ADNS-2700
Single Chip USB Optical Mouse Sensor

Data Sheet

Description

The ADNS-2700 is a compact, one chip USB optical mouse sensor designed for implementation of a non-mechanical tracking engine in computer mice.

It is based on optical navigation technology that measures changes in position by optically acquiring sequential surface images (frames) and mathematically determining the direction and magnitude of movement.

The sensor is in an 8-pin optical package that is designed to be used with the ADNS-5100-001 trim lens, LED clip and the HLMP-EG3E-xxxxx LED. Together, these parts provide a complete and compact mouse sensor. There are no moving parts and precision optical alignment is not required, thus facilitating high volume assembly.

The output format is USB. This device meets HID Revision 1.11 specification and is compatible with USB Revision 2.0 specification.

Frame rate is varied internally to the sensor to achieve tracking and speed performance, eliminating the need for the use of many registers.

Default resolution is specified as 1000 counts per inch, with rates of motion up to 30 inches per second.

A complete mouse can be built with the addition of a PC board, switches and Z-wheel, plastic case and cable.

Features

- One chip USB mouse sensor
- USB 2.0 Low Speed Compliance
- 12 bits USB motion data reporting
- Meets HID Revision 1.11
- Single 5.0 volts power supply
- High speed motion detection at 30 inches per second (ips) and acceleration up to 8 g
- Input buttons: 3 buttons
- Z-Wheel interface for vertical scroll
- Integrated oscillator
- Integrated USB D- pull-up resistor
- Product string is set to “USB Optical Mouse”
- On-chip OTP memory for device configuration flexibility without any external software driver:
  - Programmable resolution from 500 to 1250 counts per inch (cpi) with 250 cpi step.
  - Programmable sensor orientation
  - Programmable VID and PID.

Applications

- Corded optical mice
- Trackballs
- Integrated input devices

NOTE: ADNS-2700 will be referred to as “sensor”, ADNS-5100-001 as “trim lens” and HLMP-EG3E as “LED” hereafter.
**Theory of Operation**

The sensor is based on Optical Navigation Technology. It contains an Image Acquisition System (IAS), a Digital Signal Processor (DSP) and USB stream output.

The IAS acquires microscopic surface images via the lens and illumination system provided by the trim lens. The clip and LED. These images are processed by the DSP to determine the direction and distance of motion. The DSP generates the $\Delta x$ and $\Delta y$ relative displacement values which are converted to USB motion data.

**Table 1. Pin Name Description**

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Input/Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XY_LED</td>
<td>I</td>
<td>XY_LED</td>
</tr>
<tr>
<td>VDDA5</td>
<td>–</td>
<td>5-Volt Power</td>
</tr>
<tr>
<td>ZB</td>
<td>I</td>
<td>Z-Wheel quadrature input</td>
</tr>
<tr>
<td>ZA</td>
<td>I</td>
<td>Z-Wheel quadrature input</td>
</tr>
<tr>
<td>SW</td>
<td>–</td>
<td>3-in-1 button pin. Do not force any voltage into this pin</td>
</tr>
<tr>
<td>GND</td>
<td>–</td>
<td>Ground</td>
</tr>
<tr>
<td>D-</td>
<td>I/O</td>
<td>USB D- line</td>
</tr>
<tr>
<td>D+</td>
<td>I/O</td>
<td>USB D+ line</td>
</tr>
</tbody>
</table>

**Package Pinout**

**Figure 1. Package outline drawing (top view)**
CAUTION: It is advised that normal static precautions be taken in handling and assembling of this component to prevent damage and/or degradation which may be induced by ESD.
Overview of Optical Mouse Sensor Assembly

Avago Technologies provides an IGES file drawing describing the base plate molding features for lens and PCB alignment. The sensor is designed for mounting on a through-hole PCB. There is an aperture stop and features on the package that align to the lens. The lens provides optics for the imaging of the surface as well as the illumination of the surface at the optimum angle. Features on the lens align it to the sensor, base plate, and clip with the LED. The clip holds the LED in relation to the lens. The LED must be inserted into the clip and the LED's leads formed prior to loading on the PCB.

Figure 3. Recommended PCB Mechanical Cutouts and Spacing
Figure 4. 2D Assembly drawing of sensor (Top and Side View)

NOTE: Dimensions in mm/Inches
Important Note: Pin 1 of sensor should be located nearest to the LED

Figure 5. Distance from lens reference plane to tracking surface (Z)
PCB Assembly Considerations

1. Insert the sensor and all other electrical components into PCB.
2. Insert the LED into the assembly clip and bend the leads 90 degrees.
3. Insert the LED clip assembly into PCB.
4. This sensor package is only qualified for wave-solder process.
5. Wave solder the entire assembly in a no-wash solder process utilizing solder fixture. The solder fixture is needed to protect the sensor during the solder process. It also sets the correct sensor-to-PCB distance as the lead shoulders do not normally rest on the PCB surface. The fixture should be designed to expose the sensor leads to solder while shielding the optical aperture from direct solder contact.
6. Place the lens onto the base plate.
7. Remove the protective Kapton tape from optical aperture of the sensor. Care must be taken to keep contaminants from entering the aperture. Recommend not to place the PCB facing up during the entire mouse assembly process. Recommend to hold the PCB first vertically for the Kapton removal process.
8. Insert PCB assembly over the lens onto the base plate aligning post to retain PCB assembly. The sensor aperture ring should self-align to the lens.
9. The optical position reference for the PCB is set by the base plate and lens. Note that the PCB motion due to button presses must be minimized to maintain optical alignment.
10. Install mouse top case. There MUST be a feature in the top case to press down onto the PCB assembly to ensure all components are interlocked to the correct vertical height.
Design considerations for improving ESD Performance

The table below shows typical values assuming base plate construction per the Avago Technologies supplied IGES file trim lens. Stand-off of the base plate shall not be larger than 5 mm.

<table>
<thead>
<tr>
<th>Typical Value</th>
<th>Distance (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creepage</td>
<td>17.9 mm</td>
</tr>
<tr>
<td>Clearance</td>
<td>9.2 mm</td>
</tr>
</tbody>
</table>

Note that the lens material is polycarbonate or polystyrene HH30, therefore, cyanoacrylate based adhesives should not be used as they will cause lens material deformation.
Figure 8. Application Circuit with sensor

Table 1

<table>
<thead>
<tr>
<th>Rbin Value (Ohm)</th>
<th>LED Bin</th>
</tr>
</thead>
<tbody>
<tr>
<td>59.0 ~ 78.7</td>
<td>Q</td>
</tr>
<tr>
<td>59.0 ~ 93.1</td>
<td>R</td>
</tr>
<tr>
<td>59.0 ~ 110.0</td>
<td>S</td>
</tr>
<tr>
<td>59.0 ~ 143.0</td>
<td>T</td>
</tr>
</tbody>
</table>

Use ONLY 1% tolerance resistor for buttons

1. Try to shield the HWFL with a SHIELD Plane (USB pin 5) similar size of HWFL before connect to GND. Isolate the Shield island with GND plane.
2. Don’t use jumper wire. Use resistor to jump instead. Keep the trace wide. Keep (pin 5) away from noisy traces, GND plane and Power plane. The 3 button trace for SW pin need to be same width, length.
### Absolute Maximum Ratings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Units</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Temperature</td>
<td>$T_S$</td>
<td>-40</td>
<td>85</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>$T_A$</td>
<td>-15</td>
<td>55</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>Lead Solder Temperature</td>
<td></td>
<td>260</td>
<td>0</td>
<td>°C</td>
<td>For 10 seconds, 1.6 mm below seating plane.</td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>$V_{DDA}$</td>
<td>-0.5</td>
<td>5.5</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>ESD</td>
<td></td>
<td>2</td>
<td>0</td>
<td>kV</td>
<td>All pins, human body model JESD22-A114</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>$V_{IN}$</td>
<td>-0.5</td>
<td>$V_{DDA}$ +0.5</td>
<td>V</td>
<td>All I/O pins except D+, D-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1.0</td>
<td>4.6</td>
<td>V</td>
<td>D+, D-, AC waveform, see USB specification (7.1.1)</td>
</tr>
<tr>
<td>Input Short Circuit Voltage</td>
<td>$V_{SC}$</td>
<td>0</td>
<td>$V_{DDA}$</td>
<td>V</td>
<td>D+, D-, see USB specification (7.1.1)</td>
</tr>
</tbody>
</table>

### Recommended Operating Condition

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
<th>Units</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Temperature</td>
<td>$T_A$</td>
<td>0</td>
<td>40</td>
<td>40</td>
<td>°C</td>
<td>For accurate navigation and proper USB operation</td>
</tr>
<tr>
<td>Power Supply Voltage</td>
<td>$V_{DDA}$</td>
<td>4.25</td>
<td>5.0</td>
<td>5.25</td>
<td>V</td>
<td>Maintains communication to USB host and internal register contents.</td>
</tr>
<tr>
<td></td>
<td>$V_{ddm}$</td>
<td>4.0</td>
<td>5.0</td>
<td>5.25</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Power Supply Rise Time</td>
<td>$V_{RT}$</td>
<td>0.1</td>
<td>6</td>
<td></td>
<td>ms</td>
<td>Peak to peak within 0-100 MHz bandwidth</td>
</tr>
<tr>
<td>Supply Noise</td>
<td>$V_N$</td>
<td></td>
<td>100</td>
<td></td>
<td>mV</td>
<td></td>
</tr>
<tr>
<td>Velocity</td>
<td>Vel</td>
<td>30</td>
<td></td>
<td></td>
<td>ips</td>
<td></td>
</tr>
<tr>
<td>Acceleration</td>
<td>Acc</td>
<td>8</td>
<td></td>
<td></td>
<td>g</td>
<td>0.5 g from Rest</td>
</tr>
<tr>
<td>Serial Port Clock Frequency</td>
<td>$f_{SCLK}$</td>
<td>1</td>
<td></td>
<td></td>
<td>MHz</td>
<td>50% duty cycle</td>
</tr>
<tr>
<td>Distance from Lens Reference Plane to Surface</td>
<td>Z</td>
<td>2.3</td>
<td>2.4</td>
<td>2.5</td>
<td>mm</td>
<td>See Figure 9</td>
</tr>
<tr>
<td>Light Level onto IC</td>
<td>$IRR_{INC}$</td>
<td>80</td>
<td>25000</td>
<td></td>
<td>MW/m²</td>
<td>$\lambda = 639$ nm</td>
</tr>
<tr>
<td>Frame Rate</td>
<td></td>
<td></td>
<td>2400</td>
<td></td>
<td>fps</td>
<td>Internally adjusted by sensor</td>
</tr>
</tbody>
</table>

### Regulatory Requirements

- Passes FCC B and worldwide analogous emission limits when assembled into a mouse with shielded cable and following Avago Technologies recommendations.
- Passes EN61000-4-4/IEC801-4 EFT tests when assembled into a mouse with shielded cable and following Avago Technologies recommendations.
- UL flammability level UL94 V-0.
- Provides sufficient ESD creepage/clearance distance to withstand discharge up to 8 kV when assembled into a mouse with trim lens according to usage instructions above.

---

![Figure 9. Distance from lens reference plane to object surface](image-url)
## AC Electrical Specifications

Electrical Characteristics over recommended operating conditions. Typical values at 25°C, VDDA5 = 5.0 V.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
<th>Units</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wakeup Delay from Rest Mode</td>
<td>T\textsubscript{WUPP}</td>
<td>1</td>
<td>2</td>
<td></td>
<td>ms</td>
<td></td>
</tr>
<tr>
<td>Due to Motion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Up Delay</td>
<td>T\textsubscript{PUP}</td>
<td></td>
<td></td>
<td>50</td>
<td>ms</td>
<td></td>
</tr>
<tr>
<td>Debounce Delay on Button Inputs</td>
<td>T\textsubscript{DBB}</td>
<td>5</td>
<td>9</td>
<td>17</td>
<td>ms</td>
<td>&quot;Maximum&quot; specified at 8 ms polling rate.</td>
</tr>
<tr>
<td>Scroll Wheel Sampling Period</td>
<td>T\textsubscript{SW}</td>
<td>150</td>
<td>200</td>
<td>300</td>
<td>µs</td>
<td>ZA PIN</td>
</tr>
<tr>
<td>Transient Supply Current</td>
<td>I\textsubscript{DDT}</td>
<td>60</td>
<td></td>
<td></td>
<td>mA</td>
<td>Max. supply current during a VDDA5 ramp from 0 to 5.0 V with &gt; 500 µs rise time. Does not include charging currents for bypass capacitors.</td>
</tr>
</tbody>
</table>

## USB Electrical Specifications

Electrical Characteristics over recommended operating conditions.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Units</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output Signal Crossover Voltage</td>
<td>V\textsubscript{CRS}</td>
<td>1.5</td>
<td>2.0</td>
<td>V</td>
<td>(C\textsubscript{L} = 200) to 600 pF (see Figure 10)</td>
</tr>
<tr>
<td>Input Signal Crossover Voltage</td>
<td>V\textsubscript{ICRS}</td>
<td>1.2</td>
<td>2.1</td>
<td>V</td>
<td>(C\textsubscript{L} = 200) to 600 pF (see Figure 10)</td>
</tr>
<tr>
<td>Output High</td>
<td>V\textsubscript{OH}</td>
<td>2.8</td>
<td>3.6</td>
<td>V</td>
<td>with 15 kΩ to Ground and 7.5 kΩ to V\textsubscript{BUS} on D- (see Figure 11)</td>
</tr>
<tr>
<td>Output Low</td>
<td>V\textsubscript{OL}</td>
<td>0.0</td>
<td>0.3</td>
<td>V</td>
<td>with 15 kΩ to Ground and 7.5 kΩ to V\textsubscript{BUS} on D- (see Figure 11)</td>
</tr>
<tr>
<td>Single Ended Input</td>
<td>V\textsubscript{SEI}</td>
<td>0.8</td>
<td></td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Input High (Driven)</td>
<td>V\textsubscript{IH}</td>
<td>2.0</td>
<td></td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Input High (Floating)</td>
<td>V\textsubscript{IHZ}</td>
<td>2.7</td>
<td>3.6</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Input Low</td>
<td>V\textsubscript{IL}</td>
<td>0.8</td>
<td></td>
<td>V</td>
<td>7.5 kΩ to VDDA5</td>
</tr>
<tr>
<td>Differential Input Sensitivity</td>
<td>V\textsubscript{DI}</td>
<td>0.2</td>
<td></td>
<td>V</td>
<td>(</td>
</tr>
<tr>
<td>Differential Input Common Mode Range</td>
<td>V\textsubscript{CM}</td>
<td>0.8</td>
<td>2.5</td>
<td>V</td>
<td>Includes V\textsubscript{DI}, See Figure 12</td>
</tr>
<tr>
<td>Single Ended Receiver Threshold</td>
<td>V\textsubscript{SE}</td>
<td>0.8</td>
<td>2.0</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Transceiver Input Capacitance</td>
<td>C\textsubscript{IN}</td>
<td>12</td>
<td></td>
<td>pF</td>
<td>D+ to V\textsubscript{BUS}, D- to V\textsubscript{BUS}</td>
</tr>
</tbody>
</table>
### USB Timing Specifications

Timing Specifications over recommended operating conditions.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Units</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>D+/D- Transition rise time</td>
<td>$T_{LR}$</td>
<td>75</td>
<td></td>
<td>ns</td>
<td>$C_L = 200$ pF (10% to 90%), see Figure 10</td>
</tr>
<tr>
<td>D+/D- Transition rise time</td>
<td>$T_{LR}$</td>
<td>300</td>
<td></td>
<td>ns</td>
<td>$C_L = 600$ pF (10% to 90%), see Figure 10</td>
</tr>
<tr>
<td>D+/D- Transition fall time</td>
<td>$T_{LF}$</td>
<td>75</td>
<td></td>
<td>ns</td>
<td>$C_L = 200$ pF (90% to 10%), see Figure 10</td>
</tr>
<tr>
<td>D+/D- Transition fall time</td>
<td>$T_{LF}$</td>
<td>300</td>
<td></td>
<td>ns</td>
<td>$C_L = 600$ pF (90% to 10%), see Figure 10</td>
</tr>
<tr>
<td>Rise and Fall time matching</td>
<td>$T_{R/FM}$</td>
<td>80</td>
<td>125</td>
<td>%</td>
<td>$T_R/T_F; C_L = 200$ pF; Excluding the first transition from the Idle State</td>
</tr>
<tr>
<td>Wakeup delay from USB suspend mode due to buttons push</td>
<td>$T_{WUPB}$</td>
<td></td>
<td></td>
<td>ms</td>
<td>Delay from button push to USB operation Only required if remote wakeup enabled</td>
</tr>
<tr>
<td>Wakeup delay from USB suspend mode due to buttons push until accurate navigation</td>
<td>$T_{WUPN}$</td>
<td></td>
<td></td>
<td>ms</td>
<td>Delay from button push to navigation operation. Only required if remote wakeup enabled</td>
</tr>
<tr>
<td>USB reset time</td>
<td>$T_{reset}$</td>
<td>18.7</td>
<td></td>
<td>$\mu$s</td>
<td></td>
</tr>
<tr>
<td>Data Rate</td>
<td>$t_{LDRATE}$</td>
<td>1.4775</td>
<td>1.5225</td>
<td>Mb/s</td>
<td>Average bit rate, 1.5 Mb/s +/- 1.5%</td>
</tr>
<tr>
<td>Receiver Jitter Tolerance</td>
<td>$t_{DJR1}$</td>
<td>-75</td>
<td>75</td>
<td>ns</td>
<td>To next transition, see Figure 13</td>
</tr>
<tr>
<td>Receiver Jitter Tolerance</td>
<td>$t_{DJR2}$</td>
<td>-45</td>
<td>45</td>
<td>ns</td>
<td>For paired transitions, see Figure 13</td>
</tr>
<tr>
<td>Differential to EOP Transition Skew</td>
<td>$t_{LDEOP}$</td>
<td>-40</td>
<td>100</td>
<td>ns</td>
<td>See Figure 14</td>
</tr>
<tr>
<td>EOP Width at Receiver</td>
<td>$t_{LEOPR}$</td>
<td>670</td>
<td></td>
<td>ns</td>
<td>Accepts EOP, see Figure 14</td>
</tr>
<tr>
<td>Source EOP Width</td>
<td>$t_{LEOPT}$</td>
<td>1.25</td>
<td>1.50</td>
<td>$\mu$s</td>
<td></td>
</tr>
<tr>
<td>Width of SE0 interval during Differential Transition</td>
<td>$t_{LST}$</td>
<td></td>
<td>210</td>
<td>ns</td>
<td>See Figure 11</td>
</tr>
<tr>
<td>Differential Output Jitter</td>
<td>$t_{UDJ1}$</td>
<td></td>
<td>-95</td>
<td>95</td>
<td>To next transition, see Figure 15</td>
</tr>
<tr>
<td>Differential Output Jitter</td>
<td>$t_{UDJ2}$</td>
<td></td>
<td>-150</td>
<td>150</td>
<td>For paired transitions, see Figure 15</td>
</tr>
</tbody>
</table>

![Figure 10. Data Signal Rise and Fall Times](image-url)
Figure 11. Data Signal Voltage Levels

Figure 12. Differential Receiver Input Sensitivity vs. Common Mode Input Range

Figure 13. Receiver Jitter Tolerance
Figure 14. Differential to EOP Transition Skew and EOP Width

Figure 15. Differential Output Jitter
## DC Electrical Specifications

Electrical Characteristics over recommended operating conditions. Typical values at 25° C, $V_{DDAS} = 5.0$ V.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
<th>Units</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Current, Mouse Moving</td>
<td>$I_{DDS}$</td>
<td>35</td>
<td>50</td>
<td></td>
<td>mA</td>
<td>Includes $XY_LED$ current</td>
</tr>
<tr>
<td>System Current, Mouse Not Moving</td>
<td>$I_{DDS_N}$</td>
<td>15</td>
<td>30</td>
<td></td>
<td>mA</td>
<td>Includes $XY_LED$ current</td>
</tr>
<tr>
<td>System Current, USB Suspend Mode, Remote Wakeup Enabled</td>
<td>$I_{DDS_S}$</td>
<td>500</td>
<td>7.5</td>
<td>$\mu$A</td>
<td></td>
<td>Includes $XY_LED$ current and $D-$ pullup resistor.</td>
</tr>
<tr>
<td>Supply Current (Sensor only), Mouse Moving</td>
<td>$I_{DDS}$</td>
<td>4.5</td>
<td>8</td>
<td></td>
<td>mA</td>
<td>No load on $SW$, $XY_LED$, $ZA$, $ZB$, $D+$, $D-$</td>
</tr>
<tr>
<td>Supply Current (Sensor only), Mouse Not Moving</td>
<td>$I_{DDS_N}$</td>
<td>3.9</td>
<td>7.5</td>
<td></td>
<td>mA</td>
<td>No load on $SW$, $XY_LED$, $ZA$, $ZB$, $D+$, $D-$</td>
</tr>
<tr>
<td>Sensor Supply Current, USB Suspend Mode</td>
<td>$I_{DDS_S}$</td>
<td>320</td>
<td></td>
<td>$\mu$A</td>
<td></td>
<td>No load on $SW$, $XY_LED$, $ZA$, $ZB$, $D+$, $D-$</td>
</tr>
<tr>
<td>$XY_LED$ Current</td>
<td>$I_{LED}$</td>
<td>30</td>
<td></td>
<td></td>
<td>mA</td>
<td></td>
</tr>
<tr>
<td>$XY_LED$ Output Low Voltage</td>
<td>$V_{OL}$</td>
<td>1.1</td>
<td></td>
<td></td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Input Low Voltage</td>
<td>$V_{IL}$</td>
<td>0.5</td>
<td></td>
<td></td>
<td>V</td>
<td>Pins: $ZA$, $ZB$, $V_{IL\ max}\ of\ 0.5\ V_{DC}$ is at $V_{DDAS\ min}\ of\ 4\ V_{DC}$, with a typical of $0.8\ V_{DC}$ at $V_{DDAS}\ of\ 5\ V_{DC}$</td>
</tr>
<tr>
<td>Input High Voltage</td>
<td>$V_{IH}$</td>
<td>$0.6*V_{DDAS}$</td>
<td></td>
<td></td>
<td>V</td>
<td>Pins: $ZA$, $ZB$</td>
</tr>
</tbody>
</table>
One-Time-Programmable (OTP) Memory

The on chip OTP memory allows device configuration flexibility to override the default setting of sensors without any external software driver. Once the OTP operation is enabled, all OTP registers must be programmed accordingly as the default values of un-program OTP registers are always zero when L1_USE_OTP register setting is not zero value. Tips: OTP write to the OTP register can be skipped if the setting is zero value (0x00) in order to save the OTP programming time.

OTP address space is from 0xDF to 0xE8. OTP can be programmed via USB interface using Set Vendor Test and Get Vendor Test commands.

OTP Byte Write Operation

OTP write operation flow chart is shown in Figure 16.
1. Set OTP Clock enable bit in OTP_CLOCK register, 0x42: OTP_CLOCK_EN = 1.
2. Set OTP enable bit in OTP_CONFIG register, 0x51: OTP_EN = 1.
3. Write the OTP register address byte to OTP_ADDR register, 0x52.
4. Write the OTP data byte to OTP_DATA register, 0x53.
5. Set write enable bit in OTP_CTRL register, 0x54 to enable write command to OTP: WR = 1.
6. Read the write enable bit status in OTP_CTRL register, 0x54. If WR = 1, repeat reading the bit status until it is clear.
7. Read the write status bit in OTP_CTRLSTAT register, 0x58.
   a. If WR_OK = 1, OTP write operation is completed. Repeat Step 2 for more OTP byte write operations.
   b. If WR_OK = 0, repeat Step 5.
8. If Step 6b is repeated up to 10 times, OTP write operation is failed and the chip is confirmed as defective unit.

Figure 16. OTP Byte Write Flow Chart
**OTP Byte Read Operation**

OTP read operation flow chart is shown in Figure 17.

1. Set OTP Clock enable bit in OTP_CLOCK register, 0x42: OTP_CLOCK_EN = 1.
2. Set OTP enable bit in OTP_CONFIG register, 0x51: OTP_EN = 1.
3. Write the OTP register address byte to OTP_ADDR register, 0x52.
4. Set read enable bit in OTP_CTRL register, 0x54 to enable OTP read operation.
5. Read the read enable bit status in OTP_CTRL register, 0x54. If RD = 1, repeat reading the bit status until it is clear. Read the OTP data byte from OTP_DATA register, 0x53 to complete the OTP read operation.
6. Read the OTP data byte from OTP_DATA register, 0x53 to complete the OTP read operation.
7. Repeat Step 2 for more OTP read operations

**OTP Lock Operation**

OTP lock operation MUST be performed once OTP write to OTPLOCK1 register for the sensor to function. DO not reset or power up the chip right after OTP write to OTPLOCK1 register, otherwise the chip will be malfunction. The OTP lock operation flow chart is shown in Figure 18.

1. Set OTP Clock enable bit in OTP_CLOCK register, 0x42: OTP_CLOCK_EN = 1.
2. After OTP write to OTPLOCK1 register, set OTP enable bit in OTP_CONFIG register, 0x51: OTP_EN = 1.
3. Set OTP lock bit in OTP_CTRL register, 0x54 to enable OTP lock command: LOCK_L1 = 1.
4. Read the OTP lock bit status in OTP_CTRL register, 0x54. If LOCK_L1 = 1, repeat reading the bit status until it is clear.
5. Read the lock status and CRC bits in OTP_CTRLSTAT register, 0x58.
   a. If both L1_LOCK_OK and L1_CRC_OK = 1, OTP lock operation is completed.
   b. If either L1_LOCK_OK or L1_CRC_OK = 0, repeat Step 2 until both bits are set.
6. If Step 4b is repeated up to 10 times, OTP lock operation is failed and the chip is confirmed as defective unit.
7. Read the CRC result stored in register 0xE9, 0xEA, 0xEB, 0xEC, if four register values not 0x00 means CRC has been generated correctly and verified as lock operation success.

---

**Figure 17. OTP Byte Read Flow Chart**

**Figure 18. OTP Byte Lock Flow Chart**
Buttons

The minimum time between button pressed is $T_{DBB}$. The button connection is described in Figure 19.

![Button connections diagram]

Note: Use only 1% tolerance resistors

Debounce Algorithm

- Button inputs B1, B2, and B3 are sampled every 6ms.
- Two consecutive low values create a button press event.
- Three consecutive high values create a button release event.

Configuration after Power up (Data Values)

<table>
<thead>
<tr>
<th>Signal Function</th>
<th>State from Figure 9-1 of USB spec: Powered or Default Address or Configured</th>
<th>State from Figure 9-1 of USB spec: Suspended from Any Other State</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW</td>
<td>Output voltage at 1.16 V (Typ)</td>
<td>Output voltage at 2.7 V (Typ)</td>
</tr>
<tr>
<td>D-</td>
<td>USB I/O</td>
<td>Hi-Z Input</td>
</tr>
<tr>
<td>D+</td>
<td>USB I/O</td>
<td>Hi-Z Input</td>
</tr>
<tr>
<td>XY_LED</td>
<td>Always ON / Pulsing</td>
<td>Pulled HIGH (OFF)</td>
</tr>
<tr>
<td>ZB</td>
<td>Hi-Z Input</td>
<td>Output HIGH</td>
</tr>
<tr>
<td>ZA</td>
<td>Hi-Z Input</td>
<td>Output HIGH</td>
</tr>
</tbody>
</table>

Typical Performance Characteristics

Performance Characteristics over recommended operating conditions. Typical values at 25° C, $V_{DD} = 5.0$ V, 24 MHz

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
<th>Units</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path Error</td>
<td>$P_{Error}$</td>
<td>0.5</td>
<td>%</td>
<td></td>
<td></td>
<td>Average path error as percent of total 2.5&quot; travel on various standard surfaces</td>
</tr>
</tbody>
</table>

The following graphs are the typical performance of the sensor, assembled as shown in the 2D assembly drawing with trim lens, clip, and LED.
Figure 20. Typical Resolution vs. Z Height

Figure 21. Mean shutter vs Z height over white paper

Notes:
1. The sensor is designed for optimal performance when used with the specified LED.
2. Z = distance from Lens Reference Plane to Surface.
## USB Commands

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Command</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>USB_RESET</td>
<td>D+/D- low &gt; 18.7 μs</td>
<td>Device Resets; Address=0</td>
</tr>
<tr>
<td>USB_SUSPEND</td>
<td>Idle state &gt; 3 mS</td>
<td>Device enters USB low-power mode</td>
</tr>
<tr>
<td>USB_RESUME</td>
<td>Non-idle state</td>
<td>Device exits USB low-power mode</td>
</tr>
<tr>
<td>Get_Status_Device</td>
<td>80 00 00 00 00 02 00</td>
<td>Normally returns 00 00, Self powered 00 00, Remote wakeup 02 00</td>
</tr>
<tr>
<td>Get_Status_Interface</td>
<td>81 00 00 00 00 02 00</td>
<td>Normally returns 00 00</td>
</tr>
<tr>
<td>Get_Status_Endpt0</td>
<td>82 00 00 00 xx 00 02 00</td>
<td>OUT: xx=00, IN: xx=80</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Normally returns 00 00</td>
</tr>
<tr>
<td>Get_Status_Endpt1</td>
<td>82 00 00 00 81 00 02 00</td>
<td>Normally returns 00 00, Halt 00 01</td>
</tr>
<tr>
<td>Get_Configuration</td>
<td>80 08 00 00 00 01 00</td>
<td>Return: 00=not configured, 01=configured</td>
</tr>
<tr>
<td>Get_Interface</td>
<td>81 0A 00 00 00 00 01 00</td>
<td>Normally returns 00</td>
</tr>
<tr>
<td>Get_Protocol</td>
<td>A1 03 00 00 00 00 01 00</td>
<td>Normally returns 01, Boot protocol 00</td>
</tr>
<tr>
<td>Get_Desc_Device</td>
<td>80 06 00 01 00 00 nn 00</td>
<td>See USB command details</td>
</tr>
<tr>
<td>Get_Desc_Config</td>
<td>80 06 00 02 00 00 nn 00</td>
<td>See USB command details</td>
</tr>
<tr>
<td>Get_Desc_String</td>
<td>80 06 xx 03 00 00 nn 00</td>
<td>See USB command details</td>
</tr>
<tr>
<td>Get_Desc_HID</td>
<td>81 06 00 21 00 00 09 00</td>
<td>See USB command details</td>
</tr>
<tr>
<td>Get_Desc_HID_Report</td>
<td>81 06 00 22 00 00 nn 00</td>
<td>See USB command details</td>
</tr>
<tr>
<td>Get_HID_Input</td>
<td>A1 01 00 01 00 00 nn 00</td>
<td>Return depends on motion &amp; config</td>
</tr>
<tr>
<td>Get_Idle</td>
<td>A1 02 00 00 00 00 01 00</td>
<td>Returns rate in multiples of 4 ms</td>
</tr>
<tr>
<td>Get_Vendor_Test</td>
<td>C0 01 00 xx 00 01 00</td>
<td>Read register xx</td>
</tr>
<tr>
<td>Set_Address</td>
<td>00 05 xx 00 00 00 00 00</td>
<td>xx = address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Not configured: xx=00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Configured: xx=01</td>
</tr>
<tr>
<td>Set_Configuration</td>
<td>00 09 xx 00 00 00 00 00</td>
<td>Only one interface supported</td>
</tr>
<tr>
<td>Set_Interface</td>
<td>10 0B 00 00 00 00 00 00</td>
<td>Boot: xx=00, Report: xx=01</td>
</tr>
<tr>
<td>Set_Protocol</td>
<td>21 0B xx 00 00 00 00 00</td>
<td>Enable remote wakeup</td>
</tr>
<tr>
<td>Set_Feature_Device</td>
<td>00 03 01 00 00 00 00 00</td>
<td>Enable remote wakeup</td>
</tr>
<tr>
<td>Set_Feature_Endpt0</td>
<td>02 03 00 00 xx 00 00 00</td>
<td>Halt. OUT: xx=00, IN: xx=80</td>
</tr>
<tr>
<td>Set_Feature_Endpt1</td>
<td>02 03 00 00 81 00 00 00</td>
<td>Halt</td>
</tr>
<tr>
<td>Clear_Feature_Device</td>
<td>00 01 01 00 00 00 00 00</td>
<td>Disable Remote wakeup</td>
</tr>
<tr>
<td>Clear_Feature_Endpt0</td>
<td>02 01 00 00 xx 00 00 00</td>
<td>Clear Halt; OUT: xx=00, IN: xx=80</td>
</tr>
<tr>
<td>Clear_Feature_Endpt1</td>
<td>02 01 00 00 81 00 00 00</td>
<td>Clear Halt</td>
</tr>
<tr>
<td>Set_Idle</td>
<td>21 0A 00 rr 00 00 00 00</td>
<td>rr = report rate in multiples of 4 ms</td>
</tr>
<tr>
<td>Set_Vendor_Test</td>
<td>40 01 00 00 xx yy 00 00</td>
<td>Write yy to address xx</td>
</tr>
<tr>
<td>Poll_Endpt1</td>
<td></td>
<td>Read buttons, motion, &amp; Z-wheel</td>
</tr>
</tbody>
</table>

Note:
The last two bytes in a command shown as "nn 00" specify the 16-bit data size in the order of “LowByte HighByte.” For example a two-byte data size would be specified as '02 00.’ The sensor will not provide more bytes than the number requested in the command, but it will only supply up to a maximum of 8 bytes at a time. The sensor will re-send the last packet if the transfer is not acknowledged properly.
## USB Command Details

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>USB Spec:</th>
<th>Notes:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>USB_RESET</strong></td>
<td>D+/D- low for an extended period</td>
<td>A device may reset after seeing an SE0 for more than 18.7 uS, and definitely after 10 mS.</td>
<td>After power up and prior to Reset, the device will not respond to any USB commands. After the device has been given a USB Reset, the device's address will be reset to zero and the device will be in the Default state. The chip will default to Report protocol and any pending output will be flushed. All registers will be reset to a state that matches power-on-reset with the following exceptions: USB State register will be “Default” instead of “Attached”.</td>
</tr>
<tr>
<td><strong>USB_SUSPEND</strong></td>
<td>Idle state for an extended period</td>
<td>A device may suspend after seeing an idle for more than 3 mS, and definitely after 10 mS.</td>
<td>The chip will take a minimum of 5 mS to start Suspend, though will definitely start after 6 mS. The chip may finish the current frame if necessary before stopping the clock. Thus, an additional frame time may be used to reach Suspend mode.</td>
</tr>
<tr>
<td><strong>USB_RESUME</strong></td>
<td>Non-idle state</td>
<td>Remote Resume signalling from a device must be between 1 mS and 15 mS. The host is required to send Resume signaling for 20 mS plus 10 mS of resume recovery time in which it does not access any devices. This allows devices enough time to wake back up. The chip can cause a Resume if Remote Wakeup is enabled and a button has been pressed. Remote resume signalling from the chip will last 11.45 mS to 12.45 mS.</td>
<td></td>
</tr>
</tbody>
</table>

### Get_Status_Device

- **Returns:**
  - xx yy
  - xx[0] = Self Powered
  - xx[7:2] = 0
  - yy = 00 (Reserved)

- **Default:** Accept (undefined in USB Spec)
- **Addressed:** Accept
- **Configured:** Accept

- **Notes:** Use Set_Feature_Device/Clear_Feature_Device to set/clear remote wakeup.

### Get_Status_Interface

- **Returns:**
  - 00 00

- **Default:** Stall (undefined in USB Spec)
- **Addressed:** Stall
- **Configured:** Accept

- **Notes:** Both return bytes are reserved and currently 00.
<table>
<thead>
<tr>
<th>Method</th>
<th>Returns</th>
<th>Default</th>
<th>Addressed</th>
<th>Configured</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Status_Endpt0</td>
<td>82 00 00 00 xx 00 02 00&lt;br&gt;82 00 00 00 00 00 02 00&lt;br&gt;82 00 00 00 80 00 02 00&lt;br&gt;xx = 00 = Endpt0 OUT&lt;br&gt;xx = 80 = Endpt0 IN</td>
<td>Accept (undefined in USB Spec)</td>
<td>Accept</td>
<td>Accept</td>
<td>Use Set_Feature_Endpt0/Clear_Feature_Endpt0 to (try to) set/clear Halt bit. According to USB, “It is neither required or recommended that the Halt feature be implemented for the Default Control Pipe.” Since a new SETUP command will clear any Endpt0 halt bit, it is impossible to tell if there really is a halt bit.</td>
</tr>
<tr>
<td>Get_Status_Endpt1</td>
<td>82 00 00 00 81 00 02 00&lt;br&gt;xx yy&lt;br&gt;xx[0] = Halt&lt;br&gt;xx[7:1] = 0&lt;br&gt;yy = 00 (Reserved)</td>
<td>Stall (undefined in USB Spec)</td>
<td>Stall</td>
<td>Accept</td>
<td>Use Set_Feature_Endpt1/Clear_Feature_Endpt1 to set/clear Halt bit.</td>
</tr>
<tr>
<td>Get_Configuration</td>
<td>80 08 00 00 00 00 01 00&lt;br&gt;xx</td>
<td>Accept (undefined in USB Spec) – returns 00</td>
<td>Accept – returns 00</td>
<td>Accept – returns 01</td>
<td>Use Set_Configuration to change.</td>
</tr>
<tr>
<td>Get_Interface</td>
<td>81 0A 00 00 00 00 01 00</td>
<td>Stall (undefined in USB Spec)</td>
<td>Stall</td>
<td>Accept – returns 00</td>
<td>Command has no alternate interfaces, so only valid value is 00</td>
</tr>
<tr>
<td>Get_Protocol</td>
<td>A1 03 00 00 00 00 01 00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Returns:</td>
<td>xx</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>xx = 00 = Boot protocol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>xx = 01 = Report protocol</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default:</td>
<td>Accept</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Addressed:</td>
<td>Accept</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configured:</td>
<td>Accept</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notes:</td>
<td>Defaults to Report protocol after USB Reset. Use Set_Protocol to change.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Get_Desc_Device | 80 06 00 01 00 00 nn 00 |
| Returns: | 12 01 00 02 00 00 00 08 |
| vv vv pp pp dd dd mm PP ss 01 |
| Example: | vv vv = vendor id |
| pp pp = product id (vendor specified) |
| dd dd = device id (vendor specified) (bcd rev_id byte) |
| mm = iManufacturer |
| PP = iProduct |
| ss = iSerialNumber (00 – no string) |
| Example for Multi-button: | 12 01 00 02 00 00 00 08 |
| 6D 04 pp pp 00 54 01 02 |
| 00 01 |
| // Device Descriptor |
| | 12 // bLength (18 decimal) |
| | 01 // bDescriptorType |
| | 00 // bcdUSB (Release ##.## = 02.00) |
| | 02 |
| | 00 // bDeviceClass |
| | 00 // bDeviceSubClass |
| | 00 // bDeviceProtocol |
| | 08 // bMaxPacketSize0 |
| | 2F // idVendor |
| | 19 // idVendor |
| | 16 // idProduct // based on #buttons & wheel |
| | 09 // idProduct |
| | 00 // bcdDevice (Dev Rel 54.00) |
| | 54 |
| | 00 // iManufacturer |
| | 02 // iProduct |
| | 00 // iSerialNumber |
| | 01 // bNumConfigurations |
| Default: | Accept |
| Addressed: | Accept |
| Configured: | Accept |

Get_Desc_String will return “stall” if Manufacturer string is queried when iManufacturer = 0x00.
Get_Desc_Config

Returns:
- 80 06 00 02 00 00 nn 00
- 80 06 00 02 00 00 22 00
- 09 02 22 00 01 01 00 A0
- 32 09 04 00 00 01 03 01
- 02 00 09 21 11 01 00 01
- 22 rr 00 07 05 81 03 05
- 00 0A
  \(rr\) = HID Report descriptor length
  47 = 12 bit motion reporting

Default: Accept
Addressed: Accept
Configured: Accept
Notes:
- This is the concatenation of 4 descriptors:
  - Configuration
  - Interface
  - HID
  - Endpt

Get_Desc_String

Command Option:
- xx = 00 Language String
- xx = 01 Manufacturer String
- xx = 02 Product String

Returns:
- ss 03 "unicode string"
- ss = String descriptor length
- For xx = 00: 04 03 09 04 // Language ID
- For xx = 01: default: stall

Product String (xx=02)

<table>
<thead>
<tr>
<th>Product String</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>USB Optical Mouse</td>
<td>24 03 55 00 53 00 42 00 20 00 4f 00 70 00 74 00 69 00 63 00 61 00 6C 00 20 00 4D 00 6F 00 75 00 73 00 65 00</td>
</tr>
</tbody>
</table>

Synopsys cmd: No
Default: Accept
Addressed: Accept
Configured: Accept

Notes:
1. A request for any other string will STALL.
2. Returned string depends on the manufacturer string section via OTP.
### Get_Desc_HID

Returns:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>81 06 00 21 00 00 09 00</td>
<td>Returns HID Report descriptor length</td>
</tr>
</tbody>
</table>

\( r = \text{HID Report descriptor length} \)

40 = 12bit reporting

### Get_Desc_HID_Report

Returns:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>81 06 00 22 00 00 nn 00</td>
<td>Returns a report descriptor that describes how many buttons and x, y, z data.</td>
</tr>
</tbody>
</table>

12 bit reporting:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>05 01 09 02 A1 01 09 01</td>
<td>Default: Accept</td>
</tr>
<tr>
<td>A1 00 05 09 19 01 29 03</td>
<td>Addressed: Accept</td>
</tr>
<tr>
<td>15 00 25 01 75 01 95 03</td>
<td>Configured: Accept</td>
</tr>
<tr>
<td>81 02 75 05 95 01 81 01</td>
<td>This returns a report descriptor that describes how many buttons and x, y, z data.</td>
</tr>
<tr>
<td>05 01 09 30 09 31 16 01</td>
<td></td>
</tr>
<tr>
<td>F8 26 FF 07 75 0C 95 02</td>
<td></td>
</tr>
<tr>
<td>81 06 09 38 15 81 25 7F</td>
<td></td>
</tr>
<tr>
<td>75 08 95 01 81 06 C0 C0</td>
<td></td>
</tr>
</tbody>
</table>

```plaintext
// HID Report
| 05 01 // USAGE_PAGE (Generic Desktop) |
| 09 02 // USAGE (Mouse) |
| A1 01 // COLLECTION (Application) |
| 09 01 // USAGE (Pointer) |
| A1 00 // COLLECTION (Physical) |
| 05 09 // USAGE_PAGE (Button) |
| 19 01 // USAGE_MINIMUM (Button 1) |
| 29 03 // USAGE_MAXIMUM (Button #3) |
| 15 00 // LOGICAL_MINIMUM (0) |
| 25 01 // LOGICAL_MAXIMUM (1) |
| 75 01 // REPORT_SIZE (1) |
| 95 03 // REPORT_COUNT (3) |
| 81 02 // INPUT (Data,Var,Abs) |
| 75 05 // USAGE_PAGE |
| 95 01 // REPORT_COUNT(1) |
| 81 01 // INPUT (CNST,ARR,ABS) |
| 05 01 // USAGE_PAGE (Generic Desktop) |
| 09 30 // USAGE (X) |
| 09 31 // USAGE (Y) |
| 16 01 F8 // LOGICAL_MINIMUM (-127) |
| 26 FF 07 // LOGICAL_MAXIMUM (128) |
| 75 0C // REPORT_SIZE (12) |
| 95 02 // REPORT_COUNT (2) |
| 81 06 // INPUT (Data,Var,Rel) |
| 09 38 // USAGE (Zwheel) |
| 15 81 // LOGICAL_MINIMUM(-127) |
| 25 7F // LOGICAL_MAXIMUM(127) |
| 75 08 // REPORT_SIZE(8) |
| 95 01 // REPORT_COUNT(1) |
| 81 06 // INPUT(Data,Var,Rel) |
| C0 // END_COLLECTION |
| C0 // END_COLLECTION |
```

Default: Accept

Addressed: Accept

Configured: Accept

Notes: The length of this report is needed in the HID descriptor.
Get_HID_Input  A1 01 00 01 00 00 nn 00

Returns:  

bb xx yy zz
bb = button byte
xx = X motion byte
yy = XY motion byte
zz = Z motion byte

Default:  Stall
Addressed:  Stall
Configured:  Accept
Notes:  

If the device is configured, it will always respond with a report for this command, even if no motion or button changes have occurred. In this case, it would report 00 for motion and simply report the current button state. If a report is pending on endpt1, the data there will be reported and the report on endpt1 cleared.

The mouse will only create new button/motion packets when it is in the Configured state.

### USB Data Packet Format

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>B3(MB)</td>
<td>B2(RB)</td>
<td>B1(LB)</td>
</tr>
<tr>
<td>Byte 2</td>
<td>X7</td>
<td>X6</td>
<td>X5</td>
<td>X4</td>
<td>X3</td>
<td>X2</td>
<td>X1</td>
<td>X0</td>
</tr>
<tr>
<td>Byte 3</td>
<td>Y3</td>
<td>Y2</td>
<td>Y1</td>
<td>Y0</td>
<td>X11</td>
<td>X10</td>
<td>X9</td>
<td>X8</td>
</tr>
<tr>
<td>Byte 4</td>
<td>Y11</td>
<td>Y10</td>
<td>Y9</td>
<td>Y8</td>
<td>Y7</td>
<td>X6</td>
<td>X5</td>
<td>X4</td>
</tr>
<tr>
<td>Byte 5</td>
<td>Z7</td>
<td>Z6</td>
<td>Z5</td>
<td>Z4</td>
<td>Z3</td>
<td>Z2</td>
<td>Z1</td>
<td>Z0</td>
</tr>
<tr>
<td>Byte 6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Get_Idle  A1 02 00 00 00 00 01 00

Returns:  

rr
rr = rate in multiples of 4 mS

Default:  Accept
Addressed:  Accept
Configured:  Accept
Notes:  

The third byte of the command is to select the Report ID. There is only one for the mouse – so, using 00 or 01 will work. See also Set_Idle.

Get_Vendor_Test  C0 01 00 00 xx 00 01 00

Returns:  

rr (depends on register read)

Default:  Accept
Addressed:  Accept
Configured:  Accept
Notes:  

Address range (xx) is datasheet register range
<table>
<thead>
<tr>
<th>Command</th>
<th>Default</th>
<th>Addressed</th>
<th>Configured</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set_Address</strong></td>
<td><strong>Accept</strong></td>
<td><strong>Accept</strong></td>
<td><strong>Accept</strong> (undefined in USB Spec)</td>
<td>Chip gets new address, but stays in “Configured” mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>If device is not configured, the device will be given the new address and put in the addressed state (or default if new address = 00). If the device is already configured, the device will be given the new address state and remain configured.</td>
</tr>
<tr>
<td><strong>Set_Configuration</strong></td>
<td><strong>Accept (undefined in USB Spec)</strong></td>
<td><strong>Accept</strong></td>
<td><strong>Accept</strong></td>
<td><strong>Invalid config values will cause stall. Chip will stall invalid value in configured mode, and leave device in old (configured) mode.</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Set_Interface</strong></td>
<td>Stall (undefined in USB Spec)</td>
<td>Stall</td>
<td>Stall</td>
<td><strong>Mouse has only one valid interface (00) and alternate setting (00). Invalid values will cause stall. Chip retains previous (valid) interface state after executing this command in configured mode even if invalid values are given and command was stalled.</strong></td>
</tr>
<tr>
<td><strong>Set_Protocol</strong></td>
<td><strong>Accept (Not in USB Spec)</strong></td>
<td><strong>Accept</strong></td>
<td><strong>Accept</strong></td>
<td>3 byte data packets will be reported in boot mode. These bytes are button, XX data, and YY data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Set_Feature_Device</strong></td>
<td><strong>Accept (undefined in USB Spec)</strong></td>
<td><strong>Accept</strong></td>
<td><strong>Accept</strong></td>
<td>This sets the remote wakeup bit.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Set_Feature_Endpt0</strong></td>
<td>Stall (undefined in USB Spec)</td>
<td>Stall</td>
<td>Stall</td>
<td><strong>This (tries to) sets the halt bit. The chip always stalls the status stage for this command. The chip never reports the halt bit set for Endpt0 with the Get_Status_Endpt0 command, as any new SETUP command will clear Endpt0 stall.</strong></td>
</tr>
</tbody>
</table>

Set_Address: 00 05 xx 00 00 00 00 00
xx = new device address, from 00 to 7F

Set_Configuration: 00 09 xx 00 00 00 00 00
xx = 00 = not configured
xx = 01 = configured

Set_Interface: 01 0B 00 00 00 00 00 00

Set_Protocol: 21 0B xx 00 00 00 00 00
xx = 00 = Boot protocol
xx = 01 = Report protocol

Set_Feature_Device: 00 03 01 00 00 00 00 00

Set_Feature_Endpt0: 02 03 00 00 xx 00 00 00
02 03 00 00 00 00 00 00
02 03 00 00 80 00 00 00
xx = 00 = Endpt0 OUT
xx = 80 = Endpt0 IN
<table>
<thead>
<tr>
<th>Feature</th>
<th>Default</th>
<th>Addressed</th>
<th>Configured</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set_Feature_Endpt1</td>
<td>02 03 00 00 81 00 00 00</td>
<td>Stall (undefined in USB Spec)</td>
<td>Stall</td>
<td>Accept (undefined in USB Spec)</td>
</tr>
<tr>
<td>Clear_Feature_Device</td>
<td>00 01 01 00 00 00 00 00</td>
<td>Accept</td>
<td>Accept</td>
<td>This clears the remote wakeup bit.</td>
</tr>
<tr>
<td>Clear_Feature_Endpt0</td>
<td>02 01 00 00 xx 00 00 00 00</td>
<td>Stall</td>
<td>Accept</td>
<td>The chip does NOT stall like it does for Set_Feature_Endpt0.</td>
</tr>
<tr>
<td>Clear_Feature_Endpt1</td>
<td>02 01 00 00 81 00 00 00</td>
<td>Stall</td>
<td>Accept</td>
<td>See Set_Feature_Endpt1.</td>
</tr>
<tr>
<td>Set_Idle</td>
<td>21 0A 00 rr 00 00 00 00</td>
<td>Accept</td>
<td>Accept</td>
<td>The third byte of the command is to select the Report ID. There is only one for the mouse – so, using either 00 or 01 will work.</td>
</tr>
<tr>
<td>Set_Vendor_Test</td>
<td>40 01 00 00 xx yy 00 00</td>
<td>Accept</td>
<td>Accept</td>
<td>Address range for “xx” should be 0x00 to 0x3F. Addresses above this are reserved for possible future use. See also Get_Vendor_Test.</td>
</tr>
</tbody>
</table>
Poll_Endpt1

Returns:

- bb xx yy zz
- bb = button byte
- xx = X motion byte
- yy = Y motion byte
- zz = Z motion byte (if Z-Wheel)

Default:
- Ignore request

Addressed:
- Ignore request

Configured:
- Accept (NAK if no data; Send packet if available)

Notes:

See also Get_HID_Input. Endpt will only stall if halt bit is set by Set_Feature_Endpt1. Details of data packet are below.

Endpt1 should be polled at least every 10 frames (ms). It is typically polled every 8 frames on Windows machines. For internal testing, Endpt1 can be continuously polled if desired.

The chip will not generate any report packets unless in the Configured state.

If Endpt1 is currently empty, any motion or button change will be loaded into the Endpt1 buffers. Once the Endpt1 buffers are full, any further motion events will get accumulated. When the Endpt1 buffers are later polled and emptied, the current accumulated X/Y/Z values will be loaded into the Endpt1 buffers. After transferring their data, the accumulation registers are reset so they are ready to start accumulating new motion events.

Button information is handled a bit differently. If the Endpt1 buffers are empty, and a button change event occurs, the new button state is put into the Endpt1 buffers. At the same time, the button state that is put in Endpt1 is copied for later use. While Endpt1 is full, changes in button state are essentially ignored. When Endpt1 is emptied, if the current button state is different than that which was last loaded into Endpt1, then the new state will be loaded and a new copy saved. Basically, the button state that is loaded into Endpt1 is always the current button state at that point in time. It should also be noted that there is hardware on the chip to help de-bounce the buttons.

Special note on wLength: The wLength parameter in commands specifies the maximum number of bytes a device should send back. The commands listed below are not able to handle a wLength of 0 correctly.

- Get_Status_Device
- Get_Status_Interface
- Get_Status_Endpt0
- Get_Status_Endpt1
- Get_Configuration
- Get_Interface

This chip will send one byte of data rather than none when wLength = 0 is requested for the above commands.
The sensor can be programmed through registers, via the USB port, and configuration and motion data can be read from these registers. The registers will be “disabled” by $V_{DDAS}$ going low or sending a USB reset command.

<table>
<thead>
<tr>
<th>Address</th>
<th>Register Name</th>
<th>Register Type</th>
<th>Access</th>
<th>Reset Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>PROD_ID</td>
<td>Device</td>
<td>Read only</td>
<td>0x2b</td>
</tr>
<tr>
<td>0x01</td>
<td>REV_ID</td>
<td>Device</td>
<td>Read only</td>
<td>0x01</td>
</tr>
<tr>
<td>0x02</td>
<td>MOUSE_STAT</td>
<td>Device</td>
<td>Read only</td>
<td>Undefined</td>
</tr>
<tr>
<td>0x03</td>
<td>DELTA_X_L</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x04</td>
<td>DELTA_Y_L</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x05</td>
<td>DELTA_XY_H</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x06</td>
<td>DZ</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x07</td>
<td>SQUAL</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x08</td>
<td>SHUT_HI</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x09</td>
<td>SHUT_LO</td>
<td>Device</td>
<td>Read only</td>
<td>0x32</td>
</tr>
<tr>
<td>0x0A</td>
<td>PIX_MAX</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x0B</td>
<td>PIX_ACCUM</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x0C</td>
<td>PIX_MIN</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x0D</td>
<td>PIX_GRAB</td>
<td>Device</td>
<td>Read only</td>
<td>0x00</td>
</tr>
<tr>
<td>0x40</td>
<td>INV_REV_ID</td>
<td>Device</td>
<td>Read only</td>
<td>0xFE</td>
</tr>
<tr>
<td>0x42</td>
<td>OTP_CLOCK</td>
<td>Device</td>
<td>Read/Write</td>
<td>0x00</td>
</tr>
<tr>
<td>0x51</td>
<td>OTP_CONFIG</td>
<td>Device</td>
<td>Read/Write</td>
<td>0x00</td>
</tr>
<tr>
<td>0x52</td>
<td>OTP_ADDR</td>
<td>Device</td>
<td>Read/Write</td>
<td>0x00</td>
</tr>
<tr>
<td>0x53</td>
<td>OTP_DATA</td>
<td>Device</td>
<td>Read/Write</td>
<td>0x00</td>
</tr>
<tr>
<td>0x54</td>
<td>OTP_CTRL</td>
<td>Device</td>
<td>Read/Write</td>
<td>0x00</td>
</tr>
<tr>
<td>0x56</td>
<td>OTP_RUNSTAT</td>
<td>Device</td>
<td>Read only</td>
<td>Undefined</td>
</tr>
<tr>
<td>0x58</td>
<td>OTP_CTRLSTAT</td>
<td>Device</td>
<td>Read only</td>
<td>Undefined</td>
</tr>
<tr>
<td>0xDF</td>
<td>L1_USE_OTP</td>
<td>OTP</td>
<td>Read/Write</td>
<td>0x00</td>
</tr>
<tr>
<td>0xE0</td>
<td>DPI</td>
<td>OTP</td>
<td>Read/Write</td>
<td>0x04</td>
</tr>
<tr>
<td>0xE2 : 0xE3</td>
<td>PID1 : PID0</td>
<td>OTP</td>
<td>Read/Write</td>
<td>0x0916</td>
</tr>
<tr>
<td>0xE4 : 0xE5</td>
<td>VID1 : VID0</td>
<td>OTP</td>
<td>Read/Write</td>
<td>0x192F</td>
</tr>
<tr>
<td>0xE8</td>
<td>OTPLOCK1</td>
<td>OTP</td>
<td>Read/Write</td>
<td>0x00</td>
</tr>
</tbody>
</table>
**PROD_ID**

Address: 0x00  
Type: Device  
Access: Read only  
Reset Value: 0x2b

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>PID7</td>
<td>PID6</td>
<td>PID5</td>
<td>PID4</td>
<td>PID3</td>
<td>PID2</td>
<td>PID1</td>
<td>PID0</td>
</tr>
</tbody>
</table>

Data Type: 8-bit number with the product identifier.

**USAGE:** The value in this register does not change; it can be used to verify that the sensor communications link is OK.

---

**REV_ID**

Address: 0x01  
Type: Device  
Access: Read only  
Reset Value: 0x01

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>RID7</td>
<td>RID6</td>
<td>RID5</td>
<td>RID4</td>
<td>RID3</td>
<td>RID2</td>
<td>RID1</td>
<td>RID0</td>
</tr>
</tbody>
</table>

Data Type: 8-bit number with current revision of the IC.

**USAGE:** This register contains the IC revision. It is subject to change when new IC versions are released.

---

**BUT_STAT**

Address: 0x02  
Type: Device  
Access: Read only  
Reset Value: Undefined

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Reserved</td>
<td>Reserved</td>
<td>Reserved</td>
<td>Reserved</td>
<td>Reserved</td>
<td>BUT3</td>
<td>BUT2</td>
<td>BUT1</td>
</tr>
</tbody>
</table>

Data Type: Bit field.

**USAGE:** This register is included for test purposes only. For navigation use, use the USB HID defined commands. The button status bits reported are for the debounce signals.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| BUT3       | Reports the status of B3  
0 = Open  
1 = Button Pressed |
| BUT2       | Reports the status of B2  
0 = Open  
1 = Button Pressed |
| BUT1       | Reports the status of B1  
0 = Open  
1 = Button Pressed |
DELTA_X_L  
Address: 0x03  
Access: Read  
Reset Value: 0x00  

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>X7</th>
<th>X6</th>
<th>X5</th>
<th>X4</th>
<th>X3</th>
<th>X2</th>
<th>X1</th>
<th>X0</th>
</tr>
</thead>
</table>

Data Type: Bit field.

 USAGE: The value in this register reflects the last USB delta X (lower 8 bits) data output or data queued for output. This register is included for test purposes only. For navigation use, use the HID defined commands. Data is 2’s complement. Absolute value is determined by the currently set resolution.

Register 0x03 must be read before register 0x04 (Delta_Y_L) and 0x05 (Delta_XY_H)

DELTA_Y_L  
Address: 0x04  
Access: Read  
Reset Value: 0x00  

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Y7</th>
<th>Y6</th>
<th>Y5</th>
<th>Y4</th>
<th>Y3</th>
<th>Y2</th>
<th>Y1</th>
<th>Y0</th>
</tr>
</thead>
</table>

Data Type: Bit field.

 USAGE: The value in this register reflects the last USB delta Y (lower 8 bits) data output or data queued for output. This register is included for test purposes only. Register 0x03 should be read before register 0x04 (Delta_Y_L) and 0x05 (Delta_XY_H), else Delta_Y_L will return 0. For navigation use, use the HID defined commands. Data is 2’s complement. Absolute value is determined by the currently set resolution.

DELTA_XY_H  
Address: 0x05  
Access: Read  
Reset Value: 0x00  

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>X11</th>
<th>X10</th>
<th>X9</th>
<th>X8</th>
<th>Y11</th>
<th>Y10</th>
<th>Y9</th>
<th>Y8</th>
</tr>
</thead>
</table>

Data Type: Bit field.

 USAGE: The value in this register reflects the last USB delta X and Y (upper 4 bits) data output or data queued for output. This register is included for test purposes only. Register 0x03 should be read before register 0x04 (Delta_Y_L) and 0x05 (Delta_XY_H), else Delta_XY_H will return 0. For navigation use, use the HID defined commands. Data is 2’s complement. Absolute value is determined by the currently set resolution.

DZ  
Address: 0x06  
Access: Read  
Reset Value: 0x00  

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Z7</th>
<th>Z6</th>
<th>Z5</th>
<th>Z4</th>
<th>Z3</th>
<th>Z2</th>
<th>Z1</th>
<th>Z0</th>
</tr>
</thead>
</table>

Data Type: Bit field.

 USAGE: This register contains the Z-wheel count. Range is from -127 to 127 decimal.
### SQUAL

**Address:** 0x07  
**Access:** Read  
**Reset Value:** 0x00

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>SQ_7</td>
<td>SQ_6</td>
<td>SQ_5</td>
<td>SQ_4</td>
<td>SQ_3</td>
<td>SQ_2</td>
<td>SQ_1</td>
<td>SQ_0</td>
</tr>
</tbody>
</table>

**Data Type:** Eight bit number.  
**Usage:** SQUAL is a measure of the number of features visible by the sensor in the current frame. The maximum value is 128. Since small changes in the current frame can result in changes in SQUAL, slight variations in SQUAL on one surface is expected.

### SHUT_HI

**Address:** 0x08  
**Type:** Device  
**Access:** Read only  
**Reset Value:** 0x00

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>S_15</td>
<td>S_14</td>
<td>S_13</td>
<td>S_12</td>
<td>S_11</td>
<td>S_10</td>
<td>S_9</td>
<td>S_8</td>
</tr>
</tbody>
</table>

### SHUT_LO

**Address:** 0x09  
**Type:** Device  
**Access:** Read only  
**Reset Value:** 0x64

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>S_7</td>
<td>S_6</td>
<td>S_5</td>
<td>S_4</td>
<td>S_3</td>
<td>S_2</td>
<td>S_1</td>
<td>S_0</td>
</tr>
</tbody>
</table>

**Data Type:** 16-bit number.  
**Usage:** The combination of SHUT_HI and SHUT_LO is a 16-bit number. This is the number of clocks the shutter was open for the last image taken. The unit is in main clock count (nominally 12MHz). To avoid split read issues, read SHUT_HI first.

### PIX_MAX

**Address:** 0x0A  
**Type:** Device  
**Access:** Read only  
**Reset Value:** 0x00

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>0</td>
<td>MX_6</td>
<td>MX_5</td>
<td>MX_4</td>
<td>MX_3</td>
<td>MX_2</td>
<td>MX_1</td>
<td>MX_0</td>
</tr>
</tbody>
</table>

**Data Type:** Eight bit number.  
**Usage:** This is the maximum pixel value from the last image taken.
**PIX_ACCUM**  
Address: 0x0B  
Type: Device  
Access: Read only  
Reset Value: 0x00  

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AC₇</td>
<td>AC₆</td>
<td>AC₅</td>
<td>AC₄</td>
<td>AC₃</td>
<td>AC₂</td>
<td>AC₁</td>
<td>AC₀</td>
</tr>
</tbody>
</table>

Data Type: High 8bits of 17-bit unsigned integer.  

**USAGE:** This is the accumulated pixel value from the last image taken. For the 19x19 pixels, only the 8 most prominent bits are reported ([15:8]). To get the true average pixel value, divide this register value by 1.41.

**PIX_MIN**  
Address: 0x0C  
Type: Device  
Access: Read only  
Reset Value: 0x00  

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>MN₆</td>
<td>MN₅</td>
<td>MN₄</td>
<td>MN₃</td>
<td>MN₂</td>
<td>MN₁</td>
<td>MN₀</td>
</tr>
</tbody>
</table>

Data Type: 8-bit number.  

**USAGE:** This is the minimum pixel value from the last image taken.
**PIX_GRAB**

Address: 0x0D

Type: Device

Access: Read/Write

Reset Value: 0x00

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALID</td>
<td>PG6</td>
<td>PG5</td>
<td>PG4</td>
<td>PG3</td>
<td>PG2</td>
<td>PG1</td>
<td>PG0</td>
<td></td>
</tr>
</tbody>
</table>

Data Type: 8-bit number.

**USAGE:** The pixel grabber captures 1 pixel per frame. If there is a valid pixel in the grabber when this is read, the MSB will be set, an internal counter will incremented to captured the next pixel and the grabber will be armed to capture the next pixel. It will take 361 reads to upload the completed image.

Any write to this register will reset and arm the grabber to grab pixel 0 on the next image. See pixel array numbering in Figure 20.

---

**INV_REV_ID**

Address: 0x40

Access: Read

Reset Value: 0xfe

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRID7</td>
<td>RRID6</td>
<td>RRID5</td>
<td>RRID4</td>
<td>RRID3</td>
<td>RRID2</td>
<td>RRID1</td>
<td>RRID0</td>
<td></td>
</tr>
</tbody>
</table>

Data Type: Eight bit number.

**USAGE:** Contains the inverse of the revision ID which is located in register 0x01.

---

**Figure 20. Pixel Map. Sensor looking at the navigation surface through the lens from top of mouse.**
### OTP_CONFIG
Address: 0x51
Type: Device
Access: Read/Write
Reset Value: 0x00

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>OTP_EN</td>
<td>Reserved</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>3</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>4</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>5</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>6</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
<tr>
<td>7</td>
<td>Reserved</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

Data Type: Bit field.

**USAGE:** OTP commands enable/disable. Refer to OTP programming section.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OTP_EN</td>
<td>OTP commands</td>
</tr>
<tr>
<td></td>
<td>1 = Enabled</td>
</tr>
<tr>
<td></td>
<td>0 = Disabled</td>
</tr>
</tbody>
</table>

### OTP_ADDR
Address: 0x52
Type: Device
Access: Read/Write
Reset Value: 0x00

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>OTP_ADDR7</td>
</tr>
<tr>
<td>1</td>
<td>OTP_ADDR6</td>
</tr>
<tr>
<td>2</td>
<td>OTP_ADDR5</td>
</tr>
<tr>
<td>3</td>
<td>OTP_ADDR4</td>
</tr>
<tr>
<td>4</td>
<td>OTP_ADDR3</td>
</tr>
<tr>
<td>5</td>
<td>OTP_ADDR2</td>
</tr>
<tr>
<td>6</td>
<td>OTP_ADDR1</td>
</tr>
<tr>
<td>7</td>
<td>OTP_ADDR0</td>
</tr>
</tbody>
</table>

Data Type: 8-bit number.

**USAGE:** This register is the container of OTP address in OTP read/write command. Refer to OTP programming section.

### OTP_DATA
Address: 0x53
Type: Device
Access: Read/Write
Reset Value: 0x00

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>OTP_DATA7</td>
</tr>
<tr>
<td>1</td>
<td>OTP_DATA6</td>
</tr>
<tr>
<td>2</td>
<td>OTP_DATA5</td>
</tr>
<tr>
<td>3</td>
<td>OTP_DATA4</td>
</tr>
<tr>
<td>4</td>
<td>OTP_DATA3</td>
</tr>
<tr>
<td>5</td>
<td>OTP_DATA2</td>
</tr>
<tr>
<td>6</td>
<td>OTP_DATA1</td>
</tr>
<tr>
<td>7</td>
<td>OTP_DATA0</td>
</tr>
</tbody>
</table>

Data Type: 8-bit number.

**USAGE:** This register is the container of OTP data value in OTP read/write command. Refer to OTP programming section.
### OTP_CRTL

**Address:** 0x54  
**Type:** Device  
**Access:** Read/Write  
**Reset Value:** 0x00

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOCK_L1</td>
<td>Enable write command to OTP</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
<td>1 = Write to OTP</td>
</tr>
<tr>
<td>0</td>
<td>RD</td>
<td>0 = Write command is completed</td>
</tr>
<tr>
<td></td>
<td>WR</td>
<td></td>
</tr>
</tbody>
</table>

**Data Type:** Bit field.

**USAGE:** This register controls the read, write and lock commands of OTP. The commands are auto clear for status check. Refer to OTP programming section.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WR</td>
<td>Enable write command to OTP</td>
</tr>
<tr>
<td></td>
<td>1 = Write to OTP</td>
</tr>
<tr>
<td></td>
<td>0 = Write command is completed</td>
</tr>
<tr>
<td>RD</td>
<td>Enable read command to OTP</td>
</tr>
<tr>
<td></td>
<td>1 = Read from OTP</td>
</tr>
<tr>
<td></td>
<td>0 = Data is ready to be read from OTP_DATA register</td>
</tr>
<tr>
<td>LOCK_L1</td>
<td>Enable OTP lock command</td>
</tr>
<tr>
<td></td>
<td>1 = Lock OTP space</td>
</tr>
<tr>
<td></td>
<td>0 = Lock command is completed</td>
</tr>
</tbody>
</table>

### OTP_RUNSTAT

**Address:** 0x56  
**Type:** Device  
**Access:** Read only  
**Reset Value:** Undefined

<table>
<thead>
<tr>
<th>Bit</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>L1_CHECKED</td>
<td>OTP space status</td>
</tr>
<tr>
<td>1</td>
<td>L1_LOCKED</td>
<td>1 = Used</td>
</tr>
<tr>
<td>0</td>
<td>L1_USED</td>
<td>0 = Unused</td>
</tr>
</tbody>
</table>

**Data Type:** Bit field.

**USAGE:** This register shows the OTP run status. Refer to OTP programming section.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1_USED</td>
<td>OTP space status</td>
</tr>
<tr>
<td></td>
<td>1 = Used</td>
</tr>
<tr>
<td></td>
<td>0 = Unused</td>
</tr>
<tr>
<td>L1_LOCKED</td>
<td>OTP space locking status</td>
</tr>
<tr>
<td></td>
<td>1 = Locked</td>
</tr>
<tr>
<td></td>
<td>0 = Open</td>
</tr>
<tr>
<td>L1_CHECKED</td>
<td>OTP status check</td>
</tr>
<tr>
<td></td>
<td>1 = Checked</td>
</tr>
<tr>
<td></td>
<td>0 = Unchecked</td>
</tr>
</tbody>
</table>
**OTP_CTRLSTAT**  
Address: 0x58  
Type: Device  
Access: Read only  
Reset Value: Undefined

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Reserved</td>
</tr>
<tr>
<td>6</td>
<td>Reserved</td>
</tr>
<tr>
<td>5</td>
<td>L1_CRC_OK</td>
</tr>
<tr>
<td>4</td>
<td>L1_LOCK_OK</td>
</tr>
<tr>
<td>3</td>
<td>DEVICE_DONE</td>
</tr>
<tr>
<td>2</td>
<td>DEVICE_RDY</td>
</tr>
<tr>
<td>1</td>
<td>WR_DENIED</td>
</tr>
<tr>
<td>0</td>
<td>WR_OK</td>
</tr>
</tbody>
</table>

Data Type: Bit field.

**Usage:** This register shows the OTP control status. Refer to OTP programming section.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| WR_OK      | OTP write status  
  1 = OK  
  0 = Failed |
| WR_DENIED  | OTP write access status  
  1 = Denied  
  0 = OK |
| L1_LOCK_OK | OTP lock status  
  1 = OK  
  0 = Failed |
| L1_CRC_OK  | CRC test status  
  1 = OK  
  0 = Failed |

**L1_USE_OTP**  
Address: 0xDF  
Type: OTP  
Access: Read/Write  
Reset Value: 0x00

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>L1_USE_OTP7</td>
</tr>
<tr>
<td>6</td>
<td>L1_USE_OTP6</td>
</tr>
<tr>
<td>5</td>
<td>L1_USE_OTP5</td>
</tr>
<tr>
<td>4</td>
<td>L1_USE_OTP4</td>
</tr>
<tr>
<td>3</td>
<td>L1_USE_OTP3</td>
</tr>
<tr>
<td>2</td>
<td>L1_USE_OTP2</td>
</tr>
<tr>
<td>1</td>
<td>L1_USE_OTP1</td>
</tr>
<tr>
<td>0</td>
<td>L1_USE_OTP0</td>
</tr>
</tbody>
</table>

Data Type: 8-bit field.

**Usage:** Bypass OTP configuration if all bits are zero. MUST write non-zero value to this register to enable OTP operation. Once enabled, all OTP registers must be written as the default values are zero value.
DPI
Address: 0xE0 Type: OTP
Access: Read/Write Reset Value: 0x54

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SwapXY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INV_X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INV_Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPI2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPI1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPI0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data Type: Bit field.

**Usage:** These registers are used to customize the device’s DPI.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| SwapXY     | To swap x and y  
0 = no swap  
1 = swap |
| INV_X      | To invert x direction  
0 = not inverted  
1 = inverted |
| INV_Y      | To invert y direction  
0 = not inverted  
1 = inverted |
| DPI        | Device status  
010 = 500 dpi  
011 = 750 dpi  
100 = 1000 dpi (default)  
101 = 1250 dpi |

Note: Sensor orientation setting by changing bit 6, 5 and 4. The SwapXY operation is always performed before INV_X and INV_Y inversion operations.

<table>
<thead>
<tr>
<th>Rotation</th>
<th>0°</th>
<th>90°</th>
<th>180°</th>
<th>270°</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit[6:4]</td>
<td>101</td>
<td>000</td>
<td>110</td>
<td>011</td>
</tr>
</tbody>
</table>

Motion Reporting Direction

<table>
<thead>
<tr>
<th>Motion Reporting Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>A2700 XYWWWZL</td>
</tr>
<tr>
<td>+X</td>
</tr>
<tr>
<td>+X</td>
</tr>
<tr>
<td>+X</td>
</tr>
</tbody>
</table>
## VID0

**Address:** 0xE2  
**Type:** OTP  
**Access:** Read/Write  
**Reset Value:** 0x16

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>VID7</td>
<td>VID6</td>
<td>VID5</td>
<td>VID4</td>
<td>VID3</td>
<td>VID2</td>
<td>VID1</td>
<td>VID0</td>
</tr>
</tbody>
</table>

**PID1**

**Address:** 0xE3  
**Type:** OTP  
**Access:** Read/Write  
**Reset Value:** 0x09

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PID15</td>
<td>PID14</td>
<td>PID13</td>
<td>PID12</td>
<td>PID11</td>
<td>PID10</td>
<td>PID9</td>
<td>PID8</td>
</tr>
</tbody>
</table>

Data Type: 16-Bit number.  
**USAGE:** These registers are used to customize device VID. **Default is Avago’s VID = 0x192F.**

## VID1

**Address:** 0xE4  
**Type:** OTP  
**Access:** Read/Write  
**Reset Value:** 0x2F

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>VID15</td>
<td>VID14</td>
<td>VID13</td>
<td>VID12</td>
<td>VID11</td>
<td>VID10</td>
<td>VID9</td>
<td>VID8</td>
</tr>
</tbody>
</table>

## OTPLOCK1

**Address:** 0xE8  
**Type:** OTP  
**Access:** Read/Write  
**Reset Value:** 0x00

<table>
<thead>
<tr>
<th>Bit Field</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OTPLOCK17</td>
<td>OTPLOCK16</td>
<td>OTPLOCK15</td>
<td>OTPLOCK14</td>
<td>OTPLOCK13</td>
<td>OTPLOCK12</td>
<td>OTPLOCK11</td>
<td>OTPLOCK10</td>
</tr>
</tbody>
</table>

Data Type: 8-bit field.  
**USAGE:** Must write 0xFF in this register to lock the OTP configuration.

---

For product information and a complete list of distributors, please go to our web site:  
www.avagotech.com

Avago, Avago Technologies, and the A logo are trademarks of Avago Technologies in the United States and other countries.  
Data subject to change.  
Copyright © 2005-2012 Avago Technologies. All rights reserved.  
AV02-2723EN - April 17, 2012