HID Sensors Usages

Annotations for Windows HID Sensor Class Driver

October 22, 2015

Abstract

This paper provides information about the HID Sensor Class driver for Windows 8.1 and later operating systems. It provides guidelines for developing sensor hardware and firmware that take full advantage of and work correctly with the in-box driver.

This information applies to the following operating systems:   
 Windows 10

Windows 8.1

References and resources discussed here are listed at the end of this paper.

The current version of this paper is maintained on the Web at:   
 [HID Sensors Usages](http://msdn.microsoft.com/en-us/library/windows/hardware/hh975383)

**Disclaimer**: This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet website references, may change without notice. Some information relates to pre-released product which may be substantially modified before it’s commercially released. Microsoft makes no warranties, express or implied, with respect to the information provided here. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes. You may modify this document for your internal, reference purposes.

© 2015 Microsoft. All rights reserved.

C:\Users\jenlin\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Outlook\KN5ONHWU\dep_MicrosoftLogotype.png

Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Change |  |  |  |
| October 22, 2015 | Updated: Section 1.5.1 with a **Note**; **Table 5**; Section 4.2.2. Also made general edits throughout the document. | | | |
| June 14, 2013 | Removed unsupported sensors, added magnetometer accuracy usages, updated documentation of per-data-field modifiers, general edits throughout | | | |
| September 28, 2012 | Updated to reflect rebranding | | | |
| July 13, 2012 | Updated HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STAT | | | |
| May 31, 2012 | First publication | | | |

Contents

[0. Introduction 4](#_Toc358980703)

[0.1 Summary 4](#_Toc358980704)

[0.2 Background 4](#_Toc358980705)

[0.3 Approach 4](#_Toc358980706)

[1. Sensor Page (0x20) 5](#_Toc358980707)

[1.1 Sensor Device Usages 5](#_Toc358980708)

[1.2 Sensor Field Usages: Modifiers 10](#_Toc358980709)

[1.3 Sensor Field Usages: States 10](#_Toc358980710)

[1.4 Sensor Field Usages: Events 11](#_Toc358980711)

[1.5 Sensor Field Usages: Properties 11](#_Toc358980712)

[1.5.1 HID sensor property usages 12](#_Toc358980713)

[1.5.2 HID sensor property defaults 25](#_Toc358980714)

[1.5.3 HID sensor input report properties and datafield usages 33](#_Toc358980715)

[1.6 Biometric Sensor Field Usages 35](#_Toc358980716)

[1.7 Electrical Sensor Field Usages 36](#_Toc358980717)

[1.8 Environmental Sensor Field Usages 36](#_Toc358980718)

[1.9 Light Sensor Field Usages 37](#_Toc358980719)

[1.10 Location Sensor Field Usages 38](#_Toc358980720)

[1.11 Mechanical Sensor Field Usages 39](#_Toc358980721)

[1.12 Motion Sensor Field Usages 39](#_Toc358980722)

[1.13 Orientation Sensor Field Usages 41](#_Toc358980723)

[1.14 Scanner Sensor Field Usages 47](#_Toc358980724)

[1.15 Time Sensor Field Usages 47](#_Toc358980725)

[1.16 Custom Sensor Field Usages 47](#_Toc358980726)

[1.17 Generic Sensor Field Usages 54](#_Toc358980727)

[2. Sensor Backgrounder 54](#_Toc358980728)

[2.1 Glossary 55](#_Toc358980729)

[2.2 Sensor Taxonomy and Object Model 55](#_Toc358980730)

[3. Sensor Interaction via HID 55](#_Toc358980731)

[3.1 Related Documents 55](#_Toc358980732)

[3.2 Functional Overview 55](#_Toc358980733)

[3.3 HID Logical Devices 55](#_Toc358980734)

[3.4 HID Reports 56](#_Toc358980735)

[3.5 HID Report IDs 56](#_Toc358980736)

[3.6 HID Report Items 56](#_Toc358980737)

[3.6.1 HID Report Item packing options 56](#_Toc358980738)

[3.7 HID Usages 56](#_Toc358980739)

[3.7.1 HID Usage Types 56](#_Toc358980740)

[3.7.2 HID Selectors 56](#_Toc358980741)

[3.8 HID Usage Page 57](#_Toc358980742)

[3.9 HID Units 57](#_Toc358980743)

[3.10 HID Unit Exponents 58](#_Toc358980744)

[3.11 3D Coordinates and Compass Points 58](#_Toc358980745)

[4. Illustrative Examples 58](#_Toc358980746)

[4.1 Include File Definitions 58](#_Toc358980747)

[4.2 Special Constructions 83](#_Toc358980748)

[4.2.1 Values, Types, and Unit Exponents 83](#_Toc358980749)

[4.2.2 Extended Properties 85](#_Toc358980750)

[4.2.3 Modifiers: Per-datafield Properties 92](#_Toc358980751)

[4.2.4 Event Thresholds 96](#_Toc358980752)

[4.2.5 Sensor Collections 96](#_Toc358980753)

[4.2.6 Custom Sensor 99](#_Toc358980754)

[4.3 Illustrative Sensor Report Descriptors 110](#_Toc358980755)

[4.3.1 Biometric: Human Presence 111](#_Toc358980756)

[4.3.2 Biometric: Human Proximity 113](#_Toc358980757)

[4.3.3 Environmental: Atmospheric Pressure 116](#_Toc358980758)

[4.3.4 Environmental: Humidity 119](#_Toc358980759)

[4.3.5 Environmental: Temperature 122](#_Toc358980760)

[4.3.6 Light: Ambient Light 125](#_Toc358980761)

[4.3.7 Location: GPS 129](#_Toc358980762)

[4.3.8 Motion: Accelerometer 129](#_Toc358980763)

[4.3.9 Motion: Gyrometer 132](#_Toc358980764)

[4.3.10 Orientation: Compass 135](#_Toc358980765)

[4.3.11 Orientation: Inclinometer 140](#_Toc358980766)

[4.3.12 Orientation: Device Orientation 143](#_Toc358980767)

[5. Sensor Implementation and Debugging – Tips & Tricks 147](#_Toc358980768)

[5.1 Using required and optional datafields 147](#_Toc358980769)

[5.1.1 Out-of-range datafield values 148](#_Toc358980770)

[5.1.2 Controlling datafield values 148](#_Toc358980771)

[5.2 Using dynamic datafields 149](#_Toc358980772)

[5.3 Using dynamic properties 149](#_Toc358980773)

[5.4 Using Custom sensors and datafields 150](#_Toc358980774)

[5.5 Setting device properties 150](#_Toc358980775)

[5.6 Debugging a sensor or sensor collection 151](#_Toc358980776)

[5.7 Using sensor logging 152](#_Toc358980777)

# Introduction

## Summary

This is a companion document to the HID Sensor Usage Tables and HID Accuracy documents. Its purpose is to provide an annotated guide to the HID Sensor Page and explain what is and is not supported by the Windows 8.1 Preview HID Sensor Class Driver and how that support is used by a sensor device.

## Background

The [HID Sensor Usage Tables](http://www.usb.org/developers/hidpage/HUTRR39b.pdf) and [HID Accuracy](http://www.usb.org/developers/hidpage/HUTRR44Final-MagnetometerAccuracy.pdf) documents describes HID Usages for the Sensor Page. While this document goes into great detail about the various usage definitions, it is less prescriptive about how hardware developers use the information to build sensor devices that are compliant with the HID specification and with device drivers that support those sensors. This companion document is intended as a supplement to provide prescriptive guidance specifically for the Windows HID Sensor Class Driver and how to access the sensors defined by this specification from the Windows Sensor API.

For more information, see the Windows [Sensor API](http://msdn.microsoft.com/en-us/library/dd318953(v=VS.85).aspx) documentation.

This document does not specify the means by which the sensor gathers the data reported to the driver. For guidance on how to gather sensor data that is useful to the Windows operating system, see the following documents:

* Motion Sensor applications

<http://msdn.microsoft.com/en-us/library/windows/hardware/br259127.aspx>

* Ambient Light Sensor applications

<http://msdn.microsoft.com/en-us/windows/hardware/gg463470>

* More information on the HID Sensor Class Driver

http://msdn.microsoft.com/en-us/library/windows/hardware/jj128407(v=vs.85).aspx

## Approach

The organization of the HID Sensor Usage Tables document has been preserved in this document but, for the most part, the content of that document is not repeated. Rather, this companion document provides additional clarifying content and implementation recommendations.

Throughout this document, the following terms are used for the sake of brevity:

* **API**: The Windows Sensor Platform API
* **Client**: An application communicating with a sensor through the API
* **Device**: A sensor device implemented to conform to the Specification and the API
* **Driver**: The Windows HID Sensor Class driver
* **Implementer**: The organization implementing a device
* **PC**: The system hosting the Windows operating system
* **Sensor**: An individual sensor implemented on a device. A device may support more than one sensor, and may support more than one sensor of the same category and type
* **Specification**: The HID Sensor Usage Tables document

# Sensor Page (0x20)

The sensor page defined in the Specification provides usages for sensors. There are no sensor page additions to those defined in the Specification.

## Sensor Device Usages

The Windows Sensor API describes sensors as being comprised of a Sensory Category (a GUID), a Sensor Type (a GUID), and a collection of Sensor Datafields (each a PROPERTYKEY.) There is no explicit mapping between the Sensor Type and the Sensor Datafield. As far as the API is concerned, any Sensor Type can contain any collection of Sensor Datafields. Table 1 provides the recommended mapping for each supported Sensor Category/Type and the collection of supported Sensor Datafields. This table further extends the mapping to include the relationship between the HID Sensor Usage and the Sensor Category/Type.

When it is stated that a datafield is required for the sensor at the API, the equivalent HID Usage must be available in the input report as defined by the input report descriptor. If it is not present, the value for that datafield at the API is VT\_EMPTY. When it is stated that a datafield is optional for the sensor at the API, the equivalent HID Usage may be available in the input report as defined by the input report descriptor. If it is not present in the input report, that datafield does not appear at the API.

Table 1. HID sensor usage to Windows sensor datafield mapping

|  |  |  |
| --- | --- | --- |
| **Driver supports the following Sensor Device Usages: Usage ID** | **Windows Sensor Category and Type** | **Windows Sensor Datafields** |
| **Time**  No equivalent HID Usage | All Sensor Categories  All Sensor Types | Required datafields  SENSOR\_DATA\_TYPE\_TIMESTAMP  Note: this datafield will be present in all sensors. The value of this is assigned by the driver when an input packet is received by the driver.  The driver does not support the “time” usages in the specification. See section 1.15. |
| **Biometric: Human Presence**  **0x11**  HID\_USAGE\_SENSOR\_TYPE\_BIOMETRIC\_PRESENCE | SENSOR\_CATEGORY\_BIOMETRIC  SENSOR\_TYPE\_HUMAN\_PRESENCE | Required datafields:  SENSOR\_DATA\_TYPE\_HUMAN\_PRESENCE  Optional datafields: none |
| **Biometric: Human Proximity**  **0x12**  HID\_USAGE\_SENSOR\_TYPE\_BIOMETRIC\_PROXIMITY | SENSOR\_CATEGORY\_BIOMETRIC  SENSOR\_TYPE\_HUMAN\_PROXIMITY | Required datafields:  SENSOR\_DATA\_TYPE\_HUMAN\_PROXIMITY\_METERS  Optional datafields: none |
| **Environmental: Atmospheric Pressure**  **0x31**  HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE | SENSOR\_CATEGORY\_ENVIRONMENTAL  SENSOR\_TYPE\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE | Required datafields:  SENSOR\_DATA\_TYPE\_ATMOSPHERIC\_PRESSURE\_BAR  Optional datafields: none |
| **Environmental: Humidity**  **0x32**  HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_HUMIDITY | SENSOR\_CATEGORY\_ENVIRONMENTAL  SENSOR\_TYPE\_ENVIRONMENTAL\_HUMIDITY | Required datafields:  SENSOR\_DATA\_TYPE\_RELATIVE\_HUMIDITY\_PERCENT  Optional datafields: none |
| **Environmental: Temperature**  **0x33**  HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_TEMPERATURE | SENSOR\_CATEGORY\_ENVIRONMENTAL  SENSOR\_TYPE\_ENVIRONMENTAL\_TEMPERATURE | Required datafields:  SENSOR\_DATA\_TYPE\_TEMPERATURE\_CELSIUS  Optional datafields: none |
| **Light: Ambient Light**  **0x41**  HID\_USAGE\_SENSOR\_TYPE\_LIGHT\_AMBIENTLIGHT | SENSOR\_CATEGORY\_LIGHT  SENSOR\_TYPE\_AMBIENT\_LIGHT | Required datafields:  SENSOR\_DATA\_TYPE\_LIGHT\_LEVEL\_LUX  Optional datafields:  SENSOR\_DATA\_TYPE\_LIGHT\_TEMPERATURE\_KELVIN  SENSOR\_DATA\_TYPE\_LIGHT\_CHROMACITY |
| **Motion: Accelerometer 3D**  **0x73**  HID\_USAGE\_SENSOR\_TYPE\_MOTION\_ACCELEROMETER\_3D | SENSOR\_CATEGORY\_MOTION  SENSOR\_TYPE\_ACCELEROMETER\_3D | Required datafields:  SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Y\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Z\_G  Optional datafields: none |
| **Motion: Gyrometer 3D**  **0x76**  HID\_USAGE\_SENSOR\_TYPE\_MOTION\_GYROMETER\_3D | SENSOR\_CATEGORY\_MOTION  SENSOR\_TYPE\_GYROMETER\_3D | Required datafields:  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_X\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Y\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Z\_DEGREES\_PER\_SECOND  Optional datafields: none |
| **Orientation: Compass 3D**  **0x83**  HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_COMPASS\_3D | SENSOR\_CATEGORY\_ORIENTATION  SENSOR\_TYPE\_COMPASS\_3D | Required datafields:  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_MAGNETIC\_NORTH\_DEGREES  Optional datafields:  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_TRUE\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_MAGNETIC\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_TRUE\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_X\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Y\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Z\_MILLIGAUSS |
| **Orientation: Inclinometer 3D**  **0x86**  HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_INCLINOMETER\_3D | SENSOR\_CATEGORY\_ORIENTATION  SENSOR\_TYPE\_INCLINOMETER\_3D | Required datafields:  SENSOR\_DATA\_TYPE\_TILT\_X\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Y\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Z\_DEGREES  Optional datafields: none |
| **Orientation: Device Orientation**  **0x8A**  HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_DEVICE\_ORIENTATION | SENSOR\_CATEGORY\_ORIENTATION  SENSOR\_TYPE\_AGGREGATED\_DEVICE\_ORIENTATION | Required datafields:  SENSOR\_DATA\_TYPE\_QUATERNION  Optional datafields:  SENSOR\_DATA\_TYPE\_ROTATION\_MATRIX |
| **Other: Custom**  **0xE1**  HID\_USAGE\_SENSOR\_TYPE\_OTHER\_CUSTOM | SENSOR\_CATEGORY\_OTHER  SENSOR\_TYPE\_CUSTOM | Required datafields:  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE1  Optional datafields:  SENSOR\_DATA\_TYPE\_CUSTOM\_USAGE  SENSOR\_DATA\_TYPE\_CUSTOM\_BOOLEAN\_ARRAY  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE2  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE3  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE4  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE5  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE6 |

## Sensor Field Usages: Modifiers

The fields listed in Table 2 are optionally supported by the driver for all sensors.

Table 2. Selection values for Sensor Modifier Usage

|  |  |
| --- | --- |
| **Modifier** | **Description** |
| **Modifier: None** | US – The information contained in the data field is the unmodified meaning for that data field. |
| **Modifier: Change Sensitivity Absolute** | US – Specifies the change sensitivity set for a particular data field. Units are the same as the data field being modified. For example, if the data field is “Temperature, Degrees Celsius”, and the absolute sensitivity is “3” then that would mean “change of ±3 Degrees Celsius”. |
| **Modifier: Maximum** | US – The information contained in the data field is the maximum value for that data field. |
| **Modifier: Minimum** | US – The information contained in the data field is the minimum value for that data field. |
| **Modifier: Accuracy** | US – The information contained in the data field specifies the absolute accuracy with which that data field is reported. |
| **Modifier: Resolution** | US – The information contained in the data field specifies the absolute precision with which that data field is reported. |
| **Modifier: Change Sensitivity Percent Relative** | US – Specifies the change sensitivity set for a particular data field. Units are a percentage of the “prior reading”. For example, if the data field is “Temperature, Degrees Celsius”, the prior reading was +24.0, and the percent relative sensitivity is “4” then that would mean “change of 4% from 24.0 Degrees Celsius”, (i.e., ±0.96 Degrees Celsius). |

## Sensor Field Usages: States

The fields listed in Table 3 are supported by the driver for all sensors. The meaning is common for all sensors.

Table 3. Selection values for Sensor State Usage

|  |  |  |
| --- | --- | --- |
| **Sel Usage** | **State Name** | **Comment** |
| 0x 0800 | Unknown | The sensor state is unknown. |
| 0x 0801 | Not Available | The sensor not available. |
| 0x 0802 | Ready | Sensor is able to provide new complete and accurate data. |
| 0x 0803 | No Data | The sensor is available, but is not yet providing data. It is not known in what timeframe data will, if ever, be provided. |
| 0x 0804 | Initializing | The sensor is available, but is not yet providing data due to initialization activities. It is expected the sensor will provide data, but the timeframe in which that data will be available is not known. |
| 0x 0805 | Access Denied | In the case where an ID must be provided to access sensor data, and the requester fails to match the ID, this state will be returned. |
| 0x 0806 | Error | The sensor has encountered a major error. The sensor may recover from the state, but the time frame for recovery is unknown. |

## Sensor Field Usages: Events

The fields listed in Table 4 are supported by the driver for all sensors. The meaning is common for all sensors.

Table 4. Selection values for Sensor Event Usage

|  |  |  |
| --- | --- | --- |
| Sel Usage | Event Name | Comment |
| 0x 0810 | Unknown | The sensor event type is not known |
| 0x 0811 | State Changed | The sensor state as specified in (1.3) has changed |
| 0x 0812 | Property Changed | A property value has changed |
| 0x 0813 | Data Updated | A data field has changed |
| 0x 0814 | Poll Response | The most current sensor data is being returned as the result of a poll request (Get Input) |
| 0x 0815 | Change Sensitivity | The change sensitivity has been exceeded for a data field |

## Sensor Field Usages: Properties

These fields are supported by the driver for all sensors. The meaning is common for all sensors except where noted. From the perspective of the sensor device, all properties are optional: a valid sensor can be created that supports no properties whatsoever (for example, no defined Feature Report) but a sensor defined in this manner will not pass Windows Hardware Certification requirements.

However, from the perspective of the API, all the listed properties are required. In those cases, where a property is not present in the feature report, a default value is created by the driver and is present for that sensor at the API.

In order to pass Windows Hardware Certification requirements, a property with the annotation “Usage is required for certification” must be present in the feature report as defined by the feature report descriptor for that sensor.

### HID sensor property usages

Each HID usage that describes a property maps to an equivalent Windows Sensor API property. This section specifies that mapping.

**NOTE:** If you are developing sensors and/or firmware for Windows 10, *and* you are working with a custom sensor, then the following usage properties are mandatory:

* HID\_USAGE\_SENSOR\_PROPERTY\_FRIENDLY\_NAME
* HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MANUFACTURER
* HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MODEL

Table 5. HID sensor usage to Windows sensor property mapping

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Property** |
| **Friendly Name**  **0x0301**  HID\_USAGE\_SENSOR\_PROPERTY\_FRIENDLY\_NAME | Wide character zero-terminated string up to 31 characters  Usage is optional.  Usage is required for *custom sensors* on Windows 10. | SENSOR\_PROPERTY\_FRIENDLY\_NAME  Default value is provided by the driver.  Value is read-only |
| **Persistent Unique ID**  **0x0302**  HID\_USAGE\_SENSOR\_PROPERTY\_PERSISTENT\_UNIQUE\_ID | Wide character zero-terminated string up to 31 characters  Usage is optional. | SENSOR\_PROPERTY\_PERSISTENT\_UNIQUE\_ID  Default value is provided by the driver.  Value is read-only |
| **Sensor State**  **0x0201**  HID\_USAGE\_SENSOR\_STATE | Enumerated values as described in section 1.3  Usage is required for Windows Hardware Certification. | SENSOR\_PROPERTY\_STATE |
| **Minimum Report Interval**  **0x0304**  HID\_USAGE\_SENSOR\_PROPERTY\_MINIMUM\_REPORT\_INTERVAL | 8-bit, 16-bit or 32-bit unsigned integer value  Default Unit is milliseconds  Usage is optional. | SENSOR\_PROPERTY\_MIN\_REPORT\_INTERVAL  Default value is provided by the driver.  Value is read-only |
| **Sensor Manufacturer**  **0x0305**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MANUFACTURER | Wide character zero-terminated string up to 31 characters  Usage is optional.  Usage is required for *custom sensors* on Windows 10. | SENSOR\_PROPERTY\_MANUFACTURER  Default value is provided by the driver.  Value is read-only |
| **Sensor Model**  **0x0306**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MODEL | Wide character zero-terminated string up to 31 characters  Usage is optional.  Usage is required for *custom sensors* on Windows 10. | SENSOR\_PROPERTY\_MODEL  Default value is provided by the driver.  Value is read-only |
| **Sensor Serial Number**  **0x0307**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_SERIAL\_NUMBER | Wide character zero-terminated string up to 31 characters  Usage is optional. | SENSOR\_PROPERTY\_SERIAL\_NUMBER  Default value is provided by the driver.  Value is read-only |
| **Sensor Description**  **0x0308**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_DESCRIPTION | Wide character zero-terminated string up to 31 characters  Usage is optional. | SENSOR\_PROPERTY\_DESCRIPTION  Default value is provided by the driver.  Value is read-only |
| **Sensor Connection Type**  **0x0309**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE | Enumerated values as described in the clarifying notes at the end of this section.  Usage is required for Windows Hardware certification. | SENSOR\_PROPERTY\_CONNECTION\_TYPE  Default value is provided by the driver.  Value is read-only |
| **Report Interval**  **0x030E**  HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL | 32-bit unsigned value  Default Unit is milliseconds  Usage is required for Windows Hardware certification. | SENSOR\_PROPERTY\_CURRENT\_REPORT\_INTERVAL  Default value is provided by the driver.  Value is read/write. |
| **Reporting State**  **0x0316**  HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE | Enumerated values as described in the clarifying notes of this section.  Usage is required for Windows Hardware certification. See notes on using Reporting State in the clarifying notes of this section | Note: no corresponding API property. Refer to the clarifying notes of this section  Value is read/write. |
| **Response Curve**  **0x0318**  HID\_USAGE\_SENSOR\_PROPERTY\_RESPONSE\_CURVE | An array of pairs of 16-bit integer or fixed point values. See the clarifying notes of this section.  Default Unit is not specified  Usage is optional. See notes on specifying the response curve in the clarifying notes of this section. | SENSOR\_PROPERTY\_LIGHT\_RESPONSE\_CURVE  Note: Response Curve is only supported as a property by Ambient Light Sensors  Default values are provided by the Sensor API.  Value is read-only |
| **Power State**  **0x0319**  HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE | Enumerated values as described in the clarifying notes of this section.  Usage is required for Windows Hardware certification. See notes on using Power State in the clarifying notes of this section. | Note: no corresponding API property. Refer to the clarifying notes of this section.  Value is read/write. |
| **HID Usage**  **0x0541**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE | NOTE: the behavior here is specific to this datafield being used as a dynamic property.  To be available as a property at the API, this datafield must be present in an **input** report.  8-bit, 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Usage is optional. See notes on dynamic properties in section 5.3. | SENSOR\_PROPERTY\_HID\_USAGE  Default value is n/a. If this field is not specified in the input report, it is not present at the API.  Value is read-only. |

Clarification about how the driver supports several of these usages follows in Table 6. This list is ordered in the likelihood of use, with the most likely usages at the top of the table and the least likely usages at the bottom.

Table 6. Sensor property usage clarifying notes

|  |  |
| --- | --- |
| **Usage ID** | **Clarifying Notes** |
| **Sensor Connection Type**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE | Read-only.  This property is a means by which the driver can detect the method by which the device is connected to the PC. This matters for certain applications.  The value is an 8-bit enumerated value representing the Sensor Connection Type. This can be one of three values:  1 = Sensor is internal to the physical body of the PC. This is the required connection type for any sensor that is expected to support the User Experience for sensor fusion applications on Windows 8.1 Preview. This includes the Accelerometer, Gyrometer, Inclinometer, Compass, Device Orientation and Ambient Light sensors.  2 = Sensor is external to the PC but attached by a cable or wireless connection (such as Bluetooth)  3 = Sensor is external to the PC but not directly attached. This might, for example, be a network connection.  The Driver will read this value from the device and expose a corresponding value at the Sensor API.  While each sensor must specify this value, typically this value is common for all Sensors supported by a single Device. |
| **Reporting State**  HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE | Read/Write.  This property is a means by which the driver can minimize the power consumption of the Sensor, the device and the PC.  The value is an 8-bit enumerated value representing the Sensor Reporting State. This can be one of two supported values, though more values are defined by the Specification:  1 = Sensor will not send any Input reports asynchronously  2 = Sensor will send Input reports asynchronously  The Driver will both read and write this value. Under normal circumstances, the driver will write a “1” to the Sensor, and the Sensor is expected to comply by ceasing to send asynchronous input reports. This indicates that the driver has no clients interested in periodic asynchronous data from the Sensor. This state should persist until the driver writes a “2” to the Sensor indicating asynchronous Input reports should be sent.  It is important to note that even if the driver has requested the Sensor cease sending asynchronous input reports, the driver still expects the Sensor to be able to respond synchronously to a GET\_INPUT command, also referred to as a polled response. This creates a requirement that, even if the Sensor is in a lower power-state it is still expected to respond in a manner to a GET\_INPUT command. A further hint is provided by the Power State. If Reporting State is “2” the Power State will set by the driver will be “Full Power.” If the Reporting State is “1” and there are clients connected to the API that can be expected to request data at any time via a GET\_INPUT report, the Power State set by the driver will be “Low Power.”  Though other values are defined by the Specification, the driver will send no other values than those noted above.  The best practice here is that the Reporting State should be controlled by the device on a per-Sensor basis. This permits fine-grained control over Device power use by quiescing those Sensors for which there is no Client interest. |
| **Power State**  HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE | Read/Write.  This property is a means by which the driver can minimize the power consumption of the Sensor, the device and the PC.  The value is an 8-bit enumerated value representing the Sensor Power State. This can be one of three supported values, though more values are defined by the Specification:  2 = Full Power. This is selected by the driver when the driver also requests the Sensor send Input reports asynchronously  3 = Low Power. This is selected by the driver when the driver also requests the Sensor to not send Input reports asynchronously. In this power state, the Sensor is expected to respond to synchronous requests for Input reports.  6 = Power Off. This is selected by the driver when there are no clients at the API for this Sensor.  The Driver will both read and write this value.  This value is used in conjunction with the Reporting State. If the Reporting State is “2” the driver will always accompany this with setting the Power State to “2”, or Full Power. If the Reporting State is “1” the driver can accompany this with the Power State set to either “3” (Low Power) if Clients remain connected at the API and can be expected to request data at any time, or to “6” (Power Off) if there are no Clients connected at the API.  If all Sensors on a Device are set by the driver to “Power Off” then the device should take steps to minimize power consumption until it receives a “Low Power” or “Full Power” request for at least one sensor. |
| **Sensor State**  HID\_USAGE\_SENSOR\_STATE | Read-only.  This property is a means by which the driver can determine the current state of the Sensor. Strictly speaking this is not a property defined as are the rest of the properties for use in a Feature report. Rather, this is a property that is defined for use in the Input report. The use in a Feature report is identical to the use in an Input report.  The value is an 8-bit enumerated value representing the current state of the device. This can be one of these seven supported values:  1 = Sensor State Unknown  2 = Sensor State Ready  3 = Sensor State Unavailable  4 = Sensor State No Data  5 = Sensor State Initializing  6 = Sensor State Access Denied  7 = Sensor State Error  The Driver will read this value from the Sensor and expose a corresponding value at the Sensor API.  As noted, this same property and values must be available in the Input report. The difference in use is that this property carried in an Input report is sent asynchronously so can communicate a change in the Sensor state; this property carried in a Feature report can only be read synchronously by the driver, and this typically happens only during Sensor initialization and during a change in Client count or requests by Clients for changes to the Report Interval or Change Sensitivities for a Sensor.  The Sensor should send an asynchronous Input report whenever the state of the sensor changes (e.g. from Initializing to Ready.) This Input report should be sent no matter the current state of Reporting State or the Power State (unless the power is physically turned off the sensor and it is unable to respond.) |
| **Report Interval**  HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL | Read/Write.  This property is a means by which the driver can control the rate of events passing through the system. This is not meant to control the rate at which sampling occurs on the Sensor, only the rate at which asynchronous events are sent by the Sensor to the PC.  The value is a 32-bit unsigned integer representing in milliseconds the minimum time between successive asynchronous Input reports sent by the device to the driver. The value is meant for each individual Sensor, but the device may interpret this value as for the collection of Sensors on the device.  When sending a Feature report, the Sensor should send the value of the current Report Interval.  When receiving a Feature report, the Sensor should evaluate the requested Report Interval and set the Sensor value to the nearest value below the requested value the Sensor is able to support and immediately place that value in feature report buffer for an anticipated request for a Feature report. If the requested value is less than the Minimum Report Interval supported by the device, the device should set the Current Report Interval to the Minimum Report Interval.  From the driver side, the driver will choose a requested Report Interval from amongst the Report Intervals requested by the connected Clients. The value chosen will be the lowest value requested by all currently connected Clients that have actually specified a Report Interval. If there are only connected Clients that have not specified a Report Interval (i.e. that have accepted the default value) then the driver will choose a default value. The Driver will then write that value to the Sensor with a SET\_FEATURE call and then immediately issue a GET\_FEATURE to read back the value the device was able to support. That value is then available at the API.  It is important to note that when testing the implementation of this property that the value written to the Sensor is dependent upon the values requested by the current Sensor Clients. It is best when testing this property to have only a single Client connected, and that Client is the Client evaluating the proper function of this property from the API. |
| **Minimum Report Interval**  HID\_USAGE\_SENSOR\_PROPERTY\_MINIMUM\_REPORT\_INTERVAL | Read-only.  This property is a means by which the driver can determine the shortest supported Report Interval. By knowing this, the driver can avoid sending unsupported Report Interval requests to the device.  The value is an 8-bit, 16-bit or 32-bit unsigned integer value representing in milliseconds the minimum supported time between successive asynchronous Input reports sent by the device to the driver. The value is meant for an individual Sensor supported by the device. |
| **Change Sensitivity Absolute**  HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS  **Change Sensitivity relative percent**  HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_REL\_PCT | Read/Write.  Change Sensitivity is one of five per-datafield properties supported by the driver; the other four (minimum, maximum, accuracy and resolution,) described in another section of this table, are read-only.  Note that this section describes change sensitivity as a property for completeness; however, change sensitivity is specified as a data field modifier in the feature report. Further details about the *bulk* and *specific* use of this modifier are described in section 4.2.3.  When receiving a Feature report, the Sensor should evaluate the requested Change Sensitivity and set the Sensor value to the nearest value below the requested value the Sensor is able to support and immediate place that value in the buffer for an anticipated request for a Feature report. For most Sensors, the Sensor should avoid setting the Change Sensitivity to a value of ‘0’ as this may cause an excessive rate of events reported to the driver. (Exceptions to this include Sensors such as Switches, which are binary in nature.)  From the driver side, the driver will choose a requested Change Sensitivity from amongst the Change Sensitivities requested by the connected Clients for a particular datafield. The value chosen will be the lowest value requested by all currently connected Clients that have actually specified a Change Sensitivity for that datafield. If there are only connected Clients that have not specified a Change Sensitivity (i.e. that have accepted the default value for that datafield) then the default value will be selected. The Driver may either choose its own default value or select the default value provided in the feature report during enumeration. The Driver will then write that value to the Sensor with a SET\_FEATURE call and then immediately issue a GET\_FEATURE to read back the value the device was able to support. That value is then available at the API.  It is important to note that when testing the implementation of this property that the value written to the Sensor is dependent upon the values requested by the current Sensor Clients. It is best when testing this property to have only a single Client connected, and that Client is the Client evaluating the proper function of this property from the API.  Care should be taken when defining the Report Descriptor for the fields that no greater precision is implied than can actually be achieved by the Sensor. For example, an 8-bit analog-to-digital converter is unlikely to support the precision supported by a 32-bit fixed point value, but is better represented by a 16-bit fixed point value. |
| **Sensor Accuracy**  HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY  **Sensor Resolution**  HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION  **Range Maximum**  HID\_USAGE\_SENSOR\_ DATA\_MOD\_RANGE\_MAXIMUM  **Range Minimum**  HID\_USAGE\_SENSOR\_ DATA\_MOD\_RANGE\_MINIMUM | Read-only.  These four per-datafield properties are handled identically though their individual meanings vary.  Note that this section describes accuracy, resolution, and range as properties as data field modifiers in the feature report. Further details about the *bulk* and *specific* use of these modifiers are described in section 4.2.3.  The Sensor may support these properties at its discretion after due consideration of the Client needs.  Maximum and Minimum should be supported if the Client may have some interest in the Maximum and Minimum range of the valid Sensor datafield values. As an example, this permits the Client to determine if the Sensor is close to or exceeded its measurement range, enabling the Client to act appropriately. An inappropriate use of Maximum and Minimum would be for any Sensor that only supports a binary value, such as Switches.  Accuracy (how close is the measured value to the actual value) should be supported if it is important for the Client to know the Accuracy with which a Sensor measurement is being made.  Resolution (to what degree can the Sensor analog-to-digital conversion process resolve a value) should be supported if it is important for the Client to know the Resolution supported by the underlying Sensor hardware.  Care should be taken when defining the Report Descriptor for the fields that no greater precision is implied than can actually be achieved by the Sensor. An 8-bit analog-to-digital converter is unlikely to support the precision supported by a 32-bit fixed point value, but is better represented by a 16-bit fixed point value. |
| **Response Curve**  HID\_USAGE\_SENSOR\_PROPERTY\_RESPONSE\_CURVE | Read-only.  The means by which a Sensor can specify a response curve mapping for the driver to be consumed at the API. The Driver only uses this property for the Sensor category Light type Ambient Light.  This collection of values is an n x 2 (ex. int v[3][2]) array of either 16-bit unsigned integers, or of 16-bit fixed point numbers. Whichever Units Exponent is chosen for value specification applies to all values in the array.  On the Sensor side, the buffer constructed to hold the array of values should be in the following format:  If the buffer is specified as v[3][2] the elements are stored (in increasing addresses) as  v[0][0], v[0][1], v[1][0], v[1][1], v[2][0], v[2][1]  Arranged in this way, the array is correctly interpreted by the driver and presented at the API.  Note that this property is optional. If it is not present, then the API will use the default values specified for the Ambient Light sensor. |
| **Friendly Name**  HID\_USAGE\_SENSOR\_PROPERTY\_FRIENDLY\_NAME  **Persistent Unique ID**  HID\_USAGE\_SENSOR\_PROPERTY\_PERSISTENT\_UNIQUE\_ID  **Sensor Manufacturer**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MANUFACTURER  **Sensor Model**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MODEL  **Sensor Serial Number**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_SERIAL\_NUMBER  **Sensor Description**  HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_DESCRIPTION | Read only.  The means by which a Sensor can specify an alternative to the default values for these properties generated by the driver.  Each property is a zero-terminated wide-character array of no more than 31 wide characters followed by a zero termination. No Unit or Exponent is used. The character set used is the default wide character set for the PC.  Care should be used when using these properties and specifying them in the Report Descriptor. The largest Feature Report that is supported by the driver is a total of 64-bytes (63-bytes of payload plus the 1-byte Report ID.) Note that specifying a 31 character string will exceed this maximum Feature Report size. The Report Descriptor must take into account the definition for all other specified properties and only then specify a string length that will not exceed this maximum Feature Report size. |
| **HID Usage**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE | Read only.  A vendor-specific means by which a Sensor can distinguish itself from a sensor of the same category and type.  This property is available dynamically by means of its inclusion in the **input** report of any sensor. |

### HID sensor property defaults

Table 7 provides a detailed description of the various property defaults for each Sensor supported by the driver. The default Sensor values are used as follows:

* **Report Interval**: The Driver chooses the lowest Report Interval value from among the connected Clients that have specified a Report Interval. If there are only connected Clients that have not specified a Report Interval (i.e., that have accepted the default value), the driver will choose its own default value. That chosen value is written to the Sensor by means of a SET\_FEATURE report. The Sensor should examine this requested value and immediately respond to an anticipated GET\_FEATURE request from the driver by setting the value for Report Interval to a value that can be supported by the Sensor. If the Sensor cannot support the requested value, the value should be set to the next lowest supported value. For example, if the request is for 6ms, but the Sensor can only support 5ms, the value should be set to 5ms for the driver to retrieve by means of a GET\_FEATURE report.
* **Change Sensitivity**: The Driver chooses the lowest Change Sensitivity value for that datafield from among the connected Clients that have specified a Change Sensitivity for that datafield. If there are only connected Clients that have not specified a Change Sensitivity for that datafield (i.e., that have accepted the default value), then the driver may either choose its own default value or select the default value provided in the feature report during enumeration. That chosen value is written to the Sensor by means of a SET\_FEATURE report. The Sensor should examine this requested value and immediately respond to an anticipated GET\_FEATURE request from the driver by setting the value for Change Sensitivity to a value that can be supported by the Sensor. If the Sensor cannot support the requested value, the value should be set to the next lowest supported value. For example, if the request is for 0.12g, but the Sensor can only support 0.1g, the value should be set to 0.1g for the driver to retrieve by means of a GET\_FEATURE report.
* **Range Maximum**: The default value is used at the API unless a value is provided by the Sensor Device by means of a valid response to a GET\_FEATURE request from the driver.
* **Range Minimum**: The default value is used at the API unless a value is provided by the Sensor Device by means of a valid response to a GET\_FEATURE request from the driver.
* **Accuracy**: The default value is used at the API unless a value is provided by the Sensor Device by means of a valid response to a GET\_FEATURE request from the driver.
* **Resolution**: The default value is used at the API unless a value is provided by the Sensor Device by means of a valid response to a GET\_FEATURE request from the driver.

The default values are typically chosen to be a value that minimizes CPU and power use for that Sensor type while optimizing the value for a range of applications. Many applications simply opt for the default value, while those applications to which the value is important choose a specific value and over-ride the default.

**Note**: Datafield properties Change Sensitivity, Range Maximum, Range Minimum, Accuracy, and Resolution are all treated by the driver (and at the API) as per-datafield properties. The Sensor, however, can choose how these per-datafield properties are supported, either as Bulk or Specific:

* **Bulk**: The value understood by and supplied to a Sensor is common to those datafields that are related by the use of a Bulk datafield usage
* **Specific**: The value understood by and supplied to a Sensor is used for only a single datafield

Which of the three forms of per-datafield property support is used for a Sensor is up to the Implementer.

Table 7. Sensor properties default values

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sensor Category, Type and Datafield** | **Per-Sensor property** | **Per-datafield properties** | | |
| Default Report Interval | Default Change Sensitivity | Default Range Maximum and Minimum | Default Accuracy and Resolution |
| Biometric: Human Presence  SENSOR\_CATEGORY\_BIOMETRIC  SENSOR\_TYPE\_HUMAN\_PRESENCE | Report Interval = 100 mS  Default Min Report Interval = 50 mS | Not supported | Not supported | Not supported |
| Biometric: Human Proximity  SENSOR\_CATEGORY\_BIOMETRIC  SENSOR\_TYPE\_HUMAN\_PROXIMITY | Report Interval = 100 mS  Default Min Report Interval = 50 mS | SENSOR\_DATA\_TYPE\_HUMAN\_PROXIMITY\_METERS  Sensitivity = 0.01 M | SENSOR\_DATA\_TYPE\_HUMAN\_PROXIMITY\_METERS  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_HUMAN\_PROXIMITY\_METERS  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| Environmental: Atmospheric Pressure  SENSOR\_CATEGORY\_ENVIRONMENTAL  SENSOR\_TYPE\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE | Report Interval = 100 mS  Default Min Report Interval = 50 mS | SENSOR\_DATA\_TYPE\_ATMOSPHERIC\_PRESSURE\_BAR  Sensitivity = 0.00001 Bar | SENSOR\_DATA\_TYPE\_ATMOSPHERIC\_PRESSURE\_BAR  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_ATMOSPHERIC\_PRESSURE\_BAR  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| Environmental: Humidity  SENSOR\_CATEGORY\_ENVIRONMENTAL  SENSOR\_TYPE\_ENVIRONMENTAL\_HUMIDITY | Report Interval = 100 mS  Default Min Report Interval = 50 mS | SENSOR\_DATA\_TYPE\_RELATIVE\_HUMIDITY\_PERCENT  Sensitivity = 0.25 Percent | SENSOR\_DATA\_TYPE\_RELATIVE\_HUMIDITY\_PERCENT  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_RELATIVE\_HUMIDITY\_PERCENT  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| Environmental: Temperature  SENSOR\_CATEGORY\_ENVIRONMENTAL  SENSOR\_TYPE\_ENVIRONMENTAL\_TEMPERATURE | Report Interval = 100 mS  Default Min Report Interval = 50 mS | SENSOR\_DATA\_TYPE\_TEMPERATURE\_CELSIUS  Sensitivity = 0.25 C | SENSOR\_DATA\_TYPE\_TEMPERATURE\_CELSIUS  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_TEMPERATURE\_CELSIUS  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| Light: Ambient Light  SENSOR\_CATEGORY\_LIGHT  SENSOR\_TYPE\_AMBIENT\_LIGHT | Report Interval = 100 mS  Default Min Report Interval = 50 mS | SENSOR\_DATA\_TYPE\_LIGHT\_LEVEL\_LUX  Sensitivity = 1.0 Percent  SENSOR\_DATA\_TYPE\_LIGHT\_TEMPERATURE\_KELVIN  Sensitivity = 0.2 K  SENSOR\_DATA\_TYPE\_LIGHT\_CHROMACITY  Sensitivity = 0.2 | SENSOR\_DATA\_TYPE\_LIGHT\_LEVEL\_LUX  Max = FLT\_MAX  Min = -FLT\_MAX  SENSOR\_DATA\_TYPE\_LIGHT\_TEMPERATURE\_KELVIN  Max = FLT\_MAX  Min = -FLT\_MAX  SENSOR\_DATA\_TYPE\_LIGHT\_CHROMACITY  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_LIGHT\_LEVEL\_LUX  Accuracy = FLT\_MAX  Resolution = FLT\_MAX  SENSOR\_DATA\_TYPE\_LIGHT\_TEMPERATURE\_KELVIN  Accuracy = FLT\_MAX  Resolution = FLT\_MAX  SENSOR\_DATA\_TYPE\_LIGHT\_CHROMACITY  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| Motion: Accelerometer 3D  SENSOR\_CATEGORY\_MOTION  SENSOR\_TYPE\_ACCELEROMETER\_3D | Report Interval = 100 mS  Default Min Report Interval = 16 mS | SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Y\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Z\_G  Sensitivity = 0.02 G | SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Y\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Z\_G  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Y\_G  SENSOR\_DATA\_TYPE\_ACCELERATION\_Z\_G  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| Motion: Gyrometer 3D  SENSOR\_CATEGORY\_MOTION  SENSOR\_TYPE\_GYROMETER\_3D | Report Interval = 100 mS  Default Min Report Interval = 16 mS | SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_X\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Y\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Z\_DEGREES\_PER\_SECOND  Sensitivity = 0.5 Degrees-per-Second | SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_X\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Y\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Z\_DEGREES\_PER\_SECOND  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_X\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Y\_DEGREES\_PER\_SECOND  SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Z\_DEGREES\_PER\_SECOND  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| Orientation: Compass 3D  SENSOR\_CATEGORY\_ORIENTATION  SENSOR\_TYPE\_COMPASS\_3D | Report Interval = 100 mS  Default Min Report Interval = 50 mS | SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_MAGNETIC\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_TRUE\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_MAGNETIC\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_TRUE\_NORTH\_DEGREES  Sensitivity = 0.2 Degree  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_X\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Y\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Z\_MILLIGAUSS  Sensitivity = 0.2 Milligauss | SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_MAGNETIC\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_TRUE\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_MAGNETIC\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_TRUE\_NORTH\_DEGREES  Max = FLT\_MAX  Min = -FLT\_MAX  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_X\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Y\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Z\_MILLIGAUSS  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_MAGNETIC\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_TRUE\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_MAGNETIC\_NORTH\_DEGREES  SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_TRUE\_NORTH\_DEGREES  Accuracy = FLT\_MAX  Resolution = FLT\_MAX  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_X\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Y\_MILLIGAUSS  SENSOR\_DATA\_TYPE\_MAGNETIC\_FIELD\_STRENGTH\_Z\_MILLIGAUSS  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| **Orientation: Inclinometer 3D**  SENSOR\_CATEGORY\_ORIENTATION  SENSOR\_TYPE\_INCLINOMETER\_3D | Report Interval = 50 mS  Default Min Report Interval = 16 mS | SENSOR\_DATA\_TYPE\_TILT\_X\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Y\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Z\_DEGREES  Sensitivity = 0.5 Degree | SENSOR\_DATA\_TYPE\_TILT\_X\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Y\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Z\_DEGREES  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_TILT\_X\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Y\_DEGREES  SENSOR\_DATA\_TYPE\_TILT\_Z\_DEGREES  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| **Orientation: Device Orientation**  SENSOR\_CATEGORY\_ORIENTATION  SENSOR\_TYPE\_AGGREGATED\_DEVICE\_ORIENTATION | Report Interval = 50 mS  Default Min Report Interval = 16 mS | SENSOR\_DATA\_TYPE\_QUATERNION  Sensitivity = 0.2  SENSOR\_DATA\_TYPE\_ROTATION\_MATRIX  Sensitivity = 0.2 | SENSOR\_DATA\_TYPE\_QUATERNION  Max = FLT\_MAX  Min = -FLT\_MAX  SENSOR\_DATA\_TYPE\_ROTATION\_MATRIX  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_QUATERNION  Accuracy = FLT\_MAX  Resolution = FLT\_MAX  SENSOR\_DATA\_TYPE\_ROTATION\_MATRIX  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |
| **Other: Custom**  SENSOR\_CATEGORY\_OTHER  SENSOR\_TYPE\_CUSTOM | Report Interval = 100 mS  Default Min Report Interval = 50 mS | SENSOR\_DATA\_TYPE\_CUSTOM\_USAGE  Sensitivity = 0.05 (Note that this value is not used except to provide a placeholder for the required per-datafield property)  SENSOR\_DATA\_TYPE\_CUSTOM\_BOOLEAN\_ARRAY  Sensitivity = 0.05 (Note that this value is not used except to provide a placeholder for the required per-datafield property)  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE1  through  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE28  Sensitivity = 0.05 | SENSOR\_DATA\_TYPE\_CUSTOM\_USAGE  Max = FLT\_MAX  Min = -FLT\_MAX  SENSOR\_DATA\_TYPE\_CUSTOM\_BOOLEAN\_ARRAY  Max = FLT\_MAX  Min = -FLT\_MAX  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE1  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE2  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE3  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE4  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE5  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE6  Max = FLT\_MAX  Min = -FLT\_MAX | SENSOR\_DATA\_TYPE\_CUSTOM\_USAGE  Accuracy = FLT\_MAX  Resolution = FLT\_MAX  SENSOR\_DATA\_TYPE\_CUSTOM\_BOOLEAN\_ARRAY  Accuracy = FLT\_MAX  Resolution = FLT\_MAX  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE1  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE2  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE3  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE4  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE5  SENSOR\_DATA\_TYPE\_CUSTOM\_VALUE6  Accuracy = FLT\_MAX  Resolution = FLT\_MAX |

### HID sensor input report properties and datafield usages

There is no single section in the Specification that defines usages for datafields. Rather, these usages are described in sections 1.6 through 1.17. Regarding the datafield usages described in sections 1.6 through 1.17, there is some background information required how the driver handles datafields.

The Driver supports three kinds of datafields:

1. **Required** datafields: Required for that particular Sensor in order for the driver to be able to support that Sensor up through the API. The required datafields for each sensor are noted in section 1.1 of this document. If a datafield is required for a sensor and the corresponding HID usage for that sensor is not present in the input report as defined by the report descriptor, that datafield will have a value of VT\_EMPTY at the API.
2. **Optional** datafields: Optional for a specific Sensor. The Sensor that does not include these optional datafields will be correctly supported by the driver up through the API. Even though certain datafields are optional, the implementer should be sure that the correct optional fields are supported if Hardware Certification compliance is required. The optional datafields for each sensor are noted in section 1.1 of this document. If a datafield is optional for a sensor and the corresponding HID usage for that sensor is not present in the input report as defined by the report descriptor, that datafield will not be present at the API. For further information on using optional datafields see section 5.1.
3. **Dynamic** datafields: Essentially any Sensor can include any supported datafield in the report descriptor and have that datafield exposed by the driver for that Sensor at the API. There are restrictions on certain datafields as well as on how per-datafield properties should be defined in order for the driver to provide support. For specific information on which datafields cannot be dynamic for specific sensors, refer to sections 1.6 through 1.17. If the HID usage for a dynamic datafield is included for a sensor in the input report as defined by the report descriptor, that datafield will be present at the API. If it is not included, it will not be present. For further information on using dynamic datafields, see section 5.2.

The Driver, upon initialization or upon the first connection of a Client, will issue a GET\_INPUT request to the Sensor. The Sensor must respond synchronously with the most recent valid value for that sensor. If the Sensor does not do so, the SENSOR\_PROPERTY\_STATE at the API will be NO\_DATA, and this may prevent interested Clients from connecting to the Sensor. If a Client is able to connect to the Sensor, another GET\_INPUT is issued by the driver in order to acquire the most recent data available from the Sensor.

In addition to the datafields supported in the Input Report for each Sensor, a Sensor Input Report should support the following two usages in Table 8.

Note that a valid Sensor can be created that does not support these usages in the Input Report (i.e. only datafields are supported) but also note that a Sensor defined in this manner will not pass Hardware Certification requirements.

Table 8. Input report properties

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Property** |
| **Sensor State**  **0x0201**  HID\_USAGE\_SENSOR\_STATE | Enumerated values as described in section 1.3 | SENSOR\_PROPERTY\_STATE  Default value is SENSOR\_STATE\_NO\_DATA.  Value is read-only |
| **Event Type**  **0x0202**  HID\_USAGE\_SENSOR\_EVENT | Enumerated values as described in section 1.3 | There is no corresponding Windows Sensor Property. Event types are made known to Clients by subscription to events for a particular sensor. |

Clarification of the specific way in which the driver supports these usages follows in Table 9.

Table 9. Input report properties clarifying notes

|  |  |
| --- | --- |
| **Usage ID** | **Clarifying Notes** |
| Sensor State  0x0201  HID\_USAGE\_SENSOR\_STATE | Read-only.  This property is a means by which the driver can determine the current state of the Sensor. This property is also used in the Feature report.  The value is an 8-bit enumerated value representing the current state of the device. This can be one of 7 supported values:  1 = Sensor State Unknown  2 = Sensor State Unavailable  3 = Sensor State Ready  4 = Sensor State No Data  5 = Sensor State Initializing  6 = Sensor State Access Denied  7 = Sensor State Error  The Driver will read this value from the Sensor and expose a corresponding value at the Sensor API. |
| Event Type  0x0202  HID\_USAGE\_SENSOR\_EVENT | Read-only.  This property is a means by which the driver can determine the reason the driver is being sent an asynchronous Input Report.  The value is an 8-bit enumerated value representing the current state of the device. This can be one of 6 supported values:  1 = Sensor Event Unknown  2 = Sensor Event State Changed  3 = Sensor Event Property Changed  4 = Sensor Event Data Updated  5 = Sensor Event Poll Response (GET\_INPUT response)  6 = Sensor Event Change Sensitivity  Other values are defined in the Specification, but only these values are supported by the driver. The Driver will read this value from the Sensor and expose a corresponding value at the Sensor API.  This value is only used in an Input report.  Note that the sensor should send an asynchronous Input report on any change of Sensor state (e.g. from Initializing to Ready) regardless of the state of Reporting State. |

## Biometric Sensor Field Usages

The fields listed in Table 10 are supported by the driver for biometric sensors.

Table 10. Biometric sensor field usages

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Datafield** |
| **Human Presence**  **0x04B1**  HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PRESENCE | Boolean value | SENSOR\_DATA\_TYPE\_HUMAN\_PRESENCE  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Human Proximity Range**  **0x04B2**  HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_RANGE | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is meters.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_HUMAN\_PROXIMITY\_METERS  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Human Proximity Out-of-Range**  **0x04B3**  HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_OUT\_OF\_RANGE | Boolean value  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is used internally by the driver and is not available at the API  Dynamic datafield support:  Not supported in any Sensor but Human Proximity Range. |

## Electrical Sensor Field Usages

No electrical sensor fields are supported by the driver.

## Environmental Sensor Field Usages

The fields listed in Table 12 are supported by the driver for environmental sensors.

Table 12. Environmental Sensor Field Usages

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Datafield** |
| Atmospheric Pressure  0x0431  HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Bar.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ATMOSPHERIC\_PRESSURE\_BAR  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| Relative Humidity Percent  0x0433  HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_RELATIVE\_HUMIDITY | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Percent.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_RELATIVE\_HUMIDITY\_PERCENT  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| Temperature  0x0434  HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Celsius.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_TEMPERATURE\_CELSIUS  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |

## Light Sensor Field Usages

The fields listed in Table 13 are supported by the driver for light sensors.

Table 13. Light sensor field usages

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Datafield** |
| **Illuminance**  **0x04D1**  HID\_USAGE\_SENSOR\_DATA\_LIGHT\_ILLUMINANCE | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Lux.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_LIGHT\_LEVEL\_LUX  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor, but the Change Sensitivity for any Sensor other than Ambient Light will be interpreted as Change Sensitivity Absolute. |
| **Color Temperature**  **0x04D2**  HID\_USAGE\_SENSOR\_DATA\_LIGHT\_COLOR\_TEMPERATURE | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Kelvin.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_LIGHT\_TEMPERATURE\_KELVIN  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  Not supported in any Sensor but Ambient Light Sensor. |
| **Chromaticity**  **0x04D3**  HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is not available at the API. It is only used with a data modifier to specify a per-datafield property. See section 4.2.3 |
| **Chromaticity X**  **0x04D4**  HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY\_X  **Chromaticity Y**  **0x04D5**  HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY\_Y | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED.  If either Chromaticity X or Chromaticity Y values are used, both must be used to result in a valid value at the API | SENSOR\_DATA\_TYPE\_LIGHT\_CHROMACITY  Type = [VT\_VECTOR | VT\_UI1]  Default value is VT\_EMPTY  Note that at the API, the values for the two axes of Chromaticity (X & Y) are combined into a 2-value array.  Dynamic datafield support:  Not supported in any Sensor but Ambient Light Sensor. |

## Location Sensor Field Usages

No location sensor fields are supported by the driver.

## Mechanical Sensor Field Usages

No mechanical sensor fields are supported by the driver.

## Motion Sensor Field Usages

The fields listed in Table 15 are supported by the driver for motion sensors.

Table 15. Motion sensor field usages

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Datafield** |
| **Motion State**  **0x0451**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_STATE | Boolean value  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED.  Note that this value if present in an Accelerometer Input report is interpreted by the driver as a “shake” event, and as such should be sent in an asynchronous Input report (along with the current accelerometer datafield values) whenever a “shake sequence” has been detected. This report should be sent immediately without regard to the current Report Interval as long as the Reporting State is set to All Events. | SENSOR\_DATA\_TYPE\_MOTION\_STATE  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Motion Intensity**  **0x045F**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_INTENSITY | 8-bit, 16-bit or 32-bit value representing motion intensity value.  Default Unit is Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED.  This value will be treated as a Boolean value at the API.  Note that this value if present in an Accelerometer Input report is interpreted by the driver as a “shake” event, and as such should be sent in an asynchronous Input report (along with the current accelerometer datafield values) whenever a “shake sequence” has been detected. This report should be sent immediately without regard to the current Report Interval as long as the Reporting State is set to All Events. | SENSOR\_DATA\_TYPE\_MOTION\_STATE  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Acceleration**  **0x0452**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is G.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is not available at the API. It is only used with a data modifier to specify a per-datafield property. See section 4.2.3  Dynamic datafield support:  Not valid for use in any Sensor but Accelerometer |
| **Acceleration Axis X**  **0x0453**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is G.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Acceleration Axis Y**  **0x0454**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is G.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ACCELERATION\_Y\_G  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Acceleration Axis Z**  **0x0455**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is G.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ACCELERATION\_Z\_G  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Angular Velocity**  **0x0456**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees-per-Second.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is not available at the API. It is only used with a data modifier to specify a per-datafield property. See section 4.2.3  Dynamic datafield support:  Not valid for use in any Sensor but Gyrometer |
| **Angular Velocity X Axis**  **0x0457**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_X\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees-per-Second.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_X\_DEGREES\_PER\_SECOND  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Angular Velocity Y Axis**  **0x0458**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_Y\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees-per-Second.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Y\_DEGREES\_PER\_SECOND  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Angular Velocity Z Axis**  **0x0459**  HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_Z\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees-per-Second.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ANGULAR\_VELOCITY\_Z\_DEGREES\_PER\_SECOND  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |

## Orientation Sensor Field Usages

The fields listed in Table 16 are supported by the driver for orientation sensors.

Table 16. Orientation sensor field usages

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Datafield** |
| **Heading**  **0x0471**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is not available at the API. It is only used with a data modifier to specify a per-datafield property. See section 4.2.3  Dynamic datafield support:  Not valid for use in any Sensor but Compass |
| **Heading Compensated Magnetic North**  **0x0475**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_COMPENSATED\_MAGNETIC\_NORTH | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_MAGNETIC\_NORTH\_DEGREES  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Heading Compensated True North**  **0x0476**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_COMPENSATED\_TRUE\_NORTH | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_COMPENSATED\_TRUE\_NORTH\_DEGREES  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Heading Magnetic North**  **0x0477**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_NORTH | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_MAGNETIC\_NORTH\_DEGREES  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Heading True North**  **0x0478**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TRUE\_NORTH | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_MAGNETIC\_HEADING\_TRUE\_NORTH\_DEGREES  Type = VT\_R8  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Tilt**  **0x047E**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is not available at the API. It is only used with a data modifier to specify a per-datafield property. See section 4.2.3  Dynamic datafield support:  Not valid for use in any Sensor but Inclinometer |
| **Tilt X Axis**  **0x047F**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_X | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_TILT\_X\_DEGREES  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Tilt Y Axis**  **0x0480**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_Y | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_TILT\_Y\_DEGREES  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Tilt Z Axis**  **0x0481**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_Z | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Degrees.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_DISTANCE\_Z\_METERS  Type = VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Rotation Matrix**  **0x0482**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_ROTATION\_MATRIX | A 3 x 3 array of 16-bit fixed point values.  Default Unit is Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED.  This collection of values is a 3 x 3 (ex. int v[3][3]) array of 16-bit fixed point numbers. Whichever Units Exponent is chosen for value specification applies to all values in the array.  On the Sensor side, the buffer constructed to hold the array of values should be in the following format:  If the buffer is specified as v[3][3] the elements are stored (in increasing addresses) as  v[0][0], v[0][1], v[0][2], v[1][0], v[1][1], v[1][2], v[2][0], v[2][1], v[2][2]  Arranged in this way, the array is correctly interpreted by the driver and presented at the API.  See section 4.3.21 for more information. | SENSOR\_DATA\_TYPE\_ROTATION\_MATRIX  Type = [VT\_VECTOR | VT\_UI1]  Default value is VT\_EMPTY  Dynamic datafield support:  Not valid for use in any Sensor but Orientation |
| **Quaternion**  **0x0483**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_QUATERNION | A 4 x 1 array of 16-bit fixed point values.  Default Unit is Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED.  This collection of values is a 4 x 1 (ex. int v[4]) array of 16-bit fixed point numbers. Whichever Units Exponent is chosen for value specification applies to all values in the array.  On the Sensor side, the buffer constructed to hold the array of values should be in the following format:  If the buffer is specified as v[4] the elements are stored (in increasing addresses) as  v[0], v[1], v[2], v[3]  Arranged in this way, the array is correctly interpreted by the driver and presented at the API.  See section 4.3.21 for more information. | SENSOR\_DATA\_TYPE\_QUATERNION  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  Not valid for use in any Sensor but Orientation |
| **Magnetic Flux**  **0x0484**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Milligauss.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is not available at the API. It is only used with a data modifier to specify a per-datafield property. See section 4.2.3  Dynamic datafield support:  Not valid for use in any Sensor but Compass |
| **Magnetic Flux X Axis**  **0x0485**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_X\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Milligauss.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Magnetic Flux Y Axis**  **0x0486**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_Y\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Milligauss.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Magnetic Flux Z Axis**  **0x0487**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_Z\_AXIS | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Default Unit is Milligauss.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | SENSOR\_DATA\_TYPE\_ACCELERATION\_X\_G  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Magnetometer Accuracy**  **0x488**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY | One of the Selector values described in the clarifying notes of this section. | SENSOR\_DATA\_TYPE\_MAGNETOMETER\_ACCURACY  Type = VT\_I4  Default value is VT\_EMPTY  Dynamic datafield support:  Not valid for use in any Sensor but Orientation |

Clarification about how the driver supports several of these usages follows.

Table 17. Orientation sensor field usage clarifying notes

|  |  |
| --- | --- |
| **Usage ID** | **Clarifying Notes** |
| **Magnetometer Accuracy**  HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY | This data field indicates how closely the data represents the heading of a magnetically-calibrated device with respect to a horizontal plane. It should be included in the input reports of all magnetometer-derived sensors, including compass, inclinometer, and device orientation. Here magnetometer accuracy relates to the component of the fused sensor data impacted by the magnetic field and not just the raw magnetic field vector. Specifically magnetometer accuracy refers to the accuracy of the heading for compass and the accuracy of the yaw component, or rotation about the Z axis, for inclinometer and device orientation. The sensor solution should report an up-to-date magnetometer accuracy in each input report, conveying the accuracy of the data with respect to heading. Magnetometer accuracy should not be included in any per-datafield properties.  This data field is an Nary usage with one of the following Selector values:  **0x8E0** – Low. The current heading is between 25 and 180 degrees of the heading of a perfectly calibrated sensor.  **0x8E1** – Medium. The current heading is between 10 and 25 degrees of the heading of a perfectly calibrated sensor.  **0x8E2** – High. The current heading is between 0 and 25 degrees of the heading of a perfectly calibrated sensor.  The Driver will read this value from the device and expose a corresponding MagnetometerAccuracy enum value at the Sensor API. The mapping is as follows:  Low – MAGNETOMETER\_ACCURACY\_UNRELIABLE  Medium – MAGNETOMETER\_ACCURACY\_APPROXIMATE  High – MAGNETOMETER\_ACCURACY\_HIGH |

## Scanner Sensor Field Usages

No Scanner fields are supported by the driver.

## Time Sensor Field Usages

No Time Sensor fields are supported by the driver. The Driver does assign a timestamp to the input report sent by a sensor at the time the input report is received. This assigned value is available in the datafield SENSOR\_DATA\_TYPE\_TIMESTAMP.

## Custom Sensor Field Usages

The fields listed in Table 17 are supported by the driver for custom sensors.

Table 17. Custom sensor field usages

|  |  |  |
| --- | --- | --- |
| **Usage ID** | **Supported Values** | **Windows Sensor Datafield** |
| **Custom Usage**  **0x0541**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE | Boolean value  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED.  This datafield may be included in the input report of any sensor. If it is present, the property SENSOR\_PROPERTY\_HID\_USAGE will appear as a property supported by that sensor. | SENSOR\_DATA\_TYPE\_CUSTOM\_USAGE  Type = VT\_UI4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Boolean Array**  **0x0542**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_BOOLEAN\_ARRAY | 8-bit, 16-bit or 32-bit array of Boolean values.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_BOOLEAN\_ARRAY  Type = VT\_BOOL  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value**  **0x0543**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE | 16-bit or 32-bit fixed point value. See section 4.2.1 for use of fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | This value is not available at the API. It is only used with a data modifier to specify a per-datafield property. See section 4.2.3  Dynamic datafield support:  Not valid for use in any Sensor but Custom |
| **Custom Value 1**  **0x0544**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_1 | 8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_1  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 2**  **0x0545**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_2 | 8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_2  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 3**  **0x0546**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_3 | 8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_3  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 4**  **0x0547**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_4 | 8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_4  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 5**  **0x0548**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_5 | 8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_5  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 6**  **0x0549**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_6 | 8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_6  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 7**  **0x054A**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_7 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_7  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 8**  **0x054B**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_8 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_8  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 9**  **0x054C**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_9 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_9  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 10**  **0x054D**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_10 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_10  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 11**  **0x054E**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_11 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_11  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 12**  **0x054F**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_12 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_12  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 13**  **0x0550**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_13 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_13  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 14**  **0x0551**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_14 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_14  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 15**  **0x0552**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_15 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_15  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 16**  **0x0553**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_16 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_16  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 17**  **0x0554**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_17 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_17  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 18**  **0x0555**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_18 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_18  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 19**  **0x0556**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_19 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_19  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 20**  **0x0557**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_20 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_20  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 21**  **0x0558**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_21 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_21  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 22**  **0x0559**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_22 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_22  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 23**  **0x055A**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_23 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_23  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 24**  **0x055B**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_24 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_24  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 25**  **0x055C**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_25 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_25  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 26**  **0x055D**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_26 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_26  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 27**  **0x055E**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_27 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_27  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |
| **Custom Value 28**  **0x055F**  **Vendor-specific**  HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_28 | NOTE: Use of this is vendor-specific and not defined in the specification.  8-bit, 16-bit or 32-bit unsigned integer value or 16-bit or 32-bit fixed point value. See section 4.2.1 for use of integer and fixed-point values  Unit may be Not Specified.  No unit may be specified other than HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED. | HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_28  Type = VT\_UI4 or VT\_R4  Default value is VT\_EMPTY  Dynamic datafield support:  May be used in any Sensor. |

## Generic Sensor Field Usages

No Generic fields are supported by the driver.

# Sensor Backgrounder

This section describes Sensor terminology and the conceptual object model associated with the HID Sensor Usages. This section is informative and meant for orientation and guidance only.

## Glossary

A number of additional terms specific to the Sensor, Windows Sensor Platform, and Windows HID Sensor Class Driver subject matter are used in the context of this document. The following list defines these terms:

**API:** The Windows Sensor Platform API

**Client:** An application communicating with a Sensor through the API

**Device:** A sensor device implemented to conform to the Specification and the API

**Driver:** The Windows HID Sensor Class Driver

**Implementer:** The organization implementing a Device

**PC:** The system hosting the Windows operating system

**Sensor:** An individual sensor implemented on a Device. A Device may support more than one sensor, and may support more than one sensor of the same category and type

**Specification:** The HID Sensor Usage Tables document

## Sensor Taxonomy and Object Model

No further clarification required.

# Sensor Interaction via HID

This section describes how communication with Sensors is mapped to HID mechanisms. This section is informative and meant only for orientation and guidance.

## Related Documents

No additional clarification required.

## Functional Overview

The Specification states that for Sensor Devices, the HID Input Pipe is mandatory, and the HID Output Pipe is optional. This is true, except that for sensors that are expected to meet Windows Hardware Certification specifications, the HID Output Pipe is required as certain sensor properties communicated in the HID Feature Report as described in the HID class Report Descriptor are required.

## HID Logical Devices

The Driver will support collections either with or without nesting. A sensor with nesting will appear in Windows as a single Device Object with a collection of sensors. A sensor without nesting will appear as a single Device Object and as an individual sensor. Multiple sensors without nesting will appear as a set of Device Objects with each appearing as an individual sensors.

## HID Reports

As previously noted, it is not necessary to support SET\_REPORT in order to build a sensor that will work with the driver, though not doing so would render the senor unable to be Windows Hardware Certification compliant.

It is also not necessary to support a HID\_GET\_INPUT\_REPORT to build a sensor that will work with the driver, though not doing so will result in no data being available at the API until the first asynchronous Input report is received. This may result in an inability to be Windows Hardware Certification compliant.

## HID Report IDs

The driver only supports one Report ID per sensor per report type. The feature and input Report IDs do not necessarily need to be the same. The Driver interprets the Report ID in an asynchronous Input Report as the specific Sensor from which that Input Report was received.

## HID Report Items

The Driver does not support HID Report Items.

### HID Report Item packing options

The Driver does not support HID Report Item packing options.

## HID Usages

No additional clarification required.

### HID Usage Types

There is a clarification required with respect to the use of Static and Dynamic flags (SF and DF, respectively.)

The Driver expects these to be specified in the Report Descriptor as 8-bit values, of which only two values are used: ‘0’ and ‘1.’ For further information see the Report Descriptor examples.

### HID Selectors

Some clarification is required as to how to use this mechanism. This section is based on the example in the Specification.

The Report Descriptor example in the Specification is used in conjunction with the following C-language enumerated set of values.

enum

{

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL\_ENUM = 1,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL\_ENUM,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL\_ENUM,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL\_ENUM,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL\_ENUM,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL\_ENUM,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL\_ENUM,

} HID\_USAGE\_SENSOR\_STATE\_VALUES;

Note that the enumerated values above begin at “1”, whereas the enumerated values in the specification incorrectly start at “0.” For further details of HID selectors and named arrays, see sections 3.4.2.1, 3.4.2.1, and A.4 of the HID usage specification.

On the Sensor side, the implementer will insert the enumerated value into the location in the Feature or Input buffer that represents HID\_INPUT(Data\_Arr\_Abs) or HID\_FEATURE(Data\_Arr\_Abs) for that usage. On the driver side, this will be interpreted according to the position of the selector value in the list of selects, so the enumerated value order and the selector usage order must be the same.

As an example, if the device implementer inserts a “2” into the Input buffer, the driver will actually receive the Usage for HID\_USAGE\_SENSOR\_STATE\_READY\_SEL when the Input report is parsed.

When a selector is to be written to the device by the driver, the selector Usage is specified by the driver and the device will receive the corresponding enumerated value in the buffer.

## HID Usage Page

No additional clarification required.

## HID Units

As noted in the Specification, Usages have by definition a default Unit of Measure, which will be mentioned in the Usages Table. These default units are noted in sections 1.6 through 1.17.

In no case may the default unit for a sensor supported by the driver be overridden. If units are explicitly specified in the descriptor, note the documented persistence behavior in section 3.9 of the HID usage specification. Units are global items that persist across items until disabling them.

## HID Unit Exponents

No additional clarification is required.

## 3D Coordinates and Compass Points

Please see [Integrating Motion and Orientation Sensors](http://msdn.microsoft.com/en-us/library/windows/hardware/br259127.aspx).

# Illustrative Examples

This section clarifies what is and is not supported by the driver and the specifics about how to make a Sensor compliant with the driver.

Section 4.1 provides a sample “C” language “include file” the version of which in this document should be used by implementers building Sensors compliant with the driver. Implementers are strongly encouraged to use this include file when creating report descriptors.

Section 4.2 provides descriptions of the special constructions supported by the driver such as: Modifiers, Thresholds, Custom Sensors, and Generic Sensors.

Section 4.3 provides prescriptive examples of HID Report Descriptors for all sensor types compliant with the driver. All of these examples employ use of the “include file” described in Section 4.1 of this companion document to the Specification.

## Include File Definitions

These definitions are based on the Specification definitions, but have been extended where clarification was needed, particularly for the “enum” values that accompany the “selector” usages. Implementers should use this version of the definitions when building a Device to work with the driver.

Implementers are strongly encouraged to use this include file when defining report descriptors.

Note that the #defines in this include file are not identical to the similar include file in the Specification; certain enhancements have been included for ease in creating report descriptors.

This include file is compatible with all of the sample Sensor Report Descriptors in Section 4.3. This include file is also used for the examples in Section 4.2.

Examples of these macros are listed below:

hid\_sensor\_spec.h.

////////////////////////////////////////////////////////////////////////////////////

//

// HidSensorSpec.h : Defines compliant with released HID Sensor Usages.

//

////////////////////////////////////////////////////////////////////////////////////

#ifndef \_HIDSENSORSPEC\_H\_

#define \_HIDSENSORSPEC\_H\_

#define HID\_USAGE\_PAGE\_SENSOR 0x05,0x20

//sensor category usages

#define HID\_USAGE\_SENSOR\_TYPE\_COLLECTION 0x09,0x01

//sensor category biometric

#define HID\_USAGE\_SENSOR\_CATEGORY\_BIOMETRIC 0x09,0x10

#define HID\_USAGE\_SENSOR\_TYPE\_BIOMETRIC\_PRESENCE 0x09,0x11

#define HID\_USAGE\_SENSOR\_TYPE\_BIOMETRIC\_PROXIMITY 0x09,0x12

#define HID\_USAGE\_SENSOR\_TYPE\_BIOMETRIC\_TOUCH 0x09,0x13

//sensor category electrical

#define HID\_USAGE\_SENSOR\_CATEGORY\_ELECTRICAL 0x09,0x20

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_CAPACITANCE 0x09,0x21

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_CURRENT 0x09,0x22

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_POWER 0x09,0x23

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_INDUCTANCE 0x09,0x24

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_RESISTANCE 0x09,0x25

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_VOLTAGE 0x09,0x26

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_POTENTIOMETER 0x09,0x27

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_FREQUENCY 0x09,0x28

#define HID\_USAGE\_SENSOR\_TYPE\_ELECTRICAL\_PERIOD 0x09,0x29

//sensor category environmental

#define HID\_USAGE\_SENSOR\_CATEGORY\_ENVIRONMENTAL 0x09,0x30

#define HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE 0x09,0x31

#define HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_HUMIDITY 0x09,0x32

#define HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_TEMPERATURE 0x09,0x33

#define HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_WIND\_DIRECTION 0x09,0x34

#define HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_WIND\_SPEED 0x09,0x35

//sensor category light

#define HID\_USAGE\_SENSOR\_CATEGORY\_LIGHT 0x09,0x40

#define HID\_USAGE\_SENSOR\_TYPE\_LIGHT\_AMBIENTLIGHT 0x09,0x41

#define HID\_USAGE\_SENSOR\_TYPE\_LIGHT\_CONSUMER\_INFRARED 0x09,0x42

//sensor category location

#define HID\_USAGE\_SENSOR\_CATEGORY\_LOCATION 0x09,0x50

#define HID\_USAGE\_SENSOR\_TYPE\_LOCATION\_BROADCAST 0x09,0x51

#define HID\_USAGE\_SENSOR\_TYPE\_LOCATION\_DEAD\_RECKONING 0x09,0x52

#define HID\_USAGE\_SENSOR\_TYPE\_LOCATION\_GPS 0x09,0x53

#define HID\_USAGE\_SENSOR\_TYPE\_LOCATION\_LOOKUP 0x09,0x54

#define HID\_USAGE\_SENSOR\_TYPE\_LOCATION\_OTHER 0x09,0x55

#define HID\_USAGE\_SENSOR\_TYPE\_LOCATION\_STATIC 0x09,0x56

#define HID\_USAGE\_SENSOR\_TYPE\_LOCATION\_TRIANGULATION 0x09,0x57

//sensor category mechanical

#define HID\_USAGE\_SENSOR\_CATEGORY\_MECHANICAL 0x09,0x60

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_BOOLEAN\_SWITCH 0x09,0x61

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_BOOLEAN\_SWITCH\_ARRAY 0x09,0x62

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_MULTIVALUE\_SWITCH 0x09,0x63

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_FORCE 0x09,0x64

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_PRESSURE 0x09,0x65

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_STRAIN 0x09,0x66

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_SCALE\_WEIGHT 0x09,0x67

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_VIBRATOR 0x09,0x68

#define HID\_USAGE\_SENSOR\_TYPE\_MECHANICAL\_HALL\_EFFECT\_SWITCH 0x09,0x69

//sensor category motion

#define HID\_USAGE\_SENSOR\_CATEGORY\_MOTION 0x09,0x70

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_ACCELEROMETER\_1D 0x09,0x71

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_ACCELEROMETER\_2D 0x09,0x72

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_ACCELEROMETER\_3D 0x09,0x73

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_GYROMETER\_1D 0x09,0x74

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_GYROMETER\_2D 0x09,0x75

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_GYROMETER\_3D 0x09,0x76

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_MOTION\_DETECTOR 0x09,0x77

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_SPEEDOMETER 0x09,0x78

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_ACCELEROMETER 0x09,0x79

#define HID\_USAGE\_SENSOR\_TYPE\_MOTION\_GYROMETER 0x09,0x7A

//sensor category orientation

#define HID\_USAGE\_SENSOR\_CATEGORY\_ORIENTATION 0x09,0x80

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_COMPASS\_1D 0x09,0x81

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_COMPASS\_2D 0x09,0x82

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_COMPASS\_3D 0x09,0x83

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_INCLINOMETER\_1D 0x09,0x84

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_INCLINOMETER\_2D 0x09,0x85

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_INCLINOMETER\_3D 0x09,0x86

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_DISTANCE\_1D 0x09,0x87

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_DISTANCE\_2D 0x09,0x88

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_DISTANCE\_3D 0x09,0x89

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_DEVICE\_ORIENTATION 0x09,0x8A

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_COMPASS 0x09,0x8B

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_INCLINOMETER 0x09,0x8C

#define HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_DISTANCE 0x09,0x8D

//sensor category scanner

#define HID\_USAGE\_SENSOR\_CATEGORY\_SCANNER 0x09,0x90

#define HID\_USAGE\_SENSOR\_TYPE\_SCANNER\_BARCODE 0x09,0x91

#define HID\_USAGE\_SENSOR\_TYPE\_SCANNER\_RFID 0x09,0x92

#define HID\_USAGE\_SENSOR\_TYPE\_SCANNER\_NFC 0x09,0x93

//sensor category time

#define HID\_USAGE\_SENSOR\_CATEGORY\_TIME 0x09,0xA0

#define HID\_USAGE\_SENSOR\_TYPE\_TIME\_ALARM 0x09,0xA1

#define HID\_USAGE\_SENSOR\_TYPE\_TIME\_RTC 0x09,0xA2

//sensor category other

#define HID\_USAGE\_SENSOR\_CATEGORY\_OTHER 0x09,0xE0

#define HID\_USAGE\_SENSOR\_TYPE\_OTHER\_CUSTOM 0x09,0xE1

#define HID\_USAGE\_SENSOR\_TYPE\_OTHER\_GENERIC 0x09,0xE2

#define HID\_USAGE\_SENSOR\_TYPE\_OTHER\_GENERIC\_ENUMERATOR 0x09,0xE3

//unit usages

#define HID\_USAGE\_SENSOR\_UNITS\_NOT\_SPECIFIED 0x65,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_LUX 0x67,0xE1,0x00,0x00,0x01 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_KELVIN 0x67,0x01,0x00,0x01,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_FAHRENHEIT 0x67,0x03,0x00,0x01,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_PASCAL 0x66,0xF1,0xE1 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_NEWTON 0x66,0x11,0xE1 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_METERS\_PER\_SECOND 0x66,0x11,0xF0 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_METERS\_PER\_SEC\_SQRD 0x66,0x11,0xE0 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_FARAD 0x67,0xE1,0x4F,0x20,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_AMPERE 0x67,0x01,0x00,0x10,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_WATT 0x66,0x21,0xD1 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_HENRY 0x67,0x21,0xE1,0xE0,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_OHM 0x67,0x21,0xD1,0xE0,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_VOLT 0x67,0x21,0xD1,0xF0,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_HERTZ 0x66,0x01,0xF0 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_DEGREES 0x65,0x14 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_DEGREES\_PER\_SECOND 0x66,0x14,0xF0 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_DEGREES\_PER\_SEC\_SQRD 0x66,0x14,0xE0 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_RADIANS 0x65,0x12 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_RADIANS\_PER\_SECOND 0x66,0x12,0xF0 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_RADIANS\_PER\_SEC\_SQRD 0x66,0x12,0xE0 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_SECOND 0x66,0x01,0x10 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_GAUSS 0x67,0x01,0xE1,0xF0,0x00 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_GRAM 0x66,0x01,0x01 // Unit

#define HID\_USAGE\_SENSOR\_UNITS\_CENTIMETER 0x65,0x11 // Unit

#ifdef DEFINE\_NON\_HID\_UNITS

#define HID\_USAGE\_SENSOR\_UNITS\_CELSIUS “Use Unit(Kelvin) and subtract 273.15”

#define HID\_USAGE\_SENSOR\_UNITS\_KILOGRAM “Use Unit(gram) and UnitExponent(0x03)”

#define HID\_USAGE\_SENSOR\_UNITS\_METER “Use Unit(centimeter) and UnitExponent(0x02)”

#define HID\_USAGE\_SENSOR\_UNITS\_BAR “Use Unit(Pascal) and UnitExponent(0x05)”

#define HID\_USAGE\_SENSOR\_UNITS\_KNOT “Use Unit(m/s) and multiply by 1852/3600”

#define HID\_USAGE\_SENSOR\_UNITS\_PERCENT “Use Unit(Not\_Specified)”

#define HID\_USAGE\_SENSOR\_UNITS\_G “Use Unit(m/s2) and divide by 9.8”

#define HID\_USAGE\_SENSOR\_UNITS\_MILLISECOND “Use Unit(second) and UnitExponent(0x0D)”

#define HID\_USAGE\_SENSOR\_UNITS\_MILLIGAUSS “Use Unit(Gauss) and UnitExponent(0x0D)”

#endif

//unit deprecated usages

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_LUX 0x01

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_KELVIN 0x02

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_CELSIUS 0x03

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_PASCAL 0x04

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_NEWTON 0x05

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_METERS\_PER\_SECOND 0x06

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_KILOGRAM 0x07

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_METER 0x08

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_METERS\_PER\_SEC\_SQRD 0x09

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_FARAD 0x0A

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_AMPERE 0x0B

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_WATT 0x0C

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_HENRY 0x0D

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_OHM 0x0E

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_VOLT 0x0F

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_HERTZ 0x10

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_BAR 0x11

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_DEGREES\_ANTI\_CLOCKWISE 0x12

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_DEGREES\_CLOCKWISE 0x13

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_DEGREE 0x14

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_DEGREES\_PER\_SECOND 0x15

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_KNOT 0x16

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_PERCENT 0x17

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_SECOND 0x18

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_MILLISECOND 0x19

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_G 0x1A

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_BYTES 0x1B

#define HID\_USAGE\_SENSOR\_UNITS\_DEPRECATED\_MILLIGAUSS 0x1C

//data type usage modifiers -- we use them as modifiers for sensor properties & data fields

//to create thresholds, for example.

//NOTE: the usage tables actually define these as two bytes, but in order

//to get the define macros to work so these are ‘or-ed’ these are defined

//here as only one byte.

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_NONE 0x00 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS 0x10 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX 0x20 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN 0x30 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY 0x40 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION 0x50 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_THRESHOLD\_HIGH 0x60 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_THRESHOLD\_LOW 0x70 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_CALIBRATION\_OFFSET 0x80 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_CALIBRATION\_MULTIPLIER 0x90 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_REPORT\_INTERVAL 0xA0 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_FREQUENCY\_MAX 0xB0 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_PERIOD\_MAX 0xC0 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_RANGE\_PCT 0xD0 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_REL\_PCT 0xE0 // US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_VENDOR\_RESERVED 0xF0 // US

//state usages

#define HID\_USAGE\_SENSOR\_STATE 0x0A,0x01,0x02 // NAry

//state selectors

#define HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL\_SEL 0x0A,0x00,0x08 // Sel

#define HID\_USAGE\_SENSOR\_STATE\_READY\_SEL\_SEL 0x0A,0x01,0x08 // Sel

#define HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL\_SEL 0x0A,0x02,0x08 // Sel

#define HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL\_SEL 0x0A,0x03,0x08 // Sel

#define HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL\_SEL 0x0A,0x04,0x08 // Sel

#define HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL\_SEL 0x0A,0x05,0x08 // Sel

#define HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL\_SEL 0x0A,0x06,0x08 // Sel

//state enums

#define HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL\_ENUM 0x01 // Enum

#define HID\_USAGE\_SENSOR\_STATE\_READY\_SEL\_ENUM 0x02 // Enum

#define HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL\_ENUM 0x03 // Enum

#define HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL\_ENUM 0x04 // Enum

#define HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL\_ENUM 0x05 // Enum

#define HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL\_ENUM 0x06 // Enum

#define HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL\_ENUM 0x07 // Enum

//state deprecated enums

#define HID\_USAGE\_SENSOR\_STATE\_DEPRECATED\_UNKNOWN\_ENUM 0x00

#define HID\_USAGE\_SENSOR\_STATE\_DEPRECATED\_NOT\_AVAILABLE\_ENUM 0x01

#define HID\_USAGE\_SENSOR\_STATE\_DEPRECATED\_READY\_ENUM 0x02

#define HID\_USAGE\_SENSOR\_STATE\_DEPRECATED\_NO\_DATA\_ENUM 0x03

#define HID\_USAGE\_SENSOR\_STATE\_DEPRECATED\_INITIALIZING\_ENUM 0x04

#define HID\_USAGE\_SENSOR\_STATE\_DEPRECATED\_ACCESS\_DENIED\_ENUM 0x05

#define HID\_USAGE\_SENSOR\_STATE\_DEPRECATED\_ERROR\_ENUM 0x06

//event usages

#define HID\_USAGE\_SENSOR\_EVENT 0x0A,0x02,0x02 // NAry

//event selectors

#define HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL\_SEL 0x0A,0x10,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL\_SEL 0x0A,0x11,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL\_SEL 0x0A,0x12,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL\_SEL 0x0A,0x13,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL\_SEL 0x0A,0x14,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL\_SEL 0x0A,0x15,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_MAX\_REACHED\_SEL 0x0A,0x16,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_MIN\_REACHED\_SEL 0x0A,0x17,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_UPWARD\_SEL 0x0A,0x18,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THESHOLD\_CROSS\_ABOVE\_SEL HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_UPWARD\_SEL

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_DOWNWARD\_SEL 0x0A,0x19,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_BELOW\_SEL HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_DOWNWARD\_SEL

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_UPWARD\_SEL 0x0A,0x1A,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_ABOVE\_SEL HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_UPWARD\_SEL

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_DOWNWARD\_SEL 0x0A,0x1B,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_BELOW\_SEL HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_DOWNWARD\_SEL

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_UPWARD\_SEL 0x0A,0x1C,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_ABOVE\_SEL HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_UPWARD\_SEL

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_DOWNWARD\_SEL 0x0A,0x1D,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_BELOW\_SEL HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_DOWNWARD\_SEL

#define HID\_USAGE\_SENSOR\_EVENT\_PERIOD\_EXCEEDED\_SEL 0x0A,0x1E,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_FREQUENCY\_EXCEEDED\_SEL 0x0A,0x1F,0x08 // Sel

#define HID\_USAGE\_SENSOR\_EVENT\_COMPLEX\_TRIGGER\_SEL 0x0A,0x20,0x08 // Sel

//event enums

#define HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL\_ENUM 0x01 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL\_ENUM 0x02 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL\_ENUM 0x03 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL\_ENUM 0x04 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL\_ENUM 0x05 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL\_ENUM 0x06 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_MAX\_REACHED\_ENUM 0x07 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_MIN\_REACHED\_ENUM 0x08 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_UPWARD\_ENUM 0x09 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THESHOLD\_CROSS\_ABOVE\_ENUM HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_UPWARD\_ENUM

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_DOWNWARD\_ENUM 0x0A // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_BELOW\_ENUM HID\_USAGE\_SENSOR\_EVENT\_HIGH\_THRESHOLD\_CROSS\_DOWNWARD\_ENUM

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_UPWARD\_ENUM 0x0B // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_ABOVE\_ENUM HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_UPWARD\_ENUM

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_DOWNWARD\_ENUM 0x0C // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_BELOW\_ENUM HID\_USAGE\_SENSOR\_EVENT\_LOW\_THRESHOLD\_CROSS\_DOWNWARD\_ENUM

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_UPWARD\_ENUM 0x0D // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_ABOVE\_ENUM HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_UPWARD\_ENUM

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_DOWNWARD\_ENUM 0x0E // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_BELOW\_ENUM HID\_USAGE\_SENSOR\_EVENT\_ZERO\_THRESHOLD\_CROSS\_DOWNWARD\_ENUM

#define HID\_USAGE\_SENSOR\_EVENT\_PERIOD\_EXCEEDED\_ENUM 0x0F // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_FREQUENCY\_EXCEEDED\_ENUM 0x10 // Enum

#define HID\_USAGE\_SENSOR\_EVENT\_COMPLEX\_TRIGGER\_ENUM 0x11 // Enum

//event deprecated enums

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_UNKNOWN\_ENUM 0x00

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_STATE\_CHANGED\_ENUM 0x01

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_PROPERTY\_CHANGED\_ENUM 0x02

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_DATA\_UPDATE\_ENUM 0x03

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_POLL\_RESPONSE\_ENUM 0x04

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_CHANGE\_SENSITIVITY\_ENUM 0x05

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_MAX\_REACHED\_ENUM 0x06

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_MIN\_REACHED\_ENUM 0x07

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_HIGH\_THRESHHOLD\_CROSS\_ABOVE\_ENUM 0x08

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_HIGH\_THRESHHOLD\_CROSS\_BELOW\_ENUM 0x09

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_LOW\_THRESHHOLD\_CROSS\_ABOVE\_ENUM 0x0A

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_LOW\_THRESHHOLD\_CROSS\_BELOW\_ENUM 0x0B

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_ZERO\_THRESHOLD\_CROSS\_ABOVE\_ENUM 0x0C

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_ZERO\_THRESHOLD\_CROSS\_BELOW\_ENUM 0x0D

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_PERIOD\_EXCEEDED\_ENUM 0x0E

#define HID\_USAGE\_SENSOR\_EVENT\_DEPRECATED\_FREQUENCY\_EXCEEDED\_ENUM 0x0F

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY 0x0A,0x00,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_FRIENDLY\_NAME 0x0A,0x01,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_PERSISTENT\_UNIQUE\_ID 0x0A,0x02,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_STATUS 0x0A,0x03,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_MINIMUM\_REPORT\_INTERVAL 0x0A,0x04,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MANUFACTURER 0x0A,0x05,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MODEL 0x0A,0x06,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_SERIAL\_NUMBER 0x0A,0x07,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_DESCRIPTION 0x0A,0x08,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE 0x0A,0x09,0x03 // NAry

//begin connection type selectors

#define HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL 0x0A,0x30,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL\_SEL 0x0A,0x31,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL\_SEL 0x0A,0x32,0x08 // Sel

//end connection type selectors

//begin connection type enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_ENUM 0x01 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL\_ENUM 0x02 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL\_ENUM 0x03 // Enum

//end connection type enums

//begin connection type deprecated enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_CONNECTION\_TYPE\_PC\_INTEGRATED\_ENUM 0x00 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_CONNECTION\_TYPE\_PC\_ATTACHED\_ENUM 0x01 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_CONNECTION\_TYPE\_PC\_EXTERNAL\_ENUM 0x02 // Enum

//end connection type deprecated enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_DEVICE\_PATH 0x0A,0x0A,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_HARDWARE\_REVISION 0x0A,0x0B,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_FIRMWARE\_VERSION 0x0A,0x0C,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_RELEASE\_DATE 0x0A,0x0D,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL 0x0A,0x0E,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_ABS 0x0A,0x0F,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_RANGE\_PCT 0x0A,0x10,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_REL\_PCT 0x0A,0x11,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_ACCURACY 0x0A,0x12,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_RESOLUTION 0x0A,0x13,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_RANGE\_MAXIMUM 0x0A,0x14,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_RANGE\_MINIMUM 0x0A,0x15,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE 0x0A,0x16,0x03 // NAry

//begin reporting state selectors

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL\_SEL 0x0A,0x40,0x08 // Sel

#define HID\_USAGE\_REPORTING\_STATE\_ON\_NONE\_SEL HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL\_SEL

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL\_SEL 0x0A,0x41,0x08 // Sel

#define HID\_USAGE\_REPORTING\_STATE\_ON\_ALL\_SEL HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL\_SEL

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL\_SEL 0x0A,0x42,0x08 // Sel

#define HID\_USAGE\_REPORTING\_STATE\_ON\_THRESHOLD\_SEL HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL\_SEL

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL\_WAKE\_SEL 0x0A,0x43,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL\_WAKE\_SEL 0x0A,0x44,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL\_SEL 0x0A,0x45,0x08 // Sel

//end reporting state selectors

//begin reporting state enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL\_ENUM 0x01 // Enum

#define HID\_USAGE\_REPORTING\_STATE\_ON\_NONE\_ENUM HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL\_ENUM

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL\_ENUM 0x02 // Enum

#define HID\_USAGE\_REPORTING\_STATE\_ON\_ALL\_ENUM HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL\_ENUM

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL\_ENUM 0x03 // Enum

#define HID\_USAGE\_REPORTING\_STATE\_ON\_THRESHOLD\_ENUM HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL\_ENUM

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL\_WAKE\_ENUM 0x04 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL\_WAKE\_ENUM 0x05 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL\_ENUM 0x06 // Enum

//end reporting state enums

//begin reporting state deprecated enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_NO\_EVENTS\_ENUM 0x00 // Enum

#define HID\_USAGE\_DEPRECATED\_REPORTING\_STATE\_ON\_NONE\_ENUM HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_NO\_EVENTS\_ENUM

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_ALL\_EVENTS\_ENUM 0x01 // Enum

#define HID\_USAGE\_DEPRECATED\_REPORTING\_STATE\_ON\_ALL\_ENUM HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_ALL\_EVENTS\_ENUM

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_ENUM 0x02 // Enum

#define HID\_USAGE\_DEPRECATED\_REPORTING\_STATE\_ON\_THRESHOLD\_ENUM HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_ENUM

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_ENUM 0x03 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_ENUM 0x04 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_ENUM 0x05 // Enum

//end reporting state deprecated enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_SAMPLING\_RATE 0x0A,0x17,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_RESPONSE\_CURVE 0x0A,0x18,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE 0x0A,0x19,0x03 // NAry

//begin power state selectors

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL 0x0A,0x50,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL 0x0A,0x51,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL 0x0A,0x52,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL 0x0A,0x53,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL 0x0A,0x54,0x08 // Sel

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL 0x0A,0x55,0x08 // Sel

//end power state selectors

//begin power state enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_ENUM 0x01 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_ENUM 0x02 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_ENUM 0x03 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_ENUM 0x04 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_ENUM 0x05 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_ENUM 0x06 // Enum

//end power state enums

//begin deprecated power state enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_POWER\_STATE\_UNDEFINED\_ENUM 0x00 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_POWER\_STATE\_D0\_FULL\_POWER\_ENUM 0x01 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_POWER\_STATE\_D1\_LOW\_POWER\_ENUM 0x02 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_ENUM 0x03 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_ENUM 0x04 // Enum

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_POWER\_STATE\_D4\_POWER\_OFF\_ENUM 0x05 // Enum

//end deprecated power state enums

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_FEATURE\_PAGE\_COUNT 0x0A,0x1A,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_FEATURE\_PAGE\_ID 0x0A,0x1B,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_INPUT\_PAGE\_COUNT 0x0A,0x1C,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_DEPRECATED\_INPUT\_PAGE\_ID 0x0A,0x1D,0x03

//data type location

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION 0x0A,0x00,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_DESIRED\_ACCURACY 0x0A,0x01,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ALTITUDE\_ANTENNA\_SEALEVEL 0x0A,0x02,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_DIFFERENTIAL\_REFERENCE\_STATION\_ID 0x0A,0x03,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ALTITIDE\_ELIPSOID\_ERROR 0x0A,0x04,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ALTITIDE\_ELIPSOID 0x0A,0x05,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ALTITUDE\_SEALEVEL\_ERROR 0x0A,0x06,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ALTITUDE\_SEALEVEL 0x0A,0x07,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_DGPS\_DATA\_AGE 0x0A,0x08,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ERROR\_RADIUS 0x0A,0x09,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_FIX\_QUALITY 0x0A,0x0A,0x04 // NAry

//begin fix quality selectors

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_QUALITY\_NO\_FIX 0x0A,0x70,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_QUALITY\_GPS 0x0A,0x71,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_QUALITY\_DGPS 0x0A,0x72,0x08 // Sel

//end fix quality selectors

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_FIX\_TYPE 0x0A,0x0B,0x04 // NAry

//begin fix type selectors

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_NO\_FIX 0x0A,0x80,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_GPS\_SPS\_MODE\_FIX\_VALID 0x0A,0x81,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_DGPS\_SPS\_MODE\_FIX\_VALID 0x0A,0x82,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_GPS\_PPS\_MODE\_FIX\_VALID 0x0A,0x83,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_REAL\_TIME\_KINEMATIC 0x0A,0x84,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_FLOAT\_RTK 0x0A,0x85,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_ESTIMATED\_DEAD\_RECKONING 0x0A,0x86,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_MANUAL\_INPUT\_MODE 0x0A,0x87,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_FIX\_TYPE\_SIMULATOR\_MODE 0x0A,0x88,0x08 // Sel

//end fix type selectors

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_GEOIDAL\_SEPARATION 0x0A,0x0C,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_GPS\_OPERATION\_MODE 0x0A,0x0D,0x04 // NAry

//begin gps operation mode selectors

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_OP\_MODE\_MANUAL 0x0A,0x90,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_OP\_MODE\_AUTOMATIC 0x0A,0x91,0x08 // Sel

//end gps operation mode selectors

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_GPS\_SELECTION\_MODE 0x0A,0x0E,0x04 // NAry

//begin gps selection mode selectors

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_SEL\_MODE\_AUTONOMOUS 0x0A,0xA0,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_SEL\_MODE\_DGPS 0x0A,0xA1,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_SEL\_MODE\_ESTIMATED\_DEAD\_RECKONING 0x0A,0xA2,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_SEL\_MODE\_MANUAL\_INPUT 0x0A,0xA3,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_SEL\_MODE\_SIMULATOR 0x0A,0xA4,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_SEL\_MODE\_DATA\_NOT\_VALID 0x0A,0xA5,0x08 // Sel

//end gps selection mode selectors

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_GPS\_STATUS 0x0A,0x0F,0x04 // NAry

//begin gps status selectors

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_STATUS\_DATA\_VALID 0x0A,0xB0,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_GPS\_STATUS\_DATA\_NOT\_VALID 0x0A,0xB1,0x08 // Sel

//end gps status selectors

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_POSITION\_DILUTION\_OF\_PRECISION 0x0A,0x10,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_HORIZONTAL\_DILUTION\_OF\_PRECISION 0x0A,0x11,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_VERTICAL\_DILUTION\_OF\_PRECISION 0x0A,0x12,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_LATITUDE 0x0A,0x13,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_LONGITUDE 0x0A,0x14,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_TRUE\_HEADING 0x0A,0x15,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_MAGNETIC\_HEADING 0x0A,0x16,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_MAGNETIC\_VARIATION 0x0A,0x17,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SPEED 0x0A,0x18,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_IN\_VIEW 0x0A,0x19,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_IN\_VIEW\_AZIMUTH 0x0A,0x1A,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_IN\_VIEW\_ELEVATION 0x0A,0x1B,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_IN\_VIEW\_ID 0x0A,0x1C,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_IN\_VIEW\_PRNs 0x0A,0x1D,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_IN\_VIEW\_STN\_RATIO 0x0A,0x1E,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_USED\_COUNT 0x0A,0x1F,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_SATELLITES\_USED\_PRNs 0x0A,0x20,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_NMEA\_SENTENCE 0x0A,0x21,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ADDRESS\_LINE\_1 0x0A,0x22,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_ADDRESS\_LINE\_2 0x0A,0x23,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_CITY 0x0A,0x24,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_STATE\_OR\_PROVINCE 0x0A,0x25,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_COUNTRY\_OR\_REGION 0x0A,0x26,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LOCATION\_POSTAL\_CODE 0x0A,0x27,0x04

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY\_LOCATION 0x0A,0x2A,0x04

#define HID\_USAGE\_SENSOR\_PROPERTY\_LOCATION\_DESIRED\_ACCURACY 0x0A,0x2B,0x04 // NAry

//begin location desired accuracy selectors

#define HID\_USAGE\_SENSOR\_DESIRED\_ACCURACY\_DEFAULT 0x0A,0x60,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DESIRED\_ACCURACY\_HIGH 0x0A,0x61,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DESIRED\_ACCURACY\_MEDIUM 0x0A,0x62,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DESIRED\_ACCURACY\_LOW 0x0A,0x63,0x08 // Sel

//end location desired accuracy selectors

//data type environmental

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL 0x0A,0x30,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE 0x0A,0x31,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_REFERENCE\_PRESSURE 0x0A,0x32,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_RELATIVE\_HUMIDITY 0x0A,0x33,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE 0x0A,0x34,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_WIND\_DIRECTION 0x0A,0x35,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_WIND\_SPEED 0x0A,0x36,0x04

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY\_ENVIRONMENTAL 0x0A,0x40,0x04

#define HID\_USAGE\_SENSOR\_PROPERTY\_ENVIRONMENTAL\_REFERENCE\_PRESSURE 0x0A,0x41,0x04

//data type motion

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_MOTION 0x0A,0x50,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_STATE 0x0A,0x51,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION 0x0A,0x52,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS 0x0A,0x53,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS 0x0A,0x54,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS 0x0A,0x55,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY 0x0A,0x56,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_X\_AXIS 0x0A,0x57,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_Y\_AXIS 0x0A,0x58,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_Z\_AXIS 0x0A,0x59,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_POSITION 0x0A,0x5A,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_POSITION\_X\_AXIS 0x0A,0x5B,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_POSITION\_Y\_AXIS 0x0A,0x5C,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_POSITION\_Z\_AXIS 0x0A,0x5D,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED 0x0A,0x5E,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_INTENSITY 0x0A,0x5F,0x04

//data type orientation

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION 0x0A,0x70,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING 0x0A,0x71,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING\_X 0x0A,0x72,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING\_Y 0x0A,0x73,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING\_Z 0x0A,0x74,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_COMPENSATED\_MAGNETIC\_NORTH 0x0A,0x75,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_COMPENSATED\_TRUE\_NORTH 0x0A,0x76,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_NORTH 0x0A,0x77,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TRUE\_NORTH 0x0A,0x78,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_DISTANCE 0x0A,0x79,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_DISTANCE\_X 0x0A,0x7A,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_DISTANCE\_Y 0x0A,0x7B,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_DISTANCE\_Z 0x0A,0x7C,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_DISTANCE\_OUT\_OF\_RANGE 0x0A,0x7D,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT 0x0A,0x7E,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_X 0x0A,0x7F,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_Y 0x0A,0x80,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_Z 0x0A,0x81,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_ROTATION\_MATRIX 0x0A,0x82,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_QUATERNION 0x0A,0x83,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX 0x0A,0x84,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_X\_AXIS 0x0A,0x85,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_Y\_AXIS 0x0A,0x86,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_Z\_AXIS 0x0A,0x87,0x04

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY 0x0A,0x88,0x04 // NAry

//begin orientation magnetometer accuracy selectors

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_LOW 0x0A,0xE0,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_MEDIUM 0x0A,0xE1,0x08 // Sel

#define HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_HIGH 0x0A,0xE2,0x08 // Sel

//data type mechanical

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL 0x0A,0x90,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_BOOLEAN\_SWITCH\_STATE 0x0A,0x91,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_BOOLEAN\_SWITCH\_ARRAY\_STATES 0x0A,0x92,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_MULTIVALUE\_SWITCH\_VALUE 0x0A,0x93,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_FORCE 0x0A,0x94,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_ABSOLUTE\_PRESSURE 0x0A,0x95,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_GAUGE\_PRESSURE 0x0A,0x96,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_STRAIN 0x0A,0x97,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_WEIGHT 0x0A,0x98,0x04

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY\_MECHANICAL 0x0A,0xA0,0x04

#define HID\_USAGE\_SENSOR\_PROPERTY\_MECHANICAL\_VIBRATION\_STATE 0x0A,0xA1,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_VIBRATION\_SPEED\_FORWARD 0x0A,0xA2,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MECHANICAL\_VIBRATION\_SPEED\_BACKWARD 0x0A,0xA3,0x04

//data type biometric

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC 0x0A,0xB0,0x04

#define HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PRESENCE 0x0A,0xB1,0x04

#define HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_RANGE 0x0A,0xB2,0x04

#define HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_OUT\_OF\_RANGE 0x0A,0xB3,0x04

#define HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_TOUCH\_STATE 0x0A,0xB4,0x04

//data type light sensor

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_LIGHT 0x0A,0xD0,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LIGHT\_ILLUMINANCE 0x0A,0xD1,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LIGHT\_COLOR\_TEMPERATURE 0x0A,0xD2,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY 0x0A,0xD3,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY\_X 0x0A,0xD4,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY\_Y 0x0A,0xD5,0x04

#define HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CONSUMER\_IR\_SENTENCE\_RECEIVE 0x0A,0xD6,0x04

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY\_LIGHT 0x0A,0xE0,0x04

#define HID\_USAGE\_SENSOR\_PROPERTY\_LIGHT\_CONSUMER\_IR\_SENTENCE\_SEND 0x0A,0xE1,0x04

//data type scanner

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_SCANNER 0x0A,0xF0,0x04

#define HID\_USAGE\_SENSOR\_DATA\_SCANNER\_RFID\_TAG 0x0A,0xF1,0x04

#define HID\_USAGE\_SENSOR\_DATA\_SCANNER\_NFC\_SENTENCE\_RECEIVE 0x0A,0xF2,0x04

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY\_SCANNER 0x0A,0xF8,0x04

#define HID\_USAGE\_SENSOR\_PROPERTY\_SCANNER\_NFC\_SENTENCE\_SEND 0x0A,0xF9,0x04

//data type electrical

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL 0x0A,0x00,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_CAPACITANCE 0x0A,0x01,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_CURRENT 0x0A,0x02,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_POWER 0x0A,0x03,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_INDUCTANCE 0x0A,0x04,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_RESISTANCE 0x0A,0x05,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_VOLTAGE 0x0A,0x06,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_FREQUENCY 0x0A,0x07,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_PERIOD 0x0A,0x08,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ELECTRICAL\_PERCENT\_OF\_RANGE 0x0A,0x09,0x05

//data type time

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_TIME 0x0A,0x20,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_YEAR 0x0A,0x21,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_MONTH 0x0A,0x22,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_DAY 0x0A,0x23,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_DAY\_OF\_WEEK 0x0A,0x24,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_HOUR 0x0A,0x25,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_MINUTE 0x0A,0x26,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_SECOND 0x0A,0x27,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_MILLISECOND 0x0A,0x28,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_TIMESTAMP 0x0A,0x29,0x05

#define HID\_USAGE\_SENSOR\_DATA\_TIME\_JULIAN\_DAY\_OF\_YEAR 0x0A,0x2A,0x05

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY\_TIME 0x0A,0x30,0x05

#define HID\_USAGE\_SENSOR\_PROPERTY\_TIME\_TIME\_ZONE\_OFFSET\_FROM\_UTC 0x0A,0x31,0x05

#define HID\_USAGE\_SENSOR\_PROPERTY\_TIME\_TIME\_ZONE\_NAME 0x0A,0x32,0x05

#define HID\_USAGE\_SENSOR\_PROPERTY\_TIME\_DAYLIGHT\_SAVINGS\_TIME\_OBSERVED 0x0A,0x33,0x05

#define HID\_USAGE\_SENSOR\_PROPERTY\_TIME\_TIME\_TRIM\_ADJUSTMENT 0x0A,0x34,0x05

#define HID\_USAGE\_SENSOR\_PROPERTY\_TIME\_ARM\_ALARM 0x0A,0x35,0x05

//data type custom

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM 0x0A,0x40,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE 0x0A,0x41,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_BOOLEAN\_ARRAY 0x0A,0x42,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE 0x0A,0x43,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_1 0x0A,0x44,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_2 0x0A,0x45,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_3 0x0A,0x46,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_4 0x0A,0x47,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_5 0x0A,0x48,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_6 0x0A,0x49,0x05

#if 1 //define vendor-specific (non-spec) custom datafields

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_7 0x0A,0x4A,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_8 0x0A,0x4B,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_9 0x0A,0x4C,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_10 0x0A,0x4D,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_11 0x0A,0x4E,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_12 0x0A,0x4F,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_13 0x0A,0x50,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_14 0x0A,0x51,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_15 0x0A,0x52,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_16 0x0A,0x53,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_17 0x0A,0x54,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_18 0x0A,0x55,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_19 0x0A,0x56,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_20 0x0A,0x57,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_21 0x0A,0x58,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_22 0x0A,0x59,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_23 0x0A,0x5A,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_24 0x0A,0x5B,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_25 0x0A,0x5C,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_26 0x0A,0x5D,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_27 0x0A,0x5E,0x05

#define HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_28 0x0A,0x5F,0x05

#endif

//data type generic

//data field usages (input report)

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC 0x0A,0x60,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_GUID\_OR\_PROPERTYKEY 0x0A,0x61,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_CATEGORY\_GUID 0x0A,0x62,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_TYPE\_GUID 0x0A,0x63,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_EVENT\_PROPERTYKEY 0x0A,0x64,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_PROPERTY\_PROPERTYKEY 0x0A,0x65,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_DATAFIELD\_PROPERTYKEY 0x0A,0x66,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_EVENT 0x0A,0x67,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_PROPERTY 0x0A,0x68,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_DATAFIELD 0x0A,0x69,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ENUMERATOR\_TABLE\_ROW\_INDEX 0x0A,0x6A,0x05

#define HID\_USAGE\_SENSOR\_DATA\_ENUMERATOR\_TABLE\_ROW\_COUNT 0x0A,0x6B,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_GUID\_OR\_PROPERTYKEY\_KIND 0x0A,0x6C,0x05 // NAry

//begin GorPK kind selectors

#define HID\_USAGE\_SENSOR\_GORPK\_KIND\_CATEGORY 0x0A,0xD0,0x08 // Sel

#define HID\_USAGE\_SENSOR\_GORPK\_KIND\_TYPE 0x0A,0xD1,0x08 // Sel

#define HID\_USAGE\_SENSOR\_GORPK\_KIND\_EVENT 0x0A,0xD2,0x08 // Sel

#define HID\_USAGE\_SENSOR\_GORPK\_KIND\_PROPERTY 0x0A,0xD3,0x08 // Sel

#define HID\_USAGE\_SENSOR\_GORPK\_KIND\_DATAFIELD 0x0A,0xD4,0x08 // Sel

//end GorPK kind selectors

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_GUID 0x0A,0x6D,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_PROPERTYKEY 0x0A,0x6E,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_TOP\_LEVEL\_COLLECTION\_ID 0x0A,0x6F,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_REPORT\_ID 0x0A,0x70,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_REPORT\_ITEM\_POSITION\_INDEX 0x0A,0x71,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_FIRMWARE\_VARTYPE 0x0A,0x72,0x05 // NAry

//begin firmware vartype selectors

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_NULL 0x0A,0x00,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_BOOL 0x0A,0x01,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_UI1 0x0A,0x02,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_I1 0x0A,0x03,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_UI2 0x0A,0x04,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_I2 0x0A,0x05,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_UI4 0x0A,0x06,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_I4 0x0A,0x07,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_UI8 0x0A,0x08,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_I8 0x0A,0x09,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_R4 0x0A,0x0A,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_R8 0x0A,0x0B,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_WSTR 0x0A,0x0C,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_STR 0x0A,0x0D,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_CLSID 0x0A,0x0E,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_VECTOR\_VT\_UI1 0x0A,0x0F,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E0 0x0A,0x10,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E1 0x0A,0x11,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E2 0x0A,0x12,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E3 0x0A,0x13,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E4 0x0A,0x14,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E5 0x0A,0x15,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E6 0x0A,0x16,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E7 0x0A,0x17,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E8 0x0A,0x18,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16E9 0x0A,0x19,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16EA 0x0A,0x1A,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16EB 0x0A,0x1B,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16EC 0x0A,0x1C,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16ED 0x0A,0x1D,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16EE 0x0A,0x1E,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F16EF 0x0A,0x1F,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E0 0x0A,0x20,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E1 0x0A,0x21,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E2 0x0A,0x22,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E3 0x0A,0x23,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E4 0x0A,0x24,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E5 0x0A,0x25,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E6 0x0A,0x26,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E7 0x0A,0x27,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E8 0x0A,0x28,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32E9 0x0A,0x29,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32EA 0x0A,0x2A,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32EB 0x0A,0x2B,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32EC 0x0A,0x2C,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32ED 0x0A,0x2D,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32EE 0x0A,0x2E,0x09 // Sel

#define HID\_USAGE\_SENSOR\_FIRMWARE\_VARTYPE\_VT\_F32EF 0x0A,0x2F,0x09 // Sel

//end firmware vartype selectors

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_UNIT\_OF\_MEASURE 0x0A,0x73,0x05 // NAry

//begin unit of measure selectors

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_NOT\_SPECIFIED 0x0A,0x40,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_LUX 0x0A,0x41,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_DEGREES\_KELVIN 0x0A,0x42,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_DEGREES\_CELSIUS 0x0A,0x43,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_PASCAL 0x0A,0x44,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_NEWTON 0x0A,0x45,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_METERS\_PER\_SECOND 0x0A,0x46,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_KILOGRAM 0x0A,0x47,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_METER 0x0A,0x48,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_METERS\_PER\_SEC\_SQRD 0x0A,0x49,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_FARAD 0x0A,0x4A,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_AMPERE 0x0A,0x4B,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_WATT 0x0A,0x4C,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_HENRY 0x0A,0x4D,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_OHM 0x0A,0x4E,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_VOLT 0x0A,0x4F,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_HERTZ 0x0A,0x50,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_BAR 0x0A,0x51,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_DEGREES\_ANTI\_CLOCKWISE 0x0A,0x52,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_DEGREES\_CLOCKWISE 0x0A,0x53,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_DEGREES 0x0A,0x54,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_DEGREES\_PER\_SECOND 0x0A,0x55,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_DEGREES\_PER\_SEC\_SQRD 0x0A,0x56,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_KNOT 0x0A,0x57,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_PERCENT 0x0A,0x58,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_SECOND 0x0A,0x59,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_MILLISECOND 0x0A,0x5A,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_G 0x0A,0x5B,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_BYTES 0x0A,0x5C,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_MILLIGAUSS 0x0A,0x5D,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_UNIT\_BITS 0x0A,0x5E,0x09 // Sel

//end unit of measure selectors

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_UNIT\_EXPONENT 0x0A,0x74,0x05 // NAry

//begin unit exponent selectors

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_0 0x0A,0x70,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_1 0x0A,0x71,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_2 0x0A,0x72,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_3 0x0A,0x73,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_4 0x0A,0x74,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_5 0x0A,0x75,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_6 0x0A,0x76,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_7 0x0A,0x77,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_8 0x0A,0x78,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_9 0x0A,0x79,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_A 0x0A,0x7A,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_B 0x0A,0x7B,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_C 0x0A,0x7C,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_D 0x0A,0x7D,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_E 0x0A,0x7E,0x09 // Sel

#define HID\_USAGE\_SENSOR\_GENERIC\_EXPONENT\_F 0x0A,0x7F,0x09 // Sel

//end unit exponent selectors

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_REPORT\_SIZE 0x0A,0x75,0x05

#define HID\_USAGE\_SENSOR\_DATA\_GENERIC\_REPORT\_COUNT 0x0A,0x76,0x05

//property usages (get/set feature report)

#define HID\_USAGE\_SENSOR\_PROPERTY\_GENERIC 0x0A,0x80,0x05

#define HID\_USAGE\_SENSOR\_PROPERTY\_ENUMERATOR\_TABLE\_ROW\_INDEX 0x0A,0x81,0x05

#define HID\_USAGE\_SENSOR\_PROPERTY\_ENUMERATOR\_TABLE\_ROW\_COUNT 0x0A,0x82,0x05

////////////////////////////////////////////////////////////////////////////////////

//

// Other HID definitions

//

////////////////////////////////////////////////////////////////////////////////////

//NOTE: These definitions are designed to permit compiling the HID report descriptors

// with somewhat self-explanatory information to help readability and reduce errors

//input,output,feature flags

#define Data\_Arr\_Abs 0x00

#define Const\_Arr\_Abs 0x01

#define Data\_Var\_Abs 0x02

#define Const\_Var\_Abs 0x03

#define Data\_Var\_Rel 0x06

//collection flags

#define Physical 0x00

#define Application 0x01

#define Logical 0x02

#define NamedArray 0x04

#define UsageSwitch 0x05

//other

#define Undefined 0x00

#define HID\_USAGE\_PAGE(a) 0x05,a

#define HID\_USAGE(a) 0x09,a

#define HID\_USAGE16(a,b) 0x0A,a,b

#define HID\_USAGE\_SENSOR\_DATA(a,b) a|b //This or-s the mod into usage

#define HID\_COLLECTION(a) 0xA1,a

#define HID\_REPORT\_ID(a) 0x85,a

#define HID\_REPORT\_SIZE(a) 0x75,a

#define HID\_REPORT\_COUNT(a) 0x95,a

#define HID\_USAGE\_MIN\_8(a) 0x19,a

#define HID\_USAGE\_MIN\_16(a,b) 0x1A,a,b

#define HID\_USAGE\_MAX\_8(a) 0x29,a

#define HID\_USAGE\_MAX\_16(a,b) 0x2A,a,b

#define HID\_LOGICAL\_MIN\_8(a) 0x15,a

#define HID\_LOGICAL\_MIN\_16(a,b) 0x16,a,b

#define HID\_LOGICAL\_MIN\_32(a,b,c,d) 0x17,a,b,c,d

#define HID\_LOGICAL\_MAX\_8(a) 0x25,a

#define HID\_LOGICAL\_MAX\_16(a,b) 0x26,a,b

#define HID\_LOGICAL\_MAX\_32(a,b,c,d) 0x27,a,b,c,d

#define HID\_UNIT\_EXPONENT(a) 0x55,a

#define HID\_INPUT(a) 0x81,a

#define HID\_OUTPUT(a) 0x91,a

#define HID\_FEATURE(a) 0xB1,a

#define HID\_END\_COLLECTION 0xC0

#endif

## Special Constructions

### Values, Types, and Unit Exponents

The content of the Specification is included here for convenience. Additional clarifications are provided for certain special constructions.

The HID Report Descriptors in this section use the following definitions for values, units and unit

exponents.

The value communicated as part of a Report Descriptor is in terms of the Report Size and Report Count

attributes, combined with the Logical Minimum, Logical Maximum, and Units for data values associated

with that Report Item.

The value is treated in one of three ways:

* As a bitfield
* As a signed or unsigned integer value
* As a float value

**Bitfield**

A value is identified as a bitfield when the Report Size field = 1. In this section, this is expressed as HID\_REPORT\_SIZE(1). In this case, Logical Maximum, Logical Minimum, Units and Units Exponent are not used.

Additional clarification follows:

The Driver always requires a Report Size field = 8 when describing a bitfield. The Driver identifies a value as a bitfield by the Usage rather than by the Report Size field. Any value present in the UsageValue of a bitfield Usage is interpreted as a boolean value. In this case, Logical Maximum, Logical Minimum, Units and Units Exponent are not used.

This provides for some economy of expression in the Report Descriptor by eliminating the need to provide padding for the remaining unused bits in a bitfield.

All of the Report Descriptor examples in this document use this bitfield definition.

**Unsigned Integer**

A value is identified as an unsigned integer when the ReportSize field = 8, 16 or 32 while the Units Exponent value = 0. In this section, this is expressed as HID\_REPORT\_SIZE(8), HID\_REPORT\_SIZE(16), or HID\_REPORT\_SIZE(32) respectively. Logical Minimum and Logical Maximum must both be positive values. Units can be specified or remain unspecified. Units Exponent must be = 0.

**Signed Integer**

A value is identified as a signed integer when the ReportSize field = 8, 16 or 32 while the Units Exponent value = 0. In this section, this is expressed as HID\_REPORT\_SIZE(8), HID\_REPORT\_SIZE(16), or HID\_REPORT\_SIZE(32) respectively. Logical Minimum must less than Logical Maximum. If Logical Minimum is = 0, the value of the field is treated as an unsigned number. If Logical Minimum > 0, care must be taken that the sign bit (MSb) is not = ‘1’ or the value will be treated as a negative number. Units can be specified or remain unspecified. Units Exponent must be = 0.

**Float Value**

Essentially, a float is expressed as a combination of a mantissa carried in the value field, and the

exponent expressed as power of 10 carried in the Unit Exponent field. A value is identified as a

float value when the ReportSize field = 16 or 32 while the Units Exponent value is not 0. In this

section, this is expressed as HID\_REPORT\_SIZE(16) or HID\_REPORT\_SIZE(32) respectively.

Logical Minimum must be less than Logical Maximum. If Logical Minimum is = 0, the value of the field is treated as an unsigned number. If Logical Minimum > 0, care must be taken that the sign bit (MSb) is not = ‘1’ or the value will be treated as a negative number. Units

can be specified or remain unspecified. Units Exponent must not be = 0. The Unit Exponent field is translated into powers of 10 as specified in Table 19.

Table 19. HID Unit Exponent encoding and meanings

|  |  |  |
| --- | --- | --- |
| Value | Exponent | Power of Ten |
| 0x00 | 1x10E0 | 1 |
| 0x01 | 1x10E1 | 10 |
| 0x02 | 1x10E2 | 100 |
| 0x03 | 1x10E3 | 1 000 |
| 0x04 | 1x10E4 | 10 000 |
| 0x05 | 1x10E5 | 100 000 |
| 0x06 | 1x10E6 | 1 000 000 |
| 0x07 | 1x10E7 | 10 000 000 |
| 0x08 | 1x10E-8 | 0.00 000 001 |
| 0x09 | 1x10E-7 | 0.0 000 001 |
| 0x0A | 1x10E-6 | 0.000 001 |
| 0x0B | 1x10E-5 | 0.00 001 |
| 0x0C | 1x10E-4 | 0.0 001 |
| 0x0D | 1x10E-3 | 0.001 |
| 0x0E | 1x10E-2 | 0.01 |
| 0x0F | 1x10E-1 | 0.1 |

These Unit Exponent field usages are not unique to this specification, but are the same as the standard HID definitions.

### Extended Properties

The HID Report Descriptors illustrations in Section 4.3 are meant to be examples and not prescriptive, but if Windows Hardware Certification compliance is desired they should be interpreted as prescriptive.

As previously noted, a Sensor need not support any properties whatsoever in the Feature report and only datafields in the Input Report if Windows Hardware Certification compliance is not expected. This minimal version of the Report Descriptor can enable the use of Device hardware that may not support Feature Reports.

Typically, to be Windows Hardware Certification compliant the sensor must include the following sensor properties:

.

.

.

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

.

.

This pattern will be used in almost every sensor example.The examples also typically include the following per-datafield properties, like these taken from the **Barometer** report descriptor:

.

.

.

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,HID\_USAGE\_SENSOR\_DATA\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “bar” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “bar” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “bar” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

.

.

.

Change Sensitivity should be included unless it is not supported the the sensor; a few sensors do not support this property, particularly those that output Boolean values such as Switches.

Minimum and Maximum should be included if there is some reason the application using the sensor at the API may need to know the range of the sensor. See further discussion of this in section 5.1.

A further discussion of per-datafield properties is left to Section 4.2.3; this section will focus on the ‘extended’ Properties not used as explicit examples in Section 4.3.

For reference, the set of sensor Properties supported by the driver is repeated below for convenience. An indication is shown as to recommended use:

// required in Feature report for Logo certification

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE 0x0A,0x09,0x03 // NAry

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE 0x0A,0x16,0x03 // Nary

#define HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE 0x0A,0x19,0x03 // Nary

#define HID\_USAGE\_SENSOR\_STATE 0x0A,0x01,0x02 // Nary

#define HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL 0x0A,0x0E,0x03

// required by most Sensors for Logo certification (only one of the two may be required, typically ABS)

// NOTE: this should take the bulk or specific modifier form – see section the discussion in this section

#define HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_ABS 0x0A,0x0F,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_REL\_PCT 0x0A,0x11,0x03

// optional for all Sensors

#define HID\_USAGE\_SENSOR\_PROPERTY\_RANGE\_MAXIMUM 0x0A,0x14,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_RANGE\_MINIMUM 0x0A,0x15,0x03

// optional for all Sensors

#define HID\_USAGE\_SENSOR\_PROPERTY\_ACCURACY 0x0A,0x12,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_RESOLUTION 0x0A,0x13,0x03

// optional for the AmbientLight Sensor

#define HID\_USAGE\_SENSOR\_PROPERTY\_RESPONSE\_CURVE 0x0A,0x18,0x03

// for use at (careful) discretion of implementer

#define HID\_USAGE\_SENSOR\_PROPERTY\_FRIENDLY\_NAME 0x0A,0x01,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_PERSISTENT\_UNIQUE\_ID 0x0A,0x02,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_MINIMUM\_REPORT\_INTERVAL 0x0A,0x04,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MANUFACTURER 0x0A,0x05,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MODEL 0x0A,0x06,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_SERIAL\_NUMBER 0x0A,0x07,0x03

#define HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_DESCRIPTION 0x0A,0x08,0x03

// required in Input report for Logo certification

#define HID\_USAGE\_SENSOR\_STATE 0x0A,0x01,0x02 // Nary

#define HID\_USAGE\_SENSOR\_EVENT 0x0A,0x02,0x02 // NAry

And remember that if you’re developing a custom sensor to be used on Windows 10, then you must include the following sensor properties shown in the preceding list:

* HID\_USAGE\_SENSOR\_PROPERTY\_FRIENDLY\_NAME
* HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MANUFACTURER
* HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MODEL

Though not strictly a property, the following Usage included in the above list is also used in a Feature Report as well as in an Input Report:

#define HID\_USAGE\_SENSOR\_STATE 0x0A,0x01,0x02 // Nary

Finally, the following Usage included in the above list is used in an Input report:

#define HID\_USAGE\_SENSOR\_EVENT 0x0A,0x02,0x02 // NAry

The following extracts from a hypothetical HID Report Descriptor, show how to represent each of these. Note that in the case of string descriptors such as FRIENDLY\_NAME and PERSISTENT\_UNIQUE\_ID, the report count should be large enough to contain the expected value (16-bits for each wide character, plus 16-bits for a wide NULL termination) but need be not larger (16 is used here for reference, long enough to hold a 15 wide-character string – the constraint on use of these string descriptors in a Feature report is that the overall length of the Feature report, including the Report ID, cannot exceed 64 bytes):

.

.

.

HID\_USAGE\_SENSOR\_PROPERTY\_FRIENDLY\_NAME,

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(16),

HID\_FEATURE(Data\_Arr\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_PERSISTENT\_UNIQUE\_ID,

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(16),

HID\_FEATURE(Data\_Arr\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_MINIMUM\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MANUFACTURER,

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(16),

HID\_FEATURE(Data\_Arr\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_MODEL,

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(16),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_SERIAL\_NUMBER,

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(16),

HID\_FEATURE(Data\_Arr\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_DESCRIPTION,

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(16),

HID\_FEATURE(Data\_Arr\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_ABS,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_REL\_PCT, //only used for the AmbientLight sensor

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_RESPONSE\_CURVE,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(10), //as required for n pair of values

HID\_UNIT\_EXPONENT(0x0), // scale default unit to provide 0 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

.

.

.

Though not defined as a property, the following is also used in a Feature report as a property as well as in an Input report.

.

.

.

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

.

.

.

Finally, two property-like construction are used in the Input report:

.

.

.

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

.

.

.

It is up to the device to expose these as required by the particular application in which the device is used.

Note that many of these are strings. In the case of string descriptors such as FRIENDLY\_NAME and PERSISTENT\_UNIQUE\_ID the report count should be large enough to contain the expected value (16-bits for each wide character, plus 16-bits for a wide NULL termination) but need be not larger (16 is used here for reference, long enough to hold a 15 wide-character string – the constraint on use of these string descriptors in a Feature report is that the overall length of the Feature report, including the Report ID, cannot exceed 64 bytes.)

### Modifiers: Per-datafield Properties

A number of *Properties* (transferred in Feature Reports) that can be applied to *Data Fields* (transferred in Input Reports) are on a per-datafield basis. This presents some options in how these per-datafield *Properties* can be expressed using the definitions in this document.

**Global datafield properties**

*Global* datafield properties assume there is only a single type of Data Field or that the Property applies to all Data Fields no matter what their type. Use of global datafield properties are not fully supported by the driver. Instead Feature Reports should use either *bulk* or *specific* per-datafield properties described later in this section.

**Bulk and Specific per-datafield properties**

Per-data field properties allow explicit construction for a specific Data Field or a bulk set of related Data Fields. These are expressed by means of a Data Field type and a Modifier Usage Switch.

The per-datafield properties expressed as Data Field Modifiers are defined elsewhere in the document. Those defined in this document are repeated below for convenience:

//data type usages modifiers

//NOTE: the usage tables actually define these as two bytes, but in order

//to get the define macros to work so these are ‘or-ed’ these are defined

//here as only one byte.

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_NONE 0x00 //US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS 0x10 //US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX 0x20 //US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN 0x30 //US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY 0x40 //US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION 0x50 //US

#define HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_REL\_PCT 0xE0 //US

Any of these Modifiers can be applied to any Data Field. Below is an example extracted from a HID Report Descriptor that again uses the single Data Field thermometer example. This use is referred to as a *specific* property:

.

.

.

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_PRECISION),

.

.

.

Provision has been made for this syntax to apply to those cases where there are multiple Data Fields defined for the sensor. In each case where multiple Data Fields are defined, a definition has been created that refers to all of them collectively. This use is referred to as a *bulk* property. Using again the accelerometer as an example, the collective and individual Data Field defintions are below:

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION 0x0A,0x52,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS 0x0A,0x53,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS 0x0A,0x54,0x04

#define HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS 0x0A,0x55,0x04

Applying the list of Properties to the *bulk* version of the Data Field would be done as follows:

.

.

. HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_PRECISION),

.

.

.

Note that in each case the Data Field to which the Modifier applies is specified, and that in each case the Data Field specified is for the *collective* version of the Data Field. This is mostly equivalent to the following definitions presented previously and repeated here for convenience:

.

.

.

HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_ABS,

HID\_USAGE\_SENSOR\_PROPERTY\_MAXIMUM,

HID\_USAGE\_SENSOR\_PROPERTY\_MINIMUM,

HID\_USAGE\_SENSOR\_PROPERTY\_ACCURACY,

HID\_USAGE\_SENSOR\_PROPERTY\_RESOLUTION,

.

.

.

In the case of the *collective* Data Field specification, this will only apply to Data Fields of that type. In the case of the HID\_USAGE\_SENSOR\_PROPERTY\_xxx construction, this would apply to all Data Fields even if they are not of the same type.

In the case of specifying Properties that are applied per-datafield with the expectation that the Property may change depending on the Data Field, the following *specific* construction again using the accelerometer follows:

.

.

.

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION),

.

.

.

Note that in each case the Data Field is the *specific* Data Field and not the *bulk* version, and that the Property modifier only applies to that Data Field. This specificity is the most desireable way to express per-datafield properties, though this comes at some cost to the device that must support these Report Descriptors.

A heterogenous example, from a hypothetical 1D accelerometer combined with a thermometer follows:

.

.

.

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_ACCURACY),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,HID\_USAGE\_SENSOR\_DATA\_MOD\_RESOLUTION),

.

.

.

So far, the discussion applies to required datafields as well as optional datafields. One final consideration must be made for dynamic datafields.

A dynamic datafield is a datafield that is supported in any Sensor aside from that Sensor in which that datafield is required or optional. When supporting a dynamic datafield, the *bulk* per-datafield property would not apply since by definition the dynamic datafield is being implemented in a heterogenous Sensor. Instead use the *specific* per-datafield constructin described in the previous paragraphs.

There is no requirement that per-datafield Properties be supported at all for any Data Field; it follows, too, that these can be mixed and matched to express Properties that are important to the device implementer. However, if Windows Hardware Certification compliance is expected it will almost always be necessary to support Change Sensitivity.

### Event Thresholds

Aside from Change Sensitivity, event thresholds are not supported by the driver. Change Sensitivity is interpreted by the driver as a Data Updated event.

### Sensor Collections

The way in which Sensor Collection is described requires further elaboration. The implementer should be careful to use Report Descriptors for their specific device based on the examples in this document rather than the sensor collection examples in the Specification.

Sensor Collections must always take the same form, illustrated by the following skeleton example using two sensors:

// Two sensor collection skeleton example:

const unsigned char col1\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_COLLECTION,

HID\_COLLECTION(Application),

HID\_REPORT\_ID(1),

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ihv1,

HID\_COLLECTION(Physical),

//Feature Report Descriptor for Sensor Report ID = 1

//Input Report Descriptor for Sensor Report ID = 1

HID\_END\_COLLECTION, //for Report ID = 1

HID\_REPORT\_ID(2),

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ihv2,

HID\_COLLECTION(Physical),

//Feature Report Descriptor for Sensor Report ID = 2

//Input Report Descriptor for Sensor Report ID = 2

HID\_END\_COLLECTION, //for Report ID = 2

//More sensors follow using the same pattern

HID\_END\_COLLECTION //Application

};

Note that in this example the Report ID begins at “1”, and this is required by the driver. The folowing report IDs increment by “1” for each additional sensor. The commented sections for Feature and Input report descriptors are identical to those for any other sensor.

This Sensor report descriptor could be followed, or preceeded by, report descriptors for other devices such as a Mouse or a Keyboard. The following example illustrates this.

const unsigned char col2\_report\_descriptor[] = {

//keyboard

0x05U, 0x01U, // USAGE PAGE (Generic Desktop)

0x09U, 0x06U, // USAGE (Keyboard)

0xA1U, 0x01U, // COLLECTION (Application)

0x85U, 0x03U, // REPORT\_ID (1)

0x15U, 0x00U, // LOGICAL MINIMUM (0)

0x25U, 0x01U, // LOGICAL MAXIMUM (1)

0x75U, 0x01U, // REPORT SIZE (1)

0x95U, 0x08U, // REPORT COUNT (8)

0x05U, 0x07U, // USAGE PAGE (Keyboard)

0x19U, 0xE0U, // USAGE MINIMUM (Keyboard LeftControl)

0x29U, 0xE7U, // USAGE MAXIMUM (Keyboard Right GUI)

0x81U, 0x02U, // INPUT (Var)

0x75U, 0x08U, // REPORT SIZE (8)

0x95U, 0x0AU, // REPORT COUNT (10)

0x19U, 0x00U, // USAGE MINIMUM (No event)

0x29U, 0x91U, // USAGE MAXIMUM (Keyboard LANG2)

0x26U, 0xFFU, 0x00U, // LOGICAL MAXIMUM (0xFF)

0x81U, 0x00U, // INPUT (Data,Ary,Abs)

0xC0U, // END COLLECTION, //keyboard

// Two sensor collection skeleton example:

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_COLLECTION,

HID\_COLLECTION(Application),

HID\_REPORT\_ID(2),

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ihv1,

HID\_COLLECTION(Physical),

//Feature Report Descriptor for Sensor Report ID = 2

//Input Report Descriptor for Sensor Report ID = 2

HID\_END\_COLLECTION, //for Report ID = 2

HID\_REPORT\_ID(3),

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ihv2,

HID\_COLLECTION(Physical),

//Feature Report Descriptor for Sensor Report ID = 3

//Input Report Descriptor for Sensor Report ID = 3

HID\_END\_COLLECTION, //for Report ID = 3

//More sensors follow using the same pattern

HID\_END\_COLLECTION //Sensor collection

//mouse

0x05U, 0x01U, // USAGE PAGE (Generic Desktop)

0x09U, 0x02U, // USAGE (Mouse)

0xA1U, 0x01U, // COLLECTION (Application)

0x85U, 0x04U, // REPORT\_ID (4)

// 5 mouse buttons

0x05U, 0x09U, // USAGE PAGE (Button)

0x19U, 0x01U, // USAGE MINIMUM (Button 1)

0x29U, 0x05U, // USAGE MAXIMUM (Button 5)

0x15U, 0x00U, // LOGICAL MINIMUM (0)

0x25U, 0x01U, // LOGICAL MAXIMUM (1)

0x95U, 0x05U, // REPORT COUNT (5)

0x75U, 0x01U, // REPORT SIZE (1)

0x81U, 0x02U, // INPUT (Data,Var,Abs)

// 3 unused buttons:

0x95U, 0x01U, // REPORT COUNT (1)

0x75U, 0x03U, // REPORT SIZE (3)

0x81U, 0x03U, // INPUT (Cnst,Var,Abs)

// mouse (delta x, delta y) position

0x15U, 0x81U, // LOGICAL MINIMUM (-127)

0x25U, 0x7fU, // LOGICAL MAXIMUM (+127)

0x75U, 0x08U, // REPORT SIZE (8)

0x95U, 0x02U, // REPORT COUNT (2)

0x05U, 0x01U, // USAGE PAGE (Generic Desktop)

0x09U, 0x30U, // USAGE(X)

0x09U, 0x31U, // USAGE (Y)

0x81U, 0x06U, // INPUT (Data,Var,Rel)

0xC0U, // END COLLECTION //mouse

};

//end of report descriptor

Note that in this example there are three physcial devices: one keyboard, one sensor collection and one mouse. The sensor collection contains two sensors. If there is only a single sensor, it still must be enclosed within a sensor collection as shown in the following example.

const unsigned char col3\_report\_descriptor[] = {

//keyboard

0x05U, 0x01U, // USAGE PAGE (Generic Desktop)

0x09U, 0x06U, // USAGE (Keyboard)

0xA1U, 0x01U, // COLLECTION (Application)

0x85U, 0x03U, // REPORT\_ID (1)

0x15U, 0x00U, // LOGICAL MINIMUM (0)

0x25U, 0x01U, // LOGICAL MAXIMUM (1)

0x75U, 0x01U, // REPORT SIZE (1)

0x95U, 0x08U, // REPORT COUNT (8)

0x05U, 0x07U, // USAGE PAGE (Keyboard)

0x19U, 0xE0U, // USAGE MINIMUM (Keyboard LeftControl)

0x29U, 0xE7U, // USAGE MAXIMUM (Keyboard Right GUI)

0x81U, 0x02U, // INPUT (Var)

0x75U, 0x08U, // REPORT SIZE (8)

0x95U, 0x0AU, // REPORT COUNT (10)

0x19U, 0x00U, // USAGE MINIMUM (No event)

0x29U, 0x91U, // USAGE MAXIMUM (Keyboard LANG2)

0x26U, 0xFFU, 0x00U, // LOGICAL MAXIMUM (0xFF)

0x81U, 0x00U, // INPUT (Data,Ary,Abs)

0xC0U, // END COLLECTION, //keyboard

// One sensor collection skeleton example:

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_COLLECTION,

HID\_COLLECTION(Application),

HID\_REPORT\_ID(2),

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ihv2,

HID\_COLLECTION(Physical),

//Feature Report Descriptor for Sensor Report ID = 2

//Input Report Descriptor for Sensor Report ID = 2

HID\_END\_COLLECTION, //for Report ID = 2

HID\_END\_COLLECTION //Sensor collection

//mouse

0x05U, 0x01U, // USAGE PAGE (Generic Desktop)

0x09U, 0x02U, // USAGE (Mouse)

0xA1U, 0x01U, // COLLECTION (Application)

0x85U, 0x04U, // REPORT\_ID (3)

// 5 mouse buttons

0x05U, 0x09U, // USAGE PAGE (Button)

0x19U, 0x01U, // USAGE MINIMUM (Button 1)

0x29U, 0x05U, // USAGE MAXIMUM (Button 5)

0x15U, 0x00U, // LOGICAL MINIMUM (0)

0x25U, 0x01U, // LOGICAL MAXIMUM (1)

0x95U, 0x05U, // REPORT COUNT (5)

0x75U, 0x01U, // REPORT SIZE (1)

0x81U, 0x02U, // INPUT (Data,Var,Abs)

// 3 unused buttons:

0x95U, 0x01U, // REPORT COUNT (1)

0x75U, 0x03U, // REPORT SIZE (3)

0x81U, 0x03U, // INPUT (Cnst,Var,Abs)

// mouse (delta x, delta y) position

0x15U, 0x81U, // LOGICAL MINIMUM (-127)

0x25U, 0x7fU, // LOGICAL MAXIMUM (+127)

0x75U, 0x08U, // REPORT SIZE (8)

0x95U, 0x02U, // REPORT COUNT (2)

0x05U, 0x01U, // USAGE PAGE (Generic Desktop)

0x09U, 0x30U, // USAGE(X)

0x09U, 0x31U, // USAGE (Y)

0x81U, 0x06U, // INPUT (Data,Var,Rel)

0xC0U, // END COLLECTION //mouse

};

//end of report descriptor

Note that in this example they mouse report ID has been changed to “3” to reflect the fact there is only a single sensor in the sensor collection.

### Custom Sensor

Some of the custom sensor datafields, as well as the Feature report, are optional. Following are several examples. Do not omit the Feature report, nor the required fields in the Feature report, if the device is expected to be Windows Hardware Certification-compliant.

The first example is a minimally-functional custom sensor Report Descriptor that includes the only required datafield. Be advised this example is not Windows Hardware Certification-compliant.

const unsigned char cus1\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR, // USAGE\_PAGE (Sensor)

HID\_USAGE\_SENSOR\_TYPE\_OTHER\_CUSTOM, // USAGE (Simple Custom)

HID\_COLLECTION(Physical),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_1,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_END\_COLLECTION

};

The second example is a minimally-functional customer sensor Report Descriptor that includes the optional Custom Sensor datafields. Be advised this example is not Windows Hardware Certification-compliant.

const unsigned char cus2\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR, // USAGE\_PAGE (Sensor)

HID\_USAGE\_SENSOR\_TYPE\_OTHER\_CUSTOM, // USAGE (Simple Custom)

HID\_COLLECTION(Physical),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_TYPE\_MOTION\_SPEEDOMETER

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_BOOLEAN\_ARRAY,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_TYPE\_MOTION\_SPEEDOMETER

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_1,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_2,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_3,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_4,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_5,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_6,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_END\_COLLECTION

};

Following is a Custom Sensor Report Descriptor that illustrates these concepts. If we apply this example, the encapsulated fields would be populated as follows for a Speedometer sensor (HID Usage = HID\_USAGE\_SENSOR\_TYPE\_MOTION\_SPEEDOMETER).

const unsigned char cus3\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR, // USAGE\_PAGE (Sensor)

HID\_USAGE\_SENSOR\_TYPE\_OTHER\_CUSTOM, // USAGE (Simple Custom)

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_CUSTOM,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_CUSTOM,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_CUSTOM,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_TYPE\_MOTION\_SPEEDOMETER

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_1,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs), // = HID\_USAGE\_SENSOR\_DATA\_MOTION\_SPEED value

HID\_END\_COLLECTION

};

Following is a complete report descriptor that illustrates fields not used in the above example. Aside from choosing how many optional datafields to choose for a specific application, this is the recommended Report Descriptor for a Custom sensor:

// Complete HID report descriptor

const unsigned char cus4\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR, // USAGE\_PAGE (Sensor)

HID\_USAGE\_SENSOR\_TYPE\_OTHER\_CUSTOM, // USAGE (Simple Custom)

HID\_COLLECTION(Application),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_CUSTOM,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_CUSTOM,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_CUSTOM,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_BOOLEAN\_ARRAY,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_1,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_2,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_3,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_4,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_5,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_6,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

#if 1 //define vendor-specific (non-spec) custom datafields

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_7,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_8,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_9,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_10,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_11,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_12,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_13,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_14,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_15,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_16,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_17,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_18,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_19,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_20,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_21,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_22,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_23,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_24,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_25,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_26,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_27,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_VALUE\_28,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale unit to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

#endif

HID\_END\_COLLECTION

};

typedef struct \_CUS\_FEATURE\_REPORT

{

Unsigned char ucReportId;

unsigned char ucConnectionType;

unsigned char ucReportingState;

unsigned char ucPowerState;

unsigned char ucSensorState;

unsigned long ulReportInterval;

unsigned short usCustomSensitivity;

} CUS\_FEATURE\_REPORT, \*PCUS\_FEATURE\_REPORT;

typedef struct \_CUS\_INPUT\_REPORT

{

Unsigned char ucReportId;

unsigned char ucSensorState;

unsigned char ucEventType;

unsigned short usUsage;

short sValueArray[28]; //1 //6 max //28

} CUS\_INPUT\_REPORT, \*PCUS\_INPUT\_REPORT;

Additional discussion of report descriptor variations to be provided: fully described per-datafield properties, using all properties.

## Illustrative Sensor Report Descriptors

As noted in the Custom Sensor discussion (section 4.2.6) a sensor does not require a Feature Report or the two additional fields in the Input Report in order to work properly with the driver. That concept can be extended to all other supported drivers, so for the sake of brevity those minimal Report Descriptor constructions are not shown in the following examples. Also, the extended properties beyond those recommended in section 4.2.6 are not shown in the following examples though these constructs can be used if desired.

For the most part, the following examples are the recommended Report Descriptors for these Sensor types. These will work with the driver as well as, when properly implemented, be Windows Hardware Certification compliant.

These descriptions should be read carefully. Certain conditions apply to certain Sensors and not to others. The implementer should always comply with the notes for the Sensors to be implemented.

Structure typedefs that match the report descriptors for each sensor have been included in the descriptions. These include both the Feature report and the Input report. The following typedefs are used to define these structures:

typedef unsigned char HID\_UCHAR; //8-bits

typedef char HID\_CHAR; //8-bits

typedef unsigned short HID\_USHORT; //16-bits

typedef short HID\_SHORT; //16-bits

typedef unsigned long HID\_ULONG; //32-bits

typedef long HID\_LONG; //32-bits

These are intended to be compiled with an ANSI C compiler. The implementer is advised to treat those files as the source for the report descriptors in the following sections.

### Biometric: Human Presence

// Complete HID report descriptor

//Human Presence

const unsigned char pres\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_BIOMETRIC\_PRESENCE,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PRESENCE,

HID\_LOGICAL\_MIN\_8(0), // False

HID\_LOGICAL\_MAX\_8(1), // True

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_PRES\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

} PRES\_FEATURE\_REPORT, \*PPRES\_FEATURE\_REPORT;

typedef struct \_PRES\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_UCHAR ucPresenceState;

} PRES\_INPUT\_REPORT, \*PPRES\_INPUT\_REPORT;

### Biometric: Human Proximity

// For reference: Complete HID report descriptor

const unsigned char prox\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_BIOMETRIC\_PROXIMITY,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_RANGE,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0D), // scale default unit “meter” to “centimeter” to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_RANGE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0D), // scale default unit “meter” to “centimeter” to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_RANGE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0D), // scale default unit “meter” to “centimeter” to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_OUT\_OF\_RANGE,

HID\_LOGICAL\_MIN\_8(0), // False

HID\_LOGICAL\_MAX\_8(1), // True

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_BIOMETRIC\_HUMAN\_PROXIMITY\_RANGE,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0D), // scale default unit “meter” to “centimeter” to provide 2 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_PROX\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usProximityChangeSensitivity;

HID\_SHORT sProximityMaximum;

HID\_SHORT sProximityMinimum;

} PROX\_FEATURE\_REPORT, \*PPROX\_FEATURE\_REPORT;

typedef struct \_PROX\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_UCHAR ucOutOfRange;

HID\_SHORT sProximityValue;

} PROX\_INPUT\_REPORT, \*PPROX\_INPUT\_REPORT;

### Environmental: Atmospheric Pressure

// For reference: Complete HID report descriptor

const unsigned char bar\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Bar” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Bar” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,

HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Bar” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_ATMOSPHERIC\_PRESSURE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Bar” to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_BAR\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usBarChangeSensitivity;

HID\_USHORT usBarMaximum;

HID\_USHORT usBarMinimum;

} BAR\_FEATURE\_REPORT, \*PBAR\_FEATURE\_REPORT;

typedef struct \_BAR\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_USHORT usBarValue;

} BAR\_INPUT\_REPORT, \*PBAR\_INPUT\_REPORT;

### Environmental: Humidity

// For reference: Complete HID report descriptor

const unsigned char hyg\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_HUMIDITY,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_RELATIVE\_HUMIDITY,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0x10,0x27), // 10000 = 0.00 to 100.00 percent with 2 digits past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_RELATIVE\_HUMIDITY,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0x10,0x27), // 10000 = 0.00 to 100.00 percent with 2 digits past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_RELATIVE\_HUMIDITY,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0x10,0x27), // 10000 = 0.00 to 100.00 percent with 2 digits past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_RELATIVE\_HUMIDITY,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0x10,0x27), // 10000 = 0.00 to 100.00 percent with 2 digits past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “percent” to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_HYG\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usHygChangeSensitivity;

HID\_USHORT usHygMaximum;

HID\_USHORT usHygMinimum;

} HYG\_FEATURE\_REPORT, \*PHYG\_FEATURE\_REPORT;

typedef struct \_HYG\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_USHORT usHygValue;

} HYG\_INPUT\_REPORT, \*PHYG\_INPUT\_REPORT;

### Environmental: Temperature

// For reference: Complete HID report descriptor

const unsigned char temp\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ENVIRONMENTAL\_TEMPERATURE,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Celsius” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Celsius” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Celsius” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(16),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_ENVIRONMENTAL\_TEMPERATURE,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “Celsius” to provide 2 digits past the decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_TEMP\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usTempChangeSensitivity;

HID\_SHORT sTempMaximum;

HID\_SHORT sTempMinimum;

} TEMP\_FEATURE\_REPORT, \*PTEMP\_FEATURE\_REPORT;

typedef struct \_TEMP\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_SHORT sTempValue;

} TEMP\_INPUT\_REPORT, \*PTEMP\_INPUT\_REPORT;

### Light: Ambient Light

// For reference: Complete HID report descriptor

const unsigned char als\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR, // USAGE\_PAGE (Sensor)

HID\_USAGE\_SENSOR\_TYPE\_LIGHT\_AMBIENTLIGHT, // USAGE (AmbientLight)

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_LIGHT\_ILLUMINANCE,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_REL\_PCT),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0x10,0x27), // 10000 = 0.00 to 100.00 percent with 2 digits past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_LIGHT\_ILLUMINANCE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_LIGHT\_ILLUMINANCE,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_FEATURE(Data\_Var\_Abs),

//add this definition if required by the specific application

HID\_USAGE\_SENSOR\_PROPERTY\_RESPONSE\_CURVE,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(10), //as required for n pair of values

HID\_UNIT\_EXPONENT(0x0), // scale default unit to provide 0 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_LIGHT\_ILLUMINANCE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_LIGHT\_COLOR\_TEMPERATURE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_UNIT\_EXPONENT(0),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY\_X,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_UNIT\_EXPONENT(0x0C), // scale default unit to provide 4 digits past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_LIGHT\_CHROMATICITY\_Y,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_UNIT\_EXPONENT(0x0C), // scale default unit to provide 4 digits past decimal point

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_ALS\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usIlluminanceChangeSensitivity;

HID\_USHORT usIlluminanceMaximum;

HID\_USHORT usIlluminanceMinimum;

//add this definition if required by the specific application

HID\_USHORT usResponseCurve[5][2]; //10 elements matches descriptor

} ALS\_FEATURE\_REPORT, \*PALS\_FEATURE\_REPORT;

typedef struct \_ALS\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_USHORT usIlluminanceValue;

HID\_USHORT usColorTempValue;

HID\_USHORT usChromaticityXValue;

HID\_USHORT usChromaticityYValue;

} ALS\_INPUT\_REPORT, \*PALS\_INPUT\_REPORT;

### Location: GPS

Location category sensors are not supported by the driver.

### Motion: Accelerometer

// For reference: Complete HID report descriptor

// 3D Accelerometer

const unsigned char accel3\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_MOTION\_ACCELEROMETER\_3D,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit Gs to “centi-Gs” to provide 2 digits past Gs decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit Gs to “centi-Gs” to provide 2 digits past Gs decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit Gs to “centi-Gs” to provide 2 digits past Gs decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_X\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit Gs to “centi-Gs” to provide 2 digits past Gs decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Y\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit Gs to “centi-Gs” to provide 2 digits past Gs decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_MOTION\_ACCELERATION\_Z\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit Gs to “centi-Gs” to provide 2 digits past Gs decimal point

HID\_INPUT(Data\_Var\_Abs),

//include the following datafield if required to support the “shake” event

HID\_USAGE\_SENSOR\_DATA\_MOTION\_STATE,

HID\_LOGICAL\_MIN\_8(0), // False = Still

HID\_LOGICAL\_MAX\_8(1), // True = In Motion

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_ACCEL3\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usAccelChangeSensitivity;

HID\_SHORT sAccelMaximum;

HID\_SHORT sAccelMinimum;

} ACCEL3\_FEATURE\_REPORT, \*PACCEL3\_FEATURE\_REPORT;

typedef struct \_ACCEL3\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_SHORT sAccelXValue;

HID\_SHORT sAccelYValue;

HID\_SHORT sAccelZValue;

//include if required to support the "shake" event

HID\_UCHAR ucShakeDetectState;

} ACCEL3\_INPUT\_REPORT, \*PACCEL3\_INPUT\_REPORT;

### Motion: Gyrometer

// For reference: Complete HID report descriptor

// 3D Gyrometer

const unsigned char gyro3\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_MOTION\_GYROMETER\_3D,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_X\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_Y\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_MOTION\_ANGULAR\_VELOCITY\_Z\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_END\_COLLECTION

};

typedef struct \_GYRO3\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usGyroChangeSensitivity;

HID\_SHORT sGyroMaximum;

HID\_SHORT sGyroMinimum;

} GYRO3\_FEATURE\_REPORT, \*PGYRO3\_FEATURE\_REPORT;

typedef struct \_GYRO3\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_SHORT sGyroXValue;

HID\_SHORT sGyroYValue;

HID\_SHORT sGyroZValue;

} GYRO3\_INPUT\_REPORT, \*PGYRO3\_INPUT\_REPORT;

### Orientation: Compass

// For reference: Complete HID report descriptor

// 3D Compass, a 3-axis flux magnetometer

const unsigned char comp3\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_COMPASS\_3D,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_HEADING,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_COMPENSATED\_MAGNETIC\_NORTH,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_COMPENSATED\_TRUE\_NORTH,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_NORTH,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TRUE\_NORTH,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0F), // scale default unit to provide 1 digit past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_X\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0D), // scale default unit to “milliGauss”; provide 3 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_Y\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0D), // scale default unit to “milliGauss”; provide 3 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETIC\_FLUX\_Z\_AXIS,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0D), // scale default unit to “milliGauss”; provide 3 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_LOW,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_MEDIUM,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_HIGH,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_END\_COLLECTION

};

typedef struct \_COMP3\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usHeadingChangeSensitivity;

HID\_SHORT sHeadingMaximum;

HID\_SHORT sHeadingMinimum;

HID\_USHORT usFluxChangeSensitivity;

HID\_SHORT sFluxMaximum;

HID\_SHORT sFluxMinimum;

} COMP3\_FEATURE\_REPORT, \*PCOMP3\_FEATURE\_REPORT;

typedef struct \_COMP3\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_SHORT sHeadingCompensatedMagneticNorthValue;

HID\_SHORT sHeadingCompensatedTrueNorthValue;

HID\_SHORT sHeadingMagneticNorthValue;

HID\_SHORT sHeadingTrueNorthValue;

HID\_SHORT sFluxXValue;

HID\_SHORT sFluxYValue;

HID\_SHORT sFluxZValue;

HID\_UCHAR ucMagnetometerAccuracy;

} COMP3\_INPUT\_REPORT, \*PCOMP3\_INPUT\_REPORT;

### Orientation: Inclinometer

// For reference: Complete HID report descriptor

// 3D Inclinometer

const unsigned char inc3\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_INCLINOMETER\_3D,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT,HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_X,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_Y,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_TILT\_Z,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit to provide 2 digits past decimal point

HID\_INPUT(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_LOW,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_MEDIUM,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_HIGH,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_END\_COLLECTION

};

typedef struct \_INC3\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usIncChangeSensitivity;

HID\_SHORT sIncMaximum;

HID\_SHORT sIncMinimum;

} INC3\_FEATURE\_REPORT, \*PINC3\_FEATURE\_REPORT;

typedef struct \_INC3\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_SHORT sIncXValue;

HID\_SHORT sIncYValue;

HID\_SHORT sIncZValue;

HID\_UCHAR ucMagnetometerAccuracy;

} INC3\_INPUT\_REPORT, \*PINC3\_INPUT\_REPORT;

### Orientation: Device Orientation

// For reference: Complete HID report descriptor

// Device Orientation sensor

const unsigned char devor\_report\_descriptor[] = {

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_TYPE\_ORIENTATION\_DEVICE\_ORIENTATION,

HID\_COLLECTION(Physical),

//feature reports (xmit/receive)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_PROPERTY\_SENSOR\_CONNECTION\_TYPE, // NAry

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_INTEGRATED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_ATTACHED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_CONNECTION\_TYPE\_PC\_EXTERNAL\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_SEL, HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_NO\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_ALL\_EVENTS\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORTING\_STATE\_THRESHOLD\_EVENTS\_WAKE\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_UNDEFINED\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D0\_FULL\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D1\_LOW\_POWER\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D2\_STANDBY\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D3\_SLEEP\_WITH\_WAKE\_SEL,

HID\_USAGE\_SENSOR\_PROPERTY\_POWER\_STATE\_D4\_POWER\_OFF\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_FEATURE(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_PROPERTY\_REPORT\_INTERVAL,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_32(0xFF,0xFF,0xFF,0xFF),

HID\_REPORT\_SIZE(32),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_PROPERTY\_CHANGE\_SENSITIVITY\_ABS,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E), // scale default unit “meter” to provide 2 digits past the decimal point

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_QUATERNION,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_QUATERNION,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x01),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_QUATERNION,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x01),

HID\_FEATURE(Data\_Var\_Abs),

//include this if the values are calculated in firmware

//otherwise, the driver will calculate these values from the Quaternion

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_ROTATION\_MATRIX,

HID\_USAGE\_SENSOR\_DATA\_MOD\_CHANGE\_SENSITIVITY\_ABS),

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_16(0xFF,0xFF),

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_ROTATION\_MATRIX,HID\_USAGE\_SENSOR\_DATA\_MOD\_MAX),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E),

HID\_FEATURE(Data\_Var\_Abs),

HID\_USAGE\_SENSOR\_DATA(HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_ROTATION\_MATRIX,HID\_USAGE\_SENSOR\_DATA\_MOD\_MIN),

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(1),

HID\_UNIT\_EXPONENT(0x0E),

HID\_FEATURE(Data\_Var\_Abs),

//input reports (transmit)

HID\_USAGE\_PAGE\_SENSOR,

HID\_USAGE\_SENSOR\_STATE,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(6),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_STATE\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_STATE\_READY\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NOT\_AVAILABLE\_SEL,

HID\_USAGE\_SENSOR\_STATE\_NO\_DATA\_SEL,

HID\_USAGE\_SENSOR\_STATE\_INITIALIZING\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ACCESS\_DENIED\_SEL,

HID\_USAGE\_SENSOR\_STATE\_ERROR\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_EVENT,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(5),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_EVENT\_UNKNOWN\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_STATE\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_PROPERTY\_CHANGED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_DATA\_UPDATED\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_POLL\_RESPONSE\_SEL,

HID\_USAGE\_SENSOR\_EVENT\_CHANGE\_SENSITIVITY\_SEL,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_QUATERNION,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(4),

HID\_UNIT\_EXPONENT(0x0E),

HID\_INPUT(Data\_Arr\_Abs),

//include this if the values are calculated in firmware

//otherwise, the driver will calculate these values from the Quaternion

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_ROTATION\_MATRIX,

HID\_LOGICAL\_MIN\_16(0x01,0x80), // LOGICAL\_MINIMUM (-32767)

HID\_LOGICAL\_MAX\_16(0xFF,0x7F), // LOGICAL\_MAXIMUM (32767)

HID\_REPORT\_SIZE(16),

HID\_REPORT\_COUNT(9),

HID\_UNIT\_EXPONENT(0x0F),

HID\_INPUT(Data\_Arr\_Abs),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY,

HID\_LOGICAL\_MIN\_8(0),

HID\_LOGICAL\_MAX\_8(2),

HID\_REPORT\_SIZE(8),

HID\_REPORT\_COUNT(1),

HID\_COLLECTION(Logical),

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_LOW,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_MEDIUM,

HID\_USAGE\_SENSOR\_DATA\_ORIENTATION\_MAGNETOMETER\_ACCURACY\_HIGH,

HID\_INPUT(Data\_Arr\_Abs),

HID\_END\_COLLECTION,

HID\_END\_COLLECTION

};

typedef struct \_DEVOR\_FEATURE\_REPORT

{

//common properties

HID\_UCHAR ucReportId;

HID\_UCHAR ucConnectionType;

HID\_UCHAR ucReportingState;

HID\_UCHAR ucPowerState;

HID\_UCHAR ucSensorState;

HID\_ULONG ulReportInterval;

//properties specific to this sensor

HID\_USHORT usQuaternionChangeSensitivity;

HID\_SHORT sQuaternionMaximum;

HID\_SHORT sQuaternionMinimum;

//include this if the values are calculated in firmware

//otherwise, the driver will calculate these values from the Quaternion

HID\_USHORT usRotationChangeSensitivity;

HID\_SHORT sRotationMaximum;

HID\_SHORT sRotationMinimum;

} DEVOR\_FEATURE\_REPORT, \*PDEVOR\_FEATURE\_REPORT;

typedef struct \_DEVOR\_INPUT\_REPORT

{

//common values

HID\_UCHAR ucReportId;

HID\_UCHAR ucSensorState;

HID\_UCHAR ucEventType;

//values specific to this sensor

HID\_SHORT sQuaternionXValue;

HID\_SHORT sQuaternionYValue;

HID\_SHORT sQuaternionZValue;

HID\_SHORT sQuaternionWValue;

//include this if the values are calculated in firmware

//otherwise, the driver will calculate these values from the Quaternion

HID\_SHORT sRotationValue[3][3];

HID\_UCHAR ucMagnetometerAccuracy;

} DEVOR\_INPUT\_REPORT, \*PDEVOR\_INPUT\_REPORT;

# Sensor Implementation and Debugging – Tips & Tricks

This section provides additional information, supplemental to the Specification, that may be of aid to the IHV in creating a Sensor that will work correctly with the driver.

## Using required and optional datafields

The driver supports three kinds of datafields:

1. Required
2. Optional
3. Dynamic

The required datafields are those that must be available in the input report descriptor for the sensor to work correctly. For example, a 3D Accelerometer must include specifications for the X, Y and Z acceleration or it will not work correctly with the sensor platform: specifically, any value that is not provided by the sensor will remain of type VT\_EMPTY at the API, and this is interpreted by clients of the API as that datafield not being available. Most of the datafields defined in the sensors supported by driver are required.

Certain datafields are optional, and these work differently than the required datafields. These datafields are not actually available at the API until an input report bearing the data for an optional datafield defined in the input report descriptor is actually received by the driver. When such an input report is received by the driver, that datafield is added to the list of datafields supported by the driver for that sensor and is then available at the API.

The optional datafields that are supported by the driver are the following:

1. AmbientLight – color temperature and chromaticity X and Y
2. Compass – all other than compensated magnetic north, which is required
3. Orientation – rotation matrix.
4. Custom – all datafields are considered optional. If there are no datafields defined in the input report descriptor, or no input report bearing values for the optional datafields has been received by the driver, the only datafield available at the API will be for the timestamp.

Dynamic datafields are described in the section 5.2.

### Out-of-range datafield values

In almost all cases, the value of a datafield is either the value provided by the sensor device or the value of VT\_EMPTY in the cases noted above. However there is one case in which a value may be provided by the sensor device and the value at the API will be VT\_NULL.

This case will occur when the sensor is in SENSOR\_STATE\_READY with at least one valid datafield value but any of the other datafield values are out of the range specified by Range Maximum and Range Minimum. The defaults for Range Maximum and Range Minimum are FLT\_MAX and –FLT\_MAX for all datafields, respectively, so unless these default per-datafield properties have been specified by the sensor device there is little possibility of exceeding these values.

This behavior is applicable to all datafields in all sensors with the exception of those datafields that do not support Range Maximum and Range Minimum as part of the defaults for that sensor (typically the sensor datafields that are of type BOOLEAN.)

### Controlling datafield values

Consider the case where a sensor device has some value it can provide only under certain circumstances. For example, a Compass 3D sensor that supports the True North datafield which is derived from a GPS device that provides the True North heading will only have a valid value for this when the GPS has a fix and is providing valid data. The sensor device will need some way to indicate to the driver that, while the other compass datafields may be valid, the True North heading is not available.

This case is handled in the following way:

* The IHV will specify the Range Maximum and Range Minimum values in the Report Descriptor for the sensor in question. This must be specified as a *specific* datafield property unless the same range is used for all datafields that may be subject to the above behavior, in which case the *bulk* property may be used. See section 4.2.3 for more information on how to specify these per-datafield properties.
* In the case above, should the GPS be unavailable to provide the True North Heading value, the compass device would insert a value that is outside the range defined by Range Maximum and Range Minimum.
* The value available at the API for the True North Heading datafield will then be set to VT\_NULL by the driver. This is an indication to the application that the sensor is working correctly but that this specific datafield is not available.
* In the case above, when the GPS again becomes available to provide the True North Heading value, the compass device would then insert a valid in-range value and that value would again appear at the API.

## Using dynamic datafields

In general, any datafield defined for any sensor can be defined in the input report descriptor for any other sensor. So, for example, an Accelerometer could include the definition for a Custom\_1 datafield. Such datafields are considered dynamic, and there is a small performance penalty for using them as the driver must search upon the receipt of every input packet whether the dynamic datafield present in the packet has previously been recognized, if not, and added to the list of datafields supported by the driver for that sensor.

Like the optional datafield, these dynamic datafields are not actually available at the API until an input report bearing the data for a dynamic datafield defined in the input report descriptor is actually received by the driver. When such an input report is received by the driver, that datafield is added to the list of datafields supported by the driver for that sensor and is then available at the API.

## Using dynamic properties

Most of the properties defined for a sensor are static, which is to say they are defined at compile time for that sensor. Default values are chosen for these properties unless that value is over-ridden by a value received in a feature report from the sensor device.

One exception to this is the use of SENSOR\_PROPERTY\_HID\_USAGE. This property is only available when HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE is defined in an input report (yes, input report.) The presence of this datafield in an input report for any sensor will cause SENSOR\_PROPERTY\_HID\_USAGE to appear as a property for that sensor at the API. The value for this usage is vendor-specific.

This mechanism is typically used as a means by which similar sensors can be distinguished from one another if more than one sensor of a particular type is available on a sensor device.

Similar to the optional and dynamic datafields, this dynamic property is not actually available at the API until an input report bearing the data for the datafield defined in the input report descriptor is actually received by the driver.

This mechanism can be a bit confusing. The specification defines HID\_USAGE\_SENSOR\_DATA\_CUSTOM\_USAGE as a datafield, but if used only in this manner the ability to distinguish one sensor from another at the API would be dependent upon the client receiving data from a sensor and deciding on the basis of the datafield value for each of several similar sensors which sensor is of interest. By causing this datafield to show up at the API as a property, the ability to distinguish one sensor from another can be done by querying the supported properties for each sensor.

## Using Custom sensors and datafields

Custom sensors can be very useful in at least two circumstances:

1. An IHV creates a sensor that is not directly supported by the driver
2. An IHV needs to get data from individual sensors, or from the device as a whole, and make this data available to clients of the API.

The first case was the reason the Custom sensor was originally specified. Essentially, any sensor can be supported through a custom sensor. The purpose of that sensor is obscured (it is a “Custom” sensor to clients of the API, no matter what kind of sensor technology is being supported), and it is up to the IHV to make available, the information that un-obscures that sensor for clients of the API.

In some circumstances, for example the second case, the IHV may not wish to make the un-obscuring information available. For example, the IHV may wish to create custom sensors that return certain information from the device. There are two ways to return this information:

1. By means of a custom sensor
2. By means of custom datafields used with other supported sensors

For example, should the IHV wish to make raw data, timestamp or calibration information available about an accelerometer, it might be best to use method 2. If instead the IHV wishes to return information common to all sensors, it might be best to use method 1.

## Setting device properties

The Sensor Platform does not directly support anything other than being able to write the defined writeable properties. There are two of these:

1. Current Report Interval
2. Change Sensitivity for each datafield

Aside from their intended purpose, there are other ways in which these two writeable properties can be used to control the sensor device. Some possible ways are:

1. To put the device into a particular mode (firmware update, calibration, …)
2. To write specific data to the device

The means by which this can be accomplished is to use specific values of Current Report Interval that are recognized by the firmware. Current Report Interval is a 32-bit unsigned value and specifies in milliseconds the desired report interval of a sensor. It is unlikely that clients at the API will specify very large values for the report interval, so these high values could be used to trigger actions on the device. For example, a value of 0xFFFFFFFE could represent calibration mode, and a value of 0xFFFFFFFD could represent firmware update mode. For greater security against accidentally putting the device into an unwanted mode, a sequence of report interval values could be issued instead of a single value.

It is a bit more complex to write data to the device. Per-datafield change sensitivity would need to be specified in the feature report descriptor for any data to be written. The client could then issue a Current Report Interval sequence that would take the data values carried in the change sensitivity values and use them as data rather than as change sensitivities for particular datafields. This could be best accomplished by using change sensitivities specified for custom datafields for that sensor.

## Debugging a sensor or sensor collection

Suggested steps for bringing up a new sensor implentation:

1. Read this document fully – there are quite a number of details the implementor must take into account when designing a Sensor device implementation.
2. Begin with one of the sample report descriptors in this document, or better yet, use one of the report descriptors in “hid\_sensor\_spec\_report\_descriptors.h.” These have all been tested against the driver and are known to work. Use the machine-readable headers in “hid\_sensor\_spec\_macros.h” provided along with the document.
3. Start with only a single sensor – do not start with a collection of sensors. Once a single sensor is working, it can then be convereted to a collection of one or more working sensors.
4. Comment out the Feature Report section of the report descriptor – the driver will accept data from a sensor that only provides an Input report. Start the Sensor Diagnostic Tool (SDT) and expand the “Sensors” tree view to observe that the sensor is present in the left pane of the SDT. The right top (Properties) pane of the SDT will show all the device properties which, since the sensor at this point does not support Feature Reports, will be the defaults chosen for that sensor by the driver. The right middle (Data) pane of the SDT will show the datafields of the driver.
5. Cause the sensor to generate data – observe the the Data pane should now show the datafields described in the Input report. Note that the optional and dynamic datafields, if used, will not be visible in the SDT until a valid input report is received from the device.
6. Do not proceed past this point until the previous steps are successful.
7. Uncomment out the Feature Report section of the report descriptor. At this point the sensor will need to respond to GET\_FEATURE requests from the driver. A simple way to check that this is happening is to set the Sensor Connection Type property to = ‘PC External’ In the SDT Properties pane you should see that SENSOR\_PROPERTY\_CONNECTION\_TYPE is = ‘2.’ Or the acccompanying text may say “PC EXTERNAL.” The default for this value is ‘1,’ so this means the driver successfully retrieved this property from the sensor.
8. Ensure the writable properties for the sensor (Reporting State, Power State, Report Interval and Change Sensitivities) are working correctly. Referring to section 5.7 of this document, be sure that the sensor is running without errors on Trace Level Error. Pay attention to any warning seen on Trace Level Warning as these likely need attention to have a properly functioning sensor.
9. If the application requires more than one sensor, build a sensor collection with only the first two sensors to be supported. Each of these sensors should have been already proven to work individually as in step 8. Verify that these two sensors can be observed in the SDT Data pane and that all datafields described in both input reports are present.
10. Add new sensors one at a time in a manner similar to step 8. If a sensor just added causes the other sensors to break, there is a problem with the newly added sensor.

Once all sensors have been added and been observed to be working correctly, other Top Level Collection devices (ex. mouse, keyboard, vendor specific devices) can be added. At each step be sure nothing has broken from the earlier steps.

Common failures that are seen might be caused by the following:

1. No Sensor or Sensor collection appears in the device Manager under the category Sensors. The cause of this is almost certainly:
   1. A syntax mistake in the report descriptor
   2. A mismatch between the report descriptor and the buffer(s) described by the report descriptor
   3. Check to see if the device appears under Human Interface Devices in the device Manager by examining the VID/PID of each device until the device of interest is located. If it appears as a HID device, but does not appear as a Sensor device, this likely means the Usage Page or Usage is incorrect for either a sensor collection or the individual sensor.
2. Sensor appears in Device Manager under Sensors, but it banged-out. Right-click on the banged-out sensor and select Properties. There should be a code for why the sensor is not working:
   1. Code 10 – this indicates that the driver loaded, but something about the information it received in a feature or input report caused the driver to fail to start. Refer to section 5.7 regading using sensor logging and examine the generated logs for clues as to why the driver failed to start. This error can also occur in the case of a bad Report Descriptor, though Code 31 is the more common case.
   2. Code 31 – this indicates the driver did not load for some reason. This is usually because of errors in the report descriptor, or a mismatch between the report descriptor and the buffer(s) described by the report descriptor.
   3. Code 43 – this is a USB-level error and has nothing to do with the fact the device is a Sensor.

## Using sensor logging

Use TraceView.exe from the latest WDK release. This is found under:

c:\Program Files\Windows Kits\8.0\Tools\{arch}\traceview.exe

Locate the symbols (.pdb) for your specific build.

Launch TraceView.exe.

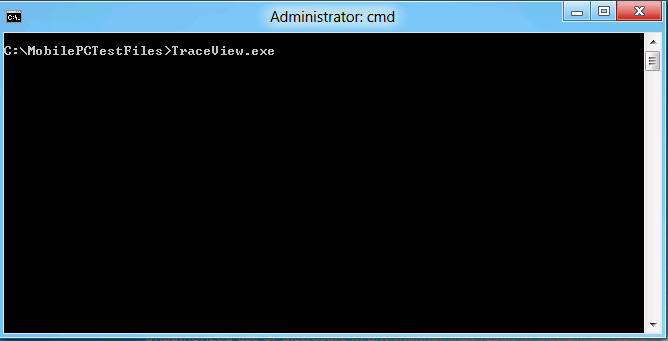


Figure 1. Administrator: cmd window with TraceView.exe

Select File > Create New Log Session.

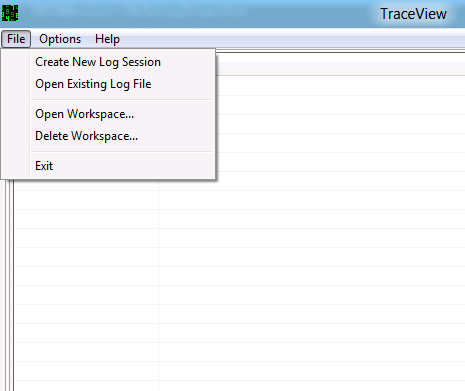


Figure 2. TraceView windows with File menu options

If you are using pdb’s, check PDB radio button and select the symbols for your build.

Click **Open**.

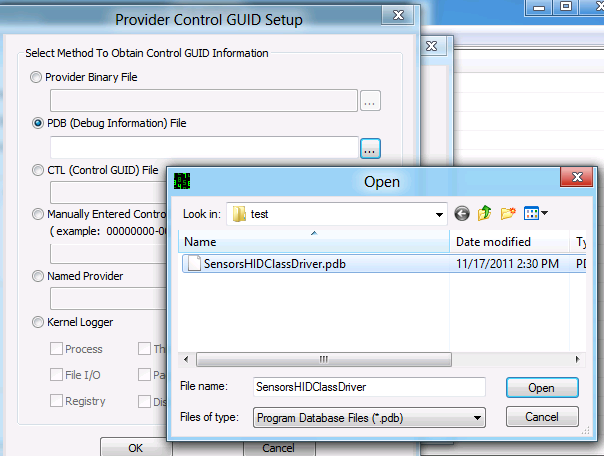


Figure 3. Pbd files

Click **OK**.

Click **Next**.

Select your logging level. Click the “>>” beside “Set Flags and Level”.

Select **Logging Level Information**.

Click **Finish**.

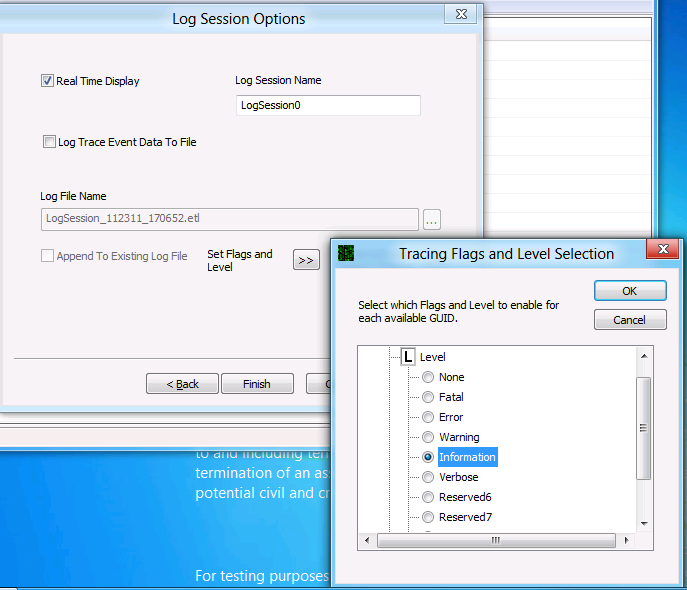


Figure 4. Tracing Flags and Level Selection window

Start using your HID Sensor Device. You will see the traces appear.

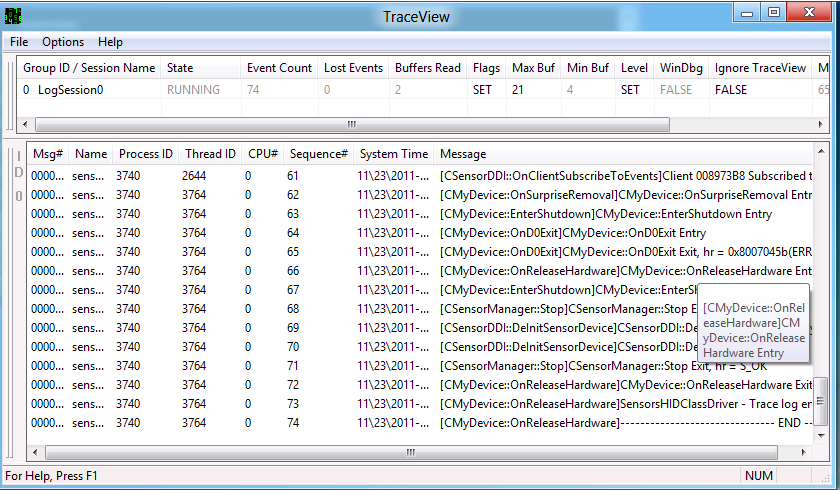


Figure 5. TraceView window with traces running

Right-click on the Logging Group to bring up a context menu. From here, you can **Stop Trace**. You can also **Save** **Workspace** so that you don’t have to keep specifying the provider info (pdb file) or logging level desired.

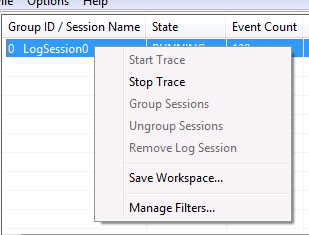


Figure 6. Context menu options

Similarly you can right-click on the **Trace Window** to bring up a context menu that allows the addition and removal of information columns. The line number column can be especially helpful.

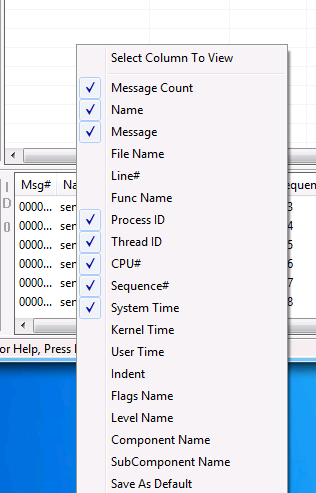


Figure 7. Select Column to View options