# IEC TR 62453-52-90

Edition 1.0 2017-06

# TECHNICAL REPORT

colour inside

## Field device tool (FDT) interface specification – Part 52-90: Communication implementation for common language infrastructure – IEC 61784 CPF 9

IEC TR 62453-52-90:2017-06(en)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch


**About the IEC**

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

**About IEC publications**

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

**IEC Catalogue - webstore.iec.ch/catalogue**

The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

**IEC publications search - www.iec.ch/searchpub**

The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

**IEC Just Published - webstore.iec.ch/justpublished**

Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

**Electropedia - www.electropedia.org**

The world's leading online dictionary of electronic and electrical terms containing 20 000 terms and definitions in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

**IEC Glossary - std.iec.ch/glossary**

65 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

**IEC Customer Service Centre - webstore.iec.ch/csc**

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

# IEC TR 62453-52-90

Edition 1.0 2017-06

# TECHNICAL REPORT

colour inside

## Field device tool (FDT) interface specification – Part 52-90: Communication implementation for common language infrastructure – IEC 61784 CPF 9

INTERNATIONAL ELECTROTECHNICAL COMMISSION

ICS 25.040.40; 35.100.05; 35.110

ISBN 978-2-8322-4333-6

# CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

# FIELD DEVICE TOOL (FDT) INTERFACE SPECIFICATION –

## Part 52-90: Communication implementation for common language infrastructure – IEC 61784 CPF 9

## FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
6) All users should ensure that they have the latest edition of this publication.
7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

The main task of IEC technical committees is to prepare International Standards. However, a technical committee may propose the publication of a technical report when it has collected data of a different kind from that which is normally published as an International Standard, for example "state of the art".

IEC TR 62453-52-90, which is a technical report, has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

Each part of the IEC 62453-52-xy series is intended to be read in conjunction with its corresponding part in the IEC 62453-3xy series. This document corresponds to IEC 62453-309.

The text of this technical report is based on the following documents:

| Enquiry draft | Report on voting |
|---|---|
| 65E/440/DTR | 65E/514/RVC |

Full information on the voting for the approval of this technical report can be found in the report on voting indicated in the above table.

This document has been drafted in accordance with the ISO/IEC Directives, Part 2.

The list of all parts of the IEC 62453 series, under the general title *Field device tool (FDT) interface specification*, can be found on the IEC website.

The committee has decided that the contents of this document will remain unchanged until the stability date indicated on the IEC website under "http://webstore.iec.ch" in the data related to the specific document. At this date, the document will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

**IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.**

# INTRODUCTION

This part of IEC 62453 is an interface specification for developers of Field Device Tool (FDT) components for function control and data access within a client/server architecture. The specification is a result of an analysis and design process to develop standard interfaces to facilitate the development of servers and clients by multiple vendors that need to interoperate seamlessly.

With the integration of fieldbuses into control systems, there are a few other tasks which need to be performed. In addition to fieldbus- and device-specific tools, there is a need to integrate these tools into higher-level system-wide planning or engineering tools. In particular, for use in extensive and heterogeneous control systems, typically in the area of the process industry, the unambiguous definition of engineering interfaces that are easy to use for all those involved is of great importance.

A device-specific software component, called Device Type Manager (DTM), is supplied by the field device manufacturer with its device. The DTM is integrated into engineering tools via the FDT interfaces defined in this specification. The approach to integration is in general open for all kind of fieldbusses and thus meets the requirements for integrating different kinds of devices into heterogeneous control systems.

Figure 1 shows how this part of the IEC TR 62453-52-xy series is aligned in the structure of the IEC 62453 series.

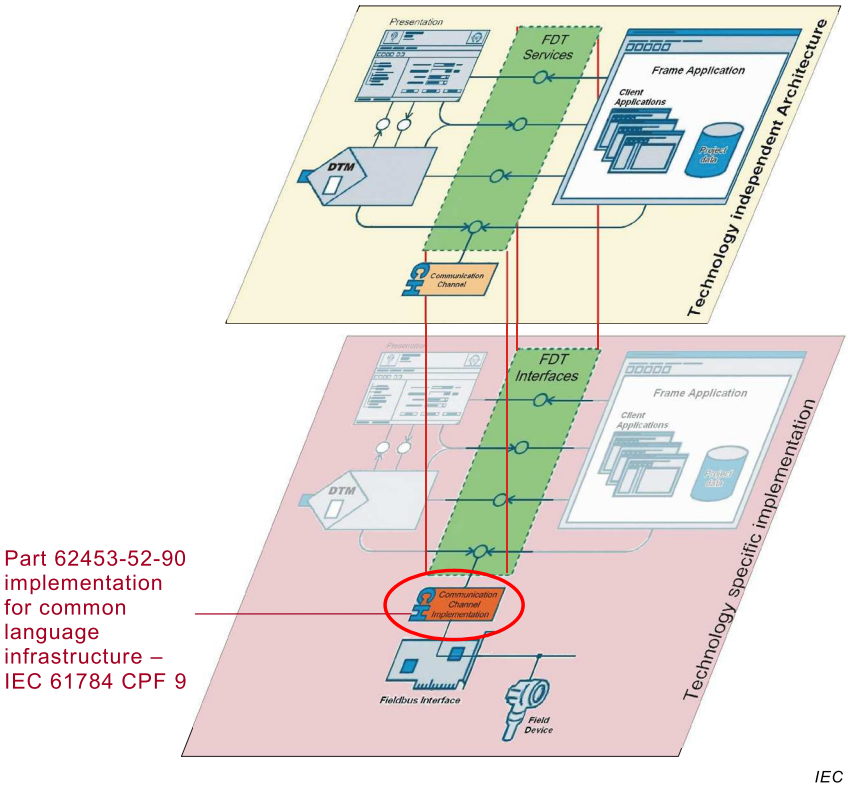



**Figure 1 – Part 52-90 of the IEC 62453 series**

# FIELD DEVICE TOOL (FDT) INTERFACE SPECIFICATION –

# Part 52-90: Communication implementation for common language infrastructure – IEC 61784 CPF 9

## 1 Scope

This part of the IEC 62453-52-xy series, which is a Technical Report, provides information for integrating the HART®[1] technology into the CLI-based implementation of FDT interface specification (IEC TR 62453-42).

This part of IEC 62453 specifies implementation of communication and other services based on IEC 62453-309.

This document neither contains the FDT specification nor modifies it.

## 2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61784-1:2014, *Industrial communication networks – Profiles – Part 1: Fieldbus profiles*

IEC 62453-1:2016, *Field device tool (FDT) interface specification – Part 1: Overview and guidance*

IEC 62453-2:2016, *Field device tool (FDT) interface specification – Part 2: Concepts and detailed description*

IEC TR 62453-42:2016, *Field device tool (FDT) interface specification – Part 42: Object model integration profile – Common language infrastructure*

IEC 62453-309:2016, *Field device tool (FDT) interface specification – Part 309: Communication profile integration – IEC 61784 CPF 9*

## 3 Terms, definitions, symbols, abbreviated terms and conventions

### 3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62453-1, IEC 62453-2, IEC TR 62453-42 and IEC 62453-309 apply.

---

[1] HART ® is the trade name of a product supplied by HART Communication Foundation. This information is given for convenience of users of this document and does not constitute an endorsement by IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

### 3.2 Symbols and abbreviated terms

For the purposes of this document, the symbols and abbreviations given in IEC 62453-1, IEC 62453-2, IEC 62453-309, and IEC TR 62453-42 apply.

### 3.3 Conventions

#### 3.3.1 Datatype names and references to datatypes

The conventions for naming and referencing of datatypes are explained in IEC 62453-2:2016, Clause A.1.

#### 3.3.2 Vocabulary for requirements

The following expressions are used when specifying requirements.

| | |
|---|---|
| Usage of “shall” or “mandatory” | No exceptions allowed. |
| Usage of “should” or “recommended” | Strong recommendation. It may make sense in special exceptional cases to differ from the described behaviour. |
| Usage of “can” or “optional” | Function or behaviour may be provided, depending on defined conditions. |

#### 3.3.3 Use of UML

Figures in this document are using UML notation as defined in Annex A of IEC 62453-1:2016.

## 4 Bus category

IEC 61784 CPF 9 protocol is identified in the attribute busCategory of the BusCategory element by the identifiers, as specified in IEC 62453-309.

## 5 Access to instance and device data

### 5.1 General

Used at interfaces:

- IInstanceData
- IDeviceData

These interfaces shall provide access to at least all parameters defined in IEC 62453-309.

### 5.2 IO signals provided by DTM

A DTM shall provide IO signal information of the device using the IProcessData interface.

To provide all information required to access the output signal information the DTM shall provide the information shown in Table 1 within its HartIOSignalInfo.

**Table 1 – Output signal info within IOSignalInfo / HartIOSignalInfo**

| Attribute | Description |
|---|---|
| Name | Name of the IO signal |
| Range | Reference to the variables providing range information |
| Unit | Reference to an enumeration variable describing the unit information |
| DeviceVariableAssignment | Constant enumeration value that can take following values<br>– unassigned: The IO signal is not assigned to dynamic variable and can only be accessed using the indexed approach (reading device variables).<br>– PV: IO signal assigned to the PV dynamic variable<br>– SV: IO signal assigned to the SV dynamic variable<br>– TV: IO signal assigned to the TV dynamic variable<br>– QV: IO signal assigned to the QV dynamic variable |
| DeviceVariableCode | Constant that specifies the device specific variable code (see [7][2] Table 34). Because of compatibility reasons to other FDT versions (e.g. IEC TR 62453-41), this variable could be set to the value 252 that stands for “unknown”. |

## 5.3 Data interfaces

### 5.3.1 General

Within HART, several command sets are defined. As part of the command set definition, HART provides a precise naming convention that is documented within the sources of the standard EDD libraries. The data provided by the access data interfaces should be named according to the HART naming convention. For backward compatibility, the semantic IDs as defined in previous versions of FDT should also be provided by the DTM.

A DTM shall provide all device data that is related to the set of Universal Commands and should provide all device data related to Common Practice CommandsIf the device supports additional command sets, like device family profiles, those data should also be exported using the naming convention as defined in the HART EDD libraries and as shown in Figure 2 and Figure 3.

### 5.3.2 Mapping HART datatypes to FDT datatypes

For a better usability of data provided by the data access interfaces IDeviceData and IInstanceData, all data from the device shall be converted into datatypes that are common to FDT. The mapping of basic datatypes is defined in Table 2.

2 Figures in square brackets refer to the Bibliography.

**Table 2 – Mapping of basic datatypes**

| HART Datatypes | FDT datatype | IEC datatype |
|---|---|---|
| Packed ASCII (see [10] 5.1.1) | String | STRING |
| ISO Latin-1 (see [10] 5.1.2) | String | STRING |
| Dates (see [10] 5.2) | DateTime | DATE_AND_TIME |
| Time (see [10] 5.3) | ULong (1/32 ms since midnight) | ULINT (1/32 ms since midnight) |
| Single Precision Floating Point (see [10] 5.4) | float | REAL |
| Double Precision Floating Point (see [10] 5.4) | double | LREAL |
| 1-4 Byte Unsigned Integer (see [10] 5.5) | UInt | UDINT |
| 5-8 Byte Unsigned Integer (see [10] 5.5) | ULong | ULINT |
| 1-4 Byte Unsigned Integer (see [10] 5.5) | UInt | UDINT |
| 5-8 Byte Unsigned Integer (see [10] 5.5) | ULong | ULINT |
| 1-4 Byte Enumerated (see [10] 5.7.1) | UInt | UDINT |
| 5-8 Byte Enumerated (see [10] 5. 7.1) | ULong | ULINT |
| 1-4 Byte Bit Fields (see [10] 5.7.2) | UInt | UDINT |
| 5-8 Byte Bit Fields (see [10] 5.7.2) | ULong | ULINT |

HCF standardized the access to device specific data using structures with the standard EDD import libraries for HART. The structure uses ARRAY and COLLECTION constructs that shall be reused when exposing data within FDT in IDeviceData and IInstanceData interfaces with elements of type StructDataGroup.

When converting a COLLECTION into a StructDataGroup, the DataItems shall be identical to the COLLECTION members with:

DataItem Name = COLLECTION member identifier

DataItem Label = COLLECTION member label

When converting an ARRAY into a StructDataGroup, the DataItem shall be identical to the ARRAY element with:

DataItem Name = ARRAY element index as string

DataItem Label = ARRAY element label

An example for such a structure is presented in Figure 2 and Figure 3.

### 5.3.3 SemanticInfo

The SemanticInfo for HART protocol related parameters is directly related to the protocol specification. The definition of the HART commands is the base for the parameter address information which shall be used in the properties ParameterReadAddress, ParameterWriteAddress and SemanticId of the SemanticInfo datatype.

The syntax of the parameter address information is as follows:

CMD<x>[Q(<r>)]B<y>B<z>L<n>

The [Q(<r>)] portion only is required to define request data.

For description of the attributes, please view Table 3.

**Table 3 – SemanticInfo attributes description**

| Attribute | Datatype | Description |
|---|---|---|
| <x> | decimal integer | command number in the range of 0 to 255 |
| <r> | hex-string | request data bytes |
| <y> | decimal integer | index of the start byte in the response data section (start index = 0) |
| <z> | decimal integer | index of the start bit in the byte referenced by <y> |
| <n> | decimal integer | length of the value in bits |

According to 6.4, commands with extended command number shall be described using “CMD31Q<r>...” wherein the first two byte of the request data section shall contain the extended command number.

The property SemanticInfo.ApplicationDomain shall contain ‘FDT_HART’ for all parameters in Table 4 and Table 5.

### 5.3.4 Data exposure using IDeviceData and IInstanceData interfaces

#### 5.3.4.1 Export of basic device parameters

Using the HART datatype mapping rules introduced in the preceding sections basic device parameters defined within the Universal and Common Practice Command sets can be exported.

Basic parameters that are not accessed using index information in the request part should for compatibility reasons provide additionally the SemanticInfo information specified in previous versions of FDT. Variables are identified by their standard identifier in HART. In Table 4, all variables are listed that shall be exported in the related data interfaces when supported by the device. HART identifier shall be assigned to the property Data.Id.

Variables with identifiers starting with “PV.”, ”SV.”, ”TV.” and ”QV.” shall be seen as short cuts to variables that shall be available too, using the structured exposure of device variables as defined in 5.3.4.2.1.

**Table 4 – Basic Variables exported in IDeviceData and IInstanceData interfaces**

| Identifier | SemanticInfo | Description |
|---|---|---|
| device_type | CMD0B1B0L16 | Expanded Device Type (see [7], subsection 5.1 and [5], section 6).<br><br>NOTE The information from CMD0 is also available via IHardwareIdentification interface. |
| request_preambles | CMD0B3B0L8 | Minimum number of Preambles required for the request message from the Master to the Slave. This number includes the two preambles used in asynchronous Physical Layers (along with the Delimiter) to detect the start of message. |
| universal_revision | CMD0B4B0L8 | HART Protocol Major Revision Number implemented by this device. For HART Revision 7, this value shall be the number 7. |
| transmitter_revision | CMD0B5B0L8 | Device Revision Level (refer to the HCF Command Summary Specification) |
| software_revision | CMD0B6B0L8 | Software Revision Level of this device. Levels 254 and 255 are reserved. |
| hardware_revision | CMD0B7B3L5 | (Most Significant 5 Bits) Hardware Revision Level of the electronics in this particular device. Does Not Necessarily Trace Individual Component Changes. Level 31 is Reserved. |
| physical_signaling_code | CMD0B7B0L3 | (Least Significant 3 Bits) Physical Signaling Code (see [7], subsection 5.10) |
| device_flags | CMD0B8B0L8 | Flags (see [7], subsection 5.11) |
| device_id | CMD0B9B0L24 | Device ID. This number shall be different for every device manufactured with a given Device Type. |
| response_preambles | CMD0B12B0L8 | Minimum number of preambles to be sent with the response message from the slave to the master. |
| max_num_device_variables | CMD0B13B0L8 | Maximum Number of Device Variables. This indicates the last Device Variable code that a host application should expect to be found in the field device (e.g., when identifying the Device Variables using Command 54). |
| config_change_counter | CMD0B14B0L16 | Configuration Change Counter |
| extended_fld_device_status | CMD0B16B0L8 | Extended Field Device Status (see [7], subsection 5.17) |
| manufacturer_id | CMD0B17B0L16 | Manufacturer Identification Code (see [7], subsection 5.8) |
| private_label_distributor | CMD0B19B0L16 | Private Label Distributor Code (see [7], subsection 5.8) |
| device_profile | CMD0B21B0L8 | Device Profile (see [7], subsection 5.57) |
| polling_address | CMD7B0B0L8 | Polling Address of Device (refer to [13], subsection 5.3.4) |
| loop_current_mode | CMD7B1B0L8 | Loop Current Mode (see [7], subsection 5.16) |
| message | CMD12B0B0L192 | Message |
| tag | CMD13B0B0L48 | Tag |
| descriptor | CMD13B6B0L96 | Descriptor |
| date | CMD13B18B0L24 | Date Code |
| PV.SENSOR_SERIAL_NUMBER | CMD14B0B0L24 | Transducer Serial Number |
| PV.DIGITAL_UNITS | CMD14B3B0L8 | Transducer Limits and Minimum Span Units Code (refer to [7], subsection 5.2) |
| PV.UPPER_SENSOR_LIMIT | CMD14B4B0L32 | Upper Transducer Limit |
| PV.LOWER_SENSOR_LIMIT | CMD14B8B0L32 | Lower Transducer Limit |
| PV.MINIMUM_SPAN | CMD14B12B0L32 | Minimum Span |

| Identifier | SemanticInfo | Description |
|---|---|---|
| PV.ALARM_CODE | CMD15B0B0L8 | PV Alarm Selection Code (see [7], subsection 5.6). The Alarm Selection Code indicates the action taken by the device under error conditions. For transmitters, the code indicates the action taken by the Loop Current. For Actuators, the action taken by the positioner is indicated. |
| PV.TRANSFER_FUNCTION | CMD15B1B0L8 | PV Transfer Function Code (see [7], subsection 5.3). The Transfer Function Code shall return "0", Linear, if transfer functions are not supported by the device. |
| PV.RANGE_UNITS | CMD15B2B0L8 | PV Upper and Lower Range Values Units Code (refer to [7], subsection 5.2) |
| PV.UPPER_RANGE_VALUE | CMD15B3B0L32 | PV Upper Range Value |
| PV.LOWER_RANGE_VALUE | CMD15B7B0L32 | PV Lower Range Value |
| PV.DAMPING_VALUE | CMD15B11B0L32 | PV Damping Value (units of seconds) |
| write_protect | CMD15B15B0L8 | Write Protect Code (see [7], subsection 5.7). The Write Protect Code shall return "251", None, when write protect is not implemented by a device. |
| PV.ANALOG_CHANNEL_FLAGS | CMD15B17B0L8 | PV Analog Channel Flags (see [7], subsection 5.26) |
| final_assembly_number | CMD16B0B0L24 | Final Assembly Number |
| longTag | CMD20B0B0L256 | Long Tag |
| PV.DIGITAL_UNITS | CMD1B0B0L8 | Primary Variable Units (refer to [7], subsection 5.2) |
| SV.DIGITAL_UNITS | CMD3B9B0L8 | Secondary Variable Units Code (refer to [7], subsection 5.2) |
| TV.DIGITAL_UNITS | CMD3B14B0L8 | Tertiary Variable Units Code (refer to [7], subsection 5.2) |
| QV.DIGITAL_UNITS | CMD3B19B0L8 | Quaternary Variable Units Code (refer to [7], subsection 5.2) |
| PV.CLASSIFICATION | CMD8B0B0L8 | Primary Variable Classification (refer to [7], subsection 5.21) |
| SV.CLASSIFICATION | CMD8B1B0L8 | Secondary Variable Classification (refer to [7], subsection 5.21) |
| TV.CLASSIFICATION | CMD8B2B0L8 | Tertiary Variable Classification (refer to [7], subsection 5.21) |
| QV.CLASSIFICATION | CMD8B3B0L8 | Quaternary Variable Classification (refer to [7], subsection 5.21) |
| lock_device_status_code | CMD76B0B0L8 | Lock Status (see HCF Common Table 25, Lock Device Status) |
| last_clock_date, | CMD90B8B0L24 | Date clock last set |
| last_clock_time | CMD90B11B0L32 | Time clock last set |
| real_time_clock_flag | CMD90B15B0L8 | RTC Flags (refer to [7], subsection 5.42) |

Table 4 contains data that can also be read via other interfaces. In this case, the DTM shall take care about the consistency of the data. E.g. the variables which are related to CMD0 shall be identical to the HartDeviceScanInfo data, which can be read via the IHardwareScan interface.

Table 5 lists all basic device variables that shall be exported only within IDeviceData interface if supported by the device.

**Table 5 – Basic Variables exported only in IDeviceData interface**

| Identifier | Semantic Info | Description |
|---|---|---|
| device_status | -/-<br>(empty) | Device status information transferred with each reply. Due to the fact that device_status is accessible by standard means in each transaction, the semantic info is empty |
| PV.DIGITAL_VALUE | CMD1B1B0L32 | Primary Variable |
| PV.ANALOG_VALUE | CMD2B0B0L32 | Primary Variable Loop Current (units of milli amperes) |
| PV.PERCENT_RANGE | CMD2B4B0L32 | Primary Variable Percent of Range (units of percent) |
| PV.DIGITAL_VALUE | CMD3B5B0L32 | Primary Variable |
| SV.DIGITAL_VALUE | CMD3B10B0L32 | Secondary Variable |
| TV.DIGITAL_VALUE | CMD3B15B0L32 | Tertiary Variable |
| QV.DIGITAL_VALUE | CMD3B20B0L32 | Quaternary Variable |
| current_date | CMD90B0B0L32 | Current Date |
| current_time | CMD90B4B0L32 | Current Time of Day |

#### 5.3.4.2 Export of structural information

##### 5.3.4.2.1 Device variables

Beside of the dynamic characteristics of dynamic variables, there exists data in the device that is related to the dynamic variable (e.g. range information, linearization functions). In HART, several categories of such data information are defined and standardized means are available that allow to document such relationships in a way that a client can access and analyze the structure without device specific knowledge. This information shall be exported in the IDeviceData and IInstanceData interfaces using StructDataGroup elements as mentioned in 5.3.2.

The DTM shall expose information in the structure as shown in Figure 2 and Figure 3.

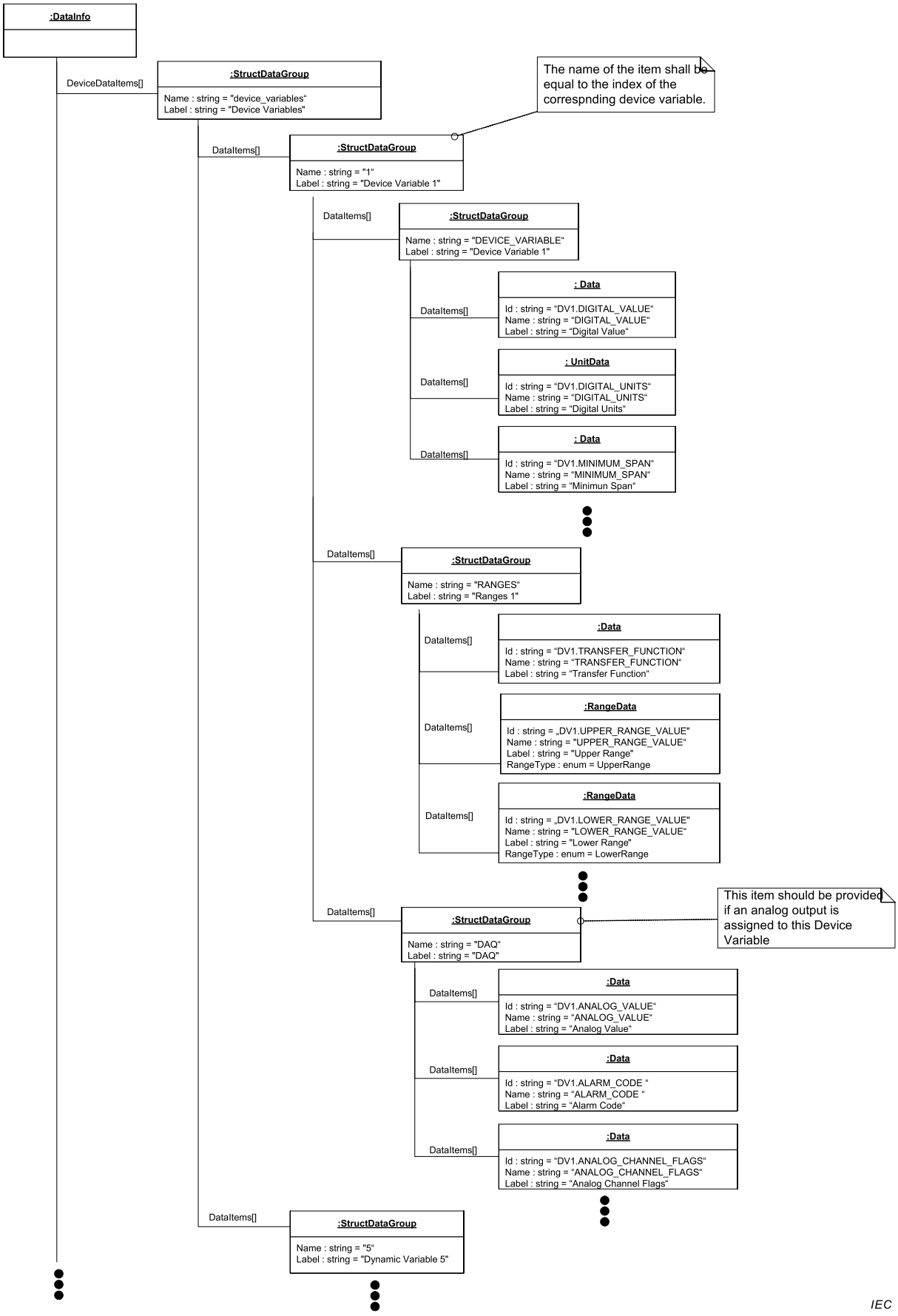



**Figure 2 – Structural information for device variables**

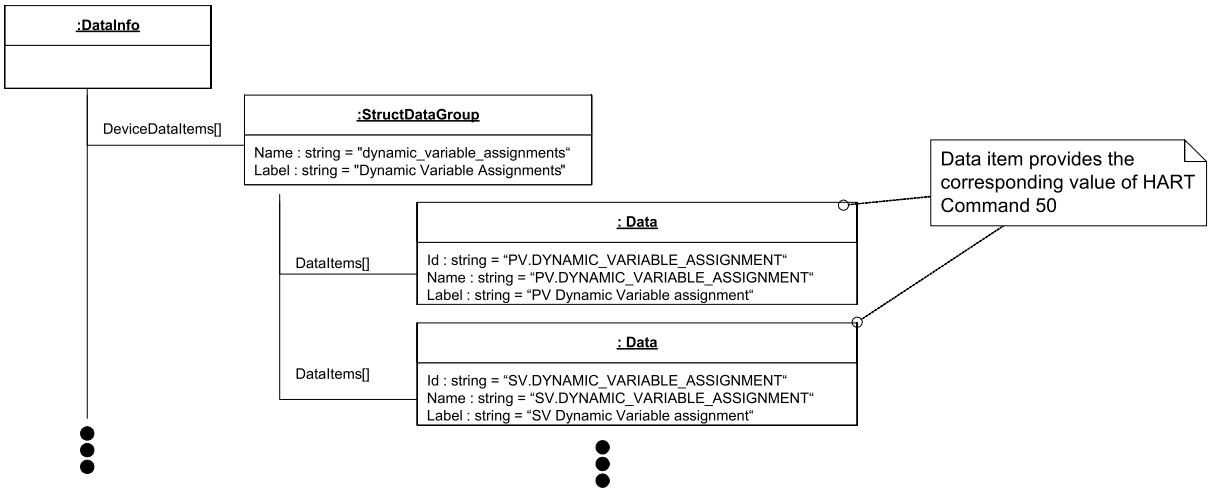



**Figure 3 – Structural information for dynamic variables**

Future changes to the structure shall be reflected similarly.

#### 5.3.4.2.2 Device specific status

HART supports two types of status information:

- device status

  This status is communicated with the response of each command and contains some flags that describe the state of the device.

  This status is already exposed in IDeviceData with variable “device_status”.

- additional device status

  This status information contains device specific information. Usually it is presented in HART command 48 when the “additional status available” flag is set in the device_status variable.

To expose additional device status information via the IDeviceData interface a new identifier is introduced here. The data is exposed with DataGroup “additional_device_status”. Elements of this data group are references to variables that contain device specific status information like shown in Figure 4.

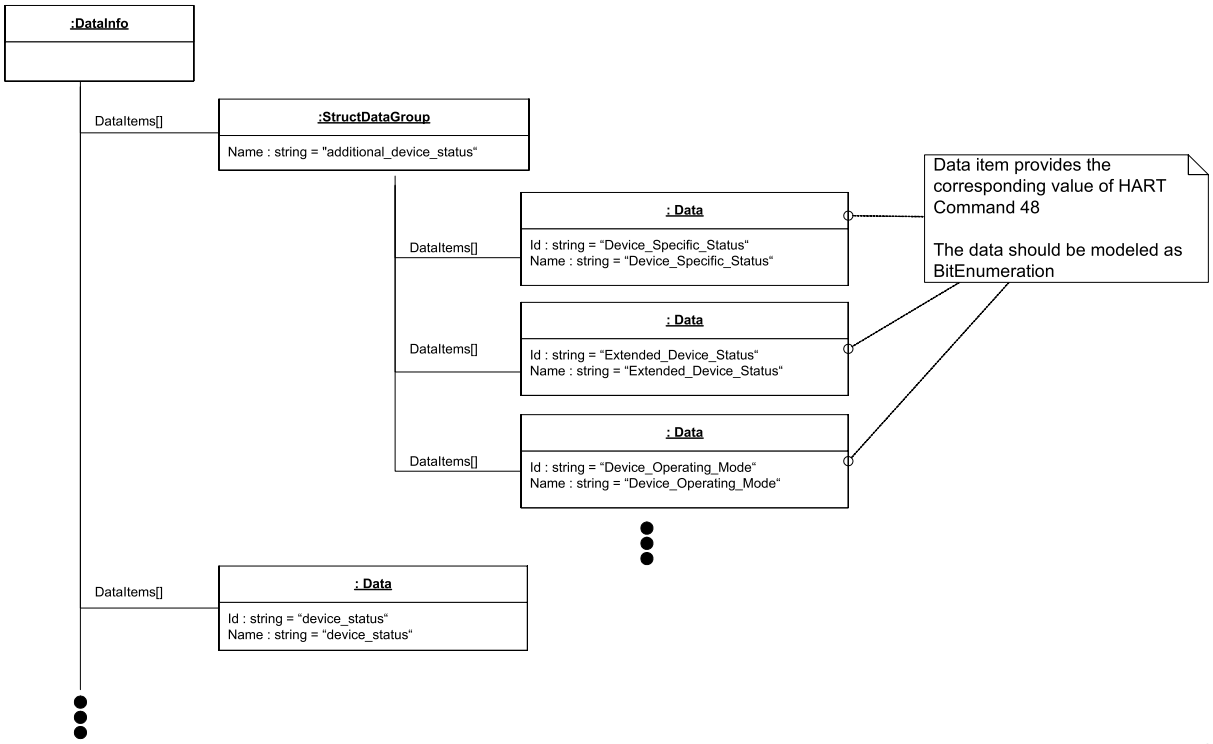



**Figure 4 – Structural information for extended device status**

SemanticInfo information shall be provided for all variables referenced in this group, allowing external tools to request status information without instantiation of the DTM.

#### 5.3.4.2.3 Extended burst and event information

HART 7 introduced new features in burst information including event notification. Data Propagation using these new burst and event management can be configured with standard means as defined by HART. The content of burst messages (especially for sub devices) can be evaluated using the exported structural information as presented in the preceding sections.

Support of this new handling is not defined in this document. It is open to the DTM developer to export additional information with the means introduced above.

## 6 Protocol specific behaviour

### 6.1 Support of burst mode

Support for burst mode defined in IEC 62453-309:2016, 6.2 shall be implemented as follows. A subscription of device initiated data transfer can be requested by calling ISubscription::<SubscriptionInitialization()>. The Communication Channel may detect if the device is already in burst mode. In HART 5 this can be detected only when HART burst frames are received from the device. In HART version 6 or later the burst mode can be detected using HART command 105. If HART Burst frames are received, the device is in burst mode and BurstModeDetected property of HartSubscribeResponse is set to TRUE. This means that device DTM will start to receive burst messages via SubscriptionCallback mechanism. In case that no burst messages were received, BurstModeDetected property is set to FALSE. It is up to the device DTM to set the device into burst mode. Then the device DTM may call ISubscription::<SubscriptionInitialization()> again in order to receive burst messages.

In order to unsubscribe, the device DTM calls ISubscription::<SubscriptionTermination()>. Device DTM will not receive any more burst information via SubscriptionCallback mechanism. The Communication Channel does not switch off the burst mode in the device. The device DTM may switch burst mode on or off by using normal TransactionRequests (HART command 109). This is independent of the subscription.

The sequence is described in Figure 5.

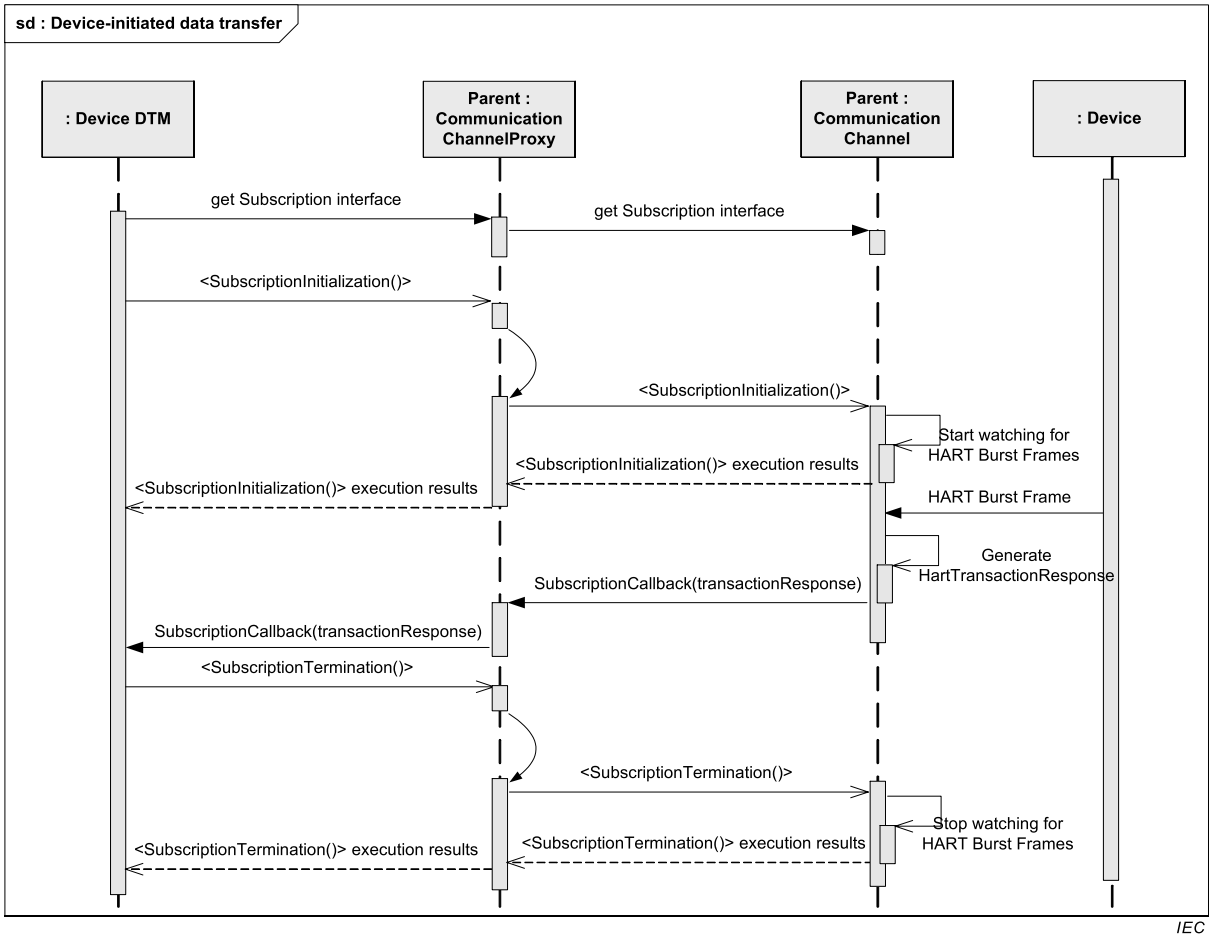



**Figure 5 – Device-initiated data transfer with burst mode**

## 6.2 Device addressing

DTMs shall support device addressing as defined in IEC 62453-309:2016, 6.3.

## 6.3 Support of scanning

DTMs shall support the IScanning interface.

## 6.4 Support of extended command numbers

DTMs shall support the extended command numbers as defined in IEC 62453-309:2016, 6.4.

## 6.5 Support for handling of communication failures and time-outs

DTMs shall support the handling of communication failures and time-outs as defined in IEC 62453-309:2016, 6.5.

### 6.6 Support for handling of Delayed Responses

DTMs shall support the handling of Delayed Responses as defined in IEC 62453-309:2016, 6.6.

### 6.7 Support for topologies with mixed HART protocols

DTMs shall support topologies with mixed HART protocols as defined in IEC 62453-309:2016, 6.7.

The implementation for the protocol check for the method ValidateAddChild() is:

- If no correlation can be found in the specific bus categories, the parent DTM shall answer the EndValidateAddChild() call with FALSE.
- If a correlation can be found, the EndValidateAddChild() call shall be answered with TRUE. During the ChildAdded() call, the parent DTM sets the ActiveProtocols property in the Child DTM.

### 6.8 Support for nested communication with multiple gateways

DTMs shall support nested communication with multiple gateways as defined in IEC 62453-309:2016, 6.8.

### 6.9 Support for topologies with WirelessHART

DTMs shall support topologies with WirelessHART as defined in IEC 62453-309, 6.9.

### 6.10 Transparent gateways

#### 6.10.1 General

HART supports various types of physical layers like HART FSK, HART over Ethernet, HART over 485, HART over Infrared, etc. Because of this range of physical layers, it can be possible that a HART Network contains a gateway device which connects nodes supporting different physical layers. One example for such gateway device is the Wireless Adapter which allows connecting HART FSK devices in a WirelessHART network.

Another example for such gateway device is a HART Infrared Adapter, which may connect a HART FSK device to a HART Infrared Network.

It is possible that some adapters appear to be transparent, i.e. they do not act as HART Nodes on the network. They act as physical layer adapter, impersonating the device they are connected to. Although, currently there are no Transparent Gateways in the market, the HART protocol does not restrict the possibility of such Transparent Gateways/Converters in the network.

IEC TR 62453-42 requires that such transparent gateways/adapters are represented by a DTM.

With the definition of new protocol Ids for each of the HART types in order to create a topology, it is required that DTMs are available for each of the participating nodes in the HART network.

The following scenarios explain the behaviour of such gateway devices in a FDT Topology.

#### 6.10.2 Scenario 1 – Manual topology creation

During manual topology creation, the HART Infrared Communication DTM only supports HART IR DTMs to be added to the topology. In this case, to add a HART FSK DTM to the

Topology, there shall be a Gateway DTM (HART Infrared to HART FSK DTM), though, the adapter is transparent.

It is mandatory that every participating node in the physical network shall have a DTM available. Without this, it is not possible to create a valid topology.

### 6.10.3 Scenario 2 – Topology scan and add

In case of scan topology, the Communication DTM may not detect the adapter (as it is transparent), and hence the scan result will not return the adapter. Instead, the scan result contains the HART FSK Device. However, the HART FSK device cannot be added to the topology due to different Protocol Ids.

In such scenario, unless the adapter acts as a HART node in the network, it is not possible to create the network topology.

## 7 Protocol specific usage of general datatypes

Table 6 shows how general datatypes are used with IEC 61784 CPF 9 devices.

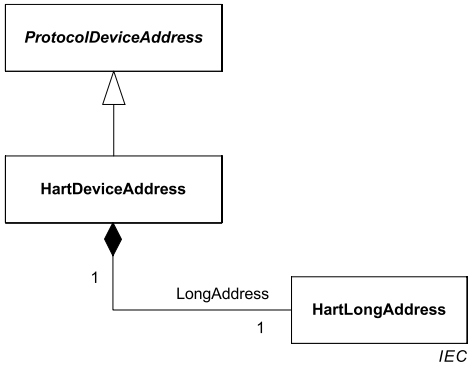
**Table 6 – Protocol specific usage of general datatypes**

| Attribute | Description for use |
|---|---|
| ProtocolId | See IEC 62453-309:2016, Table 1 |
| PhysicalLayer | See IEC 62453-309:2016, Table 2 |
| ApplicationDomain/<br>SemanticId | The application Domain is FDT_HART<br>The SemanticId shall be built as described in 5.3.3. |

## 8 Protocol specific common datatypes

### 8.1 HartDeviceAddress datatype

Figure 6 and Table 7 describe the datatype for addressing information used with protocol ID HART_Basic, HART_FSK, HART_Wireless, HART_RS485 and HART_Infrared.




**Used in:**
INetworkData::GetAddressInfo()
INetworkData::SetDeviceAddress()
ICommunication::BeginConnect()
ICommunication::EndConnect()
IScanning::EndScanRequest()

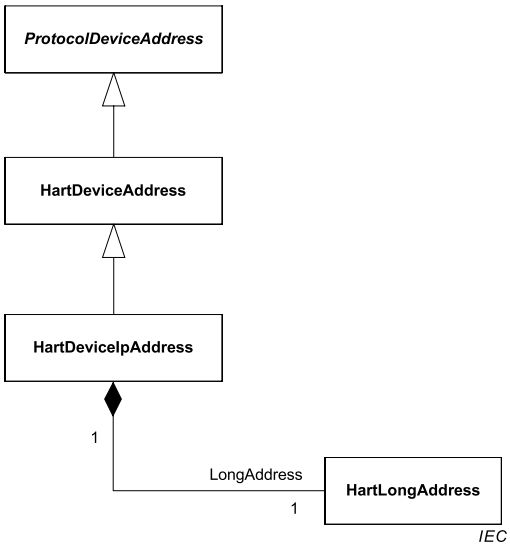
**Figure 6 – HartDeviceAddress datatype**

**Table 7 – HartDeviceAddress datatype**

| Property | Description |
|---|---|
| ShortAddress | This property is intended to store the short address (see CMD#0) information. |
| AddressingMode | With this property the parent DTM defines which address property shall be used for the connection.<br>The possible values of this enumeration are:<br>– ShortAddress<br>– ShortTag<br>– LongTag<br>– LongAddress |
| LongTag | This property is intended to store the LongTag (see CMD#20) information. |
| ShortTag | This property has stored the Tag (see CMD#13). |
| LongAddress | Long frame address information according to the HART specification (see Table 13) |

## 8.2 HartDeviceIpAddress datatype

Figure 7 and Table 8 describe the datatype derived from HartDeviceAddress with additional address information for HART TCP and UDP devices (Used with Protocol ID HART_IP).




**Used in:**

INetworkData::GetAddressInfo()
INetworkData::SetDeviceAddress()
ICommunication::BeginConnect()
ICommunication::EndConnect()
IScanning::EndScanRequest()

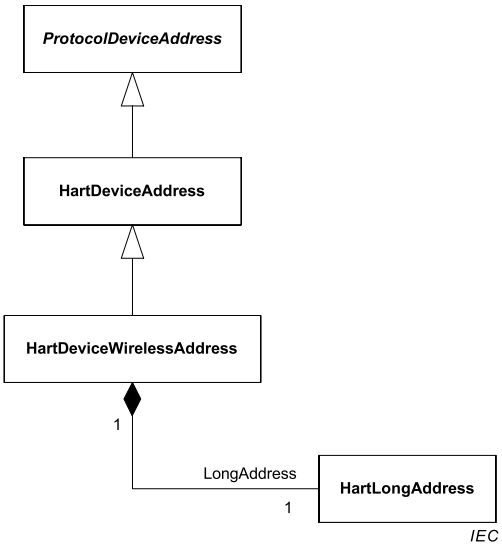
**Figure 7 – HartDeviceIpAddress datatype**

**Table 8 – HartDeviceIpAddress datatype**

| Property | Description |
|---|---|
| ShortAddress | (Inherited from HartDeviceAddress) |
| AddressingMode | (Inherited from HartDeviceAddress) |
| LongTag | (Inherited from HartDeviceAddress) |
| ShortTag | (Inherited from HartDeviceAddress) |
| IpAddress | This property stores a host name or an IP address conformant to IPv4 or IPv6 standard. |
| Port | [Optional] This property stores the port number. |
| LongAddress | Long frame address information according to the HART specification (see Table 13) |

## 8.3 HartDeviceWirelessAddress datatype

Figure 8 and Table 9 describe the datatype derived from HartDeviceAddress with additional address information for HART Wireless devices (Used with Protocol ID HART_Wireless).




**Used in:**

INetworkData::GetAddressInfo()
INetworkData::SetDeviceAddress()
ICommunication::BeginConnect()
ICommunication::EndConnect()
IScanning::EndScanRequest()

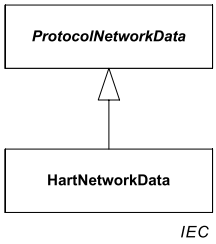
**Figure 8 – HartDeviceWirelessAddress datatype**

**Table 9 – HartDeviceWirelessAddress datatype**

| Property | Description |
|---|---|
| ShortAddress | (Inherited from HartDeviceAddress) |
| AddressingMode | (Inherited from HartDeviceAddress) |
| LongTag | (Inherited from HartDeviceAddress) |
| ShortTag | (Inherited from HartDeviceAddress) |
| NetworkID | [Optional] This property stores the Network ID for a HART Wireless network. |
| LongAddress | Long frame address information according to the HART specification (see Table 13) |

## 9 Network management datatypes

The data needed for management of the network are exposed by the Device DTM in the INetworkData interface as HartNetworkData (see Figure 9).




**Used in:**
INetworkData::GetNetworkDataInfo()

**Figure 9 – HartNetworkData datatype**

Table 10 describes the HART specific network data. This data is read only.

**Table 10 – HartNetworkData datatype**

| Attribute | Description |
|---|---|
| HartRevision | This property will be used as a informative constant that documents the HART major revision as communicated in CMD#0. |
| PollingAddressRange | This property describes the range of the polling address. |

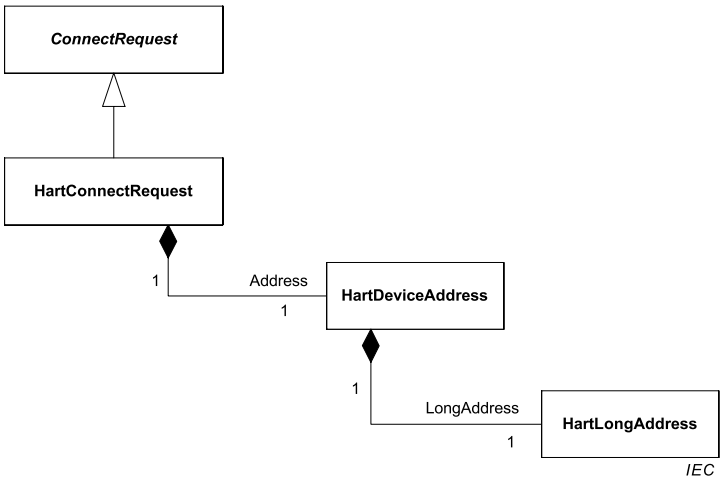
## 10 Communication datatypes

### 10.1 General

The datatypes contain the address information and the communication data required to execute the respective request or to transport the response information.

### 10.2 HartConnectRequest datatype

The structure of the datatype HartConnectRequest is shown in Figure 10.




**Used in:**

ICommunication::BeginConnect()

**Figure 10 – HartConnectRequest datatype**

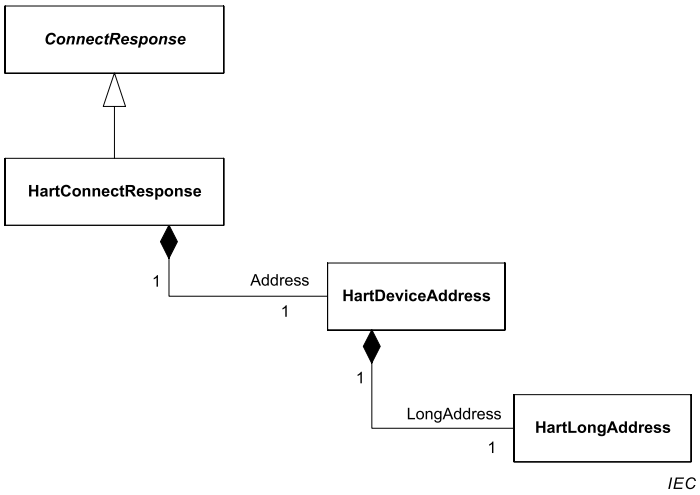
The properties of the datatype are described in Table 11.

**Table 11 – HartConnectRequest datatype**

| Property | Description |
|---|---|
| Address | In order to establish a connection the DTM shall provide address information of the HART Device (see Table 10). |
| ProtocolId | Unique identifier of the HART protocol |
| SystemTag | Unique identification of DTM within the Frame Application |
| PreambleCount | [Optional] Provides a hint for the communication component about the number of preambles, required by the device type. |
| PrimaryMaster | [Optional] Provides a hint for a communication component that a DTM requires communication as primary or secondary master. |
| LongAddress | Long frame address information according to the HART specification (see Table 13) |

## 10.3 HartConnectResponse datatype

The structure of the datatype HartConnectResponse is shown in Figure 11.




**Used in:**
ICommunication::EndConnect ()

**Figure 11 – HartConnectResponse datatype**

The properties of the datatype are described in Table 12.

**Table 12 – HartConnectResponse datatype**

| Property | Description |
|---|---|
| PreambleCount | Contains the information about the currently used preambleCount. |
| PrimaryMaster | Contains the information about the current state of the master. |
| Address | Address information including ShortAddress, ShortTag, LongTag (see Table 7) |
| LongAddress | Long frame address information according to the HART specification (see Table 13) |
| CommunicationReference | Identifier for a communication link to a device |

### 10.4 HartLongAddress datatype

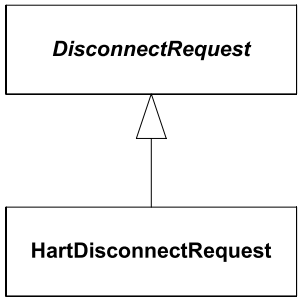
The properties of the datatype are described in Table 13.

**Table 13 – HartLongAddress datatype**

| Property | Description |
|---|---|
| Address1 | 1st byte of long address |
| Address2 | 2nd byte of long address |
| Address3 | 3rd byte of long address |
| Address4 | 4th byte of long address |
| Address5 | 5th byte of long address |

### 10.5 HartDisconnectRequest datatype

The structure of the datatype HartDisconnectRequest is shown in Figure 12.




**Used in:**

ICommunication::BeginDisconnect()

**Figure 12 – HartDisconnectRequest datatype**

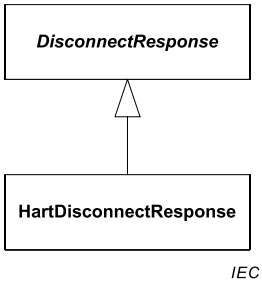
The properties of the datatype are described in Table 14.

**Table 14 – HartDisconnectRequest datatype**

| Property | Description |
|---|---|
| CommunicationReference | Identifier for a communication link to a device |
| AbortPendingTransactions | Indicates whether pending transactions shall be aborted. |

### 10.6 HartDisconnectResponse datatype

The structure of the datatype HartDisconnectResponse is shown in Figure 13.




**Used in:**

ICommunication::EndDisconnect()

**Figure 13 – HartDisconnectResponse datatype**

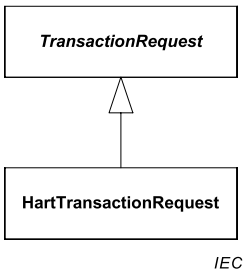
The properties of the datatype are described in Table 15.

**Table 15 – HartDisconnectResponse datatype**

| Property | Description |
|---|---|
| CommunicationReference | Identifier for a communication link to a device |

### 10.7 HartTransactionRequest datatype

The structure of the datatype HartTransactionRequest is shown in Figure 14.




**Used in:**

ICommunication::BeginCommunicationRequest ()

**Figure 14 – HartTransactionRequest datatype**

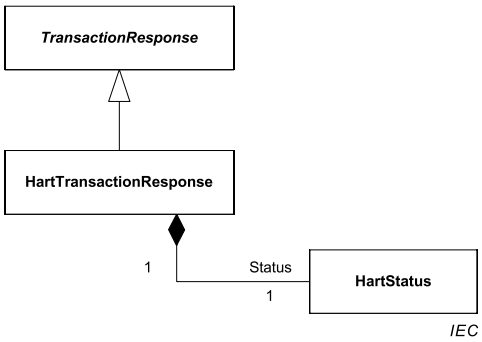
The properties of the datatype are described in Table 16.

**Table 16 – HartTransactionRequest datatype**

| Property | Description |
|---|---|
| CommandNumber | HART command number |
| CommunicationData | Communication data to be transferred to the device |
| Id | [Optional] Identifier for a single Transaction Request |

### 10.8 HartTransactionResponse datatype

The structure of the datatype HartTransactionResponse is shown in Figure 15.




**Used in:**

ICommunication::EndCommunicationRequest ()

**Figure 15 – HartTransactionResponse datatype**

The properties of the datatype are described in Table 17.

**Table 17 – HartTransactionResponse datatype**

| Property | Description |
|---|---|
| CommunicationReference | Identifier for a communication link to a device |
| ErrorInformation | [Optional] Description of a fieldbus protocol independent error occurred during communication |
| Id | [Optional] Identifier of the corresponding Transaction Request |
| CommandNumber | HART command number |
| Status | Status information according to the HART specification |
| CommunicationData | Communication data received from the device |

### 10.9 HartStatus datatype

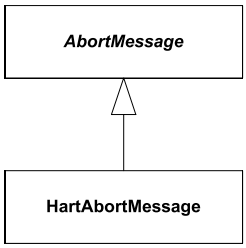
The properties of the datatype are described in Table 18.

**Table 18 – HartStatus datatype**

| Property | Description |
|---|---|
| DeviceStatus | Second status byte returned in HART command responses |
| CommandStatus | [Optional] Command Response Code, computed according HART specification from the first status byte returned in HART command responses |
| CommunicationStatus | [Optional] Communication error code, computed according HART specification from the first status byte returned in HART command responses. |

### 10.10 HartAbortMessage datatype

The structure of the datatype HartAbortMessage is shown in Figure 16.




**Used in:**
AbortCallback ()

**Figure 16 – HartAbortMessage datatype**

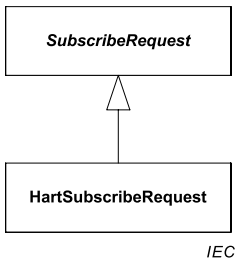
The properties of the datatype are described in Table 19.

**Table 19 – HartAbortMessage datatype**

| Property | Description |
|---|---|
| Details | Details about the cause and source of the Abort |
| CommunicationReference | Identifier for a communication link to a device |

### 10.11 HartSubscribeRequest datatype

The structure of the datatype HartSubscribeRequest is shown in Figure 17.




**Used in:**
ISubscription::BeginSubscriptionInitialization ()

**Figure 17 – HartSubscribeRequest datatype**

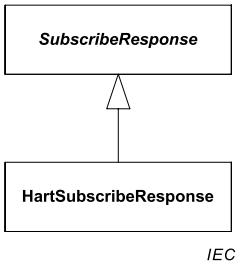
The properties of the datatype are described in Table 20.

**Table 20 – HartSubscribeRequest datatype**

| Property | Description |
|---|---|
| CommunicationReference | Identifier for a communication link to a device |

### 10.12 HartSubscribeResponse datatype

The structure of the datatype HartSubscribeResponse is shown in Figure 18.




**Used in:**
ISubscription::EndSubscriptionInitialization ()

**Figure 18 – HartSubscribeResponse datatype**

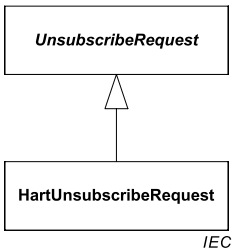
The properties of the datatype are described in Table 21.

**Table 21 – HartSubscribeResponse datatype**

| Property | Description |
|---|---|
| BurstModeDetected | Indicates whether the Communication Channel has detected that the device is already in burst mode. |
| CommunicationReference | Identifier for a communication link to a device |

### 10.13 HartUnsubscribeRequest datatype

The structure of the datatype HartSubscribeRequest is shown in Figure 19.




**Used in:**

ISubscription::BeginSubscriptionTermination ()

**Figure 19 – HartUnsubscribeRequest datatype**

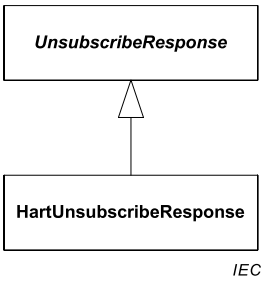
The properties of the datatype are described in Table 22.

**Table 22 – HartUnsubscribeRequest datatype**

| Property | Description |
|---|---|
| CommunicationReference | Identifier for a communication link to a device |

### 10.14 HartUnsubscribeResponse datatype

The structure of the datatype HartUnsubscribeResponse is shown in Figure 20.




**Used in:**

ISubscription::EndSubscriptionTermination ()

**Figure 20 – HartUnsubscribeResponse datatype**

The properties of the datatype are described in Table 23.

**Table 23 – HartUnsubscribeResponse datatype**

| Property | Description |
|---|---|
| CommunicationReference | Identifier for a communication link to a device |

## 11 Datatypes for process data information

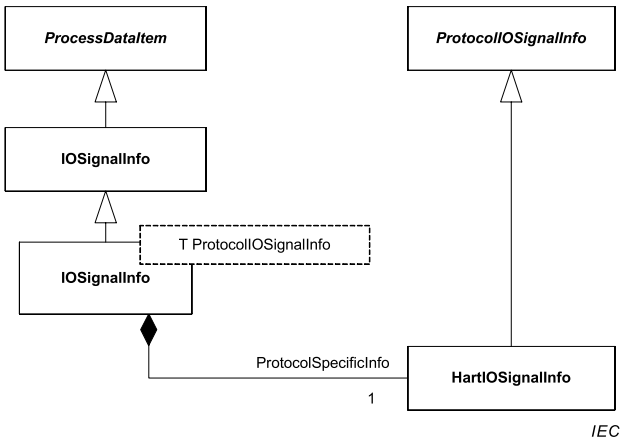
### 11.1 General

The process data information of a DTM represents the “Device Variables”, available on that device. A Process Control System (i.e. some external system which monitors values on a device) can query the DTM’s process data information via the IProcessData interface. The process data describes the process values such that an external system can use the

information to access and interpret the values from the device during normal device runtime. The external system might not use FDT to access the values.

### 11.2 HartIOSignalInfo datatype

The structure of the datatype HartIOSignalInfo is shown in Figure 21.




**Used in:**

IProcessData::EndGetProcessData ()

**Figure 21 – HartIOSignalInfo datatype**

Table 24 shows which properties of IOSignalInfo datatype should be provided or requires HART specific usage.

**Table 24 – Usage of IOSignalInfo datatype**

| Property | Description |
|---|---|
| SignalType | SignalType is always Input |
| SemanticInfos | See Table 5 for assigned variables |
| DeviceDataRef | Reference to the related variable, exposed in IDeviceData |
| UnitInfoRef | Reference to the variables providing range information |
| RangeInfoRefs | Reference to an enumeration variable describing the unit information |

The properties of the HartIOSignalInfo datatype are described in Table 25.

**Table 25 – HartIOSignalInfo datatype**

| Property | Description |
|---|---|
| DeviceVariableAssignment | Constant enumeration value that can take following values<br>– unassigned: The dynamic variable is not assigned to process variable and can only be accessed using the indexed approach.<br>– PV: Variable is assigned to the PV process variable<br>– SV: Variable is assigned to the SV process variable<br>– TV: Variable is assigned to the TV process variable<br>– QV: Variable is assigned to the QV process variable |
| DeviceVariableCode | Constant that specifies the device specific variable code (see [7] Table 34). Because of compatibility reasons to other FDT versions (e.g. IEC 62453-41), this variable could be set to the value 252 that stands for “unknown”. |

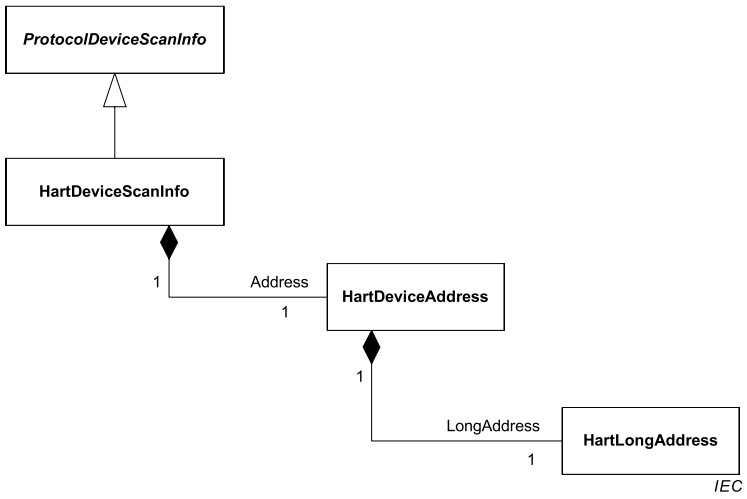
# 12 Device identification datatypes

## 12.1 General

This clause defines identification relevant protocol specific datatypes.

## 12.2 HartDeviceScanInfo datatype

The structure of the datatype HartDeviceScanInfo is shown in Figure 22.




**Used in:**
IDtmScanning::EndScanRequest ()

**Figure 22 – HartDeviceScanInfo datatype**

The properties of the datatype are described in Table 26.

**Table 26 – HartDeviceScanInfo datatype**

| Property | Description |
|---|---|
| Address | Address information of the scanned device |
| HartMajorRevisionNumber | HART protocol revision |
| ManufacturerIdentificationCode | Manufacturer specific code registered at the HART Foundation |
| DeviceTypeCode | Device type specific code registered at the HART Foundation |
| SoftwareRevision | Revision of the device embedded software |
| HardwareRevisionLevel | Hardware revision of the device |
| PhysicalSignalingCode | Physical signaling code |
| DeviceId | Individual serial number of the device |
| DeviceRevisionLevel | Device Command Revision Level supported by the device |
| DeviceFlags | Device Flags |
| ScannedPhysicalLayer | Determines the physical medium for which the scan information is provided. |
| ManufacturerSpecificExtension | [Optional] Can be used by DTM for a vendor specific device identification information, e.g. by combining a number of device parameter values into one string value. This can be used to identify a specific device variant. |

Table 27 defines the semantics of HartDeviceScanInfo properties and how this information is mapped to predefined properties of DeviceScanInfo.

The Communication Channel will read these values from the device and writes them to the properties of HartDeviceScanInfo.

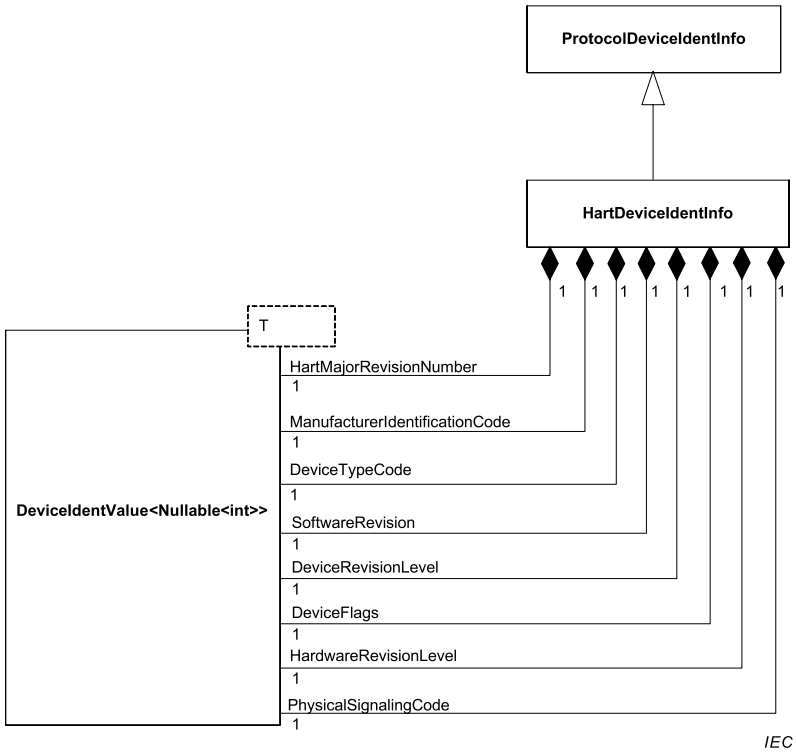
**Table 27 – Protocol specific mapping of scan information**

| HartDeviceScanInfo property name | Datatype of HartDeviceScanInfo property | Mapped to property name in DeviceScanInfo | Data request in physical device | Protocol specific name | HART data format | Specific reference |
|---|---|---|---|---|---|---|
| Address.ShortAddress | int | Address.Address | To scan for HART devices CommChannel shall poll possible address range (HART 5: [0-15], HART 6: [0-63]) by calling Cmd #0. If Cmd #0 response is available, a physical device is connected to this address.<br>Cmd #0 response does not contain short address value. If master using short address for polling receives a response, it can assume that short address of device is the same as used in polling request. In addition to this, polling address can be read from HART 6 device with Cmd #7. | Polling Address | Unsigned 8 | [5] 6.8 Command 7 Read Loop Configuration. |
| Address.ShortTag | string | Tag<br>(long or short tag depending on addressing rules) | Command 13 Bytes 0 – 5 | Tag | 6 Bytes or 8 Packed ASCII characters | [5] HART 6: HCF_Spec_127 – Universal Command Specification – |
| Address.LongTag | string | Tag | HART6 or 7:<br>Command 20 (long or short tag depending on addressing rules)<br>HART5:<br>Command 13 Bytes 6 – 17 | Long Tag Descriptor | HART6 or 7:<br>32 Bytes char<br>HART5:<br>12 Bytes or 16 Packed ASCII characters | HART6 or 7:<br>[5] 6.20 – Command 20 Read Long Tag<br>HART5:<br>[5] 6.13 – Command 13 Read Tag, Descriptor, Date |
| ProtocolId | GUID | ProtocolId | CommChannel has to pass Hart. ProtocolId in this attribute. | | | |
| ScannedPhysicalLayer | PhysicalLayer | PhysicalLayer | CommChannel returns the physical layer, via which the scanned device is connected. | | | |
| ManufacturerIdentificationCode | int | ManufacturerId | HART7:<br>Command 0 Byte 17+18.<br>HART<7:<br>Command 0 Byte 1. | Manufacturer Identification Code | 16 bit unsigned enumerator | [5] Manufacturer Identification Codes |
| DeviceTypeCode | int | DeviceTypeId | HART 7: | Device Type | 16 bit unsigned | [5] 6.1 Command |

| HartDeviceScanInfo property name | Datatype of HartDeviceScanInfo property | Mapped to property name in DeviceScanInfo | Data request in physical device | Protocol specific name | HART data format | Specific reference |
|---|---|---|---|---|---|---|
| | | | Command 0 Byte 1+2.<br>HART<7:<br>Command 0 Byte 2. | Code | enumerator | 0 Read Unique Identifier |
| SoftwareRevision | int | SoftwareRevision | Command 0 Byte 6. | Software Revision | 8 bit unsigned integer | [5] 6.1 Command 0 Read Unique Identifier |
| HardwareRevisionLevel | int | HardwareRevision | Command 0 Byte 7 (Most significant 5 bits, shifted 3 bits right). | Hardware Revision 8 bit unsigned integer | (mapped to float: xxxxx.yyy) | Last 3 bits (y) to Physical Signaling Code. Float<br>[5] 6.1 Command 0 Read Unique Identifier |
| DeviceId | int | SerialNumber | Command 0 Bytes 9 – 11 (Device Identification Number). | Device Identification Number | Unsigned 24 | [5] 6.1 Command 0 Read Unique Identifier |
| DeviceProfile | int | ProtocolIdentificationProfile | HART7:<br>Command 0 Byte 21.<br>HART<7:<br>n/a | Device Profile Codes | 8 bit unsigned enumerator | [5] 6.1 Command 0 Read Unique Identifier |
| PhysicalSignalingCode | int | Mapped to DeviceScanValues listed in ProtocolSpecificProperties with corresponding name as in HartDeviceScanInfo | Command 0 Byte 7 (Least significant 3 bits). | Physical Signaling Code | 8 bit unsigned enumerator | [5] 6.1 Command 0 Read Unique Identifier |
| HartMajorRevisionNumber | int | | Command 0 Byte 4. | HART Protocol Major Revision Number (e.g. 7 for HART 7) | 8 bit unsigned integer | [5] 6.1 Command 0 Read Unique Identifier |
| DeviceRevisionLevel | int | | Command 0 Byte 5. | Device Revision Level | 8 bit unsigned integer | Device Revision Level |
| DeviceFlags | int | | Command 0 Byte 8. | Device Flags | Bit value according Flag Assignment table. | Bit value according Flag Assignment table. |

### 12.3 HartDeviceIdentInfo datatype

HartDeviceIdentInfo (see Figure 23) provides HART specific identifications of supported device types returned by GetDeviceIdentInfo(). For DTM assignment after fieldbus-scanning, the frame application can check in a protocol independent way if the identification of a scanned device type (DeviceScanInfo) matches to the supported DeviceIdentInfo.




**Used in:**

IDtmInformation::GetDeviceIdentInfo()

**Figure 23 – HartDeviceIdentInfo datatype**

The properties of the datatype are described in Table 28.

**Table 28 – HartDeviceIdentInfo datatype**

| Property | Description |
|---|---|
| HartMajorRevisionNumber | HART protocol revision |
| ManufacturerIdentificationCode | Manufacturer specific code registered at the HART Foundation |
| DeviceTypeCode | Device type specific code registered at the HART Foundation |
| SoftwareRevision | Revision of the device embedded software |
| HardwareRevisionLevel | Hardware revision of the device |
| PhysicalSignalingCode | Physical signaling code |
| DeviceRevisionLevel | Device Command Revision Level supported by the device |
| DeviceFlags | Device Flags |
| ManufacturerSpecificExtension | [Optional] Can be used by DTM for a vendor specific device identification information, e.g. by combining a number of device parameter values into one string value. This can be used to identify a specific device variant. |

### 12.4 Mapping of information source

Table 29 defines the semantics of HartDeviceIdentInfo properties and how this information is mapped to predefined properties of DeviceIdentInfo.

**Table 29 – Protocol specific mapping of identity information**

| HartDeviceIdentInfo property name | Datatype of DeviceIdentValue in property in HartDeviceIdentInfo | Mapped to property in DeviceIdentInfo | Data request in physical device | Protocol specific name | HART data format | Specific reference |
|---|---|---|---|---|---|---|
| ProtocolId | Guid | ProtocolId | Device DTM has to pass Hart.ProtocolId in this attribute. | | | |
| ManufacturerIdentificationCode | int | ManufacturerId | HART7:<br>Command 0 Byte 17+18.<br>HART<7:<br>Command 0 Byte 1. | Manufacturer Identification Code | 16 bit unsigned enumerator | [5] Manufacturer Identification Codes |
| DeviceTypeCode | int | DeviceTypeId | HART7:<br>Command 0 Byte 1+2.<br>HART<7:<br>Command 0 Byte 2. | Device Type Code | 16 bit unsigned enumerator | [5] 6.1 Command 0 Read Unique Identifier |
| SoftwareRevision | int | SoftwareRevision | Command 0 Byte 6. | Software Revision | 8 bit unsigned integer | [5] 6.1 Command 0 Read Unique Identifier |
| HardwareRevisionLevel | int | HardwareRevision | Command 0 Byte 7 | Hardware Revision | Most significant 5 bits of 8 bit unsigned integer), shifted 3 bits right<br>Last 3 bits (y) to Physical Signaling Code. Float | [5] 6.1 Command 0 Read Unique Identifier |
| DeviceProfile | int | ProtocolIdentificationProfile | HART7:<br>Command 0 Byte 21.<br>HART<7:<br>n/a | Device Profile Codes | 8 bit unsigned enumerator | [5] 6.1 Command 0 Read Unique Identifier |
| PhysicalSignalingCode | int | Mapped to DeviceIdentValues listed in ProtocolSpecificProperties with | Command 0 Byte 7 (Least significant 3 bits). | physical_signaling_code | 8 bit unsigned enumerator | [5] 6.1 Command 0 Read Unique Identifier |
| HartMajorRevisionNumber | int | | HART Command 0 | HART Protocol Major | 8 bit unsigned | [5] 6.1 Command 0 |

| HartDeviceIdentInfo property name | Datatype of DeviceIdentValue in property in HartDeviceIdentInfo | Mapped to property in DeviceIdentInfo | Data request in physical device | Protocol specific name | HART data format | Specific reference |
|---|---|---|---|---|---|---|
| | | corresponding name as in HartDeviceIdentInfo | Byte 4. | Revision Number (e.g. 7 for HART 7) | integer | Read Unique Identifier |
| DeviceRevisionLevel | int | | | transmitter_revision | 8 bit unsigned integer Device Revision Level | |
| DeviceFlags | int | | | device_flags | Bit value according Flag Assignment table | |

The HartDeviceIndentInfo properties may have either a single value which shall exactly match the supported device or a range of matching values may be defined in regular expressions.

# Bibliography

[1] FDT 2.0 Specification v1.0, July 2012, Order No. of FDT Group: 0001-0008-000, available at http://fdtgroup.org/download/3823/ [viewed 2017-04-03]

[2] FDT Interface Specification 1.2, March 2004, Order No. of FDT Group: 0001 0001 001, available at http://fdtgroup.org/download/3866/ [viewed 2017-04-03]

[3] FDT Interface Specification 1.2.1, July 2006, Order No. of FDT Group: 0001 0001 002, available at http://fdtgroup.org/download/3840/ [viewed 2017-04-03]

[4] FDT Group Best Practice “How to create a protocol specific annex”, version 1.2, Order No. of FDT Group: 0003 0002 002

[5] HCF Universal Command Specification (HCF_SPEC 127), version 7.1, May 2008

[6] HCF Common Practice Command Specification (HCF_SPEC 151), version 9.1, May 2008

[7] HCF Common Tables (HCF_SPEC-183), version 20.3, November 2010

[8] HCF Device Description Language Specification (HCF_SPEC-500), version 12.1, October 2008,

[9] HCF DD Integrated Development Environment (HCF_KIT-225), version 4.1, July 2010

[10] HCF Command Summary Specification (HCF_SPEC-99), version 9.0, July 2007

[11] HCF HART Communication Protocol Specification (HCF_SPEC-13), version 7.3, May 2011

[12] HCF Network Management Specification (HCF_SPEC-085), version 1.1, May 2008

[13] HCF Token-Passing Data Link Layer Specification (HCF_SPEC-081), revision 9.0, May 2012

_____

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

3, rue de Varembé
PO Box 131
CH-1211 Geneva 20
Switzerland

Tel:  + 41 22 919 02 11
Fax: + 41 22 919 03 00
info@iec.ch
www.iec.ch