# PAW3204LU-TJDU
# SMD-TYPED ULTRA LOW POWER WIRELESS MOUSE SENSOR

## General Description

The PAW3204LU-TJDU is a SMD-typed high performance and ultra low power CMOS process optical mouse sensor with DSP integration chip that serves as a non-mechanical motion estimation engine for implementing a computer wireless mouse. With adaptive frame rate control, AKA AFC, this optical mouse sensor gains extra power saving during mouse moving.

## Features

- **Single power supply**
- **Precise optical motion estimation technology**
- **Complete 2-D motion sensor**
- **Accurate motion estimation over a wide range of surfaces**
- **High speed motion detection up to 30 inches/sec**
- **High resolution up to 1500 CPI**
- **Power saving mode during times of no movement**
- **Serial interface for programming and data transfer**
- **Built-in low power Timer (LPT) for sleep1/sleep2 mode**
- **MOTSWK pin to wake up mouse controller**
- **Wide operation range from 2.1V to 3.6V**
- **Low power operation under 1.98V**
- **Adaptive frame rate control for extra power saving during moving**
- **LED drive mode configuration**

## Key Specification

| | |
|---|---|
| **Power Supply** | **Operating voltage**<br>**1.73V ~ 1.98V (VDD and VDDA short)**<br>**2.1V ~ 3.6V (VDD)** |
| **Optical Lens** | **1:0.8** |
| **Speed** | **Up to 30 inches/sec** |
| **Acceleration** | **Up to 10 G** |
| **Resolution** | **500/ 650/ 750(Default)/ 1000/ 1300 /1500 CPI** |
| **Frame Rate** | **Up to 2400 frames/sec** |
| **Typical Operating Current**<br>**(without I/O toggling)** | **1.6 mA @ Mouse moving (Normal1)**<br>**1.2 mA @ Mouse moving (Normal2)**<br>**1.0 mA @ Mouse moving (Normal3)**<br>**70 uA @ Mouse not moving (Sleep1)**<br>**10 uA @ Mouse not moving (Sleep2)**<br>**9 uA @ Power down mode**<br>*including LED, typical value |
| **Package** | **SMD, 8 balls** |

## Ordering Information

| Order Number | Bundle Part Number | Part Description |
|---|---|---|
| PAW3204LU-TJDU | PAW3204LU-TJDU | SMD CMOS Optical Mouse Sensor |
| | PNSR-015-RB1 | Bundle Lens for Infrared LED |



**PixArt Imaging Inc.**
**E-mail:** fae_service@pixart.com.tw

# 1. Pin Configuration

## 1.1 Pin Description

| Pin | Name | Type | Definition |
|---|---|---|---|
| A1 | VDDA | PWR | Analog/Digital supply voltage (1.9V)<br>Power supply (1.73V~1.98V) for low power operation voltage |
| C1 | VSS | GND | Chip ground |
| E1 | LED | OUT | LED control |
| B2 | VDD | PWR | Power supply (2.1V~3.6V) for internal power regulator, VDDA (1.9V) is the power regulator output.<br>Power supply (1.73V~1.98V) for low power operation voltage |
| D2 | SDIO | I/O | Serial interface bi-direction data *Initial with input floating |
| A3 | ATPG_EN | IN | Reserved. Connect to ground |
| C3 | MOTSWK | OUT | Motion detect (active low, see Section7) *Initial with output high |
| E3 | SCLK | IN | Serial interface clock *Initial with input floating |

## 1.2 Pin Assignment

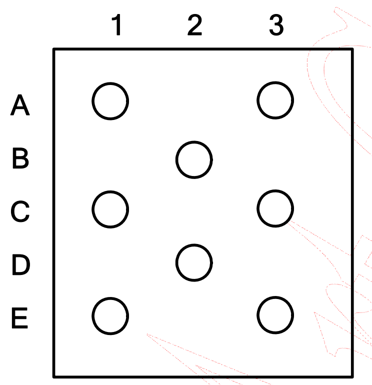



Figure 1. Top View Pinout

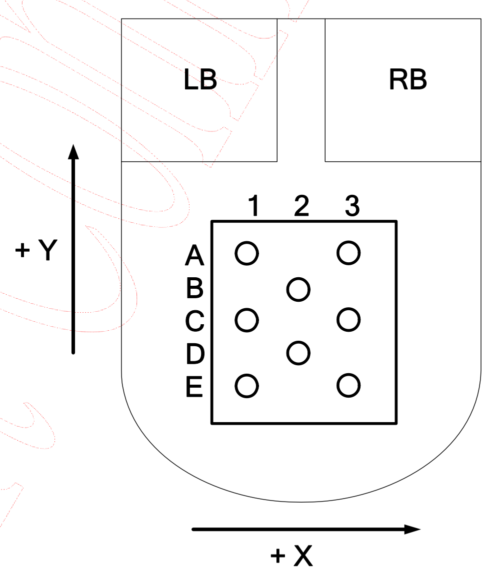



Figure 2. Top View of Mouse



PixArt Imaging Inc.
E-mail: fae_service@pixart.com.tw

## 2. Block Diagram and Operation

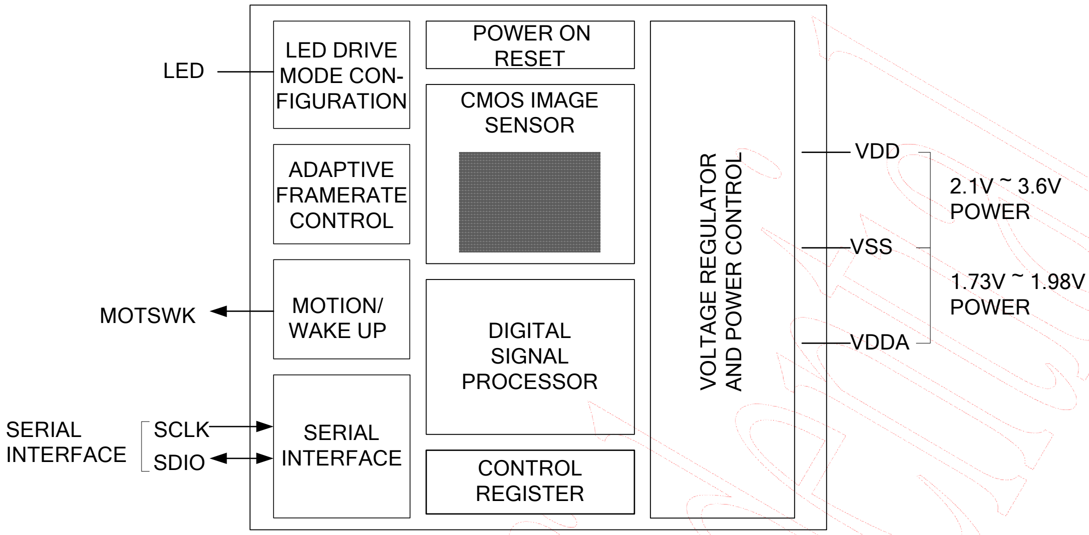



**Figure 3. Block Diagram**

The PAW3204LU-TJDU is a SMD-typed high performance and ultra low power CMOS-process optical mouse sensor with DSP integration chip that serves as a non-mechanical motion estimation engine for implementing a wireless computer mouse. It is based on new optical navigation technology, which measures changes in position by optically acquiring sequential surface images (frames) and mathematically determining the direction and magnitude of movement. The mouse sensor is in an 8-ball SMD package. The current X and Y information are available in registers accessed via a serial port. The word "mouse sensor", instead of PAW3204LU-TJDU, is used in the document.

With LED control technology, adaptive frame rate control (also known as AFC), the mouse sensor gain extra power saving during mouse moving. The AFC servers position/speed detection and then mapping to different frame rate. With lower frame rate, it leads to lower power consumption of the mouse sensor and LED. The mouse sensor is featured with THREE-level AFC which is 2400/1200/800 frame per second.

A brand-new configurable LED drive mode switch function provides flexible choice for the user to achieve different LED application. The mouse sensor provides TWO choices with Current DAC mode, and Current Switch modes. **Note that default drive is Current DAC mode**.

## 3. Registers and Operation

The mouse sensor can be programmed through registers via the serial port. Also, the DSP configuration and motion data can be read from these registers. All registers not listed are reserved, and should never be written by firmware.

### 3.1 Registers

| Address | Name | R/W | Default | Data Type |
|---|---|---|---|---|
| 0x00 | Product_ID1 | R | 0x30 | Eight bits[11:4] number with the product identifier |
| 0x01 | Product_ID2 | R | 0xFX | Upper Four bits[3:0] number with the product identifier<br>Lower Four bits[3:0] number with the product version(VID) |
| 0x02 | Motion_Status | R | - | Bit field |
| 0x03 | Delta_X | R | - | Eight bits 2's complement number |
| 0x04 | Delta_Y | R | - | Eight bits 2's complement number |
| 0x05 | Operation_Mode | R/W | 0xB8 | Bit field |
| 0x06 | Configuration | R/W | 0x02 | Bit field |
| 0x07 | Image_Quality | R | - | Eight bits unsigned integer |
| 0x08 | Operation_State | R | - | Bit field |
| 0x09 | Write_Protect | R/W | 0x00 | Bit field |
| 0x0A | Sleep1_Setting | R/W | 0x70 | Bit field |
| 0x0B | Enter_Time | R/W | 0x10 | Bit field |
| 0x0C | Sleep2_Setting | R/W | 0x70 | Bit field |
| 0x0D | Image_ Threshold | R/W | 0x14 | Eight bits unsigned integer |
| 0x0E | Image_ Recognition | R/W | 0xE5 | Bit field |
| 0x31 | AE_State_Index | R | - | Lower Six bits[5:0] unsigned integer |
| 0x37 | Frameavg | R | - | Eight bits unsigned integer |
| 0x41 | LED_DriveStrength | R/W | 0xE8 | LED drive strength selection |
| 0x43 | LED_DriveMode | R/W | 0xA9 | LED drive mode selection |

### 3.2 Register Descriptions

| **0x00** | **Product_ID1** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | PID[11:4] | | | | | | | |
| Usage | The value in this register can't be changed. It can be used to verify the serial communications link is OK. | | | | | | | |

| **0x01** | **Product_ID2** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | PID[3:0] | | | | VID[3:0] | | | |
| Usage | The value in this register can't be changed. PID[3:0] can be used to verify that the serial communications link is OK.<br>VID[3:0] is a value between 0x0 and 0xF, it represents the chip version. | | | | | | | |

| **0x02** | **Motion_Status** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Motion | Reserved[1:0] | | DYOVF | DXOVF | RES[2:0] | | |
| Usage | ***Motion_Status*** register allows the user to determine if motion has occurred since the last time it was read. If so, then the user should read ***Delta_X*** and ***Delta_Y*** registers to get the accumulated motion. It also reveals if the motion buffers have overflowed since the last reading. The current resolution is also shown.<br><br>Reading this register freezes the ***Delta_X*** and ***Delta_Y*** register values. Read this register before reading the ***Delta_X*** and ***Delta_Y*** registers. If ***Delta_X*** and ***Delta_Y*** are not read before the motion register is read a second time, the data in ***Delta_X*** and ***Delta_Y*** will be lost. | | | | | | | |

Notes:

| **Field Name** | **Description** |
|---|---|
| Motion | Motion since last report<br>**0 = No motion (Default)**<br>1 = Motion occurred, freeze motion data in ***Delta_X*** and ***Delta_Y*** registers |
| Reserved[1:0] | Reserved for future use |
| DYOVF | Motion Delta Y overflow, ΔY buffer overflowed since last report<br>**0 = No overflow (Default)**<br>1 = Overflow has occurred |
| DXOVF | Motion Delta X overflow, ΔX buffer overflowed since last report<br>**0 = No overflow (Default)**<br>1 = Overflow has occurred |
| RES[2:0] | Resolution in counts per inch<br>000 = 500<br>001 = 650<br>010 = 750 **(Default)**<br>011 = 1000<br>100 = 1300<br>101 = 1500<br>110 – 111: reserved |



**PixArt Imaging Inc.**
**E-mail:** fae_service@pixart.com.tw

| 0x03 | Delta_X | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | X7 | X6 | X5 | X4 | X3 | X2 | X1 | X0 |
| Usage | X movement since last data frozen by reading ***Motion_Status*** register. Absolute value is determined by resolution. A reading clears the register. Report range –128 ~ +127.<br>The MSB bit represents as sign bit. | | | | | | | |

| 0x04 | Delta_Y | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 |
| Usage | Y movement since last data frozen by reading ***Motion_Status*** register. Absolute value is determined by resolution. A reading clears the register. Report range –128 ~ +127.<br>The MSB bit represents as sign bit. | | | | | | | |

| 0x05 | Operation_Mode | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | LEDsht_enh | **0** | **1** | Slp_enh | Slp2_enh | Slp2For | Slp1For | Wakeup |
| Usage | ***Operation_Mode*** register allows the user to change the mouse sensor operation modes. Shown below are the bits, their default values, and optional values.<br>Operation_Mode[4:0]<br>"0xxxx" = Disable sleep mode<br>"10xxx" = Enable sleep1 mode[1]<br>"11xxx" = Enable sleep2 mode[2]<br>"11100" = Force entering sleep2[3]<br>"1x010" = Force entering sleep1[3]<br>"1x001" = Force wakeup from sleep mode[3]<br>Notes:<br>1. Enable sleep mode, but disable automatic entering sleep2 mode. In this case, only **2** modes are available, normal mode and sleep1 mode. After 256 ms (typical) not moving during normal mode, the mouse sensor will enter sleep1 mode, and keep on sleep1 mode until motion detected or wakeup bit asserted. Note that the entering time depends on the setting of ***Enter_Time*** register.<br>2. Enable sleep mode full function. In this case, **3** modes are available, normal mode, sleep1 mode and sleep2 mode. After 256 ms (typical) not moving during normal mode, the mouse sensor will enter sleep1 mode, and keep on sleep1 mode until motion detected or wakeup bit asserted.<br>After 20 sec (typical) not moving during sleep1 mode, the mouse sensor will enter sleep2 mode, and keep on sleep2 mode until motion detected or force wakeup to normal mode. Note that the entering time depends on the setting of ***Enter_Time*** register.<br>3. Only ONE of these three bits, slp2mu_enh/slp1mu_enh/wakeup, can be set to 1 at a single register write, others MUST be 0. After function works, the asserted bit will be reset to 0 by internal signal.<br>4. To force entering normal mode, clear ***Slp_enh/Slp2_enh*** bit when the mouse sensor is in normal mode; otherwise, in sleep mode, clear ***Slp_enh/Slp2_enh*** bit, and then assert ***Wakeup*** bit. | | | | | | | |

Notes:

| Field Name | Description |
|---|---|
| LEDsht_enh | LED shutter enable/disable<br>0 = Disable<br>**1 = Enable (Default)** |
| Bit [6:5] | MUST always be **01** |
| Slp_enh | Sleep mode enable/disable<br>0 = Disable<br>**1 = Enable (Default)**<br>*Note that disable sleep mode will also disable AFC function |
| Slp2_enh | Automatic enter sleep2 mode enable/disable<br>0 = Disable<br>**1 = Enable (Default)** |

| | | |
|---|---|---|
| | Slp2For | Force entering sleep2 mode. Set "1" to enter sleep2, and then it will be reset to "0" |
| | Slp1For | Force entering sleep1 mode. Set "1" to enter sleep1, and then it will be reset to "0" |
| | Wakeup | Manual wake up from sleep mode, set "1" to wakeup and then it will be reset to "0" |

| **0x06** | **Configuration** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Reset | Mot0Swk1 | **0** | **0** | PD_enh | CPI [2:0] | | |
| Usage | The ***Configuration*** register allows the user to change the configuration of the mouse sensor. Shown below are the bits, their default values, and optional values.<br><br>With ***Mot0Swk1*** bit is clear, the MOTSWK pin is "level-sensitive". The pin level remains low when motion has occurred; The mouse controller can read ***Motion_Status*** register, ***Delta_X*** register, then ***Delta_Y*** register sequentially to acquire motion data. After the mouse controller reads all data, ***Delta_X*** and ***Delta_Y*** are both zero, the pin level will be high (see Section7).<br><br>With ***Mot0Swk1*** bit is set, the MOTSWK pin is "edge-sensitive". The pin will send a low pulse to trigger the mouse controller when the mouse sensor entering normal mode from sleep mode. The mouse controller can read ***Motion_Status*** register, ***Delta_X*** register, then ***Delta_Y*** register sequentially to acquire motion data (see Section7). | | | | | | | |

| Notes | **Field Name** | **Description** |
|---|---|---|
| | Reset | Full chip reset<br>**0 = Normal operation mode (Default)**<br>1 = Full chip reset |
| | Mot0Swk1 | MOTSWK pin output selection (see Section 7)<br>**0 = Motion function output (Default)**<br>1 = SWKINT function output |
| | Bit [5:4] | MUST always be **00** |
| | PD_enh | Power down mode<br>**0 = Normal operation (Default)**<br>1 = Power down mode |
| | CPI[2:0] | Output resolution setting, setting with CPI mode select bit<br>000 = 500<br>001 = 650<br>010 = 750 **(Default)**<br>011 = 1000<br>100 = 1300<br>101 = 1500<br>110 – 111: reserved |

| **0x07** | **Image_Quality** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Imgqa[7:0] | | | | | | | |
| Usage | Image Quality is a quality level of the mouse sensor in the current frame. Report range 0 ~ 255. The minimum required level is to be larger than the value of ***Image_ Threshold*** register. otherwise, the DSP would not process the motion data. | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | |
| | Imgqa[7:0] | Image quality report range: 0(worst) ~ 255(best). | | | | | | |

| **0x08** | **Operation_State** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Reserved[3:0] | | | | Slp_state | Op_state[2:0] | | |
| Usage | ***Operation_State*** register allows the user to read the operation state of the sensor. | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | |
| | Reserved[3:0] | Reserved for future use | | | | | | |
| | Slp_state | Sleep state (If Op_state[2:0] is 110, the Slp_state bit is effective.)<br>0 = LPT sleep1<br>1 = LPT sleep2 | | | | | | |
| | Op_state[2:0] | 000 = Normal state, 2400 FPS (with sleep function disable)<br>001 = Normal state, 2400 FPS (with sleep function enable)<br>011 = Normal state, 1200 FPS (with sleep function enable)<br>101 = Normal state, 800 FPS (with sleep function enable)<br>110 = Sleep mode (see Slp_state bit to get sleep state.)<br>Other values as transition state. | | | | | | |

| **0x09** | **Write_Protect** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | WP[7:0] | | | | | | | |
| Usage | Write protect for the register 0x0A ~ 0x7F. | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | |
| | WP[7:0] | Write protect enable/disable for the address after 0x09<br>**0x00 = Enable (Default)**, register 0x0A ~ 0x7F are read only<br>0x5A = Disable, register 0x0A ~ 0x7F can be read/written | | | | | | |

| **0x0A** | **Sleep1_Setting** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Bit | 7 | | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Slp1_freq[3:0] | | | | | **0** | **0** | **0** | **0** |
| Usage | ***Sleep1_Setting*** register allows the user to set frequency time for the sleep1 mode. | | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | | |
| | Slp1_freq[3:0] | Setting frequency time for the sleep1 mode.<br>A scale is 4ms. Relative to its value 0 ~ 15, the frequency time is 4ms ~ 64ms.<br>Default is 32ms. (slp1_freq[3:0] = 0111) | | | | | | | |
| | Bit [3:0] | MUST always be **0000** | | | | | | | |

| **0x0B** | **Enter_Time** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Bit | 7 | | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Slp1_etm[3:0] | | | | | Slp2_etm[3:0] | | | |
| Usage | ***Enter_Time*** register allows the user to set enter time for the sleep1 and sleep2 mode. | | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | | |
| | Slp1_etm[3:0] | Setting sleep1 enter time.<br>A scale is 128ms. Relative to its value 0 ~ 15, the frequency time is 128ms ~ 2048ms. Default is 256ms. (slp1_etm[3:0] = 0001) | | | | | | | |
| | Slp2_etm[3:0] | Setting sleep2 enter time.<br>A scale is 20480ms. Relative to its value 0 ~ 15, the frequency time is 20480ms ~ 327680ms. Default is 20480ms (about 20 sec). (slp2_etm[3:0] = 0000) | | | | | | | |

| **0x0C** | **Sleep2_Setting** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Bit | 7 | | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Slp2_freq[3:0] | | | | | **0** | **0** | **0** | **0** |
| Usage | ***Sleep2_Setting*** register allows the user to set frequency time for the sleep2 mode. | | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | | |
| | Slp2_freq[3:0] | Setting frequency time for the sleep2 mode.<br>A scale is 64ms. Relative to its value 0 ~ 15, the frequency time is 64ms ~ 1024ms.<br>Default is 512ms. (slp2_freq[3:0] = 0111) | | | | | | | |
| | Bit [3:0] | MUST always be **0000** | | | | | | | |



**PixArt Imaging Inc.**
**E-mail:** fae_service@pixart.com.tw

| 0x0D | Image_ Threshold | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | Imgqa_th[7:0] | | | | | | | |
| Usage | ***Image_ Threshold*** register allows the user to set image threshold. The mouse sensor calculates data to ***Delta_X*** and ***Delta_Y*** registers when image quality (please see ***Image_Quality*** register) is larger than image threshold. | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | |
| | Imgqa_th[7:0] | Image threshold: 0 (High recognition rate) ~ 255 (Low recognition rate). The minimum level for normally working is 20. Default is 00010100. | | | | | | |

| 0x0E | Image_Recognition | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | pk_wt[2:0] | | | **0** | Imgqa_df[3:0] | | | |
| Usage | ***Image_Recognition*** register allows the user to set recognition rate. | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | |
| | pk_wt[2:0] | Peak threshold weighting: 0 (Low recognition rate) ~ 7 (High recognition rate). Default is 111. | | | | | | |
| | Bit 4 | MUST always be **0** | | | | | | |
| | Imgqa_df[3:0] | Image qualification threshold difference: 0 (High recognition rate) ~ 15 (Low recognition rate). Default is 0101. | | | | | | |

| 0x41 | LED_DriveStrength | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | **1** | **1** | LED_Strength[2:0] | | | **0** | **0** | **0** |
| Usage | ***Frameave*** register allows the user to read average intensity of all frame pixels. | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | |
| | Bit 7:6 | Must be **11** | | | | | | |
| | LED_ Strength[2:0] | The value is to adjust LED current DAC strength. The driving ability is approximately: 2.5mA*Strehgth[2:0]<br>*The value works only at Reg0x43[6:5]=1, meaning LED drive mode is at LED Current DAC mode. | | | | | | |
| | Bit 2:0 | Must be **000** | | | | | | |

| 0x43 | LED_DriveMode | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Field | **0** | LED_Mode[1:0] | | **0** | **1** | **0** | **0** | **1** |
| Usage | ***Frameave*** register allows the user to read average intensity of all frame pixels. | | | | | | | |
| Notes | **Field Name** | **Description** | | | | | | |
| | Bit 7 | Must be **0** | | | | | | |
| | LED_Mode[1:0] | The value is to select LED drive mode<br>00: Current Switch mode with external resistor to control LED strength<br>01: Current DAC mode to control LED strength **(Default)**<br>10 – 11: reserved | | | | | | |
| | Bit 4:0 | Must be **01001** | | | | | | |

# 4. Specifications

## 4.1 Absolute Maximum Ratings

Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. These are stress ratings only. Functional operation of this device at these or any other conditions above those indicated in the operational sections of this specification is not implied and exposure to absolute maximum rating conditions for extended periods may affect device reliability.

| Symbol | Parameter | Min | Max | Unit | Notes |
|---|---|---|---|---|---|
| $T_{STG}$ | Storage Temperature | -40 | 85 | °C | |
| TA | Operating Temperature | -15 | 55 | °C | |
| $V_{DC}$ | DC Supply Voltage | -0.2 | $V_{dd1}$ + 0.2 | V | |
| | | -0.3 | $V_{dd2}$ + 0.3 | V | |
| $V_{IN}$ | DC Input Voltage | -0.3 | $V_{DC}$ | V | All I/O pin |
| | Lead Solder Temp | - | 260 | °C | For 10 seconds, 1.6mm below seating plane. |
| ESD | | - | 2 | kV | All pins, human body model MIL 883 Method 3015 |

## 4.2 Recommend Operating Condition

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Notes |
|---|---|---|---|---|---|---|
| $T_A$ | Operating Temperature | 0 | - | 40 | °C | |
| $V_{dd1}$ | Power Supply Voltage | 1.73 | 1.9 | 1.98 | V | VDDA, VDD short |
| $V_{dd2}$ | | 2.1 | 2.7 | 3.6 | | VDD |
| $V_{Npp}$ | Supply Noise | - | - | 150 | mV | Peak to peak within 10K - 80 MHz |
| Z | Distance From lens Reference Plane to Surface | 2.3 | 2.4 | 2.5 | mm | Refer to Figure 4. |
| Y1 | Distance from PCB surface to object surface | 5.75 | 5.85 | 5.95 | mm | PCB to Lens housing surface **must be contact**. |
| Y2 | PCB THICKNESS | | 1.6 | | mm | 1.6mm is the recommended PCB thickness. |
| R | Resolution | 500 | 750 | 1500 | CPI | |
| SCLK | Serial Port Clock Frequency | 0.08 | - | 1 | MHz | |
| FR | Frame Rate | 800 | 1200 | 2400 | frames/s | @ Normal mode, +/- 5% tolerance |
| S | Speed | 0 | - | 30 | inches/s | |
| A | Acceleration | 0 | - | 10 | g | |

## 4.3 AC Operating Condition (1.9V / 2.7V)

Electrical Characteristics over recommended operating conditions. Typical values at 25 °C, $V_{DD}$ = 2.7 V for 2.7V application and $V_{DD}$ = $V_{DDA}$ = 1.9 V for 1.9V application.

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Notes |
|---|---|---|---|---|---|---|
| $t_{PDR}$ | PD Pulse Register | - | - | 836 | us | Two frames time maximum after setting ***PD_enh*** bit in the ***Configuration*** register @2400frame/sec (refer to Figure 15). |
| $t_{PU}$ | Power Up from $V_{DD}$↑ | 10 | - | 38 | ms | From $V_{DD}$↑ to valid motion signals. And also for valid register read/write after HW/SW reset.<br>500usec + 90 frames. |
| $t_{HOLD}$ | SDIO Read Hold Time | 3 | 3 | - | us | Minimum hold time for valid data (refer to Figure 10). |
| $t_{RESYNC}$ | Serial Interface RESYNC. | 1 | - | - | us | @2400 frame/sec (refer to Figure 14) |
| $t_{SIWTT}$ | Serial Interface Watchdog Timer Timeout | 1.7<br>32<br>512 | - | - | ms | @2400 frame/sec (refer to Figure 14)<br>1.7ms for normal mode,<br>32ms (typical) for sleep1 mode,<br>512ms (typical) for sleep2 mode.<br>Note that the value depends on the setting of ***Sleep1_Setting*** register and ***Sleep2_Setting*** register. |
| $t_{ForSlp\text{-}EN}$ | Force Entering Sleep Mode | - | - | 2 | ms | From Normal mode to target Sleep mode by bits setting (refer to Register 0x05) |
| $t_{ForSlp\text{-}DIS}$ | Force Waking from Sleep Mode | - | - | 38 | ms | From Sleep mode to Normal mode by Wakeup bit setting (refer to Register 0x05) |
| $t_{DLY\text{-}SDIO}$ | SDIO delay after SCLK | - | - | 120 | ns | From SCLK falling edge to SDIO data valid, no load conditions |
| $t_{SWW}$ | SPI Time between two Write Commands | 25 | - | - | us | From rising SCLK for last bit of the first data byte, Commands to rising SCLK for last bit of the second data byte (refer to Figure 11) |
| $t_{SWR}$ | SPI Time between Write and Read Commands | 15 | - | - | us | From rising SCLK f or last bit of the first data byte, to rising SCLK for last bit of the second address byte (refer to Figure 12) |
| $t_{SRW}$<br>$t_{SRR}$ | SPI Time between Read and Subsequent Commands | 500 | - | - | ns | From rising SCLK for last bit of the first data byte, to falling SCLK for the first bit of the next address (refer to Figure 13) |
| $t_{SRAD}$ | SPI Read Address-Data Delay | 3 | - | - | us | From rising SCLK for last bit of the address byte, to falling SCLK for first bit of data being read (refer to Figure 13) |
| $t_{SWKINT}$ | Wakeup Interrupt Time | - | 418 | - | us | |
| $t_r$ , $t_f$ | Rise and Fall Times: SDIO | - | 30, 30 | - | ns | $C_L$ = 30 pF |

### 4.4 DC Electrical Characteristics (1.9V)

Electrical Characteristics over recommended operating conditions. Typical values at 25 °C, $V_{DD} = V_{DDA} = 1.9$ V

| Symbol | Parameter | Min. | Typ. | Max. | Unit | |
|---|---|---|---|---|---|---|
| **Type: Power** (Including LED current) | | | | | | |
| $I_{DDN1}$ | Supply Current<br>Mouse Moving (Normal1) | - | 1.6 | - | mA | @ 2400 FPS |
| $I_{DDN2}$ | Supply Current<br>Mouse Moving (Normal2) | - | 1.2 | - | mA | @ 1200 FPS |
| $I_{DDN3}$ | Supply Current<br>Mouse Moving (Normal3) | - | 1.0 | - | mA | @ 800 FPS |
| $I_{DDS1}$ | Supply Current<br>Mouse Not Moving (Sleep1) | - | 70 | - | uA | |
| $I_{DDS2}$ | Supply Current<br>Mouse Not Moving (Sleep2) | - | 10 | - | uA | |
| $I_{DDPD}$ | Supply Current<br>(Power Down) | - | 9 | - | uA | |
| **Type: SCLK, SDIO, PD, MOTSWK** | | | | | | |
| $V_{IH}$ | Input Voltage HIGH | VDD*0.7 | - | - | V | |
| $V_{IL}$ | Input Voltage LOW | - | - | VDD*0.3 | V | |
| $V_{OH}$ | Output Voltage HIGH | VDD-0.4 | - | - | V | @$I_{OH}$ = 2mA |
| $V_{OL}$ | Output Voltage LOW | - | - | 0.4 | V | @$I_{OL}$ = 2mA |
| **Type: LED** | | | | | | |
| $V_{OL}$ | Output Voltage LOW | - | - | 100 | mV | @$I_{OL}$ = 10mA |
| $I_{LEDS}$ | LED Sink Current | - | - | 50 | mA | |



PixArt Imaging Inc.
E-mail: fae_service@pixart.com.tw

## 4.5 DC Electrical Characteristics (2.7V)

Electrical Characteristics over recommended operating conditions. Typical values at 25 °C, $V_{DD}$ = 2.7 V

| Symbol | Parameter | Min. | Typ. | Max. | Unit | |
|---|---|---|---|---|---|---|
| **Type: Power** (Including LED current) | | | | | | |
| $I_{DDN1}$ | Supply Current<br>Mouse Moving (Normal1) | - | 1.6 | - | mA | @ 2400 FPS |
| $I_{DDN2}$ | Supply Current<br>Mouse Moving (Normal2) | - | 1.2 | - | mA | @ 1200 FPS |
| $I_{DDN3}$ | Supply Current<br>Mouse Moving (Normal3) | - | 1.0 | - | mA | @ 800 FPS |
| $I_{DDS1}$ | Supply Current<br>Mouse Not Moving (Sleep1) | - | 70 | - | uA | |
| $I_{DDS2}$ | Supply Current<br>Mouse Not Moving (Sleep2) | - | 10 | - | uA | |
| $I_{DDPD}$ | Supply Current<br>(Power Down) | - | 9 | - | uA | |
| **Type: SCLK, SDIO, PD, MOTSWK** | | | | | | |
| $V_{IH}$ | Input Voltage HIGH | VDD*0.7 | - | - | V | |
| $V_{IL}$ | Input Voltage LOW | - | - | VDD*0.3 | V | |
| $V_{OH}$ | Output Voltage HIGH | VDD-0.4 | - | - | V | @$I_{OH}$ = 2mA |
| $V_{OL}$ | Output Voltage LOW | - | - | 0.4 | V | @$I_{OL}$ = 2mA |
| **Type: LED** | | | | | | |
| $V_{OL}$ | Output Voltage LOW | - | - | 100 | mV | @$I_{OL}$ = 10mA |
| $I_{LEDS}$ | LED Sink Current | - | - | 50 | mA | |

## 5. Z, 2D/3D Assembly, PCB Position

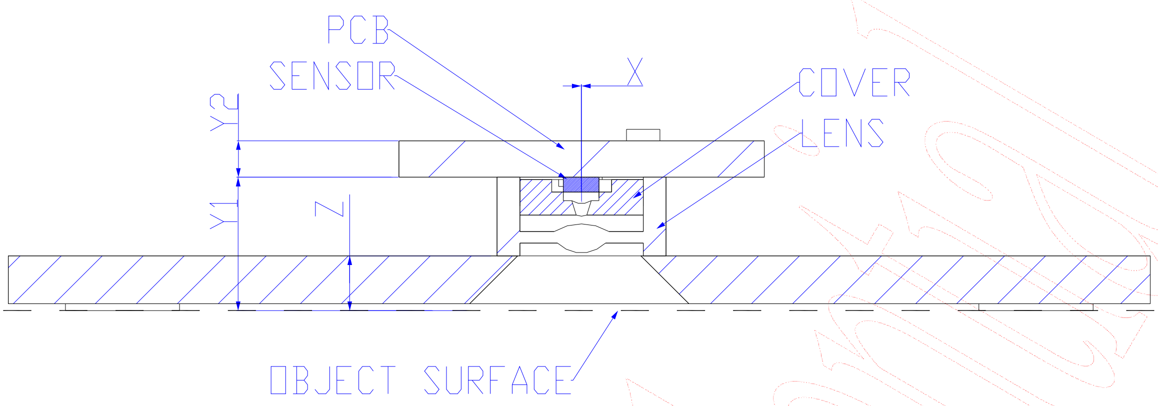



**Figure 4. Distance from Lens Reference Plane to Surface**

| PARAMETERS | SYMBOL | MIN | TYP | MAX | UNIT | CONDITIONS |
|---|---|---|---|---|---|---|
| Distance from center of IC to center of Aperture stop | X | | | 0.076 | mm | Center of Aperture stop is close to center of IC by self-align housing. |
| Distance from PCB surface to object surface | Y1 | 5.72 | 5.85 | 5.98 | mm | PCB to Lens housing surface **must be contact**. |
| PCB THICKNESS | Y2 | | (1.6) | | mm | 1.6mm is the recommended PCB thickness. |
| Distance from object surface to lens reference plane | Z | 2.3 | 2.4 | 2.5 | mm | |



PixArt Imaging Inc.
**E-mail:** fae_service@pixart.com.tw

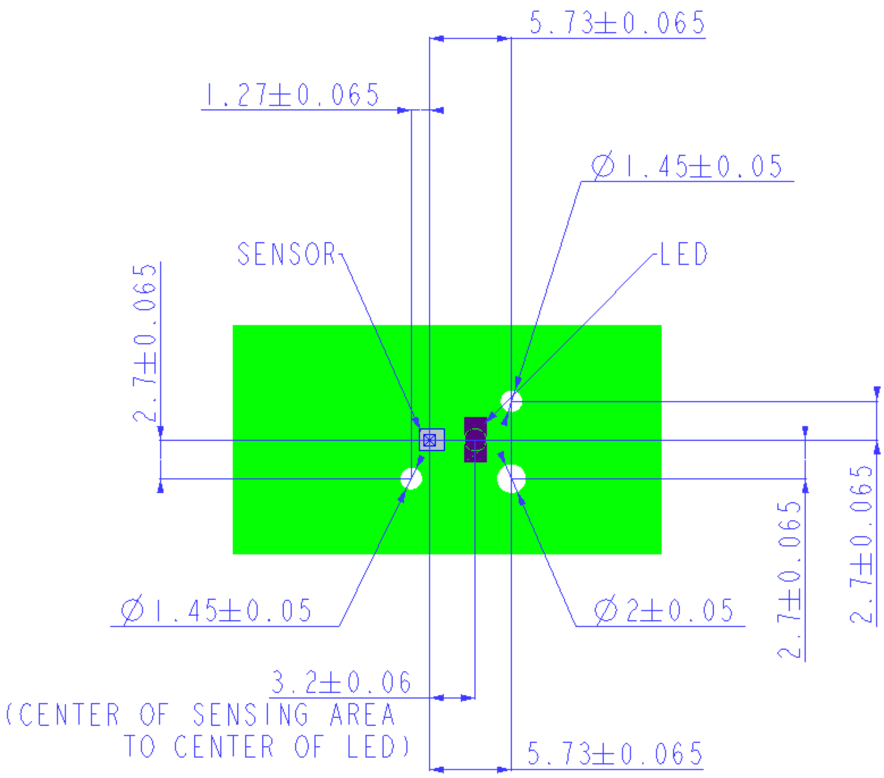



**Figure 6. PCB Position for Layout/Mounting Instructions**

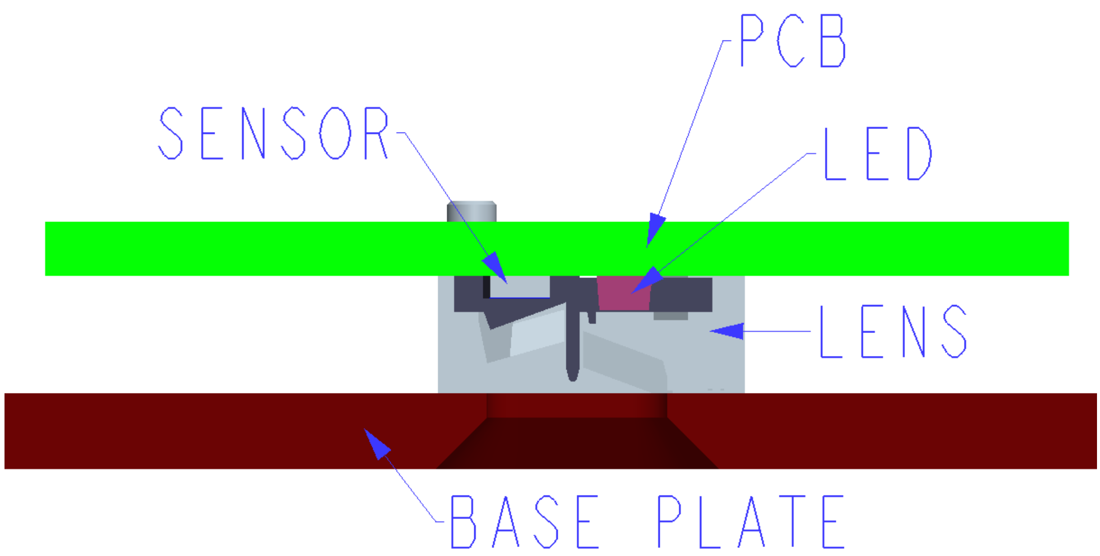



**Figure 7. 3D Assembly for Mounting Instructions**

# 6. Serial Interface

The synchronous serial port is used to set and read parameters in the mouse sensor.

**SCLK:** The serial clock line. It is always generated by the mouse controller.

**SDIO:** The serial data line is used to write and read data.

## 6.1 Transmission Protocol

The transmission protocol is a two-wire link, half duplex protocol between the micro-controller and the mouse sensor. All data changes on SDIO are initiated by the falling edge on SCLK. The mouse controller always initiates communication; the mouse sensor never initiates data transfers.

The transmission protocol consists of the two operation modes:

- Write Operation.

- Read Operation.

Both of the two operation modes consist of two bytes. The first byte contains the address (seven bits) and has a bit 7 as its MSB to indicate data direction. The second byte contains the data.

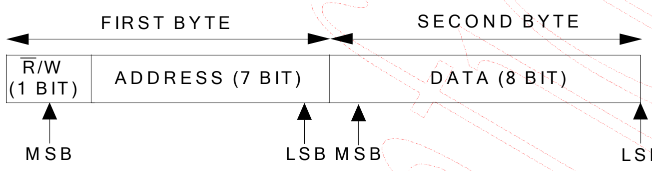



**Figure 8. Transmission Protocol**

### 6.1.1 Write Operation

A write operation, which means that data is going from the mouse controller to the mouse sensor, is always initiated by the mouse controller and consists of two bytes. The first byte contains the address (seven bits) and has a "1" as its MSB to indicate data direction. The second byte contains the data. The transfer is synchronized by SCLK. The mouse controller changes SDIO on falling edges of SCLK. The mouse sensor reads SDIO on rising edges of SCLK.

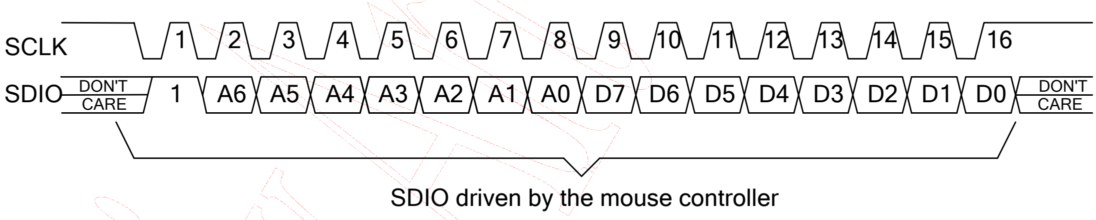



**Figure 9. Write Operation**

### 6.1.2 Read Operation

A read operation, which means that data is going from the mouse sensor to the mouse controller, is always initiated by the mouse controller and consists of two bytes. The first byte contains the address, is written by the mouse controller, and has a "0" as its MSB to indicate data direction. The second byte contains the data and is driven by the mouse sensor. The transfer is synchronized by SCLK. SDIO is changed on falling edges of SCLK and read on every rising edge of SCLK. The mouse controller must go to a high Z state after the last address data bit. The mouse sensor will go to the high Z state after the last data bit.

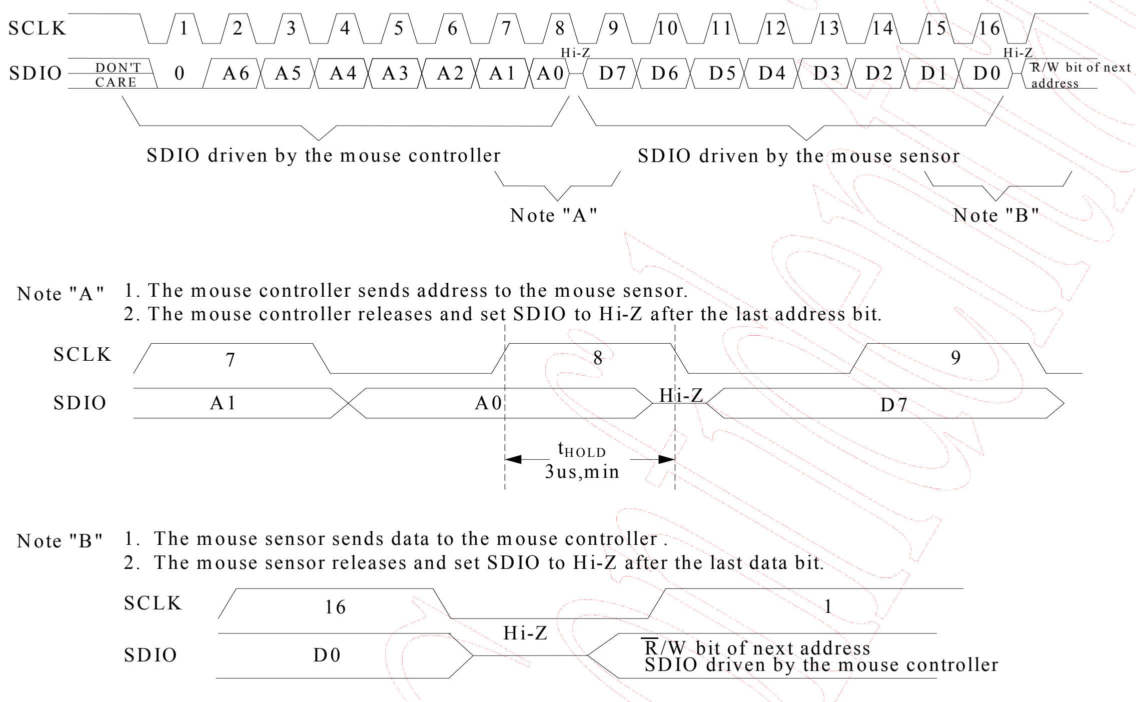



**Figure 10. Read Operation**

### 6.1.3 Required timing between Read and Write Commands

There are minimum timing requirements between read and write commands on the serial port.

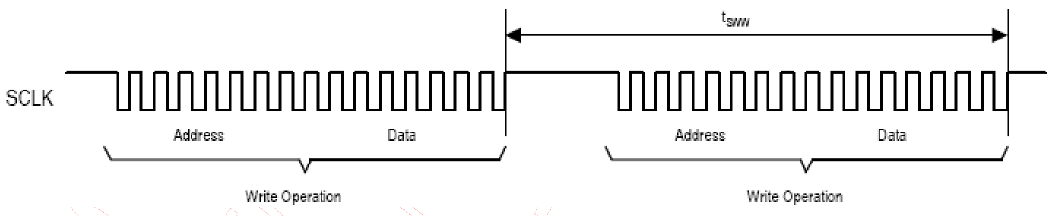



**Figure 11. Timing between two write commands**

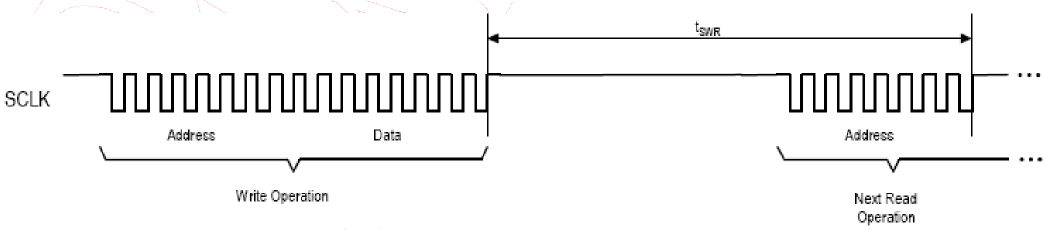



**Figure 12. Timing between write and read commands**

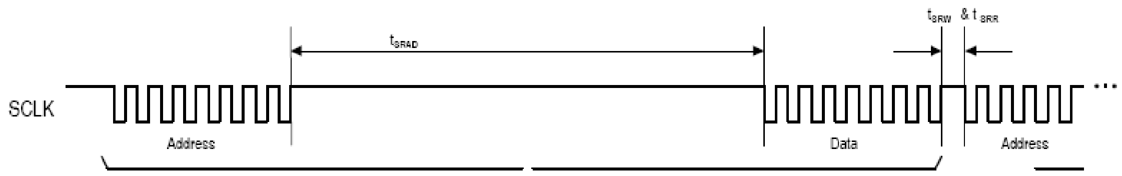



**Figure 13. Timing between read and either write or subsequent read commands**

## 6.2 Re-Synchronous Serial Interface

If the mouse controller and the mouse sensor get out of synchronization, then the data either written or read from the registers will be incorrect. In such a case, an easy way to solve this condition is to toggle the SCLK line from high to low for least $t_{RESYNC}$, and then MUST toggle it from low to high to wait at least $t_{SIWTT}$ to reach re-synchronous the serial port. This method is called by "watchdog timer timeout". The mouse sensor will reset the serial port without resetting the registers and be prepared for the beginning of a new transmission.

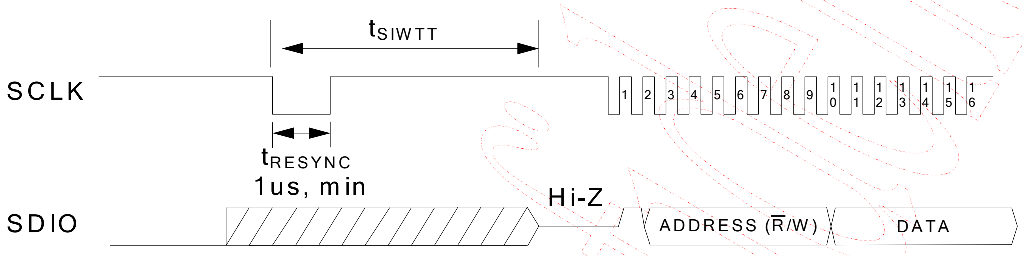



**Figure 14. Re-synchronous Serial Interface Using Watchdog Timer Timeout**

Note that this function is disabled when the mouse sensor is in the power down mode. If the user uses this function during the power down mode, it will get out of synchronization. The mouse sensor and the mouse controller also might get out of synchronization due to following conditions.

- Power On Problem - The problem occurs if the mouse sensor powers up before the mouse controller sets the SCLK and SDIO lines to be output. The mouse sensor and the mouse controller might get out of synchronization due to power on problem. An easy way to solve this is to use "watchdog timer timeout".
- ESD Events - The mouse sensor and the mouse controller might get out of synchronization due to ESD events. An easy way to solve this is to use "watchdog timer timeout".

## 6.3 Collision Detection on SDIO

The only time that the mouse sensor drives the SDIO line is during a READ operation. To avoid data collisions, the mouse controller should release SDIO before the falling edge of SCLK after the last address bit. The mouse sensor begins to drive SDIO after the next falling edge of SCLK. The mouse sensor releases SDIO of the rising SCLK edge after the last data bit. The mouse controller can begin driving SDIO any time after that. In order to maintain low power consumption in normal operation or when the PD pin is pulled high, the mouse controller should not leave SDIO floating until the next transmission (although that will not cause any communication difficulties).

## 6.4 Power Down Mode

The mouse sensor can be placed in a power-down mode by setting ***PD_enh*** bit in the ***Configuration*** register via a serial port write operation. After setting the ***Configuration*** register, wait at most 2 frames times. To get the chip out of the power down mode, clear ***PD_enh*** bit in the ***Configuration*** register via a serial port write operation. In the power down mode, the serial interface watchdog timer (see Section 6.2) is not available. But, the serial interface still can read/write normally. For an accurate report after leave the power down mode, wait about 3ms before the mouse controller is able to issue any write/read operation to the mouse sensor.

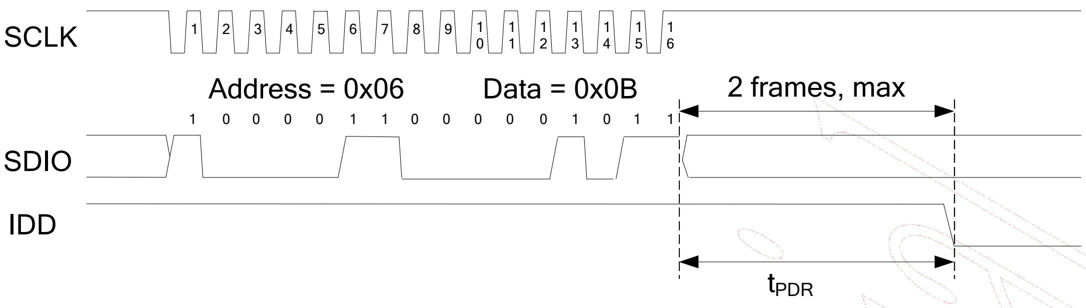



**Figure 15. Power-down Configuration Register Writing Operation**

## 6.5 Error Detection

1. The mouse controller can verify success of write operations by issuing a read command to the same address and comparing written data to read data.
2. The mouse controller can verify the synchronization of the serial port by periodically reading the product ID register

# 7. MOTSWK function

## 7.1 Motion function

To use Motion function, the ***Mot0Swk1*** bit in the ***Configuration*** register must be set to zero. Motion is used to monitor if the mouse sensor data is clear. If motion data are not clear, MOTSWK pin level will remain low. After the mouse controller reads all motion data from the mouse sensor, the mouse sensor will set MOTSWK pin level to high.

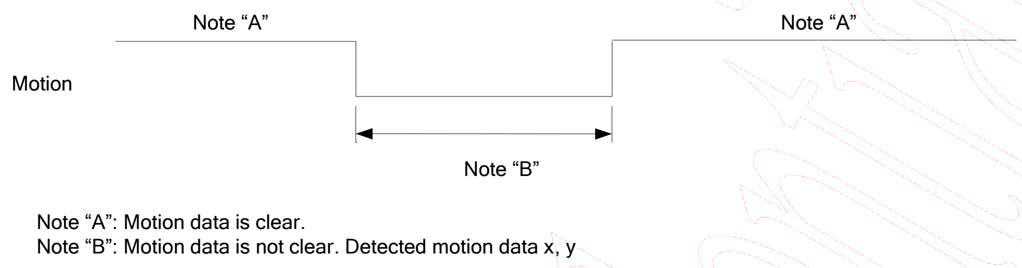



**Figure 16. Motion function**

## 7.2 SWKINT function

To use SWKINT function, the ***Mot0Swk1*** bit in the ***Configuration*** register must be set to one. SWKINT works when the mouse sensor is in the sleep mode and the mouse controller is also in the sleep mode. If the mouse sensor detects any motion occurrence at this moment, the mouse sensor will wake the mouse controller up promptly via MOTSWK pin. The mouse sensor will trigger the mouse controller at the rising/falling edge of MOTSWK pin.

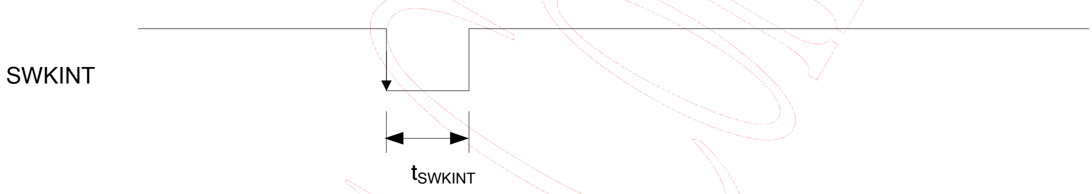



**Figure 17. SWKINT function**

# 8. LED Drive Mode Selection

## 8.1 Current DAC Mode

To use Current DAC function, the ***LED_Mode*** in the ***LED_DriveMode*** register must be set to 1. The driving strength can be adjusted by modify the ***LED_Strength*** in the ***LED_DriveStrength*** register.

***This function works ONLY at VDD from 2.1V to 3.6V. It does NOT support 1.9V application.**

## 8.2 Current Switch Mode (with external resistor)

To use Current Switch with external resistor function, the ***LED_Mode*** in the ***LED_DriveMode*** register must be set to 0. The driving strength can be adjusted by decided by the resistor value.

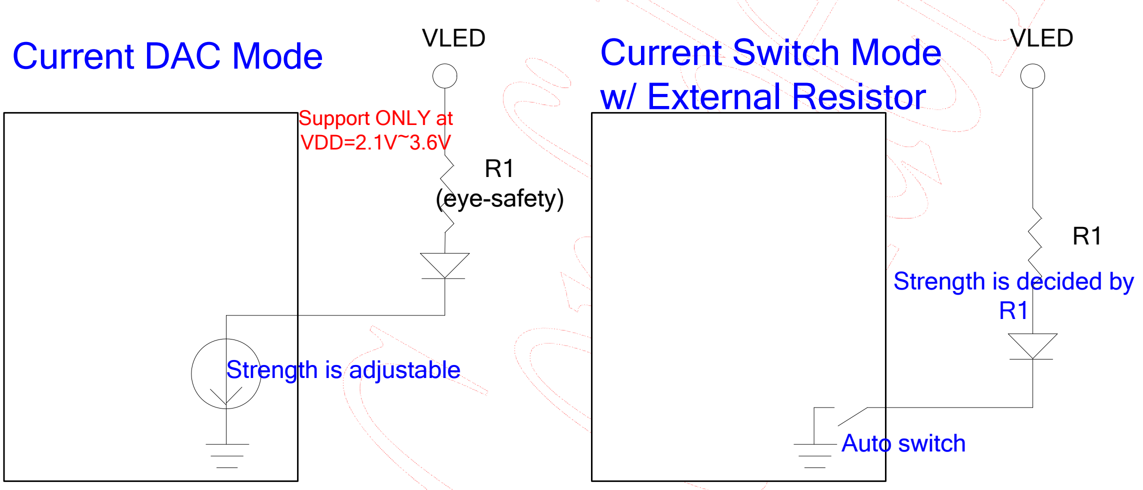



**Figure 18. LED drive mode**



PixArt Imaging Inc.
**E-mail:** fae_service@pixart.com.tw

## 9. Referencing Application Circuit

## 9.1 Power 2.7V Application Circuit, no DC/DC (with Infrared LED, 2.4GHz Transceiver)

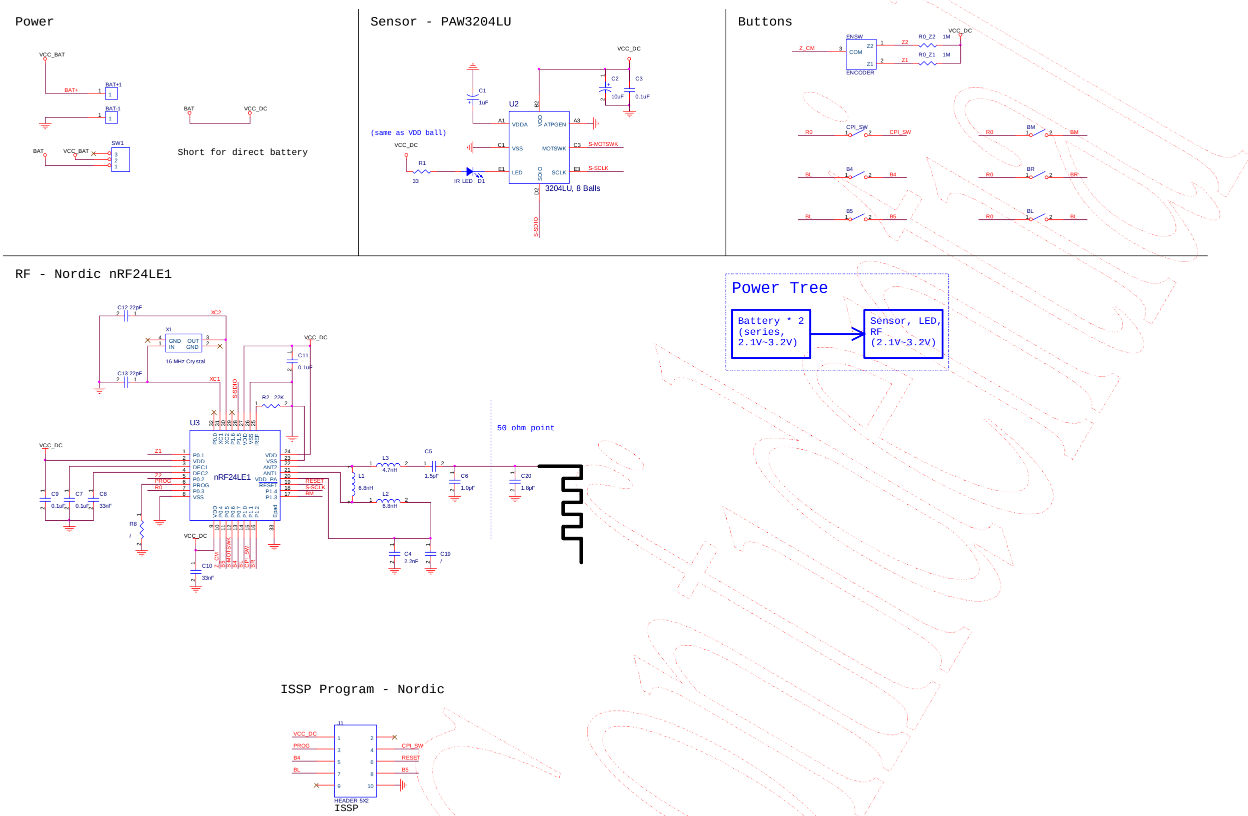



**Figure 19. Application Circuit for 2.7V, no DC/DC (with Infrared LED)**

## 9.2 Power 1.9V Application Circuit, one DC/DC (with Infrared LED, 2.4GHz Transceiver)

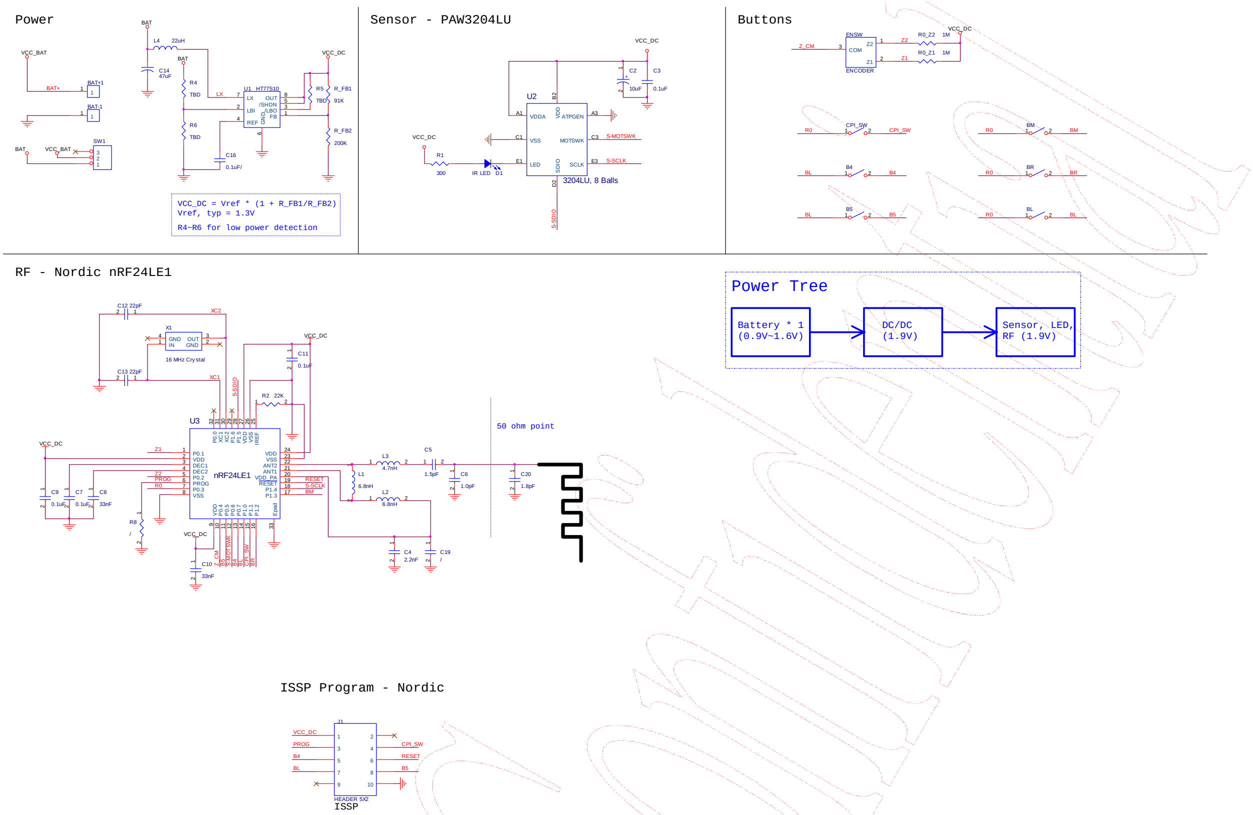



Figure 20. Application Circuit for 1.9V, one DC/DC (with Infrared LED)

## 9.3 PCB Layout Consideration

- Caps for balls A1, B2 must have trace lengths less than **5mm**.

## 9.4 Recommended Value for R1

| Light Source | $V_{LED}$ | R1 | | |
|---|---|---|---|---|
| | | Min. | Typ. | Max. |
| Infrared LED | 2.1~3.6 | - | 33 | - |
| | 1.73~1.98 | - | 300 | - |

* Recommended using LTE-C216-P-W LED.

** Recommends using internal current DAC for 2.1V~3.6V, and R1 is unnecessary.

# 10. Optical Criterion

## 10.1 Recommended Infrared LED Angle Criterion

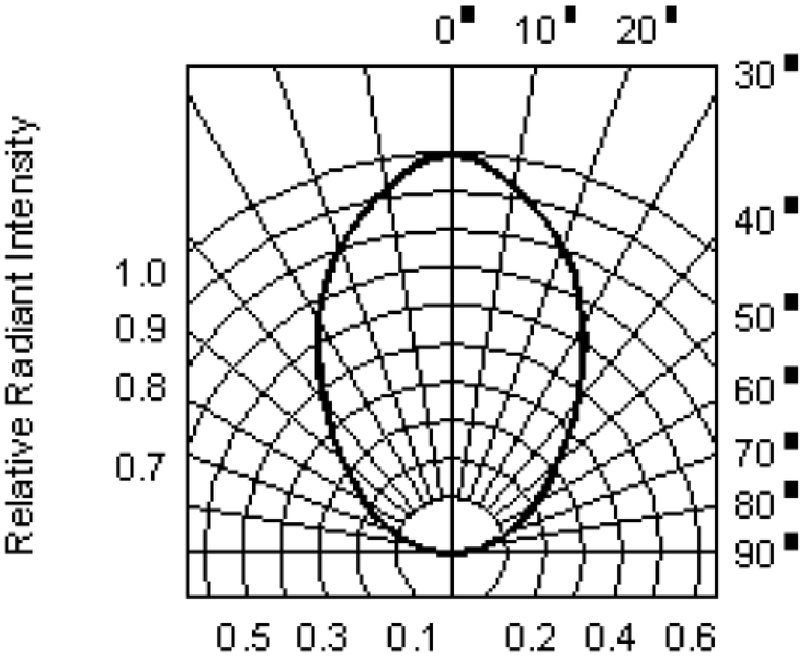



**Figure 21. Radiation Characteristics**

| LED Viewing Angle | Min. | Typ. | Max. |
|---|---|---|---|
| $2\theta_{1/2}$ | - | 100 | - |

- Recommended using Chang-Yu LED goniophotometer V110 to measure the LED viewing angle.

## 10.2 Recommended Value for Optical Power

- In order to balance tracking performance of PAW3204LU-TJDU and lower power consumption of LED, PixArt recommended a value for optical power. The power MUST fit in the following table by adjusting R1 value when LED source is not recommended one. Optical power is measured from base plate rectangle hole with LED in DC mode. (Please see optical power measurement method AP note). Recommended using ADCMT power meter 8230E to measure the optical power.

| Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|
| **Optical Power** | 200 | - | 1000 | uW |



PixArt Imaging Inc.
E-mail: fae_service@pixart.com.tw

# 11. Package Information

## 11.1 Package Outline Drawing

| | Symbol | Nominal | Min. | Max. |
|---|---|---|---|---|
| | | µm | | |
| Package Body Dimension X | A | 1567 | 1542 | 1592 |
| Package Body Dimension Y | B | 1751.8 | 1726.8 | 1776.8 |
| Package Height | C | 665 | 605 | 725 |
| Ball Height | C1 | 130 | 100 | 160 |
| Package Body Thickness | C2 | 535 | 490 | 580 |
| Thickness of Glass surface to wafer | C3 | 405 | 385 | 425 |
| Ball Diameter | D | 250 | 220 | 280 |
| Total Pin Count | N | 8 | | |
| Pin Count X axis | N1 | 3 | | |
| Pin Count Y axis | N2 | 5 | | |
| Pins Pitch X axis | J1 | 440 | | |
| Pins Pitch Y axis | J2 | 300 | | |
| Edge to Pin Center Distance along X | S1 | 343.5 | 313.5 | 373.5 |
| Edge to Pin Center Distance along Y | S2 | 275.9 | 245.9 | 305.9 |

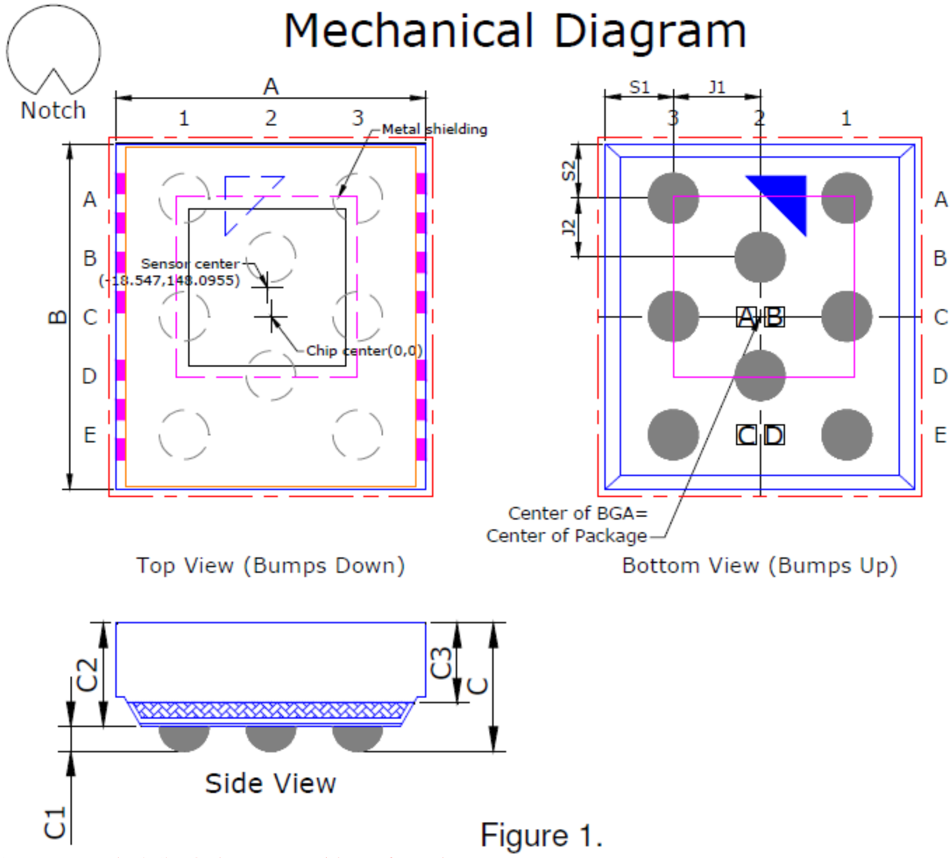



Figure 22. Package Outline Drawing

## 12. AFC Configuration

### 12.1 To Force AFC Stay at Specific Frame Rate

Below is common register setting for specific frame rate:

Function1: To stay at 2400 frame rate

Register : At initial setting, and then write 0 to Register 0x05[4] (disable sleep mode function)

Function2: To stay at 1200 frame rate

Register : At initial setting, and then (1)write 0 to Register 0x05[5],
(2)write 0x5A to Register 0x09,
(3)write 1 to Register 0x17[6]

Function3: To stay at 800 frame rate

Register : At initial setting, and then (1)write 0 to Register 0x05[5],
(2)write 0x5A to Register 0x09,
(3)write 1 to Register 0x17[7]

### 12.2 To Disable AFC Function

Register : At initial setting, and then (1)write 0x5A to Register 0x09,
(2)write 1 to Register 0x7A[7]

## Update History

| Version | Update | Date |
|---|---|---|
| V0.1 | Creation, Preliminary 1st version | Aug/01/2012 |
| V0.2 | Content Revised | Oct/15/2012 |
| V0.3 | Content Revised | Mar/06/2013 |
| V0.4 | Content Revised | Jun/20/2013 |
| V1.0 | First public version | Mar/25/2015 |



Note: The Part No. of the Mouse Product with Prefix "PAN" shall NOT be made, sold, offered to sell, imported or used in or into USA, Canada, Japan and EU. For "PAN", PixArt has only gained territory-limited patent license from Avago. Avago reserve right to take legal action against our customers who fails to comply the above term. PLEASE NOTE THAT PixArt will NOT defend, indemnify, or provide any assistance to our customers who fail to comply the term. IF YOU DO NOT AGREE THE TERM, PIXART WILL NOT DELIVER "PAN" PRODUCTS TO YOU.

# Appendix A: Performance Optimization

## Prerequisite

To get the best sensor performance, please make sure firmware behavior as following:

1. Regular polling time, ex: 4ms per poll
2. SPI clock MUST faster than 80KHz

## A-1. Register Setting (for initialization)

The following recommended setting is for even better performance on PAW3204LU-TJDU.

*Setting is for 2.7v application. When apply 1.73v~1.98v, please use current switch mode to drive LED.

| Register Address | VID=0 Value (Default) | Opt. Value (Modify) | Note |
|---|---|---|---|
| 0x09 | 0x00 | 0x5A | (Disable write protect) |
| 0x06 | 0x02 | 0x03 | |
| 0x13 | 0x0B | 0x09 | |
| 0x1D | 0xE3 | 0x23 | |
| 0x21 | 0x23 | 0x1F | |
| 0x25 | 0x0D | 0x0F | |
| 0x2A | 0x96 | 0x78 | |
| 0x2B | 0x8C | 0x6E | |
| 0x2C | 0x5A | 0x50 | |
| 0x2D | 0x50 | 0x46 | |
| 0x41 | 0xE8 | 0xC8 | |
| 0x43 | 0xA9 | 0x29 | |
| 0x47 | 0xE8 | 0x88 | |
| 0x4E | 0xA3 | 0x9F | |
| 0x50 | 0xF2 | 0x72 | |
| 0x56 | 0x63 | 0x60 | |
| 0x65 | 0x4C | 0x8C | |
| 0x70 | 0xE0 | 0x60 | |
| 0x75 | 0xC4 | 0x52 | |
| 0x76 | 0xC6 | 0x42 | |
| 0x78 | 0x05 | 0x23 | |
| 0x79 | 0x05 | 0x46 | |
| 0x7C | 0x94 | 0xD2 | |
| 0x7D | 0xD9 | 0x55 | |
| 0x05 | | 0xA9 | Must-be process! To adjust sensor exposure timing. |
| 0x20 | | 0x2E | |
| Delay 50ms | | | |
| 0x20 | | 0x2F | |
| 0x05 | | 0xB8 | |
| 0x7F | 0x00 | 0x01 | **(Switch to Bank1)** |
| 0x14 | 0x14 | 0x28 | |
| 0x19 | 0xA6 | 0x99 | |
| 0x1A | 0x89 | 0x09 | |
| 0x26 | 0x0C | 0x10 | |
| 0x7F | 0x00 | 0x00 | **(Switch to Bank0)** |
| 0x09 | 0x00 | 0x00 | **(Enable write protect)** |

***Please reserve memory space for future.**

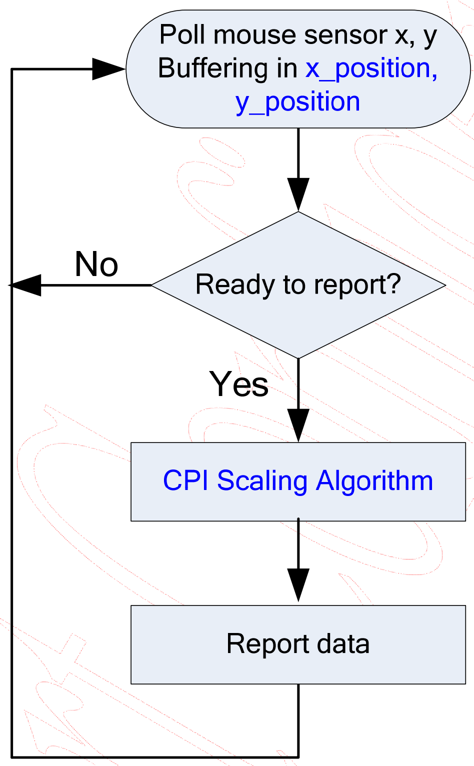
## Appendix B: Algorithm Enhancement

### Applies: PAW3204LU-TJDU

The following algorithm is for performance enhancement. PixArt provides these algorithms for the user to implement in firmware.

| Algorithm | Reference |
|---|---|
| CPI Scaling | Appendix C |

*Recommends follow the flow chart to apply CPI Scaling Algorithm.



# Appendix C: CPI Scaling Algorithm

## Summary:

For critical count per inch, known as CPI, requirement, PixArt provides this algorithm to scale CPI.

**Here is the reference code:**
(parameters definition)

```
INT iMCU_SumX = 0;
INT iMCU_SumY = 0;
INT iMCU_RX = 0;
INT iMCU_RY = 0;
INT iScalingDivX = 128;
INT iScalingDivY = 128;
extern void CPI_Scaling(INT, INT);
```

(CPI scaling algorithm code)

```
void CPI_Scaling(INT iScalingMulX, INT iScalingMulY)
{
        iMCU_SumX = iMCU_RX + x_postion*iScalingMulX;
        iMCU_SumY = iMCU_RY + y_postion*iScalingMulY;

        x_postion = iMCU_SumX/iScalingDivX;
        y_postion = iMCU_SumY/iScalingDivY;

        iMCU_RX = iMCU_SumX - x_postion*iScalingDivX;
        iMCU_RY = iMCU_SumY - y_postion*iScalingDivY;
}
```

(Call CPI scaling algorithm)

```
CPI_Scaling(120, 128);    //adjustable
```





PixArt Imaging Inc.
E-mail: fae_service@pixart.com.tw