Tracker Firmware Specification

Nirav Patel

Revision 0.20
22 February 2013

Contents

1 Revision History

Revision	Date	Description
0.1	05-10-2012	Preliminary specification.
0.2	30-10-2012	Add Feature Reports.
0.3	02-11-2012	Add Register Feature Report.
0.4	08-11-2012	Add calibration specification.
0.5	15-11-2012	Use correct Vendor ID.
0.6	25-11-2012	Update calibration procedure.
0.7	26-11-2012	Update calibration procedure and calibration report.
0.8	01-12-2012	Update calibration commands and configuration report.
0.9	04-12-2012	Change magnetometer range settings.
0.10	05-12-2012	Rate table control commands.
0.11	07-12-2012	Table RPM specifying command.
0.12	20-12-2012	Six side calibration and updated reports.
0.13	21-12-2012	OVR coordinates, CommandKeepAlive report.
0.14	23-12-2012	SampleRate in Config report.
0.15	25-12-2012	DisplayInfo report
0.16	28-12-2012	Set default coordinates to device orientation.
0.17	11-1-2013	Calibration report size and GPIO report.
0.18	15-1-2013	Serial report.
0.19	16-1-2013	Update product name.
0.20	22-2-2013	Update display info.

2 Introduction

Tracker is an HID class Full Speed USB device which reports data from onboard 3 axis gyroscopes, accelerometers, and magnetometers at up to 1000 Hz. For more information about HID, see the Device Class Definition for Human Interface Devices[1]

- **Vendor ID:** *0x2833*
- **Product ID:** *0x0001*
- **Vendor String:** *Oculus VR, Inc.*
- **Product String:** *Tracker DK*
- **Serial String:** *Custom generated per device*

[1] http://www.usb.org/developers/devclass_docs/HID1_11.pdf

3 HID In Report

The 62 byte IN report on Endpoint 1 contains the sensor data. All data is in little-endian format. The gyro and accelerometer report data at a rate of 1000 Hz, which is also the rate at which the timestamp is incremented. While the gyro may be physically running at a data rate of up to 8000 Hz, the data is filtered to 1000 Hz before being reported.

While the target rate for the host polling the device for the IN report is 1000 Hz, system or bus load can cause the polling to happen on longer intervals, dropping the reporting rate below 1000 Hz. For this reason, the report contains fields for up to 3 samples of gyro and accelerometer data as well as a field stating the number of samples being reported. The behavior is as follows:

- If the number of recorded samples is <= 3, the corresponding number of samples is returned with SampleCount set accordingly.
- If the number of recorded samples is > 3, the first (SampleCount-2) samples are averaged into Samples[0], while Samples[1] and Samples[2] contain the most recent two recorded samples. SampleCount is set to the total original number of recorded samples and Timestamp is set to that of the first sample; this allows for PC software to compensate the integration.
- The Tracker does not accumulate more then 254 samples; beyond that the SampleCount is reset to 3 and only the most recent 3 samples are reported, with the Timestamp field properly incremented to indicate the loss of samples.

	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0	ReportID = 1	SampleCount	Timestamp
4	LastCommandID		Temperature
8	Samples[0]		
24	Samples[1]		
40	Samples[2]		
56	MagX		MagY
60	MagZ		

- **ReportID** (8 bits): The USB Report ID for this report is 1.
- **SampleCount** (8 bits): The number of samples the report represents as described above.
- **Timestamp** (16 bits): The timestamp of the first sample in the report. As the internal sampling rate is 1000 Hz, this unit is equivalent to milliseconds.
- **LastCommandID** (16 bits): Contains the CommandID from the last Feature Report set on the Tracker.
- **Temperature** (16 bits): The most recent internal sensor temperature recorded by the Tracker. The value is in two's complement format, in units of centidegrees Celsius.

- **Samples** (128 bits): Each sample contains X, Y, and Z data for the gyro and accelerometer. Specified below.
- **MagX, MagY, MagZ** (16 bits each): The most recent data available from the magnetometer. The three axes are sampled simultaneously at up to 220 Hz. Note that consecutive reports may contain the same magnetometer data. The data is in units of 10^{-4} gauss.

3.1 Accelerometer and Gyro Sample

The X, Y, and Z axes of the gyro and accelerometer are sampled simultaneously. The accelerometer fields are two's complement format and 21 bits each. They are in units of $\frac{m}{s^2} \cdot 10^4$. The gyroscope fields are two's complement and 21 bits each. They are in units of $\frac{\theta}{s} \cdot 10^4$ in radians.

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><td>0</td><td colspan="21">AccelX</td><td colspan="11">AccelY[0:10]</td></tr>
<tr><td>4</td><td colspan="10">AccelY[11:20]</td><td colspan="21">AccelZ</td><td>▨</td></tr>
<tr><td>8</td><td colspan="21">GyroX</td><td colspan="11">GyroY[0:10]</td></tr>
<tr><td>128</td><td colspan="10">GyroY[11:20]</td><td colspan="21">GyroZ</td><td>▨</td></tr>
</table>

4 Feature Reports

Feature Reports are read from and written to Endpoint 0, which is the Control Endpoint.

4.1 Configuration

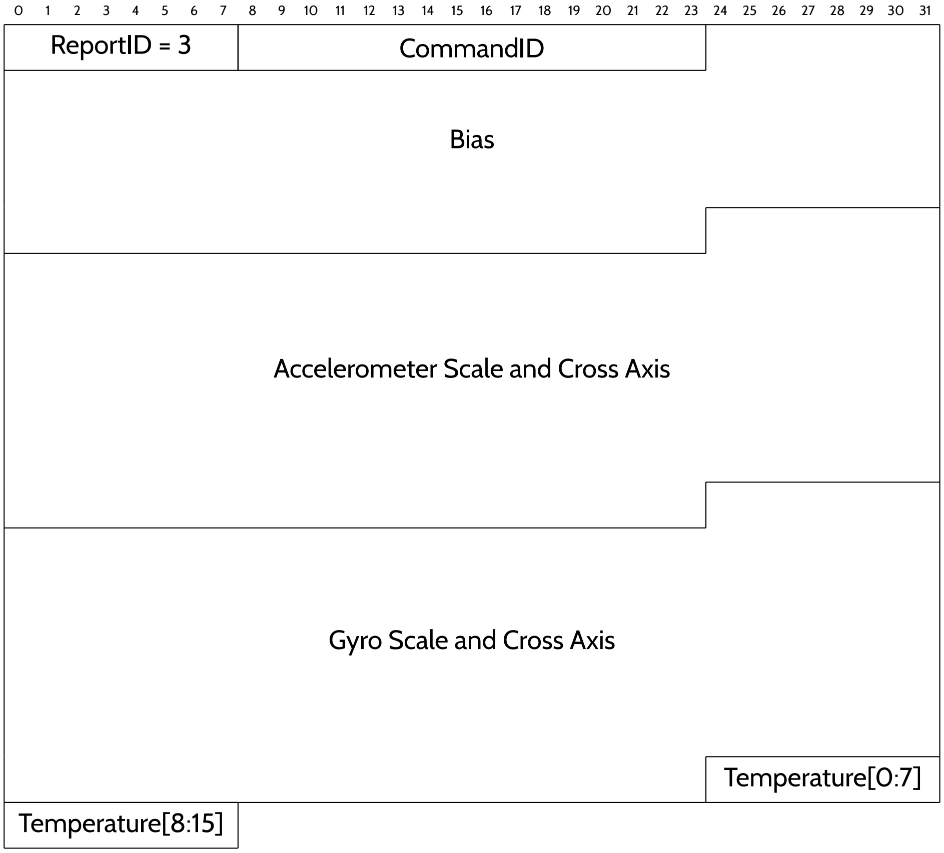
The 7 byte Configuration report has a ReportID of 2. It contains a bitfield used to query or alter the operation of the Tracker. All bits default to 0.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	ReportID = 2								CommandID																a	b	c	d	e	f	g	
4	PacketInterval								SampleRate																							

- **ReportID** (8 bits): The USB Report ID for this report is 2.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **(a) UseRaw** (1 bit): A debugging mode that outputs the samples raw from the sensors without first converting them into known units.
- **(b) InternalCalibration** (1 bit): An internal test mode for calibrating zero rate drift on the gyro. This setting should not be enabled.
- **(c) UseCalibration** (1 bit): Apply the calibration parameters stored on the Tracker or set using the Calibration Feature Report. This feature is enabled at power on.
- **(d) Autocalibration** (1 bit): Autocalibration recalibrates the gyro zero rate offset when the Tracker experiences no motion for an extended period. This feature is enabled at power on.
- **(e) UseMotionKeepAlive** (1 bit): If enabled, after the Interval value in milliseconds set in the KeepAlive report passes with no motion, the Tracker will stop sending IN Reports. This feature is enabled at power on.
- **(f) UseCommandKeepAlive** (1 bit): If enabled, after the Interval value in milliseconds set in the KeepAlive report passes with no feature reports received, the Tracker will stop sending IN Reports.
- **(g) UseSensorCoordinates** (1 bit): Output the IN Report data in the coordinate system used by LibOVR relative to the Tracker itself. This setting is disabled at power on, which means the data is reported in the coordinate system of the device it is in.
- **PacketInterval** (8 bits): The IN Report rate can be reduced from 1000 Hz by increasing the PacketInterval. The rate is $1000/(1 + PacketInterval)$. For example, setting it to 1 results in a theoretical rate of 500 Hz, while setting it to 3 results in 250 Hz.
- **SampleRate** (16 bits): The sample rate of the gyro and accelerometer in Hz. $(1/SampleRate)$ is the unit of time used in the Timestamp field of the IN Report.

4.2 Calibration

The 69 byte Calibration report has a ReportID of 3. It is used to query or set calibration parameters for bias, scale, and cross axis sensitivity for the accelerometer and gyroscope.



- **ReportID** (8 bits): The USB Report ID for this report is 3.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **Bias** (128 bits): Formatted as an Accelerometer and Gyro Sample, as specified in Section 3.1. Each gyro field represents the zero rate offset to be subtracted from the samples before being put in the IN Report, when UseCalibration is enabled. The accelerometer fields represent the offset at zero acceleration.
- **Accelerometer Scale and Cross Axis** (192 bits): Formatted as one and a half Accelerometer and Gyro Samples, as specified in Section 3.1. This nine element matrix defines both the scale values and cross axis sensitivities for the

accelerometer. The diagonal of the matrix is scale, while the remaining entries are cross axis. For example, the value at (0,1) in the matrix is the sensitivity of the X axis while the Y axis is at 1 G. Defined as a conversion from a floating point scaling multiplier to int, using the following formula for scale: $((int32\_t)((x - 1.0f) * (float)SENSOR\_MAX))$. The formula for cross axis is: $((int32\_t)((x) * (float)SENSOR\_MAX))$

- **Gyro Scale and Cross Axis** (192 bits): Formatted as one and a half Accelerometer and Gyro Samples, as specified in Section 3.1. This nine element matrix defines both the scale values and cross axis sensitivities for the gyro. The diagonal of the matrix is scale, while the remaining entries are cross axis. For example, the value at (0,1) in the matrix is the sensitivity of the X axis while the Y axis is rotating. Defined as a conversion from a floating point scaling multiplier to int, using the following formula for scale: $((int32\_t)((x-1.0f) * (float)SENSOR\_MAX))$. The formula for cross axis is: $((int32\_t)((x) * (float)SENSOR\_MAX))$
- **Temperature** (16 bits): The temperature at which the calibration data was recorded. The value is in two·s complement format, in units of centidegrees Celsius.

4.3 Full Scale Range

The 8 byte Full Scale Range report has a ReportID of 4. It contains the settings for the full scale range of the accelerometer, gyroscope, and magnetometer. Typically this report should not be set, and the default values should be used. If the values set are not supported, the nearest supported will be chosen.

<table>
<tr><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td></tr>
<tr><td>0</td><td colspan="8">ReportID = 4</td><td colspan="16">CommandID</td><td colspan="8">Accel Range</td></tr>
<tr><td>4</td><td colspan="16">Gyro Range</td><td colspan="16">Magnetometer Range</td></tr>
</table>

- **ReportID** (8 bits): The USB Report ID for this report is 2.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **Accel Range** (8 bits): The full scale range of the accelerometer in units of G. This defaults to ±2. Supported values are ±2, ±4, ±8, and ±16.
- **Gyro Range** (16 bits): The full scale range of the gyro in units of $\frac{\circ}{sec}$. This defaults to ±2000. Supported values are ±250, ±500, ±1000, and ±2000.
- **Magnetometer Range** (16 bits): The full scale range of the magnetometer in units of milliGauss. This defaults to ±1300. Supported values are ±880, ±1300, ±1900, and ±2500.

4.4 Register

The 6 byte Register report has a ReportID of 5. It is an interface to directly write to the registers of the Invensense MPU-6000[2] and Honeywell HMC5983[3] sensors on the Tracker. This report should only be used for debugging purposes. Caution should be taken when writing to the registers, as incorrect values can cause sensor malfunction.

	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31
0	ReportID = 5	CommandID		Device
4	Register	Payload		

- **ReportID** (8 bits): The USB Report ID for this report is 5.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **Device** (8 bits): The ID of the device being written to. The MPU-6000 is defined as 1, and the HMC5983 is defined as 2.
- **Register** (8 bits): The register being written to. The datasheets of the sensors should be referenced for this value.
- **Payload** (8 bits): The value being written to the register.

[2] http://invensense.com/mems/gyro/documents/RM-MPU-6000A.pdf

[3] http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/Defense_Brochures-documents/HMC5983_3_Axis_Compass_IC.pdf

4.5 Bootload

The 4 byte Bootload report has a ReportID of 6. Setting the payload to non-zero reboots the device into a bootloading mode where new firmware can be loaded onto the Tracker over USB.

<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><td>0</td><td colspan="8">ReportID = 6</td><td colspan="16">CommandID</td><td colspan="8">Bootload</td></tr>
</table>

- **ReportID** (8 bits): The USB Report ID for this report is 6.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **Bootload** (8 bits): Setting this to a non-zero value reboots the device into bootloader mode immediately.

4.6 GPIO

The 5 byte GPIO report has a ReportID of 7. It is used for internal debugging purposes to set the pins on the GPIO header to input or output. On input pins, setting the value sets the pin to be either pull up or pull down. On output pins, setting the value sets a high or low output. On reading the report, the value is the input value on input pins or the previously set output value on output pins.

	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31
0	ReportID = 7	CommandID	Direction
4	Value		

- **ReportID** (8 bits): The USB Report ID for this report is 7.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **Direction** (8 bits): A bitmask setting each pin to input or output.
- **Value** (8 bits): A bitmask setting high or low on each pin.

4.7 KeepAlive

The 5 byte KeepAlive report has a ReportID of 8. It is used to specify the interval in milliseconds allowed between feature reports when UseCommandKeepAlive and/or UseMotionKeepAlive are enabled in the Configuration report. This can also conveniently be used as the report sent to keep the Tracker alive when UseCommandKeepAlive is active. After the specified interval has passed, the Tracker will stop sending IN reports until any feature report is set or UseCommandKeepAlive is disabled.

	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31
0	ReportID = 8	CommandID	Interval[0:7]
4	Interval[8:15]		

- **ReportID** (8 bits): The USB Report ID for this report is 8.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **Interval** (16 bits): The interval in milliseconds used as the threshold for UseCommandKeepAlive and UseMotionKeepAlive. The default interval is 10000.

4.8 DisplayInfo

The 56 byte DisplayInfo report has a ReportID of 9. It is used to describe the physical and optical characteristics of the Oculus head mount display the Tracker is inside of.

	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31
0	ReportID = 9	CommandID		DistortionType
4	ResolutionX		ResolutionY	
8	DisplayX			
12	DisplayY			
16	CenterV			
20	LensSeparation			
24	LensDistanceL			
28	LensDistanceR			
32	DistortionK1			
36	DistortionK2			
40	DistortionK3			
44	DistortionK4			
48	DistortionK5			
52	DistortionK6			

- **ReportID** (8 bits): The USB Report ID for this report is 9.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **DistortionType** (8 bits): The DistortionType field defines how the Distortion fields are used.
- **ResolutionX** (16 bits): The horizontal screen resolution in pixels.
- **ResolutionY** (16 bits): The vertical screen resolution in pixels.
- **DisplayX** (32 bits): The width of the display in micrometers.
- **DisplayY** (32 bits): The height of the display in micrometers.
- **CenterV** (32 bits): The vertical center of the display in micrometers.
- **LensSeparation** (32 bits): The distance between the centers of the lenses in micrometers.

- **LensDistanceL:** (32 bits): The distance from optical center of the left lens to the screen in micrometers.
- **LensDistanceR:** (32 bits): The distance from optical center of the right lens to the screen in micrometers.
- **DistortionK1** (32 bits): The first radial distortion parameter in floating point.
- **DistortionK2** (32 bits): The second radial distortion parameter in floating point.
- **DistortionK3** (32 bits): The third radial distortion parameter in floating point.
- **DistortionK4** (32 bits): The fourth radial distortion parameter in floating point.
- **DistortionK5** (32 bits): The fifth radial distortion parameter in floating point.
- **DistortionK6** (32 bits): The sixth radial distortion parameter in floating point.

4.9 Serial

The 15 byte Serial report has a ReportID of 10. It is used internally for manufacturing purposes, when the USB serial number is set to a default number. Outside of manufacturing, the number is identical to the USB serial number. Setting this report switches the USB serial number to the value read from this report. The value being set is ignored.

	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31
0	ReportID = 10	CommandID		Serial[0]
4	Serial[1:3]			
8	Serial[4:7]			
12	Serial[8:11]			

- **ReportID** (8 bits): The USB Report ID for this report is 10.
- **CommandID** (16 bits): A sequence number that is then repeated in the LastCommandID field of the HID IN Report.
- **Serial** (12 bytes): The unique ID of the device.

5 Descriptor

The HID descriptor for the data described above is as follows:

```
    0x05, 0x03,                    // USAGE_PAGE (VR Controls)
    0x09, 0x05,                    // USAGE (Head Tracker)
    0xa1, 0x01,                    // COLLECTION (Application)

    // Sub-Collection for DK set of reports
   0x06, 0x00, 0xff,          //  USAGE_PAGE (Vendor Defined Page 1)
   0x09, 0x01,             //  USAGE (Vendor Defined - DK compatibility)
    0xa1, 0x02,                    //   COLLECTION (Logical)
    0x05, 0x01,                  //   USAGE_PAGE (Generic Desktop)

    // IN Report for sensor data
    0x85, 0x01,                    //   REPORT_ID (1)
    // Sample Count
    0x09, 0x3b,                    //   USAGE (Byte Count)
    0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
    0x75, 0x08,                    //   REPORT_SIZE (8)
    0x95, 0x01,                    //   REPORT_COUNT (1)
    0x81, 0x03,                    //   INPUT (Cnst,Var,Abs)
    // Timestamp
    0x09, 0x3b,                    //   USAGE (Byte Count)
    0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
    0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                    //   REPORT_SIZE (16)
    0x95, 0x01,                    //   REPORT_COUNT (1)
    0x81, 0x03,                    //   INPUT (Cnst,Var,Abs)
    // LastCommandID
    0x09, 0x3b,                    //   USAGE (Byte Count)
    0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
    0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                    //   REPORT_SIZE (16)
    0x95, 0x01,                    //   REPORT_COUNT (1)
    0x81, 0x03,                    //   INPUT (Cnst,Var,Abs)
    // Temperature
    0x09, 0x3b,                    //   USAGE (Byte Count)
    0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
    0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                    //   REPORT_SIZE (16)
    0x95, 0x01,                    //   REPORT_COUNT (1)
    0x81, 0x03,                    //   INPUT (Cnst,Var,Abs)
    // Gyro/Accel Sample 0
    0x09, 0x3b,                    //   USAGE (Byte Count)
    0x15, 0x00,                     //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,               //   LOGICAL_MAXIMUM (255)
    0x75, 0x08,                     //   REPORT_SIZE (8)
    0x95, 0x10,                     //   REPORT_COUNT (16)
    0x81, 0x03,                     //   INPUT (Cnst,Var,Abs)
    // Gyro/Accel Sample 1
    0x09, 0x3b,                     //   USAGE (Byte Count)
    0x15, 0x00,                     //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,               //   LOGICAL_MAXIMUM (255)
    0x75, 0x08,                     //   REPORT_SIZE (8)
    0x95, 0x10,                     //   REPORT_COUNT (16)
    0x81, 0x03,                     //   INPUT (Cnst,Var,Abs)
    // Gyro/Accel Sample 2
    0x09, 0x3b,                     //   USAGE (Byte Count)
    0x15, 0x00,                     //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,               //   LOGICAL_MAXIMUM (255)
    0x75, 0x08,                     //   REPORT_SIZE (8)
    0x95, 0x10,                     //   REPORT_COUNT (16)
    0x81, 0x03,                     //   INPUT (Cnst,Var,Abs)
    // Mag Sample
    0x09, 0x3b,                     //   USAGE (Byte Count)
    0x15, 0x00,                     //   LOGICAL_MINIMUM (0)
    0x27, 0xff, 0xff, 0x00, 0x00,   //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                     //   REPORT_SIZE (16)
    0x95, 0x03,                     //   REPORT_COUNT (3)
    0x81, 0x03,                     //   INPUT (Cnst,Var,Abs)

    // Config Report
    0x85, 0x02,                     //   REPORT_ID (2)
    0x09, 0x3b,                     //   USAGE (Byte Count)
    0x15, 0x00,                     //   LOGICAL_MINIMUM (0)
    0x27, 0xff, 0xff, 0x00, 0x00,   //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                     //   REPORT_SIZE (16)
    0x95, 0x01,                     //   REPORT_COUNT (1)
    0xb1, 0x03,                     //   FEATURE (Cnst,Var,Abs)
    0x09, 0x3b,                     //   USAGE (Byte Count)
    0x15, 0x00,                     //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,               //   LOGICAL_MAXIMUM (255)
    0x75, 0x08,                     //   REPORT_SIZE (8)
    0x95, 0x04,                     //   REPORT_COUNT (4)
    0xb1, 0x03,                     //   FEATURE (Cnst,Var,Abs)

    // Calibration Report
    0x85, 0x03,                     //   REPORT_ID (3)
    0x09, 0x3b,                     //   USAGE (Byte Count)
    0x15, 0x00,                     //   LOGICAL_MINIMUM (0)
```

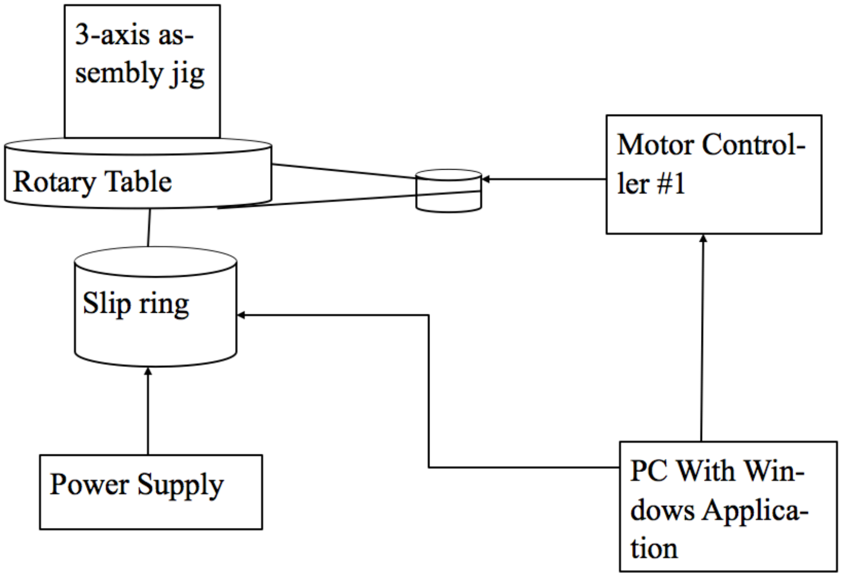
```
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
0x75, 0x10,                    //   REPORT_SIZE (16)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x42,                    //   REPORT_COUNT (66)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

// Full Scale Range Report
0x85, 0x04,                    //   REPORT_ID (4)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
0x75, 0x10,                    //   REPORT_SIZE (16)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x05,                    //   REPORT_COUNT (5)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

// Register Report
0x85, 0x05,                    //   REPORT_ID (5)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
0x75, 0x10,                    //   REPORT_SIZE (16)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x03,                    //   REPORT_COUNT (3)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

// DFU Report
0x85, 0x06,                    //   REPORT_ID (6)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
```

```
0x75, 0x10,                    //   REPORT_SIZE (16)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

// GPIO Report
0x85, 0x07,                    //   REPORT_ID (7)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
0x75, 0x10,                    //   REPORT_SIZE (16)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x02,                    //   REPORT_COUNT (2)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

// Keep Alive Report
0x85, 0x08,                    //   REPORT_ID (8)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
0x75, 0x10,                    //   REPORT_SIZE (16)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x02,                    //   REPORT_COUNT (2)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

// Display Info Report
0x85, 0x09,                    //   REPORT_ID (9)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
0x75, 0x10,                    //   REPORT_SIZE (16)
```

```
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x35,                    //   REPORT_COUNT (53)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

// Serial Report
0x85, 0x0A,                    //   REPORT_ID (10)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x27, 0xff, 0xff, 0x00, 0x00,  //   LOGICAL_MAXIMUM (65535)
0x75, 0x10,                    //   REPORT_SIZE (16)
0x95, 0x01,                    //   REPORT_COUNT (1)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)
0x09, 0x3b,                    //   USAGE (Byte Count)
0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00,              //   LOGICAL_MAXIMUM (255)
0x75, 0x08,                    //   REPORT_SIZE (8)
0x95, 0x0C,                    //   REPORT_COUNT (12)
0xb1, 0x03,                    //   FEATURE (Cnst,Var,Abs)

0xc0,                          //   END_COLLECTION

0xc0                           // END_COLLECTION
```

6 Factory Calibration

The gyroscope and accelerometer on the Tracker require factory calibration for scale and bias on each axis. The calibration and test jig consists of a single axis rate table, a Tracker holder capable of being secured to the rate table on three orthogonal sides, and a PC which interfaces to both the rate table and the Tracker.



6.1 Physical Interfaces

There are two serial ports on the PC, U1 and U2. U1 is connected to the Tracker via a Bluetooth virtual serial port and U2 goes to the motor controller on the rate table.

Wire Name	Comes from	Goes to	Purpose
U1-tx	PC Virtual Serial port	Tracker at 3.3VTTL signal levels	Calibration state control
U1-rx	PC Virtual Serial port	Tracker at 3.3VTTL signal levels	Calibration state feedback
U2-tx	PC Virtual Serial port	Rate table at RS-232 levels	Rate table commands
U2-rx	PC Virtual Serial port	Rate table at RS-232 levels	Rate table feedback
Power Supply + Agilent E3647A	Programmable supply	Slip ring supply for Tracker	5V power for Tracker
Power Supply Ground - Agilent E3647A	Programmable supply	Slip ring supply for Tracker	Ground

6.2 Tracker Interface Protocol

The Tracker interface is a USART configured at 115200,N,8,1. Commands are single byte values.

Stage Commands	Value	Purpose
NONE	'0'	No effect when sent to Tracker. Sent from Tracker when in state when asked for Status.
ZDOWN	'1'	Puts Tracker in Z-down calibration mode. Returned by Tracker as acknowledgement.
ZUP	'2'	Puts Tracker in Z-up calibration mode. Returned by Tracker as acknowledgement.
ZROT	'3'	Puts Tracker in Z-rotation calibration mode. Returned by Tracker as acknowledgement.
YDOWN	'4'	Puts Tracker in Y-down calibration mode. Returned by Tracker as acknowledgement.
YUP	'5'	Puts Tracker in Y-up calibration mode. Returned by Tracker as acknowledgement.
YROT	'6'	Puts Tracker in Y-rotation calibration mode. Returned by Tracker as acknowledgement.
XDOWN	'7'	Puts Tracker in X-down calibration mode. Returned by Tracker as acknowledgement.
XUP	'8'	Puts Tracker in X-up calibration mode. Returned by Tracker as acknowledgement.
XROT	'9'	Puts Tracker in X-rotation calibration mode. Returned by Tracker as acknowledgement.
COMPLETE	'c'	Tells the Tracker to compute and store calibration parameters. Returned by Tracker as acknowledgement.

Returned by Tracker	Value	Purpose
BUSY	’b’	Returned when the Tracker receives a stage change command while the current stage is still running.
OUT_OF_ORDER	’o’	Returned when the Tracker receives an out of order stage change command.
OUT_OF_RANGE	’r’	Returned when the Tracker computes a calibration parameter out of expected range.
UNKNOWN	’u’	Returned when the Tracker receives an unknown command.
FINISHED	’f’	Sent by the Tracker when the calibration stage being run completes.

PC to Tracker	Value	Purpose
STATUS	’s’	Asks the Tracker for the current stage, which it sends back.
DEBUG	’d’	Toggles debug printout of sensor data from the Tracker.
TABLE_RPM	”t78.01\r\n”	Sets the RPM the table will be running at. Returns ’t’ on receiving ’t’. Returns UNKNOWN if a character that is not a period or digit is received before the newline. After receiving the newline, returns FINISHED after successfully parsing the rpm. Returns OUT_OF_RANGE if the parsed number is unexpected.

6.3 Rate Table Interface Protocol

The rate table being used is a Stanton ST-150 turntable modified to be controllable over a USB Serial interface. The interface is configured to 9600,N,8,1.

Commands	**Value**	**Purpose**
33RPM	'3'	Puts the table into 33 rpm mode. Returned on success.
45RPM	'4'	Puts the table into 45 rpm mode. Returned on success.
78RPM	'7'	Puts the table into 78 rpm mode. Returned on success.
POWER	'p'	Powers on the table. Returned on success.
OFF	'o'	Powers off the table. Returned on success.
GO	'g'	Tells the table to start rotation. Returned on success.
STOP	's'	Tells the table to stop rotation. Returned on success.
Returned by Table	**Value**	**Purpose**
ERROR	'e'	Returned when the table attempts to set a speed and fails.
UNKNOWN	'u'	Returned when the table receives an unknown command.

After starting table rotation or changing table speed while rotation is going, the calibration application should wait for 5 seconds to allow the table to stabilize at the proper speed.

6.4 Calibration Procedure

The Tracker calibration procedure consists of six stages, ZUP, ZROT, YUP, YROT, XUP, and XROT. These stages must be completed in order, and all stages must be complete before issuing a COMPLETE command which computes the final calibration parameters and commits them to non-volatile storage. The calibration procedure is as follows:

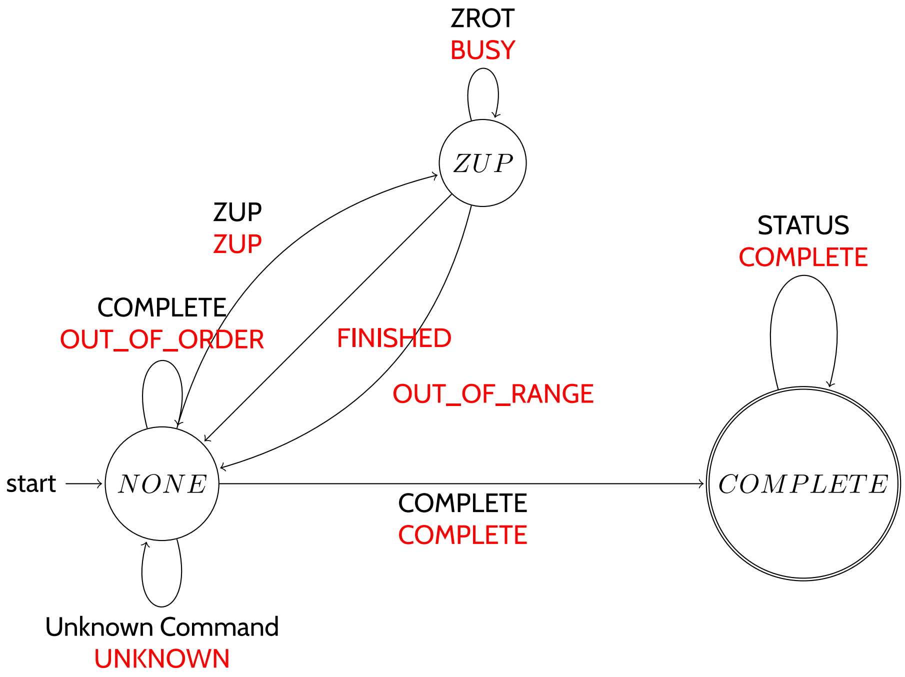
1. The user places the Tracker in the calibration box and closes the holder door.
2. The user places the box in the center of the rate table with the Z axis down.
3. The user presses the start calibration button in the application.
4. The application sends ON and 78RPM to the table.
5. The application sends a ZDOWN command to the Tracker.
6. After the Tracker returns FINISHED, the application prompts the user to place the box with the Z axis up.
7. The user presses the start calibration button, which sends a ZUP command to the Tracker.
8. After the Tracker returns FINISHED, the application sends GO to the table.
9. After a 5 second delay, the application sends the ZROT command to the Tracker.
10. After the Tracker returns FINISHED, the application sends STOP to the table and notifies the user to place the box with the Y axis down.
11. The user presses the start calibration button, which sends a YDOWN command to the Tracker.
12. After the Tracker returns FINISHED, the application prompts the user to place the box with the Y axis up.
13. The user presses the start calibration button, which sends a YUP command to the Tracker.
14. After the Tracker returns FINISHED, the application sends GO to the table.
15. After a 5 second delay, the application sends the YROT command to the Tracker.
16. After the Tracker returns FINISHED, the application sends STOP to the table and notifies the user to place the box with the X axis down.
17. The user presses the start calibration button, which sends a XDOWN command to the Tracker.
18. After the Tracker returns FINISHED, the application prompts the user to place the box with the X axis up.

19. The user presses the start calibration button, which sends a XUP command to the Tracker.
20. After the Tracker returns FINISHED, the application sends GO to the table.
21. After a 5 second delay, the application sends the XROT command to the Tracker.
22. After the Tracker returns FINISHED, the application sends STOP to the table.
23. The application sends the COMPLETE command to the Tracker.
24. After the Tracker returns COMPLETE, the application sends OFF to the table and notifies the user to remove the Tracker from the box and restart at step 1 with a new Tracker.

If during a calibration stage, the Tracker returns OUT_OF_RANGE, the application should prompt the user to check that the calibration box is placed correctly. The user should correct any placement error and press the start calibration button. The application should then run the same stage again. If the stage again fails, the Tracker should be removed from calibration and set aside for examination by engineering.

If at any point the Tracker returns OUT_OF_ORDER, the Tracker should be reset and calibration should restart from step 1.

The simplified state diagram below shows how the Tracker responds to various commands. The ZUP state represents all of the calibration stages in the diagram. The text in black represents commands sent from the PC to the Tracker. The text in red represents responses and notifications from the Tracker to the PC.


ZROT
BUSY
ZUP
ZUP
ZUP
STATUS
COMPLETE
COMPLETE
OUT_OF_ORDER
FINISHED
OUT_OF_RANGE
start
NONE
COMPLETE
COMPLETE
COMPLETE
Unknown Command
UNKNOWN