ABB	Technical Information Common Diagnosis Subsystem Mapping Guideline			TI
Responsibility: **APR / ITKET**	Date: 2008 December 17	Language: **en**	Filing system :	Revision: **0.5**	Page: **1/14**
Issued by: Giovanni Invernizzi	Approved:		Released:	Area of validity: BUI

Title: **Common Diagnosis Subsystem Configuration**

References:

[1] Requirement Specification Common Diagnosis Sub-System, Rev. 0.4
[2] Design Description Common Diagnosis Sub-System, Rev 0.4
[3] NE107 - Self-monitoring and diagnosis of field devices - Version: 10.02.2006
[4] IEC 61508 Part 3 and Part 7 First edition 1998-12
[5] MoMs Lenno 7 June 2006
[6] MoMs Lenno 8 June 2006
[7] HART Communication Foundation Document Number HCF_SPEC-99, Revision 8.0
[8] FOUNDATION™ Specification Function Block Application Process Part 1, FF-890 Rev. FS 1.7
[9] PROFIBUS Specification -Profile for Process Control Devices Version 3.01 December 2004
[10] PROFIBUS Specification - Amendment 2 to the Profibus Profile for Process Control Devices V 3.01 - Condensed Status and Diagnostic Messages V 1.0 June 2005

Distribution

Author Giovanni Invernizzi Date: 2008 December 17

Remarks:

Contents

1 Introduction

An important goal reached by the Process 5k methodology is the harmonization of the customer oriented part of each device, from housing to data presentation (via HMI or digital communication protocol). We recognize that a Common Diagnosis SubSystem will help in the harmonization process and in finding out a common look and feel though a common integrated way of managing alarms.

Field devices producing common diagnostic conditions can be integrated in the overall systems/tools in an easier and better-harmonized way only if these diagnostic conditions have been mapped in the same way within the field devices. Additional device specific diagnostic conditions can be individually identified for each device but their handling and mapping must be specified and implemented following a common rule.

Whenever all the devices will satisfy the above requirements, it will be possible to treat part of their diagnosis capability with:

- Same reading mechanism/logic
- Same specification for the Asset Monitor (Optimize IT)
- Same representation in the DTM
- Same description in the DD
- Same naming and meaning

The above requirement enforces treatment of each diagnosis condition not as a simple bit but as a combination of information covering several aspects that were traditionally implemented/assembled within the maintenance stations (ASSET MONITORs). The advantage is that it should be no longer necessary to produce specific device integrations for different Hosts because all the relevant information is directly produced by the field devices.

The scope of this document is to help a developer to:

- Manage the alarms conditions and map them into the NAMUR representation used into the diagnosis subsystem, to achieve the harmonization process and the common look and feel discussed above.
- Map diagnosis subsystem result into the communication subsystems (i.e. Communication Protocol, HMI, etc), to achieve harmonization of the customer oriented part.

2 Diagnosis Mapping

Field devices producing common diagnostic conditions can be integrated in the overall systems/tools in an easier and better-harmonized way only if these diagnostic conditions have been mapped in the same way within the field devices. Additional device specific diagnostic conditions can be individually identified for each device but their handling and mapping must be specified and implemented following a common rule.

The Table 2 is an estimation of the common diagnosis condition that shall cover the commonalities of every BUI device. The Table 2: Common Diagnosis Condition is the result of merging the current devices diagnosis conditions management and its columns represent:

- **Mnemonics** of the diagnosis alarm condition used in the diagnosis subsystem [1].
- **Type of the Error**, it is a short explanation of the alarm condition.
- **Type of detection**, it is how the diagnosis alarm can be detected inside the device. This column is the merging of different devices types of detection and could be a useful guideline for new device.
- **Classification** of the alarm condition.
- **Group** of the alarm condition.

On the rows there is the typical subsystem in which the alarm is raised.

2.1 Firmware execution non safe state

All the alarm conditions that lead to a non safe state for the firmware execution shall not be managed though a software solution, in particular though the diagnosis subsystem. The suggested reaction to those alarm condition is to execute a software reset. This particular class of diagnosis conditions is shown in Table 1.

TYPE OF ERROR	TYPE OF DETECTION
Communication Board Microcontroller Failure	Compare RAM and NovRAM RamCheck FlashCheck
External Power Supply Alarm	Input power supply readback
Front End Board Microcontroller failure	Compare RAM and NovRAM RamCheck FlashCheck

Table 1: firmware execution non safe state conditions

2.2 Device Specific Diagnosis Conditions

	#	MNEMONIC	TYPE OF ERROR	TYPE OF DETECTION	CLASSIFICATION	GROUP
SENSOR	0	PV_SENSOR_FAILURE	Primary variable sensor failure	Device specific Device specific	FAILURE	HW_STATUS_SENSOR
	1	SV_SENSOR_FAILURE	Secondary variable sensor failure	Device specific Device specific	FAILURE / MAINTENANCE	HW_STATUS_SENSOR
	2	TV_SENSOR_FAILURE	Tertiary variable sensor failure	Device specific Device specific	FAILURE / MAINTENANCE	HW_STATUS_SENSOR
	3	FV_SENSOR_FAILURE	Fourth variable sensor failure	Device specific Device specific	FAILURE / MAINTENANCE	HW_STATUS_SENSOR
	6	PV_SENSOR_OUT_LIMS	Primary variable out of limits	PV > Sensor Limit Hi PV < Sensor Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
	7	SV_SENSOR_OUT_LIMS	Secondary variable out of limits	SV > Sensor Limit Hi SV < Sensor Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
	8	TV_SENSOR_OUT_LIMS	Tertiary variable out of limits	TV > Sensor Limit Hi TV < Sensor Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
	9	FV_SENSOR_OUT_LIMS	Fourth variable out of limits	FoV > Sensor Limit Hi FoV < Sensor Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
APPLICATION	12	PV_SENSOR_OUT_RANGE	Primary variable out of range	PV > Range Limit Hi PV < Range Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
	13	SV_SENSOR_OUT_RANGE	Secondary variable out of range	SV > Range Limit Hi SV < Range Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
	14	TV_SENSOR_OUT_RANGE	Tertiary variable out of range	TV > Range Limit Hi TV < Range Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
	15	FV_SENSOR_OUT_RANGE	Fourth variable out of range	FoV > Range Limit Hi FoV < Range Limit Lo	OFF_SPECIFICATION	OPERATING_CONDITION_PROCESS
F.E.	18	FE_BOARD_FAILURE	Front End Board Failure	ROM CRC Check Frame Timeout	FAILURE	HW_STATUS_ELECTRONICS
	19	FE_NV_FAILURE	Front End NvMem Failure	CRC Check	FAILURE	HW_STATUS_ELECTRONICS

	20	FE_NV_WARNING	Front End NvMem Warning	EEPROM write cycles exceeded	MAINTENANCE	HW_STATUS_ELECTRONICS
	21	FE_BOARD_NOT_DETECTED	Front End Not Detected	No Start-up Communication	FAILURE	HW_STATUS_ELECTRONICS
	22	FE_BOARD_COMM_ERROR	Front End Communication Error	Data CRC Frame counter	FAILURE	HW_STATUS_ELECTRONICS
NV	26	CB_NV_FAILURE	Communication Board NvMem Failure	CRC Check	FAILURE	HW_STATUS_ELECTRONICS
	27	CB_NV_WARNING	Communication Board NvMem Warning	EEPROM write cycles exceeded	MAINTENANCE	HW_STATUS_ELECTRONICS
	30	NV_STORAGE_ACTIVE	NV storage active		CHECK_FUNCTION	HW_STATUS_ELECTRONICS
	31	NV_CONCISTENCY_CHECK	EEPROM Consistency Check	NV Failures states, repair	FAILURE	CONFIG_STATUS

Table 2: Common Diagnosis Condition

3 Communication Protocols Mapping

3.1 HART

The HART protocol provides different methods to communicate diagnosis condition and the device status, those methods and the decision about them will be explained in following paragraph.

3.1.1 Command Status Bytes

All slave response messages must return two Command Status bytes in the first two bytes of Data field. The first byte is multiplexed and contains either the Communication Status or the Response Code. The second byte of a slave response message always contains Field Device Status.
Communication Status is returned if a communication error is detected by the field device.
If there are no communication errors then the Response Code gives the result of the executed command.
The Device Status represents the current state of the slave.
The first two bytes are managed directly by the HART communication subsystem, while the Device Status is managed by the Diagnosis Subsystem.

Device Status	
BIT MASK	**DEFINITION**
0x80	Device Malfunction – The device detected a serious error or failure that compromises device operation
0x40	Configuration Changed
0x20	Cold Start
0x10	More Status Available – More status information is available via Command 48, Read Additional Status Information
0x08	Loop Current Fixed
0x04	Loop Current Saturated
0x02	Non-Primary Variable Out of Limits – A Device Variable not mapped to the PV is beyond its operating limits
0x01	Primary Variable Out of Limits – The PV is beyond its operating limits

3.1.2 Command 48, Read Additional Status Information

CMD 48 Structure			
BYTE	**FORMAT**[1]	**MEANING**[1]	**COMMON DIAGNOSIS**
0-5	Bits or Enum	Device-Specific Status	Device Status (bit string)
6	Bits	Extended Device Status	Not Used
7	Bits	Device Operating Mode	Not Used
8-10	Bits	Analog Channel Saturated	Not Used
11-13	Bits	Analog Channel Fixed	Not Used
14	Bits or Enum	Device-Specific Status	High priority alarm bit number (integer)
15			Classification of high priority alarm bit (bit string)
16			Group of the worst diagnosis condition (bit string)
17–22			Alarm History (bit string)
23-24			Not Used

[1] The format is specified in
HART Communication Foundation Document Number HCF_SPEC-99, Revision 8.0

3.1.3 Command XX, Read Lower Priority Diagnosis Conditions

An additional command shall be added to extract from the device information about lower priority diagnosis conditions.

CMD XX Structure	
SEND	
BYTE	**MEANING**
0	Diagnosis Condition Number (based on device status bit string)
RETURN	
0	Diagnosis Condition Number (based on device status bit string)
1	Classification
2	Group
3	Priority
4 to 5	Occurrences counter
6 to 11	How long the diagnosis condition has been set, expressed in: • Bytes 6 to 7 day • Bytes 8 to 11 msec
12 to 17	Time stamp of the last occurrence. If a real time clock is available this represent the time. If we have an run time hour counter the meaning will be how long ago it happened Expressed in: • Bytes 12 to 13 day • Bytes 14 to 17 msec
18 to 31	Diagnosis Condition Composition (list of enumerated)

3.2 Fieldbus Foundation

Every FF device supporting common diagnosis shall have:

- A resource block containing diagnosis conditions coming from electronics.
- A device transducer block containing diagnosis conditions coming from the measurement variable production process and from the sensor electronics if it is physically divided (it could be substituted with another sensor).
- An HMI transducer block containing diagnosis conditions coming from the HMI.
- An advanced transducer block (if needed) containing diagnosis conditions coming from advanced algorithms that produce more sophisticated information.

The FF protocol provides different methods to communicate diagnosis condition and the device status, those methods and the decision about them will be explained in following paragraph.

3.2.1 Alert

Alert objects generate notification messages when alarms or events are detected. Alert objects report alarm and event occurrences using the FMS Event Notification service in a manner that ensures that alert messages are not lost. Two types of alerts are defined, events and alarms. Events are used to report a status change when a value in a function block changes state. Alarms are used to report a status change when a function block value enters an abnormal state and also when it returns to normal.
Every ABB FF devices shall have multi-alert notification.

The alert notification shall contain classification of the diagnosis condition, the variable used will be the **subcode** using this structure (if the alert notification value could be used and read by all the asset monitors the grouping could be moved inside the value):

	NAMUR		GROUPS	ERROR CODES
	Bit 15	Bit 14	Bit 13 to Bit 10	Bit 9 to bit 0
DIAGNOSIS_CLASS_CHECK_FUNCTION	0	0		0 – 1024
DIAGNOSIS_CLASS_OFF_SPECIFICATION	0	1		0 – 1024
DIAGNOSIS_CLASS_MAINTENANCE	1	0		0 – 1024
DIAGNOSIS_CLASS_FAILURE	1	1		0 – 1024

3.2.2 Additional variables

Each block (resource or transducer) shall have a variable (bit-string) to store the active diagnosis conditions.

INDEX	PARAMETER NAME		DATA TYPE	NOTES
48	DIAG_WORST_COND		UNSIGNED16	
49	DIAG_EXT		BIT_STRING	
50	DIAG_EXT_HISTORY		BIT_STRING	
51	DIAG_CONDITION_IDX		UNSIGNED8	
52	DIAG_DETAILS	COUNTER	UNSIGNED16	Custom Data Type: Diagnosis Detailed History Data Type
		TIME_COUNTER	TIME_DIFF_S	
		LAST_TIME	DATE_S	
53	DIAG_SIMUL_STATUS		UNSIGNED8	Enable/Disable Simulation
54	DIAG_SIMUL		BIT_STRING	Set simulation conditions
55	DIAG_MASK		BIT_STRING	Mask simulation conditions

The meaning of variables is explained in the following tables.

This variable is necessary to identify the worst diagnosis condition occurred to the device and to drive, inside the DTM/EDD, the description, the possible cause and the suggested action.

DIAG_WORST_COND

	NAMUR		GROUPS	ERROR CODES
	Bit 15	Bit 14	Bit 13 to Bit 10	Bit 9 to bit 0
DIAGNOSIS_CLASS_CHECK_FUNCTION	0	0		0 – 1024
DIAGNOSIS_CLASS_OFF_SPECIFICATION	0	1		0 – 1024
DIAGNOSIS_CLASS_MAINTENANCE	1	0		0 – 1024
DIAGNOSIS_CLASS_FAILURE	1	1		0 – 1024

The diagnosis extension is necessary to show which diagnosis conditions are set more than the diagnosis worst condition.

DIAG_EXT

The diagnosis extension history is necessary to remember which diagnosis conditions have been occurred till now and to drive the details reading.

DIAG_EXT_HISTORY

The details of the diagnosis condition shall be accessed writing the DIAG_CONDITION_IDX and reading the values in the following variables. This policy was chosen because, with the same variables, the user can read either common or device specific alarm condition details. The advantage is that we don't have to make any device specific variable for device specific diagnosis condition.

		Example 1	Example 2
DIAG_CONDITION_IDX		PV out of limits	Communication Error
DIAG_DETAILS	COUNTER	4	2
	TIME_COUNTER	10 min 21 sec 45 msec	14 min 1 sec 5 msec
	LAST_TIME[2]	20/11/2006 21.34	02/08/2006 11.25

[2] If a real time clock is available, else a format like DIAGNOSIS_CONDITION_DETAILS_TIME_COUNTER will be used.

3.3 Profibus PA

All the diagnosis information available on Profibus PA devices shall be stored into the physical block. Moreover the diagnosis variable defined by the profile, new variable shall be added to the standard physical block described in [8] and they are:

INDEX	PARAMETER NAME		DATA TYPE	NOTES
35	DIAG_WORST_COND		UNSIGNED16	
36	DIAG_EXT_HISTORY		OCTET_STRING	
37	DIAG_CONDITION_IDX		UNSIGNED8	
38	DIAG_DETAILS	COUNTER	UNSIGNED16	Custom Data Type: Diagnosis Detailed History Data Type
		TIME_COUNTER	TIME_DIFF_S	
		LAST_TIME	DATE_S	
39	DIAG_SIMUL_STATUS		UNSIGNED8	Enable/Disable Simulation
40	DIAG_SIMUL		OCTET_STRING	Set simulation conditions
41	DIAG_MASK		OCTET_STRING	Mask simulation conditions

The meaning of those variables is explained in the following tables.

This variable is necessary to identify the worst diagnosis condition occurred to the device and to drive, inside the DTM/EDD, the description, the possible cause and the suggested action.

DIAG_WORST_COND

	NAMUR		GROUPS	ERROR CODES
	Bit 15	Bit 14	Bit 13 to Bit 10	Bit 9 to bit 0
DIAGNOSIS_CLASS_CHECK_FUNCTION	0	0		0 – 1024
DIAGNOSIS_CLASS_OFF_SPECIFICATION	0	1		0 – 1024
DIAGNOSIS_CLASS_MAINTENANCE	1	0		0 – 1024
DIAGNOSIS_CLASS_FAILURE	1	1		0 – 1024

The diagnosis extension history is necessary to remember which diagnosis conditions have been occurred till now and to drive the details reading.

DIAG_EXT_HISTORY

The details of the diagnosis condition shall be accessed writing the DIAG_CONDITION_IDX and reading the values in the following variables. This policy was chosen because, with the same variables, the user can read either common or device specific alarm condition details. The advantage is that we don't have to make any device specific variable for device specific diagnosis condition.

		Example 1	Example 2
DIAG_CONDITION_IDX		PV out of limits	Communication Error
DIAG_DETAILS	COUNTER	4	2
	TIME_COUNTER	10 min 21 sec 45 msec	14 min 1 sec 5 msec
	LAST_TIME[3]	20/11/2006 21.34	02/08/2006 11.25

[3] If a real time clock is available, else a format like DIAGNOSIS_CONDITION_DETAILS_TIME_COUNTER will be used.

PROFIBUS Specification - Amendment 2 to the Profibus Profile for Process Control Devices V 3.01 - Condensed Status and Diagnostic Messages V 1.0 June 2005 shall be used as example for computing variable status.

3.3.1 Diagnosis

Octet	Bit	DIAGNOSIS Mnemonic	Description	EXT_DIAG to be set	DIAGNOSIS CLASS
1	0-7		reserved by PNO, fixed to 0		
2	0-2		reserved by PNO, fixed to 0		
	3	DIA_WARMSTART		no	
	4	DIA_COLDSTART		no	
	5	DIA_MAINTENANCE	Maintenance required	no	MAINTENANCE
	6		reserved by PNO, fixed to 0		
	7	IDENT_NUMBER_VIOLATION		no	
3	0	DIA_MAINTENANCE_ALARM	Failure of the device or armature	yes	FAILURE
	1	DIA_MAINTENANCE_DEMANDED	Maintenance demanded	no	Not Used
	2	DIA_FUNCTION_CHECK	Device is in function check or in simulation or under local control e.g. maintenance	no	CHECK_FUNCTION
	3	DIA_INV_PRO_COND	The process conditions don't allow returning valid values. (Set if a value has the quality Uncertain - Process related, no maintenance or Bad - Process related, no maintenance	no	OFF_SPECIFICATION
	4-7		reserved by PNO, fixed to 0		
4	0-6		reserved by PNO, fixed to 0		
	7	EXTENSION_AVAILABLE	0: There is no more information available 1: More diagnosis information is available in DIAGNOSIS_EXTENSION		

3.3.2 Diagnosis Extension

See Device Specific Diagnosis Conditions for details.

Octet	Description
1	Common Diagnosis Conditions
2	
3	
4	Device Specific Conditions
5	
6	

4 Revision Chart

Rev.	Description of Version/Changes	Primary Author(s)	Date
0.4	First Draft (the revision start from 0.4 to keep this document consistent with other Common Diagnosis Documents)	Giovanni Invernizzi	2007 March 19
0.5	Second Draft after first complete implementation, modified command XX structure.	Giovanni Invernizzi	2008 December 17