


References: In 'HART Field Communications Protocol Specification Revision 5.9' listed Specifications for HART Revision 5.9, Test Spec. (including test cases, test method, test result, ...), etc.

Review Document

Author Horst Seele **Date:** 2005-12-13


Remarks:

None

	Software Subsystem Design Description HART 5.9 Subsystem				SDD
Responsibility:	Date:	Language:	Filing system :	Revision:	Page:
APR/I	2005-12-13	en		1.0	2/20
Issued by:	Approved:	Released:	Area of validity:		
PSM/5K AP2.2			APR/I		

Contents

1	Context	3
2	Data Sheet.....	3
3	Detailed Description	7
3.1	Static Modelling	7
3.2	Dynamic Modelling	8
3.3	Class Design	9
3.4	Detailed Design	9
3.4.1	HARTLayer7CommandInterpreter, HARTCommandInterpreterEXE	9
3.4.2	HARTSpecialReadWriteObjects	11
3.4.3	HARTReadObjects	12
3.4.4	HARTGetAttribute	13
3.4.5	HARTGetObjects	13
3.4.6	HARTWriteObjects.....	14
3.4.7	HARTPutAttribute	15
3.4.8	HARTPutObjects.....	15
3.4.9	HARTConvertPASCII.....	15
3.4.10	HARTConvertType.....	16
3.4.11	HARTCheckSecRespByte	16
3.4.12	HARTFindCmdTableIndex.....	16
3.4.13	PrepareBurstBuf_LAYER7.....	17
3.4.14	Put_HART	17
3.4.15	resetConfigFlag.....	17
3.4.16	StartSelfTest	17
3.4.17	FormatEEPROM	18
3.4.18	BurnFactoryDefaultData	18
3.4.19	HARTCanAccess	18
3.5	Design Decisions and Limitations	19
3.6	Hardware Dependencies.....	19
3.7	Data Object Description	19
3.7.1	T_DATA_OBJ_ADDR	19
3.7.2	T_DATA_OBJ_CONF	19
3.8	Error Handling	19
4	Revision Chart	20

	Software Subsystem Design Description HART 5.9 Subsystem			SDD
Responsibility:	Date:	Language:	Filing system :	Revision: Page:
APR/I	2005-12-13	en	1.0	3/20
Issued by:	Approved:	Released:	Area of validity:	
PSM/5K AP2.2			APR/I	

1 Context

This Document is a representation of analysis, planing, implementation and decision-making. It consists of design information for the subsystem <HART>. One of its major goals is to support reuse of the described subsystem.


The document is divided into two main parts. First there is the data sheet in chapter one. This chapter is intended for developers that want to reuse the subsystem. Therefore it lists important issues from a black-box view. The second chapter describes the design in more detail. This is the right place to look for developer that have to maintain or expand the subsystem.

Besides this documents there are a couple of other documents that describe the requirements, test cases and the complete system the subsystem is used in. Links to those documents can be found at the end of the data sheet.


2 Data Sheet

This chapter gives an overview about all important facts of the subsystem. It can be used by a developer who would like to reuse this subsystem.


<i>Category</i>	<i>Item</i>	<i>Description</i>
Development	Version / Status	1.0.0 / Released
	Known Bugs	None
	Planned Improvements	-
HW-Platform	Type	M16C/28
	Clocking	> 1 MHz
SW-Development Environment	Compiler	IAR M16C/C++ Compiler Version 2.12E, no optimization
	Operating System	Segger for M16C emBOS 3.20D
	Case / Code-Generation Tool	HART Command generation tool 'HARTGen'.
Required Resources	Operating System	-
	HW	LAYER1: TB0 + TB0IN, TB1 + TB1IN, TB2, INT5 + debounce-filter, P8.3, P8.4, P10.0-7, P2.2-7, P7.2, P8.7, P9.2, P9.3 DMA0, DMA1, Interrupt-level 7 exclusiv LAYER2: UART2, Timer A1

	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: 4/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I


	RAM	<p>20 Bytes (FARDATA)</p> <p>72 Bytes placed in HART_STATIC_RARE_1, HART_STATIC_FREQUENT_NO_DEFAULT_1, HART_STATIC_FREQUENT_CONSTANT_DEFAULT_1, HART_STATIC_FREQUENT_STATIC_DEFAULT_1, HART_STATIC_FREQUENT_STATIC_DEFAULT_1_DEFAULT (NEARDATA)</p> <p>LAYER1:</p> <p>4 Bytes placed in STATIC_INTERRUPT_DATA (NEARDATA)</p> <p>LAYER2:</p> <p>55 Bytes (NEARDATA)</p> <p>121 Bytes placed in STATIC_INTERRUPT_DATA (NEARDATA)</p> <p>LAYER7:</p> <p>272 Bytes placed in HART_LAYER7_DATA (FARDATA)</p>
	NVRAM	-
	ROM	<p>685 Bytes (CODE), 1070 Bytes (FARCONST)</p> <p>LAYER1:</p> <p>926 Bytes (CODE), 28 Bytes (FARCONST)</p> <p>6 Byte interrupt vector table</p> <p>LAYER2:</p> <p>3845 Bytes (CODE), 45 Bytes (FARCONST)</p> <p>9 Byte interrupt vector table</p> <p>LAYER7:</p> <p>6027 Bytes (CODE), 9 Bytes (FARCONST)</p> <p>+ cmddef.c (placed in HART_COMMAND_TABLE_DATA as FARCONST)</p>
	Execution Time	<p>Not applicable.</p> <p>LAYER1:</p> <p>approx. 9% CPU Calculation Time (1 MHz)</p> <p>LAYER2:</p> <p>Average: 300µs; max.: ~1ms (1MHz)</p> <p>LAYER7:</p> <p>Depends on executed command.</p>

	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 5/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I

	Special HW	LAYER1: External hardware (e.g. filter) are necessary for the soft-modem.
	Subsystems	Coordinator = subsystem_idx.h (Declaration of 'GetSubsystemPtr' + Subsystem ID's), coordinator.h (Declaration of 'FormatAllData'). Physical Block = physicalblock.h (Declaration of T_UNIT function 'ExitSRV'). LAYER1: System = system.h (clock definitions). LAYER2: System = system.h (clock definitions). Uses some objects of HART constructor. LAYER7: CurrentOut = current_out_idx.h (Declaration of loop current mode object and diagnosis object), current_out.h (current out bit definitions). Coordinator = subsystem_idx.h (Declaration of 'GetSubsystemPtr' + Subsystem ID's). Physical Block = physicalblock.h (Declaration of T_UNIT function 'PasswordIsSetSRV') Diagnosis = diagnosis_idx.h (diagnosis object ID), diagnosis_type.h (Declaration of diagnosis object) Others as linked by command definition in cmddef.c
	Data Objects	-
Standards	Safety	For this subsystem the following safety issues are required: - PCLint level 3 - ABB Coding Conventions for embedded software, V1.8 - Code Review - Software Subsystem Safety Analysis for Write Protection LAYER1, LAYER2: This subsystem has to fulfill the safety standard SIL level 2.
	Other	

	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 6/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I

Documentation	Requirements / Use Cases	HART HCF_SPEC for Protocol Revision 5.9 LAYER1: HCF_SPEC-54, Revision 8.1, FSK Physical Layer Specification LAYER2: HCF_SPEC-81, Revision 8.0, Data Link Layer Specification
	Public Interface Description	hart.h, hart_idx.h LAYER1: layer1.h LAYER2: layer2.h LAYER7: layer7.h
	Test Specification	ModuleTest see code LAYER1: HCF_TEST-2, Revision 2.2, FSK Physical Layer Test Specification LAYER2: HCF_TEST-1, Revision 2.1, Slave Data Link Layer, Test Specification. LAYER7: HCF_TEST-3 Slave Universal Command, Test Specification, HCF_TEST-4 Slave Common Practice Command, Test Specification
Special		<ul style="list-style-type: none"> a) HART commands may have a maximum of 40 request/response data bytes. b) In hart_arm.c the write-protect detection function HARTCanAccess must be customized to the application. c) A second layer 2 can be handled by the HART subsystem if in layer2.h activated. layer2_2_template.h is a example with needed definitions and declarations. d) Allowed burst commands have to be defined in hart_constructor.c (Done by Code generation tool 'HARTGen'). e) Only the HART subsystem layer 2 is able to work correct in burst mode not the second layer 2. Other restrictions of the second layer 2 exists if the same configurable objects (e. g. polling address) are used as by the first layer 2.

	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 7/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I

3 Detailed Description

For **Layer 1** and **Layer 2** see belonging Design Descriptions there.

3.1 Static Modelling

The static design of the HART Layer 7 is shown in the following diagram. HART as a subsystem inherits from T_UNIT. The parameters to be accessible via T_UNIT interface are created here. Two special T_DATA_OBJ have been defined in order to accommodate the address setting and configuration changed flag reading and writing. Due to the fact, that they are peculiar to this subsystem, they have not been made public. HART defines four action objects:


- resetConfigFlag
- BurnFactoryDefaultData
- FormatEEPROM
- StartSelfTest

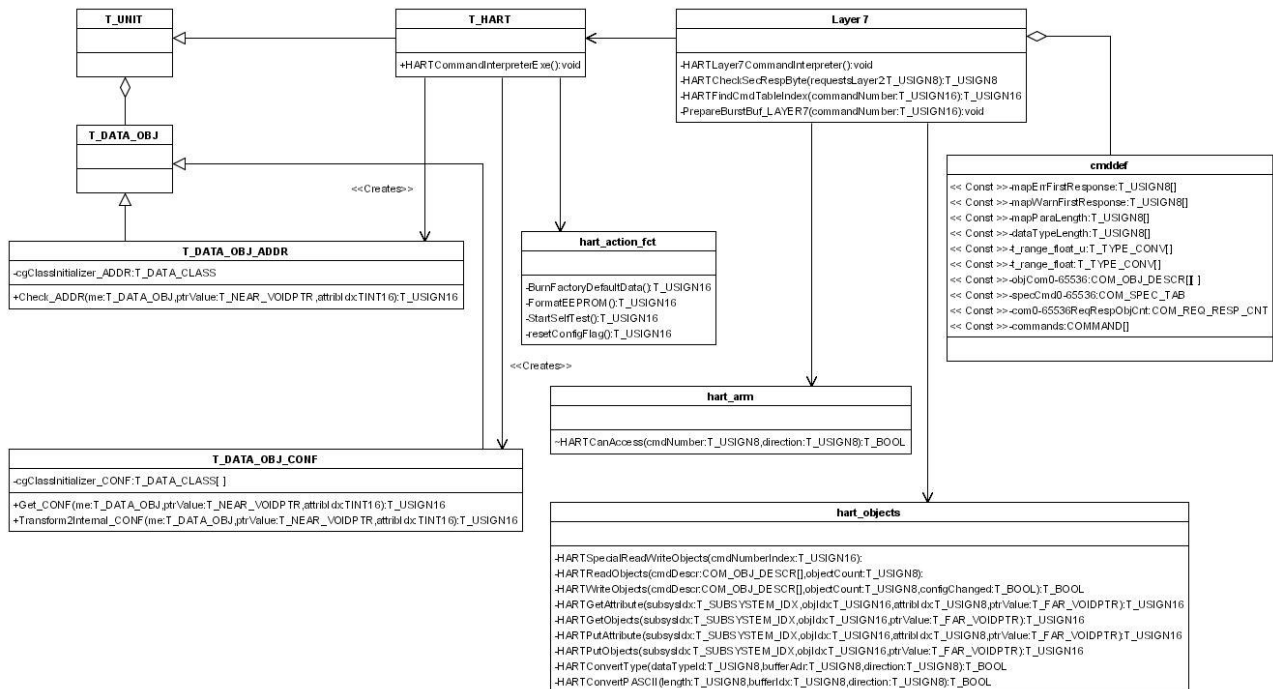
These functions are defined in hart_action_fct.c and are referenced in the dedicated constructor inside the object list. Layer 7 is only one part of this subsystem. All functions that belong to the layer 7 themselves reside in layer7.c, hart_objects.c and hart_arm.c. The class aggregates cmddef.c. This file mainly contains the command definitions and the command table that are referenced from the layer 7.

Most of this file is generated using the HART generation tool 'HARTGen':

- The table 'commands' (struct COMMAND) lists all HART commands implemented in the device.
- Normal HART commands (Action commands with 0 request data bytes, read commands with 0 request data bytes and write commands with same data length/meaning in request and response) needs only a array of descriptions (struct COM_OBJ_DESCR) of all command objects.
- Special HART commands (e. g. different request (not equal 0) and response data byte length, data field with different meaning in dependence of request slot code value, ...) need a more complex command description (struct COM_SPEC_TAB).
1 array with object descriptions of the request and 1 array with object descriptions of the response are needed (struct COM_OBJ_DESCR). Every array contains descriptions of the maximum request/response length. Arrays of a slot command (a slot selects the objects of the data field) contain default object descriptions.
One table contains the information how many request/response objects must be handled in dependence of the received data bytes number (struct COM_REQ_RESP_CNT).
A slot command needs the information which objects belong to which slot code. This information is written in a table with object descriptions of every slot code (struct COM_OBJ_DESCR).

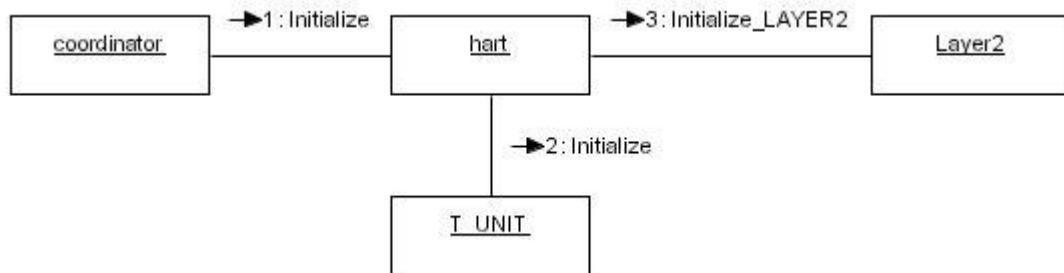
Apart from generated code, the first response mapping from T_UNIT return codes (mapErrFirstResponse for errors, mapWarnFirstResponse for warnings) and descriptors for special data structs (t_range_flt_u) are located in this file.

	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: 8/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity: APR/I	



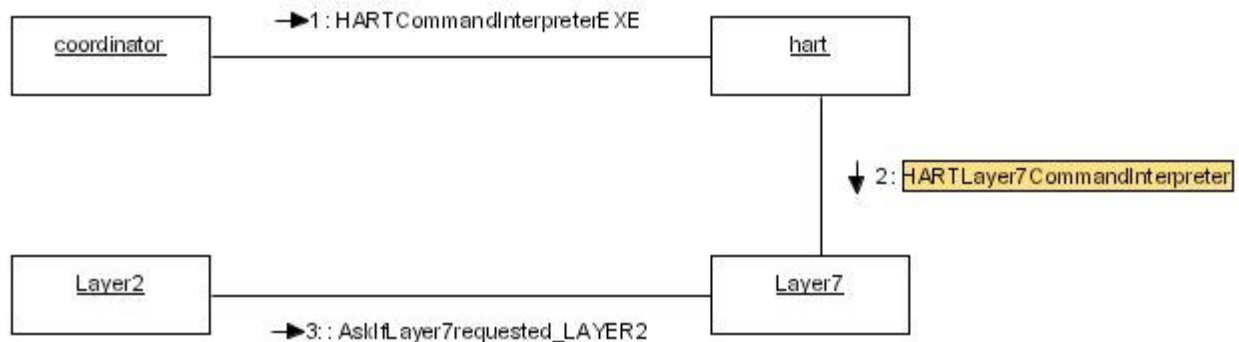
3.2 Dynamic Modelling

The collaboration diagrams below show the dynamic interaction of the classes that are referenced by the HART Layer7.



Collaboration diagram for hart.Initialize()

ABB	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 9/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I



Collaboration diagram for hart.HARTCommandInterpreterEXE()

3.3 Class Design

Due to the simplicity of this design please refer to section 'Static Modelling'.

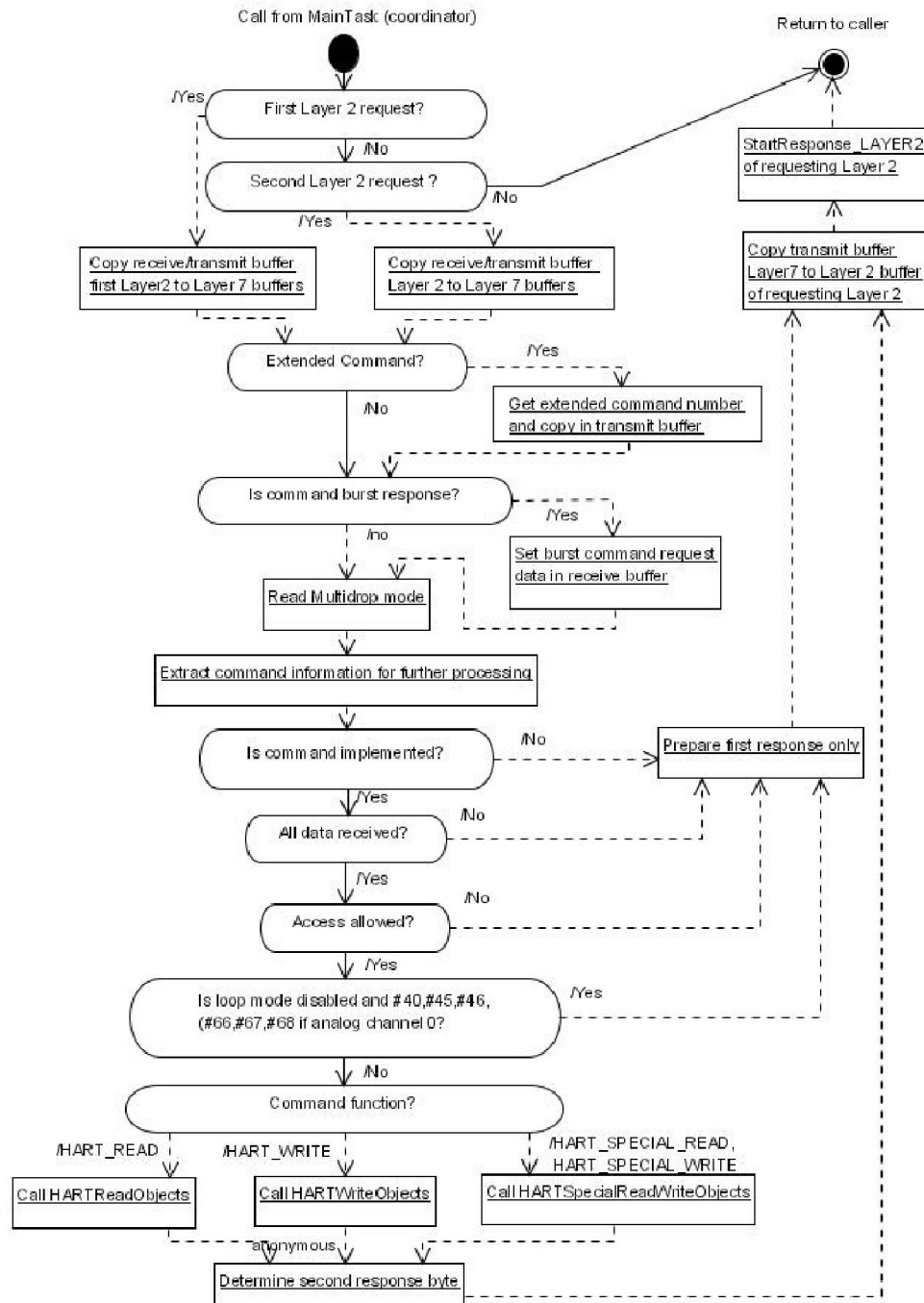
3.4 Detailed Design

3.4.1 HARTLayer7CommandInterpreter, HARTCommandInterpreterEXE

Param: void
Return: void

Command interpreter implementation. The function **HARTCommandInterpreterEXE** is called from the coordinator. It calls the Layer 7 function **HARTLayer7CommandInterpreter**. This function will return immediately if no request has been placed (**AskIfLayer7requested_LAYER2()**, **AskIfLayer7requested_LAYER2_2()**). In case a command has been received, the command interpreter does the following activities:

Responsibility:	Date:	Language:	Filing system :	Revision:	Page:
APR/I	2005-12-13	en		1.0	10/20
Issued by:	Approved:	Released:	Area of validity:		
PSM/5K AP2.2			APR/I		



If both Layer 2 have a request at the same time the first layer 2 request will be handled first. At the next calling of **HARTCommandInterpreterEXE** the second layer 2 request is handled (if no further request from the first layer 2).

Responsibility:	Date:	Language:	Filing system :	Revision:	Page:
APR/I	2005-12-13	en		1.0	11/20
Issued by:	Approved:	Released:	Area of validity:		
PSM/5K AP2.2			APR/I		

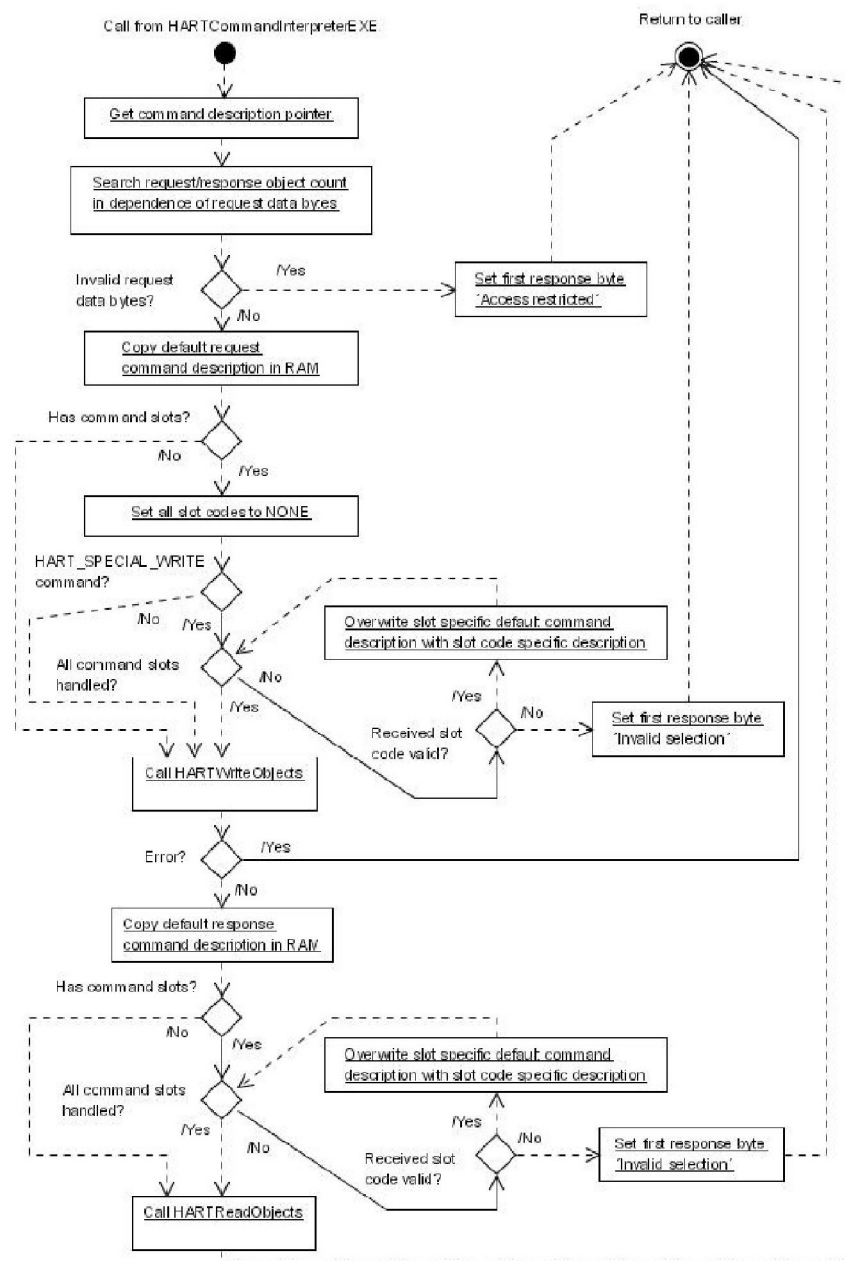
3.4.2 HARTSpecialReadWriteObjects

Param: TUSIGN16 commandNumberIndex = Index of command in command table.
Return: void

Handles special HART commands where the request/response data area has not always the same length and/or meaning.

Response code HART_RESP_ERR_ACCESS_RESTRICTED is set if a wrong number of request data bytes is received.

Response code HART_RESP_ERR_INVALID_SELECTION is set if a wrong slot code is received.

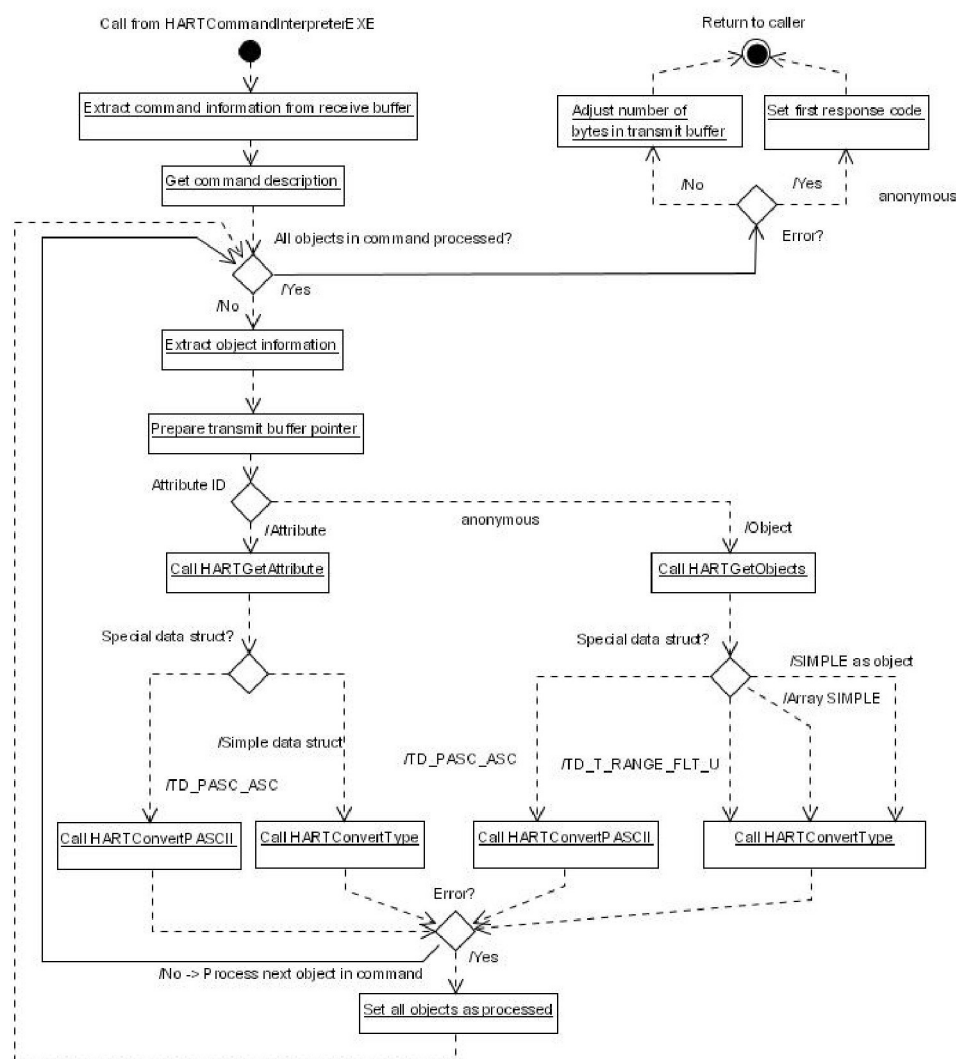



Responsibility:	Date:	Language:	Filing system :	Revision:	Page:
APR/I	2005-12-13	en		1.0	12/20
Issued by:	Approved:	Released:	Area of validity:		
PSM/5K AP2.2			APR/I		

3.4.3 HARTReadObjects

Param1: COM_OBJ_DESCR cmdDescr = Pointer to command structure.
Param2: TUSIGN8 objectCount = Number of objects in response.
Return: void

Reads all parameters in a command. The function distinguishes between Attributes and Objects. Arrays are only supported as SIMPLE data types and must be handled as Objects in the command description. The function provides a special handling for data structs that must be enhanced if new types are added.



	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 13/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I

3.4.4 HARTGetAttribute

Param1: T_SUBSYSTEM_IDX subsysIdx = Subsystem index
Param2: TUSIGN16 objIdx = Object index
Param3: TUSIGN8 attribIdx = Attribute index.
Param4: void *ptrValue = Value pointer
Return: TUSIGN16 T_DATA_OBJ error codes

This function is called when a specific attribute is to be read. The function is given the identification of the subsystem, the object index and the attribute index. The caller will find the desired value using the value pointer.

The T_UNIT interface function “GetAttribute” will be called. The response coming from this call will be returned to the caller of this function.

Due to the simplicity of this function, no diagram has been prepared.

3.4.5 HARTGetObjects

Param1: T_SUBSYSTEM_IDX subsysIdx = Subsystem index
Param2: TUSIGN16 objIdx = Object index
Param3: void *ptrValue = Value pointer
Return: TUSIGN16 T_DATA_OBJ error codes

This function is called when an object is to be read. The function is given the identification of the subsystem and the object index. The caller will find the desired value using the value pointer.

The T_UNIT interface function “GetAttribute” will be called. The response coming from this call will be returned to the caller of this function.

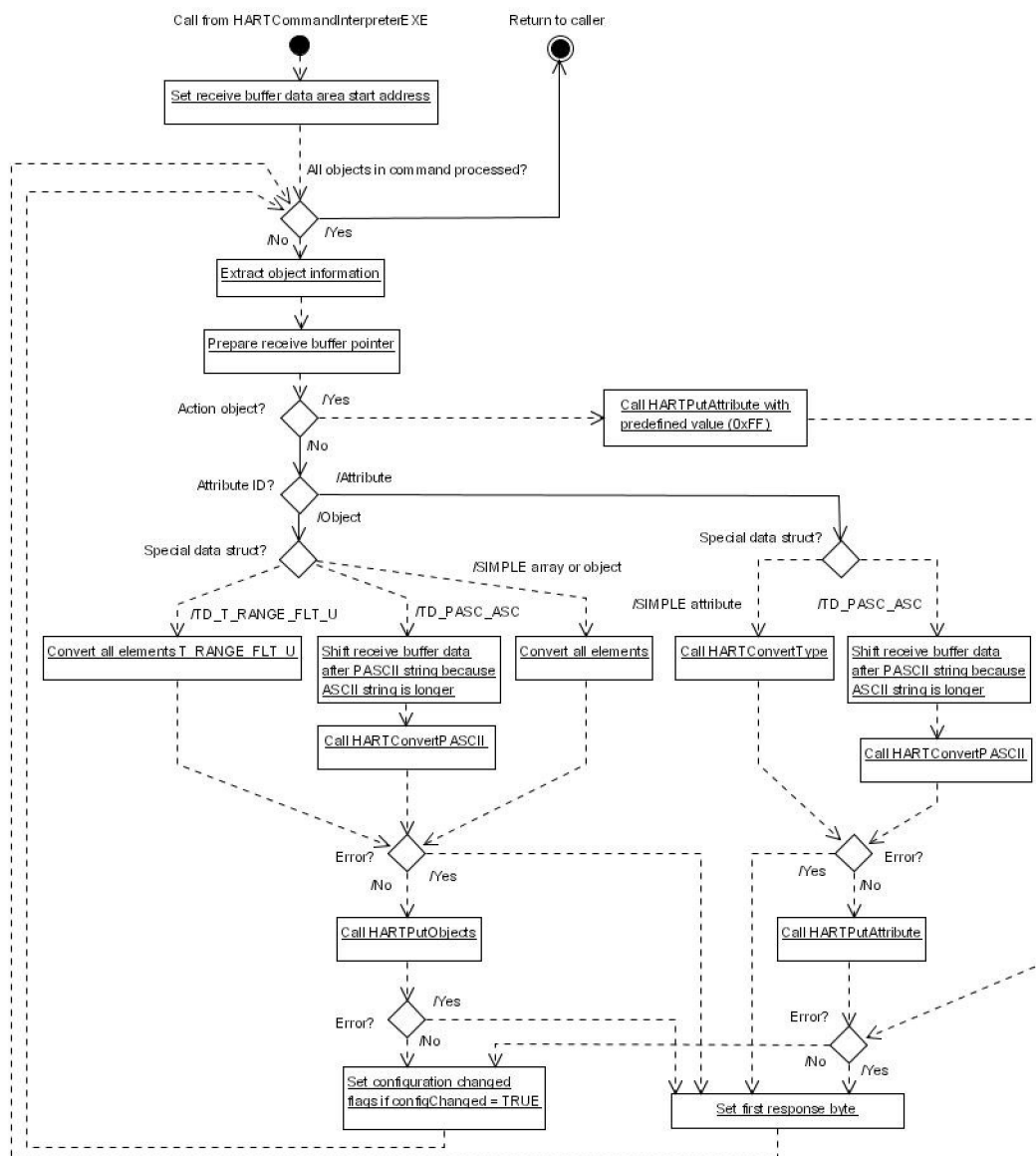
Due to the simplicity of this function, no diagram has been prepared.

ABB	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 14/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I


3.4.6 HARTWriteObjects

Param1: COM_OBJ_DESCR cmdDescr = Pointer to command structure.
Param2: TUSIGN8 objectCount = Number of objects in response.
Param3: TBOOL configChanged = Indicates if command changes config or not.
Return: TBOOL eTRUE = one or more errors occurred, eFALSE = no error occurred.

Writes all parameters in a command. The function distinguishes between Attributes and Objects. Arrays are only supported as SIMPLE data types and must be handled as Objects in the command description. The function provides a special handling for data structs that must be enhanced if new types are added.



Also writing a object of a command is not possible all other writeable objects are stored. This is not conform to the HART specification but accepted, because using the PUT function of the framework.

	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 15/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I

3.4.7 HARTPutAttribute

Param1: T_SUBSYSTEM_IDX subsysIdx = Subsystem index.
Param2: TUSIGN16 objIdx = Object index.
Param3: TUSIGN8 attribIdx = Attribute index.
Param4: void *ptrValue = Value pointer.
Return: TUSIGN16 T_DATA_OBJ error codes.

This function is called when a specific attribute is to be written. The function is given the identification of the subsystem, the object index and the attribute index. The value intended to be written must have been stored before at the location of value pointer.

The T_UNIT interface function “WriteAttribute” will be called. The response coming from this call will be returned to the caller of this function.

Due to the simplicity of this function, no diagram has been prepared.

3.4.8 HARTPutObjects

Param1: T_SUBSYSTEM_IDX subsysIdx = Subsystem index.
Param2: TUSIGN16 objIdx = Object index.
Param3: void *ptrValue = Value pointer
Return: TUSIGN16 T_DATA_OBJ error codes

This function is called when an object is to be written. The function is given the identification of the subsystem and the object index. The value intended to be written must have been stored before at the location of value pointer.

The T_UNIT interface function “WriteObject” will be called. The response coming from this call will be returned to the caller of this function.


Due to the simplicity of this function, no diagram has been prepared.

3.4.9 HARTConvertPASCII

Param1: TUSIGN8 length = HART commando length of data to be converted
Param2: TUSIGN8 bufferIdx = index in transmit or receive buffer where to read/write the data
Param3: TUSIGN8 direction = read or write
Return: TBOOL eTRUE = Conversion ok, eFALSE = conversion not possible

This function converts packed ASCII data to ASCII data (if write) or vice versa (if read). In read direction, the transmit buffer provides the source for the data to be converted. In write direction the receive buffer provides the source. Conversion will be done inside the addressed buffer itself at a specific position indicated by the parameter bufferIdx. The conversion is done if the packed ASCII string length is multiple of 3 bytes. Because 3 packed ASCII characters result to 4 ASCII characters there must be enough free space in the buffer for converting the string.

Due to the simplicity of this function, no diagram has been prepared.

	Software Subsystem Design Description HART 5.9 Subsystem			SDD
Responsibility:	Date:	Language:	Filing system :	Revision: Page:
APR/I	2005-12-13	en	1.0	16/20
Issued by:	Approved:	Released:	Area of validity:	
PSM/5K AP2.2			APR/I	

3.4.10 HARTConvertType

Param1: TUSIGN8 dataTypeld = type of data to be converted
Param2: TUSIGN8 bufferIdx = index in transmit or receive buffer where to read/write the data
Param3: TUSIGN8 direction = read or write
Return: TBOOL eTRUE = Conversion ok, eFALSE = conversion not possible

This function converts data of two or four bytes length. The conversion changes the byte order according to the direction. In read direction, the transmit buffer provides the source for the data to be converted. In write direction the receive buffer provides the source. Conversion will be done inside the addressed buffer itself at a specific position indicated by the parameter bufferIdx. The data type ID provides the length of data to be converted.

The conversion format is big endian to little endian and vice versa.

Due to the simplicity of this function, no diagram has been prepared.

3.4.11 HARTCheckSecRespByte

Param1: TUSIGN8 requestsLayer2 = Flags which Layer 2 requests Layer 7.
Return: TUSIGN8 second response byte.

This function is called from HARTCommandInterpreterEXE() after a read or write command has been placed. The function returns the bit-coded second response byte according to the table below.


Second Byte	Description	Mapping
Bit 7	Field Device Malfunction	Diagnosis Byte 4 (Operation) Bit 0 (Failure)
Bit 6	Configuration Changed	HARTGetObjects (see also chapter 'T_DATA_OBJ_CONF')
Bit 5	Cold Start	Self determined in dependence which Master (primary/secondary) from which Layer 2 send the request.
Bit 4	More Status Available	All bits out of 8 bytes in diagnosis are 0
Bit 3	Primary Variable Analog Output Fixed	CURRENT_OUT_IDX,CURR_IDX_diagnosis -> diagnosis if (diagnosis & CURR_DIAG_FIXEDMODE)
Bit 2	Primary Variable Analog Output Saturated Only if not in multidrop mode!	CURRENT_OUT_IDX,CURR_IDX_diagnosis -> diagnosis if (diagnosis & (CURR_DIAG_SPANLIMIT_LO CURR_DIAG_SPANLIMIT_HI))
Bit 1	Non-Primary Variable Out of Limits	Diagnosis Byte 1 (Process) Bit 2 (SV out of limits)
Bit 0	Primary Variable Out of Limits	Diagnosis Byte 1 (Process) Bit 1 (PV out of limits)

Due to the simplicity of this function, no diagram has been prepared.

3.4.12 HARTFindCmdTableIndex

Param1: TUSIGN16 cmdNumber = command number.
Return: TUSIGN16 Index in command table or invalid index if not found.

Searches for the index of the command in the 'commands' table in cmddef.c. An invalid index will be returned if the command is not implemented.

	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 17/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I

Due to the simplicity of this function, no diagram has been prepared.

3.4.13 PrepareBurstBuf_LAYER7

Param1: TUSIGN16 cmdNumber = command number.
Return: void.

If Command #9 or #33 are the Burst commands, this function writes the burst slot codes (previously written with command #107) as request data bytes in the receive buffer. Only used burst slots will be requested. With this information the command interpreter generates the burst response.

Due to the simplicity of this function, no diagram has been prepared.

3.4.14 Put_HART

Param1: TUSIGN16 objIdx = object index.
Param2: TINT16 attribIdx = attribute index.
Param3: void *ptrValue = pointer to object value.
Return: TUSIGN16 T_UNIT error code.

This function calls the standard T_UNIT PUT function. A special handling is done if the burst slot codes with command #107 are written. In this case the burst slots #1 - #3 are set to not used if burst slot #0 is written. This is necessary because not all burst slots must be written with command #107.

Due to the simplicity of this function, no diagram has been prepared.

3.4.15 resetConfigFlag

Param1: void
Return: TUSIGN16 T_UNIT error code.

This function will be called in response to command #38. It resets the configuration changed flag of the requesting master. (see also chapter 'T_DATA_OBJ_CONF')

Due to the simplicity of this function, no diagram has been prepared.

3.4.16 StartSelfTest

Param1: void
Return: TUSIGN16 Always OK.

This function will be called in response to command #41. As the diagnostic informations are determined in a background task, no explicit function call is needed in order to get such information out of #48. For this reason this functions simply returns OK with every call.

Due to the simplicity of this function, no diagram has been prepared.

ABB	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 18/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity: APR/I	

3.4.17 FormatEEPROM

Param1: void
Return: TUSIGN16 Always OK.

This function can be called from the production with a non-public command. The function calls the public coordinator function FormatAllData (). This calls subsequently the T_UNIT interface function FormatNV() for all subsystems. Afterwards, a device reset will be performed. The function call will respond with OK first, before the reset is carried out.

Due to the simplicity of this function, no diagram has been prepared.

3.4.18 BurnFactoryDefaultData

Param1: void
Return: TUSIGN16 T_UNIT error code.

This function can be called from the production with a non-public command. The function calls the T_UNIT interface function SetCustomDefault () for each subsystem. Modifications to custom parameters will be stored permanently to NV RAM.

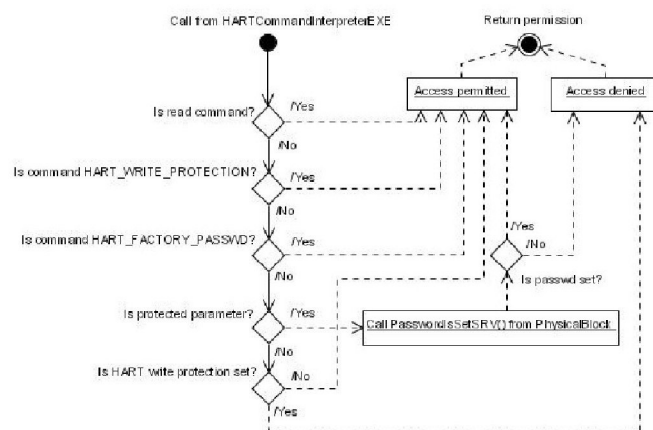
On the event of a error code coming from a subsystem being not OK, the function will forward this code to the caller.


Due to the simplicity of this function, no diagram has been prepared.

3.4.19 HARTCanAccess

Param1: TUSIGN16 cmdNumber = Command number presently read or written to.
Param2: TUSIGN8 direction = read or write.
Return: TBOOL eTRUE = Access accepted, eFALSE = Access denied.

The HART write protection is the only safety critical part in the layer 7. Access rights are encapsulated in a separate file (hart_arm.c). The function **HARTCanAccess()** return TRUE if access is allowed and FALSE if access is denied. The following diagram gives details:



	Software Subsystem Design Description HART 5.9 Subsystem		SDD	
Responsibility: APR/I	Date: 2005-12-13	Language: en	Filing system : 1.0	Revision: Page: 19/20
Issued by: PSM/5K AP2.2	Approved:	Released:	Area of validity:	APR/I

3.5 Design Decisions and Limitations

The design of the layer 7 class provides of a high degree of reuse. The parts that are device dependent have been placed in a separate file (cmddef.c) and can be generated using the HART Command generation tool 'HARTGen'.

3.6 Hardware Dependencies

See chapter 'Data Sheet.'

3.7 Data Object Description

3.7.1 T_DATA_OBJ_ADDR

The polling address is an unsigned 8 integer with valid range of 0 ...15. An address > 0 sets the current of the CurrentOut subsystem to a fixed value, while the address 0 enables the measurement dependent current.

3.7.2 T_DATA_OBJ_CONF

The configurationFlags object contains the actual state of the HART primary master/secondary master configuration changed flags of both Layer 2.

In a command response the belonging actual configuration changed flag state of the master is set in the second response byte.


All flags are set if one object of a command with set configuration flag attribut in the command description (in cmddef.c) is successful written.

The belonging configuration changed flag is reset if a master sends command #38. The configuration changed flag state of the other masters are not influenced.

3.8 Error Handling

Invalid received HART frames (e. g. communication errors defined by HART specification) will be detected by Layer 2 or Layer 7. The error response is generated inside the HART subsystem and send to the requesting master.

See also LAYER 1 and LAYER 2 design descriptions.

	Software Subsystem Design Description HART 5.9 Subsystem				SDD
Responsibility:	Date:	Language:	Filing system :	Revision:	Page:
APR/I	2005-12-13	en		1.0	20/20
Issued by:	Approved:	Released:			Area of validity:
PSM/5K AP2.2					APR/I

4 Revision Chart

Rev.	Description of Version/Changes	Primary Author(s)	Date
1.0	new	Horst Seele	2005-12-13

ABB		Software Design Review			SDR	
Responsibility:	Date:	Language:	Filing system :	Revision:	Page:	
APR/I	2006-03-23	en	-	1.0	1/2	
Template Issued by:	Template Approved:	Template Released:			Area of validity:	
PSM AP2.2	-	10/2004			APR/I	

Project:	HART 5.9,6 Subsystem Extension
Document under Review:	Software Subsystem Design Description HART 5.9 Subsystem
Revision:	1.0
Review Date:	2006-03-23

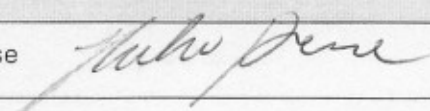
Review-Participant:

Place	Dept.	Name
MI	MIMS	Heiko Kresse
MI	MIMS	Horst Seele

Decision of the Review:

Decision	next steps
<input checked="" type="checkbox"/> Inspection passed without restrictions	Phase finished
<input type="checkbox"/> Inspection passed with restrictions	some changes must be done
<input type="checkbox"/> Inspection not passed	Inspection must be repeated

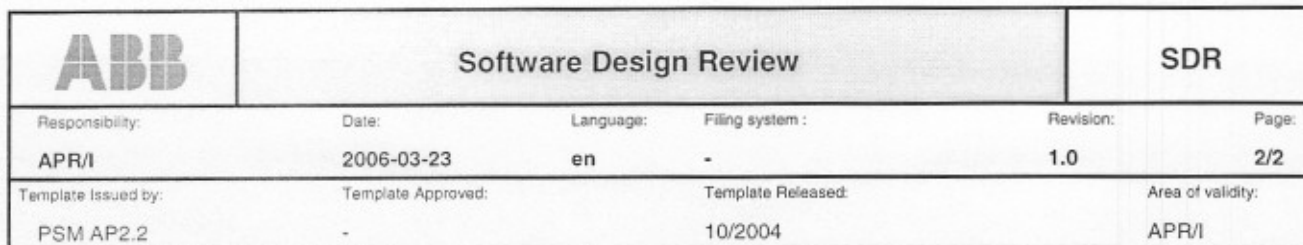
Changes are proved: The Reviewer confirms that all changes are done:

proved Rev:	Date:	Reviewer:
1.0	2006-03-23	Heiko Kresse 

Check list:

		yes	no
1.	Is the software architecture distinct and documented?	✓	
2.	Fit the modules together?	✓	
3.	Are complex algorithms/procedures explained?	✓	
4.	Is a strategy for error handling designated?	✓	
5.	Is the configuration management system well prepared?	✓	
6.	Are all open issues transferred to the defects table?	✓	

Remarks:

[illegible]