description	Implementation
0 - 6	Device-Specific Status	Variable
6	Extended Device Status	Variable: Code of Common Table 17
7	Device Operating Mode	Variable: Code of Common Table 14
8	Standardized Status 0	Variable: Code of Common Table 29
9	Standardized Status 1	Variable: Code of Common Table 30
10	Analog Channel Saturated	Variable: Code of Common Table 27
11	Standardized Status 2	Variable: Code of Common Table 31
12	Standardized Status 3	Variable: Code of Common Table 32
13	Analog Channel Fixed	Variable: Code of Common Table 28
14 -24	Device-Specific Status	Variable

All devices must support bytes 0 – 8. If device supports more than one Analog then bytes 9 – 13 must be returned as well.

Command Specific Response Code
Subsection 6.24 on Page 2 of Spec 127 Addendum

Backwards Compatibility Requirements
Request without data (means earlier than HART 7) must indicate "Success (RC #0)" and include all Response Data Bytes

2.1.12.2 Implementation

Command 48 is implemented as indexed command to get the possibility to return HART Revision specific object values corresponding to the revision number used as index.

In Entry Tool identical objects are configured for request data each with type REQUEST and for response data each with type RESPONSE to get a "HART_SPECIAL_WRITE" command. For every possible index configured with type SLOT the specific response object is defined. The data of these objects is returned at the places of the objects in response data (response data object with parameter RESPONSE).

A command 48 specific part is to add in layer7.c HARTLayer7CommandInterpreter in case of "HART_SPECIAL_WRITE". Here the function "HARTPutAttribute" with ACTION object "resetMoreStatusAvailable" as parameter is called.

In ACTION function "resetMoreStatusAvailable_HART" in "Hart_action_fct.c" the algorithm to reset signalization "More Status Available" is implemented.

HART revision specific functionality for command 48 is documented in 2.6.1.2. Additional information regarding new request/response objects is given in corresponding chapters belonging to chapter 2.3. More details regarding the signalization "More Status Available" see 2.3.19.

2.2 Common Practice Commands

In this chapter the changes and the enhancements of HART 7 Command Practice Commands, which are implemented in the EDP300 on base of HART 5.9, will be documented. Also the commands for a device that supports Burst Mode are added. Not all details are described. They are documented in the specification document which is referenced in each chapter.

2.2.1 Command 54: Read Device Variable Information

2.2.1.1 Requirements

Page 57 of Spec 151

Response with the transducer serial number, the Limits, Damping Value and Minimum Span of the selected Device Variable along with the corresponding engineering units.

Command must be supported if device supports Burst Mode.

2.2.1.2 Implementation

Precondition for realization of command 54 is the implementation of indexed commands (see Page 32 in Spec 99).

Command definition can be done with Entry Tool. With the mechanism for indexed commands in Entry Tool for every possible device variable addressed by its variable code as slot code the specific response object is defined. The data of these objects is returned at the places of the objects in response (response data object with type RESPONSE).

HART revision specific functionality for command 9 is documented in 2.6.1. For detailed information about the Device Variables see chapter 2.3.11 and 2.3.13. Additional information regarding request/response objects is given in corresponding chapters belonging to chapter 2.3.

2.2.2 Command 59: Write Number Of Response Preambles

2.2.2.1 Requirements

Page 57 of Spec 151

This value may be set to no smaller than 5 and no greater than 20.

Additional Response Code:

Code	Description	Implementation
16	Set to Nearest Possible Value	
17 - 31	Undefined	
32	Busy	
33 - 127	Undefined	

2.2.2.2 Implementation

Command definition can be done with Entry Tool.

2.2.3 Command 72: Squawk

2.2.3.1 Requirements

Command shall be newly implemented.

Page 81 of Spec 151

Command causes the addressed device to visually indicate the reception of the command on HMI.

Request Data Bytes

Byte	Description	Implementation
None		

Response Data Bytes

Byte	Description	Implementation
None		

2.2.3.2 Implementation

Not yet defined.

2.2.4 Command 73: Find Device

2.2.4.1 Requirements

Command shall be newly implemented

Page 82 of Spec 151

Slaves must only respond when physical/mechanically armed. Device is addressed via long frame address or Broadcast Address.

Request Data Bytes

Byte	Description	Implementation
None		

Response Data Bytes
Command 0 specific data bytes: look to HART 7 Command 0 specific changes!

2.2.4.2 Implementation

The functionality could be enabled using the HART object "findDeviceSwitch". If functionality is enabled the device answers to command 73 request. The enabled functionality is managed by a counter for a 5 minutes time slot, which is cyclically decremented in "LAYER7_CommandInterpreter". If time is up when counter gets 0 the functionality is disabled so a command 73 request will not be processed in IRQ_Receive_LAYER2.
Command definitions can be defined with Entry Tool (see 2.1.1.2).

2.2.5 Command 78: Read Aggregated Commands

2.2.5.1 Requirements

Command shall be newly implemented

Page 90 of Spec 151

Multiple read commands can be aggregated in one transaction. In reply the response for each command is included.

2.2.5.2 Implementation

Not yet defined.

2.2.6 Command 79: Write Device Variable

2.2.6.1 Requirements

Command shall be newly implemented

Page 92 of Spec 151

Command allows a Device Variable to be temporarily forced to a fixed value.

2.2.6.2 Implementation

2.2.7 Command 80: Read Device Variable Trim Point

2.2.7.1 Requirements

Command should be newly implemented.

Page 94 of Spec 151

Command reads the last successful trim points.

2.2.7.2 Implementation

Not yet defined.

2.2.8 Command 81: Read Device Variable Trim Guidelines

2.2.8.1 Requirements

Command should be newly implemented.

Page 95 of Spec 151

Command reads information that a Host will need to guide a user through correct selection of trim points.

2.2.8.2 Implementation

Not yet defined.

2.2.9 Command 82: Write Device Variable Trim Point

2.2.9.1 Requirements

Command should be newly implemented.

Command performs a calibration adjustment for the indicated Device Variable.

Details see Spec

2.2.9.2 Implementation

Not yet defined.

2.2.10 Command 83: Reset Device Variable Trim

2.2.10.1 Requirements

Command should be newly implemented.

Page 99 of Spec 151

Command allows the user to rest the Device Variable to the default factory trim.

2.2.10.2 Implementation

Not yet defined.

2.2.11 Command 103: Write Burst Period

2.2.11.1 Requirements

Command must be implemented if device supports Burst Mode

Page123 of Spec 151

Command selects the minimum and maximum update period of a burst message.

2.2.11.2 Implementation

Command definition can be done with Entry Tool. With the mechanism for indexed commands in Entry Tool for every possible burst message as slot code the specific response object is defined. The burst message specific data of these objects (objects with type SLOT) is returned at the places of the objects in response (response data object with type RESPONSE).

Additional information regarding request/response objects is given in chapters 2.3.51 and 2.3.52.

2.2.12 Command 104: Write Burst Trigger

2.2.12.1 Requirements

Command must be implemented if device supports Burst Mode

Page125 of Spec 151

Command configures the trigger that forces publishing of the Burst Message. Four trigger modes (Commom Table 33) are supported: Continuous, Windowed, Rising and Falling.

2.2.13 Command 105: Read Burst Mode Configuration

Command must be implemented if device supports Burst Mode

Page125 of Spec 151

Command allows the Burst Mode Configuration to be read. The Field Device responds with whether the Field Device is in Burst Mode; the command to be burst and a list of Device Variables to be transmitted, the burst minimum and maximum update time and the condition for the maximum update time.

Backward Compatibility Requirements
If the device receives a Request without data bytes the device may not respond with Response Code 5 - "Too Few Data Bytes Received" but must assume that it is read from a HART 5 or HART 6 master. The Device will return the configuration of Burst Message 0.

2.2.14 Command 107: Write Burst Device Variables

Command must be implemented if device supports Burst Mode

Page132 of Spec 151

Command selects the Device Variables that will be used by a bursting device to be returned by a Command 9 or 33 in Burst Mode.

Backward Compatibility Requirements
For backward compatibility a field device must assume that it is configured by a HART 5 or HART 6 host when it receives only 1, 2, 3 or 4 Device Variables. In such a case the device shall not return Response Code 5 - "Too Few Data Bytes Received" instead it must treat the message as a configuration of the first Burst Message with all other Device Variables set to 250 - "Not Used".

The field device shall not return "Burst Condition Conflict" but it will reconfigure the attributes associated with Command 104.

2.2.15 Command 108: Write Burst Mode Command Number

Command must be implemented if device supports Burst Mode

Page135 of Spec 151

Command selects the response message that the device transmits while in Burst Mode. Command 1, 2, 3, and 9 shall be supported in all devices implementing Burst Mode.

Backward Compatibility Requirements
If a field device receives only one data byte in the request it must assume that it is configured by a HART 5 or HART 6 host and treat is as if the first Burst Message was configured. In this case the device must not return Response Code 5 - "Too Few Data Bytes Received".

The field device shall not return "Burst Condition Conflict" but it will reconfigure the attributes associated with Command 104.

2.2.16 Command 109: Burst Mode Control

Command must be implemented if device supports Burst Mode

Page135 of Spec 151

Command is used to enter and exit the Burst Mode on the device's Token-Passing or TDMA Data-Link (Common Table 9, Burst Mode Control Codes).

Backward Compatibility Requirements
To maintain backward compatibility, the device shall only respond with "Too Few Data Bytes Received" (Response Code 5) if no request data bytes are received. If only one byte is received and the Burst Mode Control Code is greater than 1 then the device must respond with "Too Few Data Bytes Received".

When the Burst Message number is not included the device must assume that Burst Message 0 is being activated or deactivated for publishing on the Token-Passing Data-Link. When a single request byte is received only Burst Mode Control Codes 0 and 1 are valid.

Changed Request Data Bytes

Byte	Description	Implementation
1	Burst Message	

Changed Response Data Bytes

Byte	Description	Implementation
1	Burst Message	

2.3 Additional needed device data or status

In the following chapters device data objects and device status objects needed for the command definition by Entry Tool are described.

2.3.1 Minimum Number of Response Preambles

Used with Cmd 0, 11, 21
Unsigned-8 device specific constant must be defined in subsystem HART.

2.3.1.1 Implementation

HART Object “numberRespPream” is defined for 5 response preambles.

2.3.2 Maximum Number of Device Variables

Used with Cmd 0, 11, 21
Unsigned-8 device specific constant must be defined in subsystem HART.
Last valid Device Variable Code is to set.

2.3.2.1 Implementation

HART object “maxNoOfDevVariables” contains code 0x06 as highest possible index of device variables (see also 2.3.11).

2.3.3 Configuration Change Counter

Used with Cmd 0, 11, 21, 38
Counted during execution of Cmd 6, 17, 18, 39, 59, 79, 82, 83, 103, 104, 107, 108, 109
Unsigned-16 Variable must be defined in subsystem HART. Function for subsystem HART and HMI needed to increment counter.

In the description for Command 0 Spec 127 Universal Commands Page 12 is written:
The Configuration Change Counter must be incremented once for every command received that changes the devices configuration. The counter must also be incremented once for every user action that changes the device's configuration or calibration (e.g., from a local operator interface). This value is never reset or written and must be maintained even if power is removed from the device or a device reset is performed.

2.3.3.1 Implementation

HART object “confChangeCounterValue” contains the counter value. The object is defined in NV RAM to be only initialized after first device start.

2.3.4 Configuration Changed Flag

Used with Cmd 38
Primary and Secondary Master specific Flag is to be actualized.
A field device must contain a Configuration Changed bit for both the primary and the secondary master. Both bits are set when any configuration item in a field device is modified. Configuration Changed status is part of Device Status (Device Status on Page 35 in Spec 99 Command Summary.

2.3.4.1 Implementation

Master specific Configuration Changed Flag is set or reset managed with the HART CONF object “configurationFlags”. In t_data_obj_conf.c “Transform2Internal_CONF” with parameter “Set” flags for all masters are set. With parameter “Reset” only the master specific flag is reset.

2.3.5 Extended Field Device Status

Used with Cmd 0, 9, 11, 21, 48

8 Bits must be defined in subsystem (?). The Code of Common Table 17 of Page 58 in Spec 183 is used.
Andreas W. follows up how to handle the actual information.
DiagnosisServices

2.3.5.1 Implementation

2.3.6 Manufacturer Identification Code

Used with Cmd 0, 11, 21
2 Byte Code of Common Table 8 of Page 58 in Spec 183 is used.

2.3.6.1 Implementation

In HART object "mfrId" the company code 0x1A is defined. In HART object "mfrDeviceType" the device specific code (e.g. for EDP300 0x8D) is defined. For the use as response info in the commands both objects must be configured with Entry Tool.

2.3.7 Private Label Distributor Code

Used with Cmd 0, 11, 21
1 Byte Code of Common Table 8 of Page 54 in Spec 183 is used.

2.3.7.1 Implementation

In HART object "privateLabelDistributorCode" code 0x001A is defined. NV RAM is used because code should be changeable if OEM code shall be defined.

2.3.8 Device Profile

Used with Cmd 0, 11, 21
2 Byte Code of Common Table 57 of Page 58 in Spec 183 is used.

2.3.8.1 Implementation

In HART object "deviceProfile" code 0x01 for "HART Automation Device" is defined.

2.3.9 Loop Current Mode

Used with Cmd 6, 7
1 Byte Code of Common Table 16 on Page 58 in Spec 183 is used.

A Field Devices must be able to operate in multi-drop with loop current signaling disabled when it works as current source. When current signaling is disabled, the loop current is set to the minimum value required for field device operation, the field device status bit 3, Loop Current Fixed, is set, and, if appropriate, the Upscale/Downscale Alarm is disabled.

Positioner works as current sink. Therefore the change of Loop Current Mode doesn't trigger any current impact. To fulfill the HCF conformance the device status "Loop Current Fix" is set corresponding to disabled with Command 6.

2.3.9.1 Implementation

In HART object "loopCurrentMode" status "disabled" with code 0x00 and status "enabled" with code 0x01 is stored.

In procedure HARTCheckSecRespByte in layer7.c status "Loop Current Fix" is set in Device Status of a command response if Loop Current Mode is disabled.

2.3.10 Device Variable Classification

Used with Cmd 8, 9

1 Byte Code of Common Table 21 on Page 60 in Spec 183 is used.

These codes indicate the function performed by the Device Variable. This allows Masters and Host Applications to identify the type of process connection supported by the Device Variable and the Engineering Unit Code Expansion table to be used.

Classification Code 91 “Valve Actuator” could be specified. But a specific Engineering Unit Code Expansion table is not defined in Common Tables for this Classification Code.
So Classification 0 “Device Variable not classified” should be taken.

Decision

For EDP300 Classification “Device Variable not classified” will be taken.

2.3.10.1 Implementation

In HART object “devVariableClassification” code 0x00 “Device Variable not Classified” is defined.

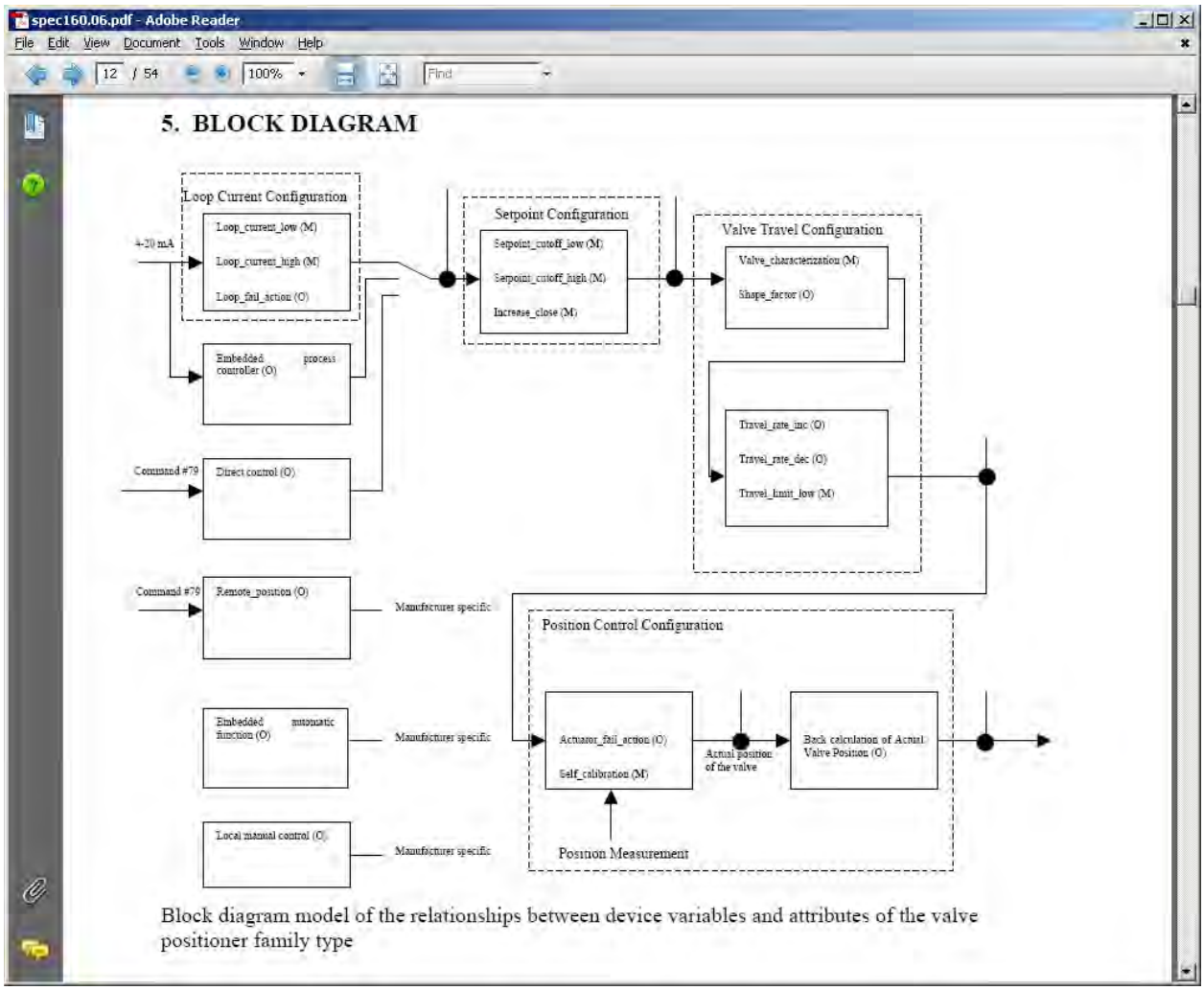
2.3.11 Device Variable

Used with Cmd 9, 79

It must be clarified if more Device Variables beside the four Dynamic Variables should be used. Code of these Device Variables must be defined. The highest code number is needed as Maximum Number (see above).

Decision

6 Device Variables are defined for EDP300 as recommended in Valve Positioner Device Family Spec (160.6 Rev 1.0: page 10):



2.3.11.1 Implementation

The following values define the possible Device Variables and are defined in the different subsystem objects as listed below:

Control Value *Coordinator.setpointIn*
Valve Setpoint *Coordinator.valveSetpoint*
Target Position *Coordinator.setpointOut*
Valve Position *Coordinator.positionExtern*
BCalc value *Coordinator.setpointBkOut*

<Pressure 1> *Pressure.pressureY1 (Druck Antriebskammer 1)*
<Pressure 2> *Pressure.pressureY2 (Druck Antriebskammer 2)*

Percent Range *CurrentIN.percentValue*
Loop Current *CurrentIN.current*

2.3.12 Dynamic Variables

Used with Cmd 1, 3, 8, 14, 15

2.3.12.1 Implementation

The following assignments of Device Variables to Dynamic Variables are done:

PV	*Primary Variable*	*Control Value*	*Coordinator.setpointIn*
SV	*Secondary Variable*	*Valve Setpoint*	*Coordinator.valveSetpoint*
TV	*Tertiary Variable*	*Target Position*	*Coordinator.setpointOut*
QV	Quaternary Variable	Valve Position	Coordinator.positionExtern

2.3.13 Device Variable Code

Used with Cmd 9, 54, 79, 105, 107
Code of Common Table 34 Page 64 of Spec 183 is used

Following Device variables are defined with Codes in Common Table 34:

244 Percent Range
245 Loop Current
246 Primary Variable
247 Secondary Variable
248 Tertiary Variable
249 Quaternary Variable

The 6 device variables listed above must be implemented in every HART 7 devices. These device variables are not reported in the Command 0 response. Please note that if the device does not support device variables, the command 0 response will indicate 0 for the last device variable code supported. The device is not required to implement more than these 6 device variables. If the device does not implement device variables other the ones listed above, the device will also need to support PV for device variable 0, SV for device variable 1, TV for device variable 2, and QV for device variable 3.

2.3.13.1 Implementation

Decision

Additional to the predefined codes the EDP300 specific Device Variables get the following codes:

0 Control Value
1 Valve Setpoint
2 Target Position
3 Valve Position
4 BCalc value
5 < Pressure 1 >
6 < Pressure 2 >

2.3.14 Units Code

Used with Cmd 9, 79
Common Table 2 on Page 32 in Spec 183 is used.

Codes for every Dynamic and Device variable is needed

2.3.14.1 Implementation

2.3.15 Device Variable Status

Used with Cmd 9, 79
Structure documented on Page 43 in Command Summary Spec 99

Status for every Dynamic and Device variable is needed

2.3.15.1 Implementation

2.3.16 Real Time Clock

Used with Cmd 9, 54

A monotonic timer counter with rollover every 24h must be available. Not required to support a settable Real Time Clock. Time stamp is needed in Command 9.

Format "Time" (Page 16 in Spec 99 Command Summary)
In the Protocol, time is contained within an unsigned 32-bit binary integer with the least significant bit of the time value representing 1/32 of a millisecond (i.e., 0.03125 milliseconds). When time data types are used to represent time of day they indicate the number of 1/32 of milliseconds since midnight. Time can span in excess of 37 hours. When a longer span is required the field device should use time in conjunction with a Date (see Section 5.2).

2.3.16.1 Implementation

2.3.17 PV Analog Channel Flags

Used with Cmd 15
Code of Common Table 26 on Page 62 in Spec 183 is used.
With Set Flag this Analog Channel is a Field Device analog input channel.

2.3.17.1 Implementation

In HART object "pvAnalogChannelFlags" code 0x01 "Analog Channel is a Field Device analog input channel" is defined.

2.3.18 Long Tag

Used with Cmd 20, 21, 22
Variable contains 32 ISO Latin-1 characters. Tag and Long Tag are completely separate data items.

2.3.18.1 Implementation

In HART object "longTag" string is set to question mark characters (0x3F) after first start of device.

2.3.19 Signalizing "More Status Available"

Used with Cmd 48

2.3.19.1 Implementation

Object "diagnosisCounter" of subsystem "DiagnosisServices" contains a counter value which is incremented if new diagnosis status info is prepared.

In layer7.c function "HARTCheckSecRespByte" compares the value of object "diagnosisCounter" with the value of subsystem "HART" object "diagCounterMoreStatusAvailable". If values are different status "More Status Available" is set in device status which is send as first data byte of every response.

With command 48 defined status objects can be read. A send of a second request containing the info of last response causes a switch off of the signalization if new response data is identical to request data. The value of object "diagnosisCounter" is stored in object "diagCounterMoreStatusAvailable".

2.3.20 Device-Specific Status

Used with Cmd 48
Refer to appropriate device-specific document for detailed information. As example: subsystem diagnosis provides EDP300 specific status.

2.3.20.1 Implementation

2.3.21 Device Operating Mode

Used with Cmd 48
Common Table 14 on Page 58 of Spec 183 is reserved for definition. Codes are not specified in table.

2.3.21.1 Implementation

2.3.22 Analog Channel Saturated

Used with Cmd 48
Code of Common Table 27 on Page 62 in Spec 183 is used.

2.3.22.1 Implementation

2.3.23 Analog Channel Fixed

Used with Cmd 48
Code of Common Table 28 on Page 62 in Spec 183 is used.

2.3.23.1 Implementation

2.3.24 Standardized Status 0

Used with Cmd 48
Code of Common Table 29 on Page 63 in Spec 183 is used

2.3.24.1 Implementation

2.3.25 Standardized Status 1

Used with Cmd 48
Code of Common Table 30 on Page 63 in Spec 183 is used

2.3.25.1 Implementation

2.3.26 Standardized Status 2

Used with Cmd 48

Common Table 31 on Page 63 in Spec 183 is reserved for definition. Codes are not specified in table.

2.3.26.1 Implementation

2.3.27 Standardized Status 3

Used with Cmd 48

Code of Common Table 32 on Page 63 in Spec 183 is used. Reserved for WirelessHART.

2.3.27.1 Implementation

2.3.28 Device Variable Transducer Serial Number

Used with Cmd 54

2.3.28.1 Implementation

2.3.29 Device Variable Limits/Minimum Span Units Code

Used with Cmd 54

Code of Common Table 2 on Page 32 in Spec 183 is used.

2.3.29.1 Implementation

2.3.30 Device Variable Upper Transducer Limit

Used with Cmd 54

2.3.30.1 Implementation

2.3.31 Device Variable Lower Transducer Limit

Used with Cmd 54

2.3.31.1 Implementation

2.3.32 Device Variable Damping Value

Used with Cmd 54

2.3.32.1 Implementation

2.3.33 Device Variable Minimum Span
Used with Cmd 54

2.3.33.1 Implementation

2.3.34 Device Variable Family
Used with Cmd 54.

2.3.34.1 Implementation

2.3.35 Write Device Variable Command Code
Used with Cmd 79
Code of Common Table 19 on Page 59 in Spec 183 is used.
Implementieren nach Simu Betrieb

2.3.35.1 Implementation

2.3.36 Trim Points Units Code
Used with Cmd 80, 81, 82
Code of Common Table 2 on Page 32 in Spec 183 is used.

2.3.36.1 Implementation

2.3.37 Trim points supported
Used with Cmd 80, 81, 82
Code of Common Table 22 on Page 61 in Spec 183 is used.

2.3.37.1 Implementation

2.3.38 Lower or Single Trim Point
Used with Cmd 80

2.3.38.1 Implementation

2.3.39 Upper Trim Point
Used with Cmd 80

2.3.39.1 Implementation

2.3.40 Minimum Lower Trim Point
Used with Cmd 81

2.3.40.1 Implementation

2.3.41 Maximum Lower Trim Point
Used with Cmd 81

2.3.41.1 Implementation

2.3.42 Minimum Upper Trim Point
Used with Cmd 81

2.3.42.1 Implementation

2.3.43 Maximum Upper Trim Point
Used with Cmd 81

2.3.43.1 Implementation

2.3.44 Minimum Differential
Used with Cmd 81

2.3.44.1 Implementation

2.3.45 Trim Point Value
Used with Cmd 82

2.3.45.1 Implementation

2.3.46 Burst Message
Used with Cmd 103, 104, 105, 107, 108, 109

2.3.46.1 Implementation
The Burst Message is the identification number used to select one of the three HART objects "burstMsg0ConfData", …, "burstMsg2ConfData" containing the configuration data or to select one of the HART objects "burstMsg0ManageData", …, "burstMsg2ManageData" which contain the management data of the specific burst message. In every object "burstMsg0ConfData"… and "burstMsg2ConfData" an attribute "burstMsgNumber" exists. The attributes of these objects are initialized with the corresponding burst message number.

2.3.47 Total Number of Burst Messages
Used with Cmd 105

2.3.47.1 Implementation
HART object bursttotalNumberOfMsg contains the value 3 to enable maximum 3 different burst messages for use.

2.3.48 Burst Mode Control Code
Used with Cmd 105, 109
Code of Common Table 9 on Page 57 in Spec 183 is used

2.3.48.1 Implementation
Burst Mode Control Code is a parameter which has to be stored for every burst message. The specific values are to store in attribute "modeControlCode" specific for the burst message in corresponding HART objects "burstMsg0ConfData"… or "burstMsg2ConfData". The attributes of these objects are initialized with value 0 corresponding to code "Off" after first start of device.

2.3.49 Command Number
Used with Cmd 105, 108

2.3.49.1 Implementation
Command Number is a parameter which has to be stored for every burst message. The specific values are to store in attribute "commandNumber" specific for the burst message in corresponding HART objects "burstMsg0ConfData"… or "burstMsg2ConfData". The attributes of these objects are initialized with value 1 corresponding to code "Command 1" after first start of device.

2.3.50 Burst Message Slots
Used with Cmd 105, 107

2.3.50.1 Implementation
For every burst message for 8 slots device variable codes could be defined. So corresponding attributes burstSlot0 … burstSlot7 are needed specific for the burst message in corresponding HART objects "burstMsg0ConfData"… or "burstMsg2ConfData". The attributes of these objects are initialized with value 250 corresponding to variable code "Not used" after first start of device.

2.3.51 Update Period
Used with Cmd 103, 105
See Format "Time" documented in 2.3.16

2.3.51.1 Implementation

Update period is a parameter which has to be stored specific for every burst message. Attribute "updatePeriod" specific for the burst message in corresponding HART objects "burstMsg0ConfData" … or "burstMsg2ConfData"are defined. The attributes of these objects are initialized with value 32000 corresponding to 1 sec after first start of device.

2.3.52 Maximum Update Period

Used with Cmd 103, 105

See Format "Time" documented in 2.3.16

2.3.52.1 Implementation

Maximum Update period is a parameter which has to be stored specific for every burst message. Attribute "maximumUpdatePeriod" specific for the burst message in corresponding HART objects "burstMsg0ConfData" … or "burstMsg2ConfData"are defined. The attributes of these objects are initialized with value 1920000 corresponding to 60 sec after first start of device.

2.3.53 Burst Trigger Mode Selection Code

Used with Cmd 104, 105

Code of Common Table 33 on Page 64 in Spec 183 is used.

2.3.53.1 Implementation

Burst Trigger Mode Selection Code is a parameter which has to be stored specific for every burst message. Attribute "triggerModeSelectionCode" specific for the burst message in corresponding HART objects "burstMsg0ConfData" … or "burstMsg2ConfData" are defined. The attributes of these objects are initialized with value 0x00 corresponding to mode "Continuous" after first start of device.

2.3.54 Device Variable Classification for Trigger Level

Used with Cmd 104, 105

1 Byte Code of Common Table 21 on Page 60 in Spec 183 is used.

2.3.54.1 Implementation

Device Variable Classification is a parameter which has to be stored specific for every burst message. Attribute "triggerDevVariableClassification" specific for the burst message in corresponding HART objects "burstMsg0ConfData" … or "burstMsg2ConfData" are defined. The attributes of these objects are initialized with value 0 corresponding to "Device Variable not Classified" after first start of device.

2.3.55 Units Code

Used with Cmd 104, 105

Code of Common Table 2 on Page 32 in Spec 183 is used.

2.3.55.1 Implementation

Units Code is a trigger parameter which has to be stored specific for every burst message. Attribute "triggerUnitsCode" specific for the burst message in corresponding HART objects "burstMsg0ConfData" … or "burstMsg2ConfData" are defined. The objects are initialized with value 250 corresponding to code "Not used" after first start of device.

2.3.56 Trigger Level

Used with Cmd 104, 105

2.3.56.1 Implementation

Trigger Level is a trigger parameter which has to be stored specific for every burst message. Attribute “triggerValue” specific for the burst message in corresponding HART objects “burstMsg0ConfData” … or “burstMsg2ConfData” are defined. The objects are initialized with value 0 after first start of device.

2.3.57 Idle Burst Message

HART object burstIdleMessage contains the number of the burst message with the shortest update period. The object is initialized with value 0 for burst message 0 after first start of device.

2.4 Additional Functionality

2.4.1 Transducer Trim Procedure

2.4.1.1 Requirements of HART specifications

Page 19 of Spec 151 Common Practice Commands
Adjustment of a Device Variable reading is one of the most common functions that instruments must support. The commands in this section constitute a trim procedure that is applicable to a variety of instruments, both transmitters and actuators.
1. Issue Command 81, Read Device Variable Trim Guidelines, to determine the number of applicable trim points and their acceptable limits.
2. Issue Command 80, Read Device Variable Trim Points, to retrieve the last trim points used. These should be used as default values for a new trim operation. If the value supports a two point trim, then perform the low trim first (step 3-6).
3. Prompt the user to set the variable input to a value within acceptable limits for the trim point.
4. Once set and stable, obtain the exact process variable value from either a calibrator or the user.
5. Issue Command 82, Write Device Variable Trim Point. Inform user of any errors.
6. If a DR_INITIATE response is received then resend the identical trim command until the operations is completed. Once completed, inform the user of the results.
7. If the variable supports a two-point trim and the low trim is completed successfully, repeat steps 3 through 6 for the upper trim point.

Note: Some devices may only support an upper trim point (see Common Table 22, Trim Point Codes)

2.4.2 Indexed Commands

2.4.2.1 Requirements of HART Specification

Page 32 of Spec 99

2.4.2.2 Implementation

In existing HART subsystem for HART Revision 5.9 “slot commands” were implemented to be able configure command specific with the Entry Tool. Normally they are executed in function HARTSpecialReadWriteObjects in hart_objects.c. The command 48 is a special case because Revision 5.9 and Revision 7.2 specific object configuration is done with the Entry Tool but a specific interpretation is done command specific.

- for commands containing variable number of slot
- for commands with not continuous slot index numbers
- for commands with leading slot object or with contained slot object
- for commands with revision specific command objects

2.4.3 Burst Mode

2.4.3.1 Requirements of HART Specifications

Page 22 of Spec 151 Common Practice Commands
If a Field Device supports Burst Mode then all of Commands 103 – 105 and Commands 107 – 109 must be implemented:

- *Command 103 Write Burst Period*
- *Command 104 Write Burst Trigger*
- *Command 105 Read Burst Mode Configuration*
- *Command 107 Write Burst Device Variables*
- *Command 108 Write Burst Mode Command Number*
- *Command 109 Burst Mode Control*

Devices implementing burst mode must support at least 3 Burst Messages. Each Burst Message must allow a different configuration. In other words, each Burst Message may be a different command, trigger, update period, set of Device Variables etc. The device must retain Burst Mode Settings through a Device Reset, Self Test or the power being removed and re-applied.

Configuring A Device For Burst Mode Operation
The procedure a Master should follow to place a Field Device into burst mode is as follows:

1. The command response is configured using Command 108, Write Burst Mode Command Number. Commands 1, 2, 3, 9 and 48 must be available for publishing and Command 33 must be supported if it is implemented in the field device.

2. For Commands 9 and 33, Command 107, Write Burst Device Variables, is used to assign Device Variables to the response data slots. Command 9 supports up to 8 slots and Command 33 supports up to 4.

3. Use Command 54 to determine the update rates for the desired Device Variables. Use Command 103 to set the Update Period and Maximum Update Period for publishing the Burst Message (see Subsection 6.9.2). The Data-Link requirements may result in the Burst Message being published more frequently then the Update Period.

4. Set the Trigger Mode for the Burst Message using Command 104. Burst Messages may be configured to publish continuously or based on a Trigger Level. The Burst Message will always be published at least as often as the Maximum Update Period.

5. Issuing Command 109, Burst Mode Control, will enter or exit Burst Mode. While in Burst Mode, the Slave will begin transmitting the responses to the command number set by Command 108 based on the Data-Link requirements and the properties configured using the other Burst Mode Commands.

Once the device is in burst mode, Command 108 can be used to change the burst command response. A device may take one burst response before the response changes to the new command number. Other Burst Mode Commands may also be used to adjust burst mode operation on-the-fly.

Update Periods
The update periods may be programmed as indicated in Table 1. Field devices must correct settings differing from these values and indicate "Set to Nearest Value" in its response message.

Table 1. Update Periods Allowed (in Seconds)

< 0.100 Not Allowed	0.500	4.000	32.000
0.100	1.000	8.000	60-3600 (Any Value)
0.250	2.000	16.000	> 3600 Not Allowed

Note: The Update Times determine only the communication rate. The acquisition period may be higher or lower. The Update Time Period for a device variable can be read using command 54.

Furthermore, the settings are constrained based on the Physical Layer being used for publishing the Burst Message (see Table 2). The table also shows the default update period. For Token-Passing based Physical Layers the default update rate is determined by the Data-Link Layer requirements.

Table 2. Minimum Update Rates Allowed by Physical Layer

Physical Layer	Minimum Value	Default Period
FSK	0.500s	N/A
PSK	0.100s	N/A
RS-485	0.100s	N/A
IEEE 802.15.4	250s	60s

The update period is set to the rate is dictated by process and application requirement and often this may be larger then the minimum. However, on Token-Passing data-Link, Burst Messages are used both to publish data and to pass the token. When multiple Burst message are enabled, the device must transmit the Burst Message with the shortest period the majority of the time and transmit the other Burst message as their periods lapse.

Trigger Mode

Command 104 configures the trigger that forces publishing of the Burst Message. Four trigger modes are supported: Continuous, Windowed, Rising, and Falling (see Common Table 33, Burst Mode Trigger Mode Codes). Unless otherwise configured by this command a Burst Message shall assume a Trigger Mode of "Continuous" and a Trigger Value of 0.00.

These trigger modes allow the device to be configured to defer the publishing of the Burst Message beyond the Update Period in Command 103. In all cases the Burst message is triggered when the Maximum Update Time in Command 103 is exceeded. In addition the trigger source is also specified and is normally the first process value returned in the command (see Table 5).

Table 5. Burst Message Trigger Source

Command	***Trigger Source Value***
1	*PV*
2	*Percent of Range*
3	*PV*
9	*Device Variable in Slot 0*
33	*Device Variable in Slot 0*

2.4.3.2 Implementation

The possibility to configure *three different Burst Messa*ges will be implemented.

Objects to store burst message configuration values

(see 2.3.48)
(see 2.3.49)

(see 2.3.50)

(see 2.3.51)
(see 2.3.52)
(see 2.3.53)
(see 2.3.54)
see 2.3.55)
(see 2.3.56)

Additional for every burst message update period counter objects (burstMsgUpdatePeriodCounter containing 3 message number indexed elements) and maximum update period counter (burstMsgMaximumUpdatePeriodCounter containing 3 message number indexed elements) are to declare where the message specific cycle time for Update Period and Maximum Update Period is managed.

Also to process the different burst trigger mode for every burst message an area object to store last sent values (burstMsgLastSentData: as so many bytes as needed for the longest data area of possible values for all used burst commands) and two objects to store lower and higher trigger limit values (burstMsgTriggerLowLimit; burstMsgTriggerHighLimit containing each 3 message number indexed elements) are needed.

Cycle time management
The cycle time management of the Burst Messages is done during the cyclically working function "HARTLayer7CommandInterpreter ". The cycle time counters (Update Period and Maximum Update Period) of all Burst Messages are decremented until they get value 0. Value 0 of the Update Period cycle time counter signalize that this burst message must be included to the burst message preparation management.

Parameterization of Cycle time management

Starting with burst message 0 the burst message with the shortest update period is identified after Update Periods parameterization with Command 103 and its message number is stored in HART object burstIdleMsg.

The valid time values for Update Period and Maximum Update Period of the command 103 request data starts with 0.500 sec because device works with FSK interface. The request entries must be checked against the entries of table "Update Periods Allowed". The time values are stored as counters of 1/32 of a millisecond. They are corrected to the table values as follow:

Update Period *[sec]*	***Correction Limits***	***Cycle Time Counter***	***Lower Limit Counter***	***Upper Limit Counter***
< 0.500 Not Allowed	**0.500**	***16000***	*16000*	*-/-*
0.500	**0.500** *<= 0.749*	***16000***	*16000*	*23968*
1.000	*0.750 <=* **1.000** *<= 1.499*	***32000***	*24000*	*47968*
2.000	*1.500 <=* **2.000** *<= 2.999*	***64000***	*48000*	*95968*
4.000	*3.000 <=* **4.000** *<= 5.999*	***128000***	*96000*	*191968*
8.000	*6.000 <=* **8.000** *<= 11.999*	***256000***	*192000*	*383968*
16.000	*12.000 <=* **16.000** *<= 23.999*	***512000***	*384000*	*767968*
32.000	*24.000 <=* **32.000** *<= 45.999*	***1024000***	*768000*	*1471968*
60.000	*45.599 <=* **60.000**	***1920000***	*1472000*	*1920000*
60 -3600 (Any Value)	*No correction*	***<value> * 32***	*-/-*	*-/-*
> 3600 Not allowed	**3600.000**	***115200000***	*-/-*	*115200000*

Burst message preparation management

The preparation of next to send burst message "PrepareBurstBuf_LAYER2" is done

- *at end of function "Initialize_LAYER2",*
- *during function "StartResponse_LAYER2" when device is in Burst Mode and received message is not Burst Frame (Frame Type not BACK),*
- *during function "StartBurst_LAYER2".*

"StartResponse_LAYER2" is executed

- *during function "IRQ_HartLayer2Timer" if timer was loaded with LAYER2_TIMER_START_COMMERR_RESP*
- *and at end of HARTLayer7CommandInterpreter after the generation of a command response*

"StartBurst_LAYER2" is executed

- *during function "IRQ_Receive_LAYER2" if device is in Burst Mode AND BACK frame from another slave was received OR ACK frame was monitored*
- *and during function "IRQ_HartLayer2Timer " if timer was loaded with LAYER2_TIMER_GAPTIME_ERROR or with LAYER2_TIMER_START_BURST1 or LAYER2_TIMER_START_BURST2.*

During burst message preparation always one burst message for sending must be identified. The needed info for that burst message (see 2.4.3.2) must be provided for its generation. Starting with burst message 0 and afterwards for every following burst message the identification workflow is processed. The condition which identifies the to send message is met,

- *if the check of maximum update period detects that its counter has expired (counter is 0)*

- *if the check of update period detects that its counter has expired (counter is 0) and*
  - *if Continuous Mode is configured:*
    In Continuous Mode the communication of the burst message is cyclically working with the time of update period.

  - *if Windowed Mode is configured and the burst value gets out of limit window:*
    In Windowed Mode around the last communicated value a symmetric window is built by a lower and an upper limit value (burstMsgTriggerLowLimit; burstMsgTriggerHighLimit) which are calculated using the value "trigger level" belonging to the trigger parameters of the burst message. As long as the actual to burst value stays inside this limited window the communication works with cycle time of maximum update period. If the burst value gets out of the limit window the message is identified as to send. Around this last communicated value a new limit window is calculated and the before described algorithm is working again.

  - *if Rising Mode is configured and the burst value gets above the upper limit (burstMsgTriggerHighLimit):*
    In Rising Mode an upper limit value using "trigger level" is activated. As long as the actual to burst value stays below this limit value the communication works with cycle time of maximum update period. If the burst value gets above the limit the message is identified as to send.

  - *if Falling Mode is configured and the burst value gets below the lower limit (burstMsgTriggerLowLimit):*
    In Failing Mode a lower limit value using "trigger level" is activated. As long as the actual to burst value stays above this limit value the communication works with cycle time of maximum Update period. If the burst value gets above the limit the message is identified as to send.

  - *if On-Change Mode is configured and any change in message happened:*
    In On-Change Mode actual message info is compared with the info of the last burst communication (burstMsgLastSentData). If change in message info is detected earliest after time of update period the message info are send and the info stored burst message specific for the next compare. If no change is detected during time of maximum update period the stored message data are sent again.

    Following objects must be stored as compare data:
    - *Command 1*
      Primary Variable Units
      Primary Variable

    - *Command 2*
      Primary Variable Lopp Current
      Primary Variable Percent of Range

    - *Command 3*
      Primary variable Loop Current
      Primary Variable Units Code
      Primary Variable

- *Command 9*
  Slot 0: Device Variable Code
  Slot 0: Device Variable Classification
  Slot 0: Units Code
  Slot 0: Device Variable Value
  Slot 0: Device Variable Status

  To store data object values of Command 9 8 bytes are needed. This is the longest data length which is needed to define area object to store last sent values (burstMsgLastSentData) for every burst message.

- *if the check update period and maximum update period has not detected any expired counter for all burst message that burst message with the shortest update period (message number stored in HART object burstIdleMessage) is identified for send.*

2.5 Preparation of Burst Message

Now the preparation of a burst message is done in HARTLayer7CommandInterpreter if no STX was received. The preparation is managed by the flags “burst request built (burstMsgReqBuilt)”, “burst response built (burstMsgRespBuilt)” and “burst frame sent to first master (burstMsg1stSent). The same burst message is to send to both masters. Not before transmit start of burst message to second master the preparation of a new burst message is enabled.

The prepared burst message is always stored in burst buffer. In normal case message is copied from there before transmit.

After burst start just after sent of command 109 response transmit of burst message has to start. This is managed in IRQ_Transmit_LAYER2 where message is taken direct out of the burst buffer. After initialization/preparation Frame start bytes (delimiter, address) in burst buffer doesn’t change, therefore it is possible to start transmit before end of burst message generation. Direct after burst start part of this additional time (minimum 11 characters * character time 9.167 ms) is needed during command 9 burst preparation if a message frame for 8 device variables is to build

2.6 Burst after Device Restart

After device restart in Initialize_LAYER2 (layer2_HW.c) flag “1rst burst message after device restart (burstBufferInit)” is set. The start of the burst timer with RT2 was deleted, because preparation of burst message didn’t finish during time period of RT2. Now the timer is loaded with RT2 after burst message generation when burst was active before restart and message was generated first time after the restart (if Flag “1rst burst message after device restart (burstBufferInit)” is set) in StartResponse_LAYER2. Afterwards flag is reset.

Other Changes

2.6.1 Selectable HART revision

2.6.1.1 EDP300 Requirement

With a configuration action the revision of the HART interface can be selected as to work conform to Rev. 5.9 or conform to Rev. 7.2

2.6.1.2 Implementation

The storage specification of the HART object univCmdRev must be changed to NVRAM variable. Now it is possible to write the wanted HART Rev. No. (5 or 7) to univCmdRev when configuration action is done.

The HART Rev. No. in univCmdRev must be evaluated during command processing:

- Command 0 and 11
  In Rev. 5.9 the command 0/11 specific response contains only 10 response objects. This is different to Rev. 7.2 where the response contains 17 objects.

  In layer7.c “HARTLayer7CommandInterpreter” before call of function “HARTReadObjects” content of univCmdRev must be evaluated. If Rev. 5 is identified the input parameter of the function “objectCount” is set to 10 to truncate the preparation of the response data after the 10th Response Object.

- Command 6: Write Polling Address
  In Rev. 5.9 the command 6 specific request contains only the polling address object.

  Independent of HART Rev. if a request containing only polling address is received the logic to calculate Current Loop Mode must be carried out. If the polling address is zero the current signaling must be enabled. If the polling address is non-zero the current signaling must be disabled.

  In layer7.c “HARTLayer7CommandInetrpreter” the request length check must be expanded that also request which contain only polling address are correct. In case of “HART_Write” a special call of function “HARTWriteObjects” is done. Here only the Polling Address is prepared as parameter. Behind processing of function “HARTWriteObjects” if device works compliant to Rev. 7 value of Loop Current Mode is added to response.

  In layer7.c HARTCheckSecRespByte the logic to add status Loop Current Fixed is implemented

  In hart_overload.c “Put_HART” a special case pollAdr is added where the logic to calculate Loop Current Mode with its write to object “loopCurrentMode” is implemented

- *Command 15*
  In Rev. 5.9 the command 15 specific response contains only 7 response objects. This is different to Rev. 7.2 where the response contains 8 objects.

  In layer7.c “HARTLayer7CommandInterpreter” before call of HARTReadObjects to do processing for Command 15 content of univCmdRev must be evaluated. If Rev. 5 is identified the input parameter “objectCount” is set to 7 to truncate the preparation of the response data after the 7th Response Object.

Additional in hart_overload.c "Get_HART" if access to object "pvt_Label_Dist_Code" and device works compliant to Rev. 7 out parameter in "ptrValue" is overwritten with value "250 (Not used)".

- *Command 38*
  In Rev. 5.9 the command 38 specific request and response contain no data. This is different to Rev. 7.2 where in request the Configuration Change Counter for comparison is contained. This value is also returned as response data.

  The ACTION Function "resetConfigFlag_HART" in "Hart_action_fct.c" compares the counter value out of the request with the counter value in device if the device works compliant to Rev. 7. If both are equal the master specific configuration change flag in HART object "configurationFlags" is reset. The reset algorithm and store of status is done with function "HARTPutAttribute" with parameter "configurationFlags".

  If counter are not equal RC #9 (Configuration Change Counter Mismatch) without response data is returned. Otherwise response is prepared.

  If device works compliant to Rev. 5 always the master specific configuration change flag is reset (see above). Response without data is prepared.

- *Command 48*
  Between HART Revision 5.9 and 7.2 the objects of the response data have changed. So with Entry Tool specific objects are to configure with type "SLOT" and slot code 5 and 7 to implement the revision specific objects. Additional in 7.2 a compare of sent request data with the prepared response data is done. Only if no difference is detect signalization "More Status Available" is switched off. In 5.9 no request data is sent and the signalization "More Status Available" is always switched off.

  In ACTION function "resetMoreStatusAvailable_HART" in "Hart_action_fct.c" the revision specific response data is prepared. If device works compliant to Rev. 7 the compare of request with prepared response data is done. If no difference detected or device works compliant to Rev. 5 the value of subsystem "DiagnosisServices" object "diagnosisCounter" is stored in subsystem "HART" object "diagCounterMoreStatusAvailable".

2.6.2 Error Correction as result of Test DLL042

2.6.2.1 Known problem

If Extended Command is identified and the additional command number is not valid or the request data are not complete then Response Data Bytes are returned.

Conformance Test DLL042 signalizes that in this case no data bytes are to response.

2.6.2.2 Implementation

A Change in layer7.c "HARTLayer7CommandInterpreter" is necessary: in error case the default number of data bytes must be set back to 2! This prevents the transmission of the Command Number prepared as data bytes in response.

3 To Do List

- *Search changes out of Bug Tracking database for TORNADO*
- *Implementation of the changes*

- *Performance Measurement of HART7 implementation*
- *Modultest of all changes*
- *DLL, UAL and CAL tests with HCF Test Kit*
- *Clarification of principles with HCF Special Burst tests*

Documentation of SW implementations

4 Log book/Logbuch

Chapter	Change	Date	Name