Diagnostic buffer layout in SVI FF

Versions 0 and 1

The diagnostic buffer is a large array of notionally int16\_t type little-endian entries.

It contains a header and data payload. Interpretation of data payload depends on the header.

# The first header

[0] = type of diagnostic data

[1] = version of diagnostic of that type

[2] = size of the header (in int16\_t, i.e. 2-byte, elements) N

…

[N-4] = high halfword of sampling interval in 5-ms ticks (normally, 0, except extreme ranges of data collection) – **version 1 only**.

[N-3] = consecutive index of the buffer use (auto-increments). Resets to 0 on device reset

[N-2][N-1] = “HART” device id of the device, as a byte sequence.

After that, a second header may follow, up to the size of the header.

Unused entries of the header are filled with a signature 0x8081.

# Types of diagnostic data

The following types are supported:

* 2 – Extended actuator signature
* 3 – Step test
* 4 – Ramp (response) test
* 0x5500 – Data collection

Note: Partial Stroke test is a subtype of data collection.

# Extended actuator signature

## Header

[0] = 2 – Extended actuator signature

[1] = 0 (version)

[2] = 24 (header size)

[3] = number of samples

[4] = Start Position, %

[5] = End Position, %

[6] = Setpoint Ramp Speed, in %/sec

[7] = Sampling interval in 5 ms ticks, low halfword

[8] = Direction: 0-both directions, 1-one direction

[9] = Control Option: 0-open loop, 1-closed loop, all others invalid

[10] = Samples in First Direction (if up/down test, where is the end of first direction)

The header is not populated until the test completes

## Sample in the data payload

Each sample is an array of two int16\_t:

[0] = position, %

[1] = main pressure, psi

# Step test

## Header

[0] = 3 – Step test

[1] = 0 (version)

[2] = 24 (header size)

[3] = number of samples

[4] = Start Position, %

[5] = End Position, %

[6] = Sampling time (per configuration)

[7] = Sampling interval in 5 ms ticks, low halfword

The header is not populated until the test completes

## Sample in the data payload

Each sample is an array of two int16\_t:

[0] = setpoint, %

[1] = position, %

# Ramp test

## Header

[0] = 4 – Ramp test

[1] = 0 (version)

[2] = 24 (header size)

[3] = number of samples

[4] = Start Position, %

[5] = End Position, %

[6] = Setpoint Ramp Speed, in %/sec

[7] = Sampling interval in 5 ms ticks, low halfword

[8] = Direction: 0-both directions, 1-one direction

[9] = Samples in First Direction (if up/down test, where is the end of first direction)

The header is not populated until the test completes

## Sample in the data payload

Each sample is an array of two int16\_t:

[0] = setpoint, %

[1] = position, %

# Data collection

## Header

[0] = 0x5500 – Data collection

[1] = 0 (version)

[2] = 24 (header size)

[3] = number of samples (incremental)

[4] = Bitmap of data in a sample

0 – position, %

1 – main pressure, psi

2 – pilot pressure, psi

3 – supply pressure, psi

4 – actuator pressure B, psi

5 – Setpoint, %

6 – I/P current, mA

7 – Analog input, mA

[5] = Sampling interval in 5 ms ticks, low halfword

[6] = Number of pre-samples

[7] = Process ID on whose behalf data is collected

0 – standalone data collection

16 – PST

The header is populated on the fly and data can be read in the background, while collection is running.

Number of samples [3] increments automatically. Care should be taken not to read more samples than the header indicates.

Data collection completion is manifested by unchanging number of samples.

## Sample in the data payload

Each sample is an array of int16\_t. The number of elements in the array is the number of 1’s in header[4] bitmap, and the n-th element must be interpreted as per n-th bit set in the bitmap.

E.g. if bitmap is 0x31, the sample contains

[0] = position, %

[1] = pressure B, psi

[2] = setpoint, %

# Partial Stroke Test

PST extends data collection header as follows:

[8] = 0 (PST version)

[9] = test result

0 – OK

1 – canceled

2 – failed

[10] = position at the start of the test, %

[11] = position at the end of the test, %

[12] = completion code

0 – OK

3 – canceled by user

4 – could not stabilize valve within timeout

5 – overall timeout running the test

6 – current faults prevent PST to run

7 – PST not allowed

[13] = Dwell (pause) time per configuration

[14] = travel magnitude, %, per configuration

[15] = ramp speed, %/s, per configuration

The PST header extension is not populated until the test ends. However, the samples – and data collection header – are valid and may be retrieved on the fly.

## Sample in the data payload

It is the same as in data collection data payload.

It is worth noting that bitmap of collected data is taken from PST configuration, not data collection configuration.

# Retrieving the diagnostic buffer from the device.

Parameter DIAGNOSTIC\_DATA is used to retrieve the content of the diagnostic buffer piecemeal.

It is an array of 27 uint16\_t elements laid out as follows:

[0] - Contains the offset in bytes of the data to read from the beginning of the diagnostic data buffer,

[1] - Skip count - number of points skipped for each point read

[2] - Data Sample Count - number of valid points read from the FW

[3] - [26] – a segment of Diagnostic data payload from the buffer.

[0] is a writable value in the array and may be used to initiate the read procedure, by writing 0, in which case the diagnostic header is read. The value is autoincremented after each read of the array with sub-index 0, provided that useful data were read.

The read routine should compare [0] with expected value (previous [0] + 48\*(skip count)) and repeat the read in case of mismatch which would normally indicate a message loss.

[1] is an optional writable value meaning how many samples of dimension 2 to skip. A non-zero value is not very useful for data collection and PST, unless exactly two variables are being collected. Its main purpose is to provide a quick preview of results of tests which do not populate the header until completion. For data collection (and PST) this must be 0, except if samples contain exactly two variables.

To guide the buffer retrieval, the [first diagnostic header](#_The_first_header) provides consecutive buffer index. It must be the same at the end of data retrieval as in the beginning – a mismatch indicates that the device was reset or the buffer meaning changed during the reads, and the content read is inconsistent and should be discarded.

The header also provides HART-style device id to help identify the source of data.