

APDS-9500

Imaging-Based Proximity and Gesture Sensor



Data Sheet

Description

The APDS-9500 provides an imaging-based gesture recognition function with an I²C-bus interface in a single 18-pin package. It can recognize nine gestures, including move up, move down, move left, move right, move forward, move backward, circle-clockwise, circle-counter-clockwise and wave. This gesture information can be easily accessed via the I²C bus.

The APDS-9500 also provides built-in proximity detection for sensing objects approaching or departing.

Applications

- Gesture Detection
- Cell Phone Touch-Screen Disable
- Mechanical Switch Replacement

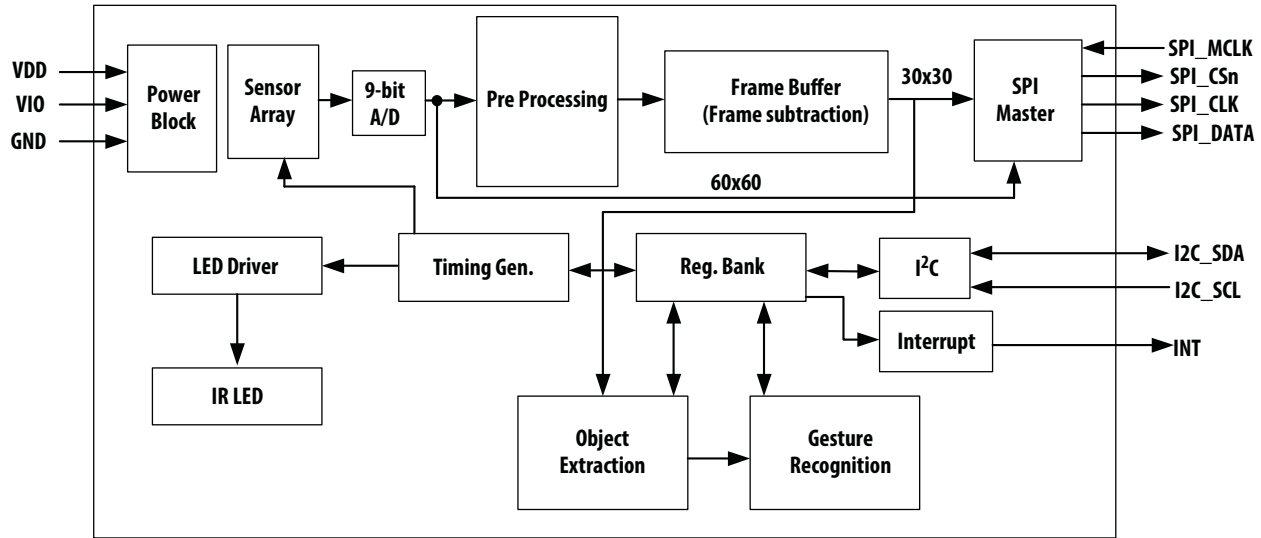
Ordering Information

Part Number	Packaging	Quantity
APDS-9500	Tape and Reel	2500 per reel

Features

- Imaging-based proximity and gesture sensor in an optical module
- Gesture output
 - Nine-gesture recognition
- Proximity output
 - Object brightness output
 - Object size output
- Cursor mode output
 - X and Y output
- Image output
 - SPI output
 - 30 × 30 pixels/60 × 60 pixels
 - 9-bit gray scale
- Ambient light immunity
- I²C-bus interface compatible
 - Data rates up to 400 kHz
 - Dedicated Interrupt pin
- Flexible power saving mode
- V_{DD} range from 2.8V to 3.3V
- I/O voltage range from 1.8V to 3.3V
- Small package: 6.87 × 3.76 × 2.86 mm (L × W × H)

Functional Block Diagram



I/O Pins Configuration

Pin	Name	Type	Description
1	NC		No connect
2	INT	O	Interrupt
3	GND		Power supply ground
4	SPI_DATA	O	SPI data
5	SPI_CLK	O	SPI clock
6	NC		No connect
7	LDR_2		LED Drive 2
8	LED-K_1		LED Cathode
9	LED-A		LED anode, connect to V _{LED} on PCB
10	VDDA_I/O		Connect to GND through decoupling capacitor
11	VDDAY_I/O		Connect to GND through decoupling capacitor
12	VDDA18_I/O		Connect to GND through decoupling capacitor
13	VDD		Power supply voltage
14	VIO		Power supply voltage
15	SPI_MCLK	I	SPI master clock
16	SPI_CS#	O	SPI chip select
17	I2C_SDA	I/O	I ² C-bus serial data I/O terminal—serial data I/O for I ² C-bus
18	I2C_SCL	I	I ² C-bus serial clock input terminal—clock signal for I ² C-bus serial data

Absolute Maximum Ratings (unless otherwise specified, $T_a = 25^\circ\text{C}$)

Parameter	Symbol	Min.	Max.	Unit
Supply voltage	V_{DD}		4.0	V
LED supply voltage	V_{LED}		4.6	V
LED pulse current ^[1]	I_{LED}		2	A
I ² C-bus pin, Int pin voltage	V_{BUS}	-0.3	$V_{DD} + 0.3$	V
I ² C-bus pin, Int pin current	I_{BUS}		10	mA

Recommended Operating Conditions (unless otherwise specified, $T_a = 25^\circ\text{C}$)

Parameter	Symbol	Min.	Typ.	Max.	Unit
Operating temperature	T_A	-20		+70	$^\circ\text{C}$
Operating supply voltage	V_{DD}	2.8		3.6	V
LED supply voltage	V_{LED}	3.0		4.2	V
Peak LED pulse current ^[1]	I_{LED}		720	860	mA
I ² C-bus pin, INT pin voltage	V_{BUS}	1.8		3.3	V
I ² C-bus pin, INT pin current	I_{BUS}			5	mA

Note:

1. Pulse width < 500 μs , duty cycle < 5%**Operating Characteristics (unless otherwise specified, $T_a = 25^\circ\text{C}$, $V_{DD} = 2.8\text{V}$)**

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Suspend current	I_{DD_SUS}		15		μA	See Figure 13
Standby state 1 current	I_{DD_ST1}		0.57		mA	See Figure 11
Standby state 2 current	I_{DD_ST2}		0.37		mA	See Figure 12
Current consumption for proximity detection			0.2		mA	1. Register R_AE_Exposure_UB, 0x48 = 0x10 2. Register R_AE_Exposure_UB, 0x49 = 0x00 3. Register R_IDLE_TIME, 0x65 = 0xCE 4. Register R_IDLE_TIME, 0x66 = 0x0B
Current consumption for gesture detection			1		mA	1. Register R_AE_Exposure_UB, 0x48 = 0x48 2. Register R_AE_Exposure_UB, 0x49 = 0x00 3. Register R_IDLE_TIME, 0x65 = 0x96 4. Register R_IDLE_TIME, 0x66 = 0x00
I ² C-bus input high voltage	V_{IH}	$0.7 \times V_{BUS}$		$V_{BUS} + 0.3$	V	
I ² C-bus input low voltage	V_{IL}	-0.3		$0.3 \times V_{BUS}$	V	
INT, I ² C_SDA output low voltage	V_{OL}			$0.1 \times V_{BUS}$	V	

Proximity Characteristics (unless otherwise specified, $T_a = 25^\circ\text{C}$, $V_{DD} = 2.8\text{V}$)

Parameter	Min.	Typ.	Max.	Unit	Test Conditions
PS ADC Count Value (No Object)	0	5		counts	1. Dedicated duo power supply, $V_{DD} = 2.8\text{V}$ and $V_{LED} = 3\text{V}$ 2. R_LensShadingComp_EnH Register, $0x25 = 0x14$ 3. R_LED1_DAC_UB Register, $0x32 = 0x14$ 4. R_AE_Exposure_UB Register, $0x48 = 0x10$ 5. R_AE_Exposure_UB Register, $0x49 = 0x00$ 6. Open view (no glass) and no reflective object above the module.
PS ADC Count Value (100 mm Distance Object)	47	56	65	counts	1. Dedicated duo power supply, $V_{DD} = 2.8\text{V}$ and $V_{LED} = 3\text{V}$ 2. Reflecting object – 73 mm × 83 mm Kodak 90% grey card, 100 mm distance 3. R_LensShadingComp_EnH Register, $0x25 = 0x14$ 4. R_LED1_DAC_UB Register, $0x32 = 0x14$ 5. R_AE_Exposure_UB Register, $0x48 = 0x10$ 6. R_AE_Exposure_UB Register, $0x49 = 0x00$ 7. Open view (no glass) above the module.

AC Electrical Characteristics, $V_{DD} = 3\text{V}$, $T_A = 25^\circ\text{C}$ (unless otherwise noted) *

Parameter	Symbol	Min.	Max.	Unit
Clock frequency (I ² C-bus only)	f_{SCL}	10	400	kHz
Bus free time between a STOP and START condition	t_{BUF}	1.3	–	μs
Hold time (repeated) START condition. After this period, the first clock pulse is generated	$t_{HD;STA}$	0.6	–	μs
Set-up time for a repeated START condition	$t_{SU;STA}$	0.6	–	μs
Set-up time for STOP condition	$t_{SU;STO}$	0.6	–	μs
Data hold time	$t_{HD;DAT}$	0	–	ns
Data set-up time	$t_{SU;DAT}$	100	–	ns
LOW period of the SCL clock	t_{LOW}	1.3	–	μs
HIGH period of the SCL clock	t_{HIGH}	0.6	–	μs
Clock/data fall time	t_f	–	300	ns
Clock/data rise time	t_r	–	300	ns
Input pin capacitance	C_i	–	10	pF

* Specified by design and characterization; not production tested.

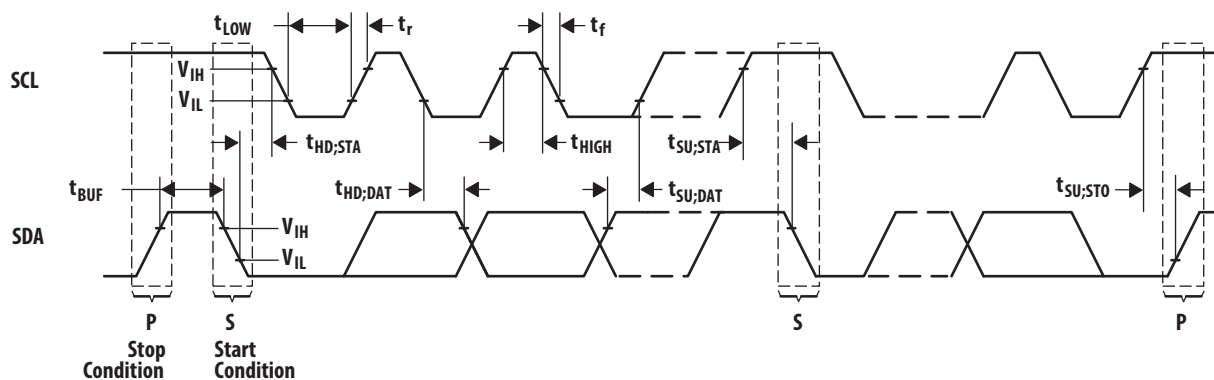


Figure 1. Timing Diagrams

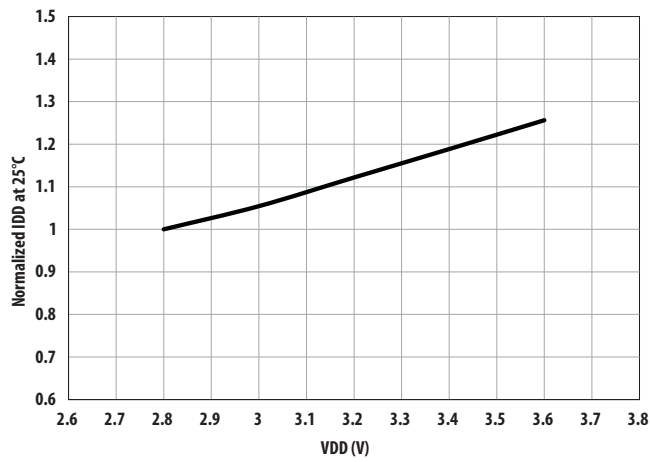


Figure 2. Normalized IDD (Suspend) vs. VDD.

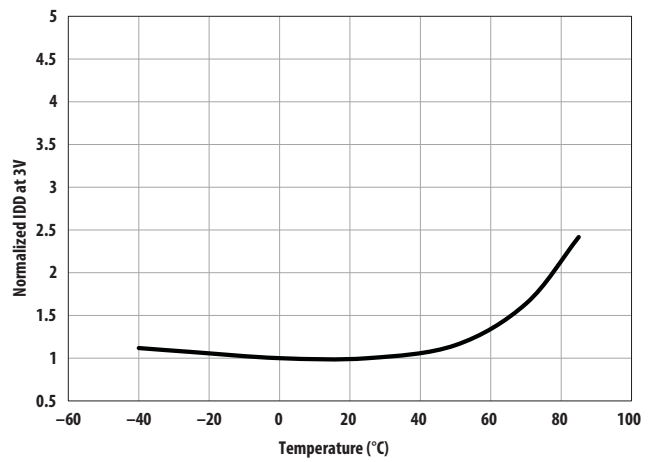


Figure 3. Normalized IDD (Suspend) vs. Temperature.

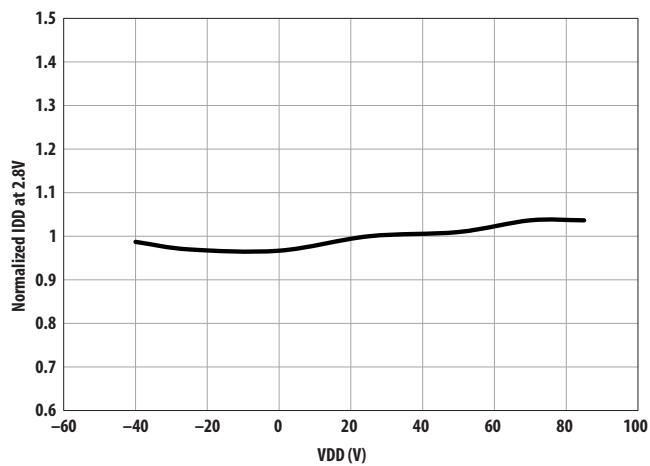


Figure 4. Normalized IDD (Standby1) vs. Temperature.

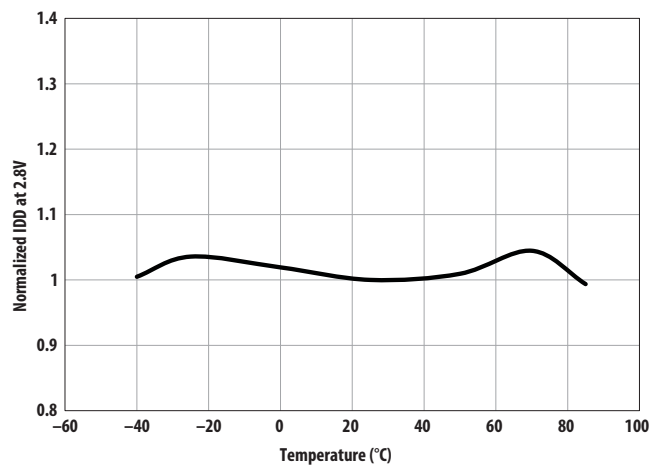


Figure 5. Normalized IDD (Standby2) vs. Temperature.

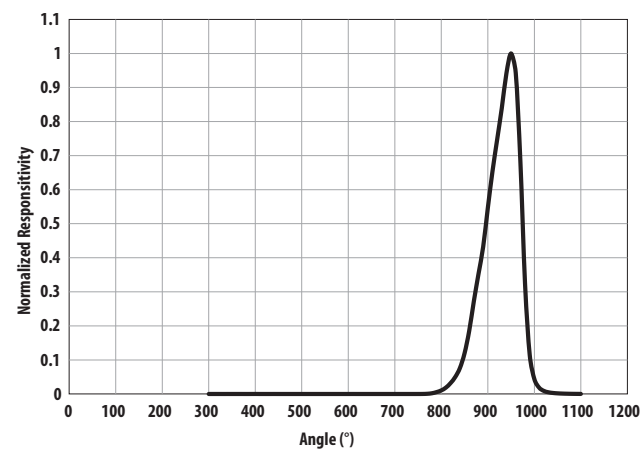


Figure 6. Normalized LED Spectral Radiant Intensity.

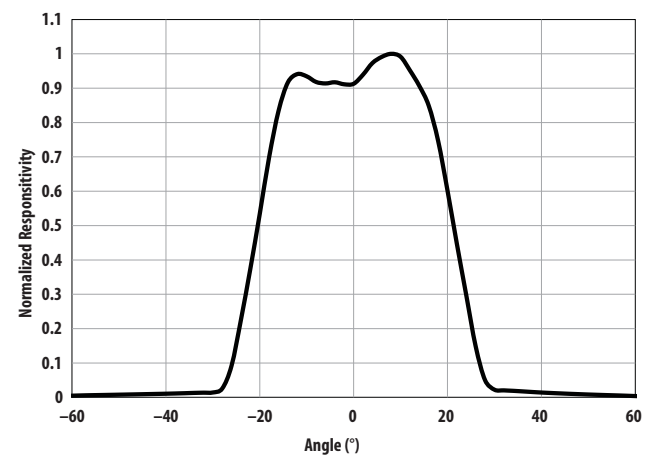


Figure 7. Normalized LED Angular Emitting Profile.

I²C-Bus Protocol

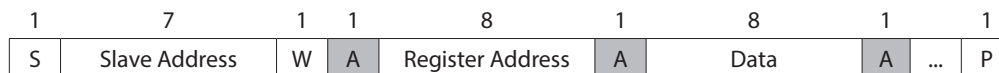
Interface and control are accomplished through an I²C-bus serial compatible interface (standard or fast mode) to a set of registers that provide access to device control functions and output data. The devices support the 7-bit I²C-bus addressing protocol.

The device supports a single slave address of 0x73 Hex using the 7-bit addressing protocol.

A	Acknowledge (0)
N	Not Acknowledged (1)
P	Stop Condition
R	Read (1)
S	Start Condition
Sr	Repeated Start Condition
W	Write (0)
...	Continuation of protocol
	Master-to-Slave
	Slave-to-Master

The I²C-bus standard provides for three types of bus transactions: read, write, and a combined protocol. During a write operation, the first byte written is a command byte followed by data. In a combined protocol, the first byte written is the command byte followed by reading a series of bytes. If a read command is issued, the register address from the previous command is used for data access. Likewise, if the MSB of the command is not set, the device writes a series of bytes at the address stored in the last valid command with a register address. The command byte contains either control information or a 5-bit register address. The control commands can also be used to clear interrupts.

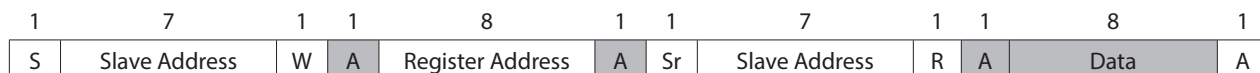
The I²C-bus protocol was developed by Philips (now NXP). For a complete description of the I²C-bus protocol, review the NXP I²C-bus design specification at <http://www.i2c-bus.org/references/>.



I²C-Bus Write Protocol



I²C-Bus Read Protocol



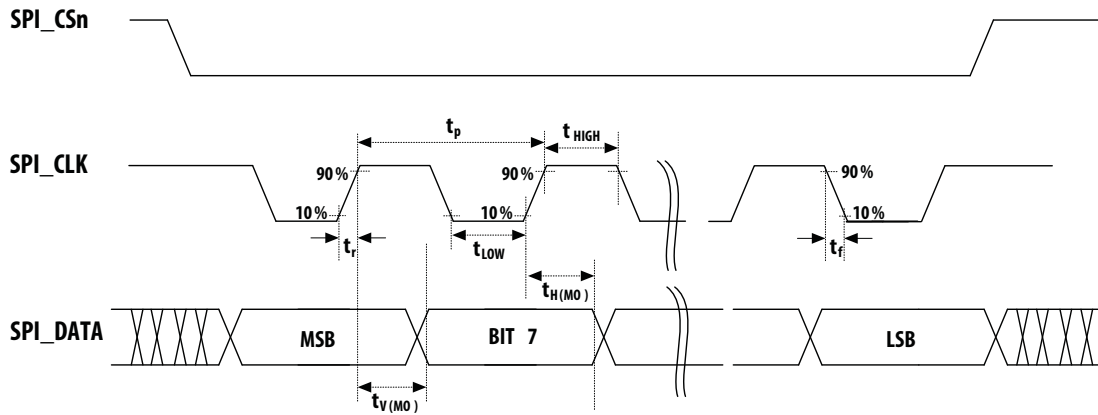
I²C-Bus Read Protocol – Combined Format

Figure 8. I²C-Bus Protocol.

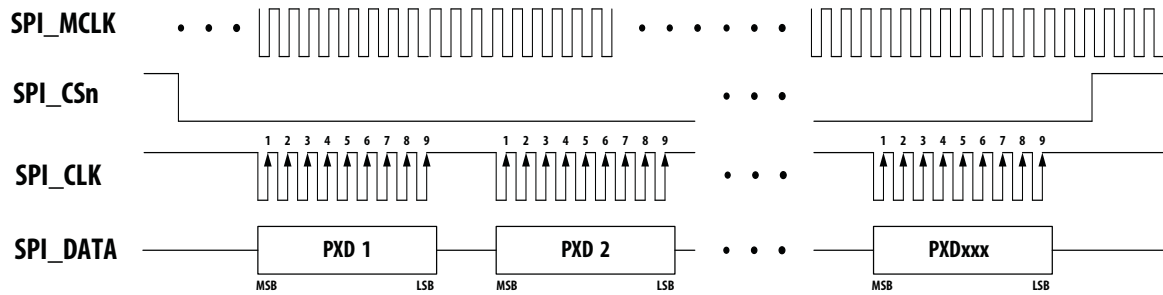
SPI Master Timing Characteristics

SPI Timing Parameter

Parameter	Symbol	Typ. (measured)	Unit
CLK clock frequency ($f_{CLK} = 1/t_p$)	f_{CLK}	24	MHz
Low period of CLK clock	t_{LOW}	15.5	ns
High period of CLK clock	t_{HIGH}	12.5	ns
Data output valid time	$t_{V(MO)}$	20.3	ns
Data output hold time	$t_{H(MO)}$	20.8	ns
Rise time of CLK clock	t_r	8	ns
Fall time of CLK clock	t_f	8	ns



SPI Master Protocol



xxx = 900 for 30x30 (Frame subtraction mode)

Power-On Sequence

In the power-on sequence, the VBUS must be powered-on before VDD. After power-on, wait T_1 μ s for the APDS-9500 to stabilize, after which write slave ID (0x73) to process I²C wake-up. After T_2 μ s, write the initial settings and the different modes settings to APDS-9500. Lastly, enable APDS-9500 by writing Register Bank1, Addr0x72 with 0x01. The gesture data can now be accessed through the I²C bus.

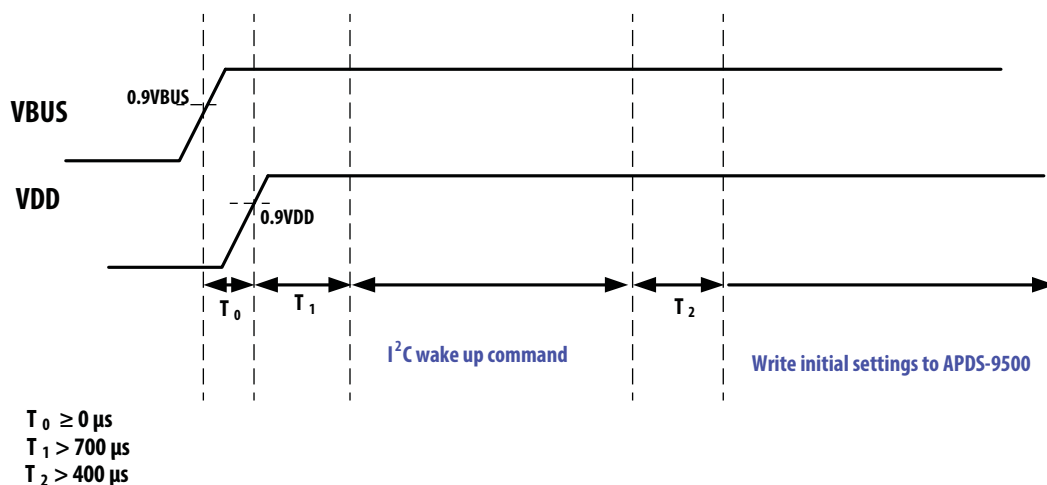


Figure 9. Power-On Timing Diagram.

Gesture Detection Operating State and State Machine

When in gesture detection, the state machine of APDS-9500 is in Figure 14. Following is a detailed description of each state.

Operation State (OP state)

When in operation state, the gesture update rate is 120 Hz for Normal Mode and 240 Hz for Gaming Mode. The gesture result can be accessed by interrupt mechanism or by continuously polling the gesture detection interrupt flag (Register Bank 0, Addr0x43).

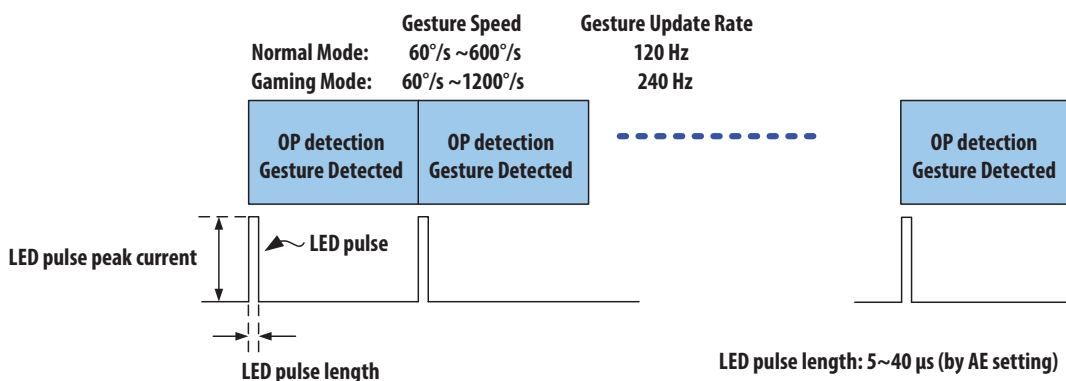


Figure 10. Operation State (OP state) Diagram.

Standby 1 State (S1 state)

When in Standby1 state, the object detection rate equals $S_{1, \text{Response Factor}}$ multiplied by the gesture update rate of Normal Mode or Gaming Mode.

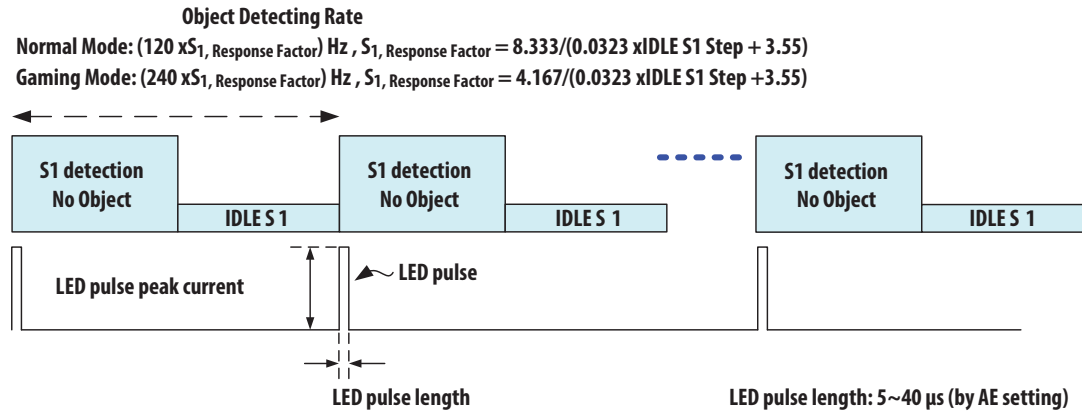


Figure 11. Standby 1 State (S1 state) Diagram.

Standby 2 State (S2 state)

When in Standby 2 state, the object detection rate equals $S_{2, \text{Response Factor}}$ multiplied by the gesture update rate of Normal Mode or Gaming Mode.

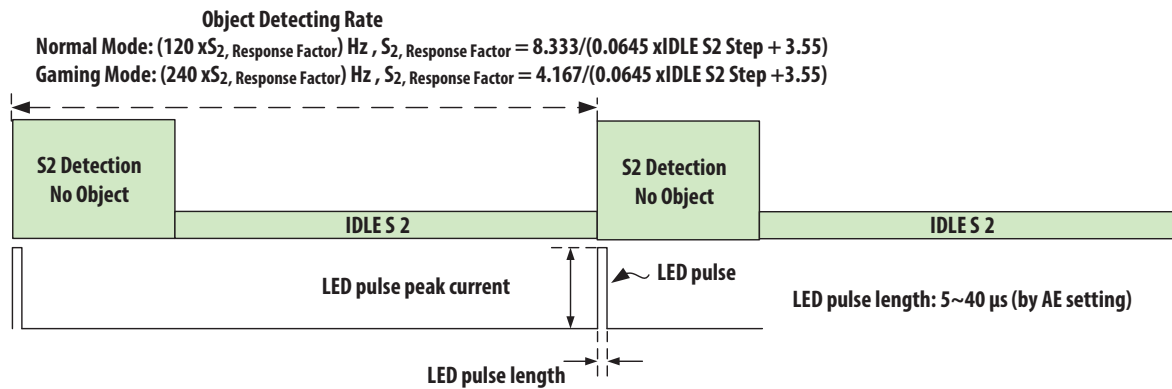


Figure 12. Standby 2 State (S2 state) Diagram.

Suspend State (SUS state)

To enter suspend state, disable the APDS-9500 by writing Register Bank 1. Then, write ADDR 0x72 with 0x00 and process the I²C suspend command by writing Register Bank 0, ADDR 0x03 with 0x01.

To exit suspend state, process the I²C wake-up command by writing the slave ID. Then, enable the APDS-9500 by writing Register Bank 1, ADDR 0x72 with 0x01.

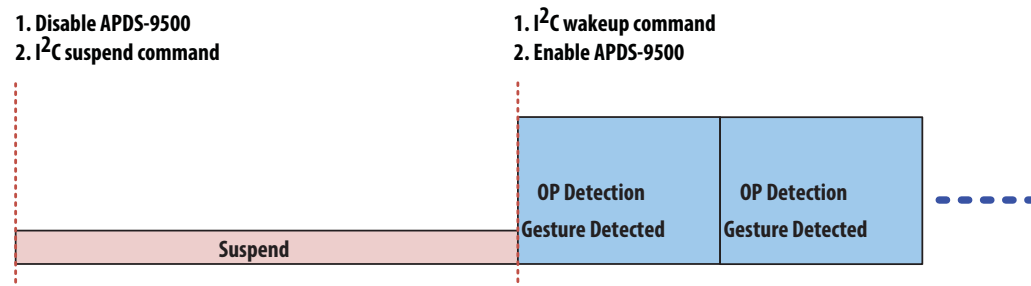
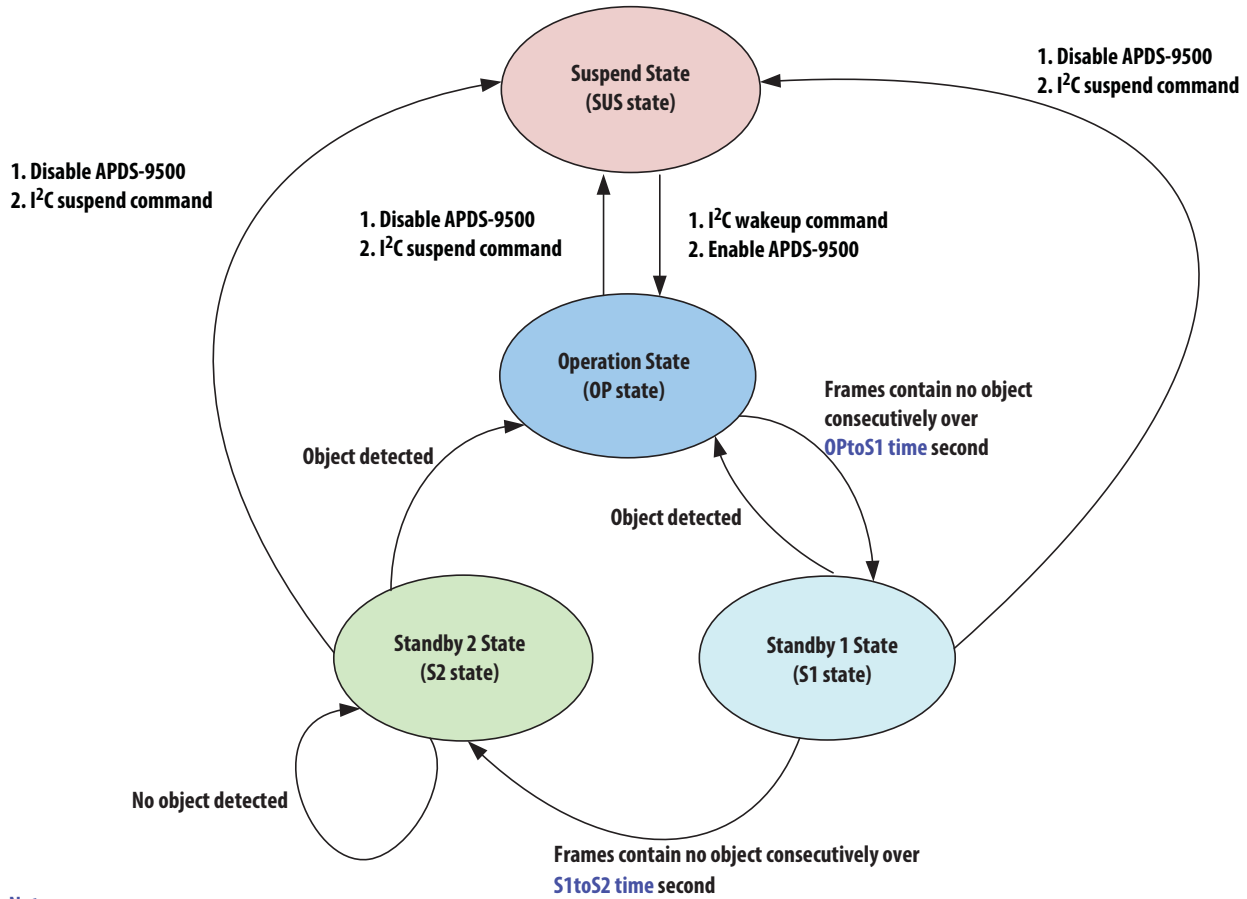


Figure 13. Suspend State (SUS state) Diagram.

State Machine



Note :

OPtoS1 time = OPtoS1 step/120 at Normal Mode
= OPtoS1 step/240 at Gaming Mode

S1toS2 time = S1toS2 step/(60 × S_{1, Response Factor}) at Normal Mode
= S1toS2 step/(120 × S_{1, Response Factor}) at Gaming Mode

Figure 14. State Machine of Gesture Detection.

Image Mode

Power-On Sequence

After power-on, wait T_1 μ s for APDS-9500 to stabilize. Then, write slave ID (0x73) to process I²C wake-up. After T_2 μ s, write the initial settings. Lastly, enable APDS-9500 by writing Register Bank1, Addr0x72 with 0x01 and then enable the SPI output by writing Register Bank 1, ADDR 0x7E with 0x01. The image data can now be accessed through the SPI bus.

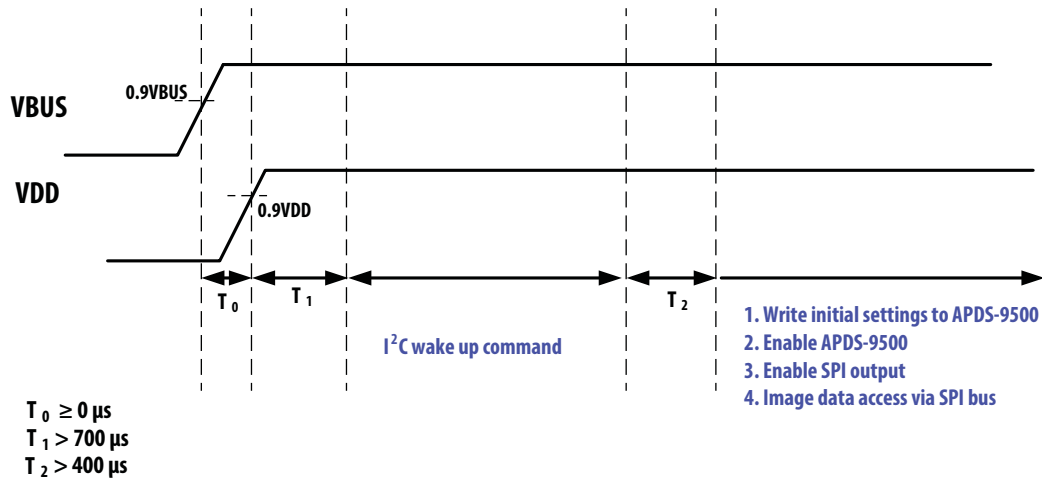


Figure 15. Power-On Timing Diagram.

Timing of Output Image

30 x 30, Frame Subtraction Mode

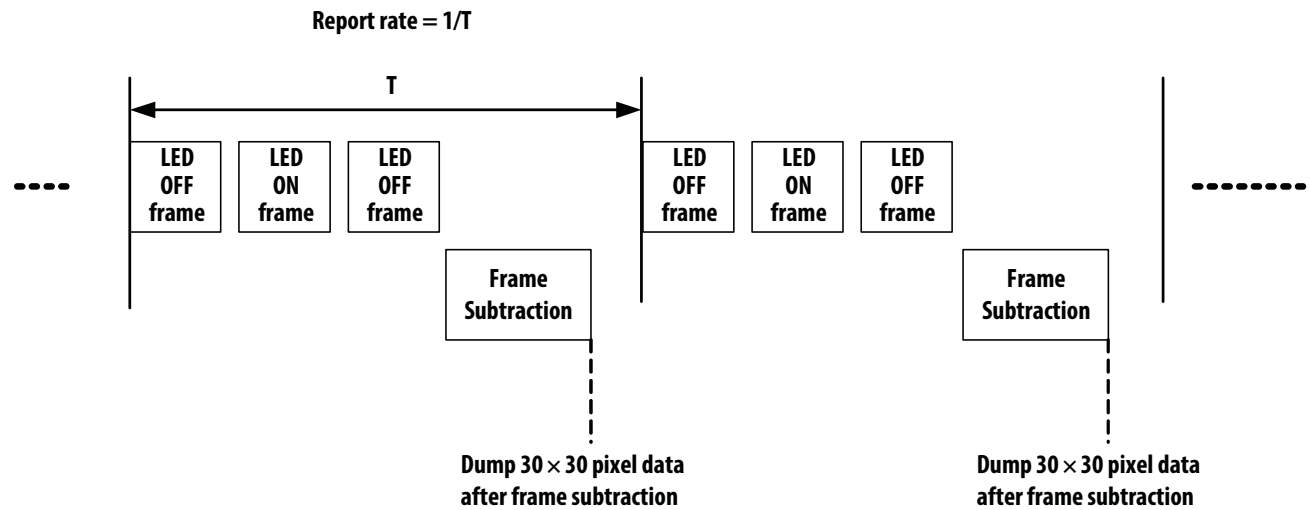


Figure 16. Timing of Output Image for 30 x 30 pixels.

60 x 60, Raw Data Mode

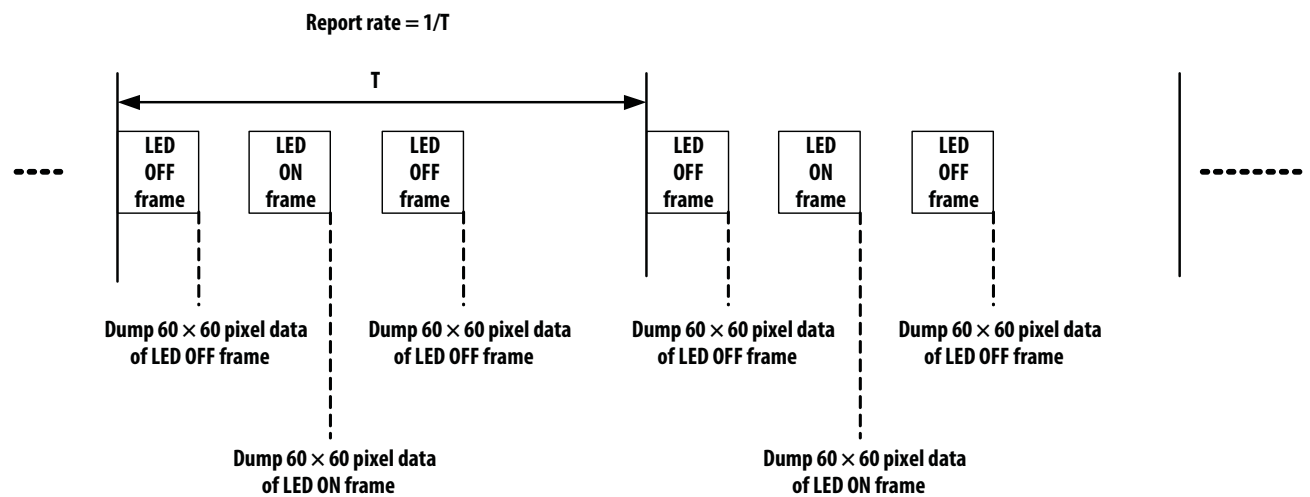


Figure 17. Timing of Output Image for 60 x 60 pixels.

Suspend State

To enter the suspend state, first disable the SPI output by writing Register Bank 1, ADDR 0x7E with 0x00.

Secondly, disable APDS-9500 by writing Register Bank 1, ADDR 0x72 with 0x00. Then, process the I²C suspend command by writing Register Bank 0, ADDR 0x03 with 0x00.

To exit the suspend state, first process the I²C wake-up command by writing the slave ID. Secondly, enable APDS-9500 by writing Register Bank 1, ADDR 0x72 with 0x01. Then, enable the SPI output by writing Register Bank 1, ADDR 0x7E with 0x01.

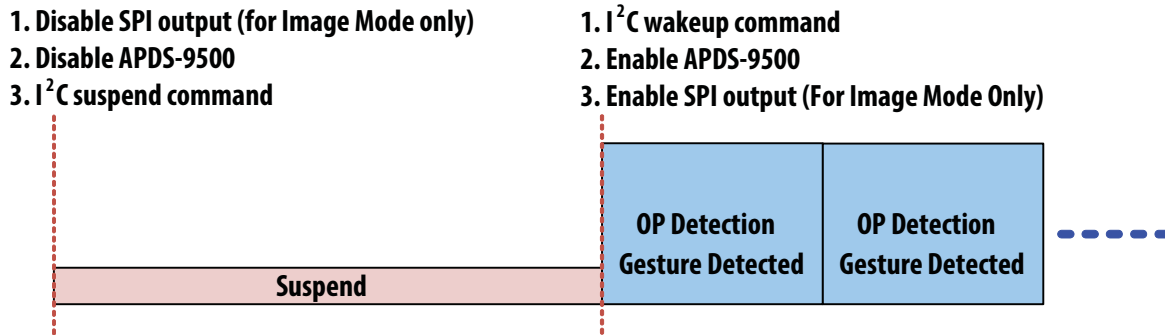


Figure 18. Suspend State (SUS State) Diagram.

Proximity Detection Mode

See Figure 19 to set the PS hysteresis window and the interrupt mechanism of proximity detection.

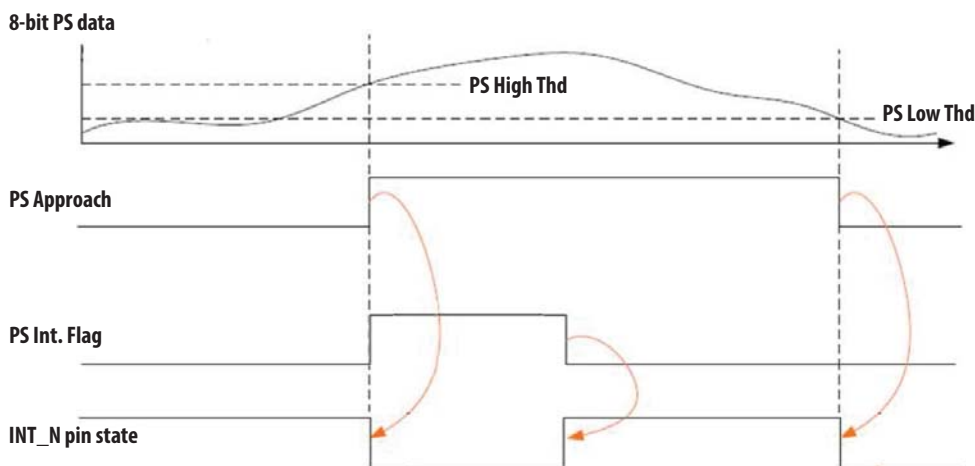


Figure 19. Proximity Sensing Functional Diagram.

When in proximity detection, the state machine of APDS-9500 is in Figure 21. Following is the detailed description of each state.

Proximity Operation State (PS OP state)

When in operation state, the update rate is 10 Hz and the LED on time is 8 μ s. The LED peak current is 760 mA.

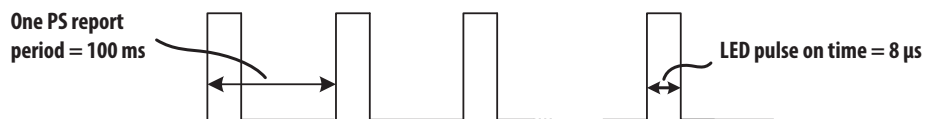


Figure 20. Proximity Operation State (PS OP state) Diagram.

Suspend State (SUS state)

1. Disable APDS-9500
2. I²C suspend command

1. I²C wakeup command
2. Enable APDS-9500

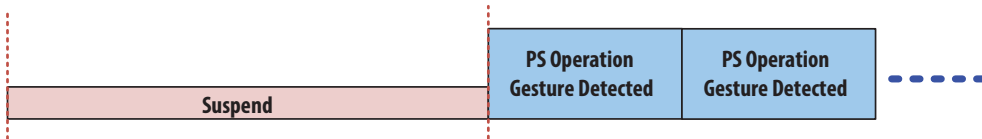


Figure 21. Suspend State (SUS) Diagram.

State Machine

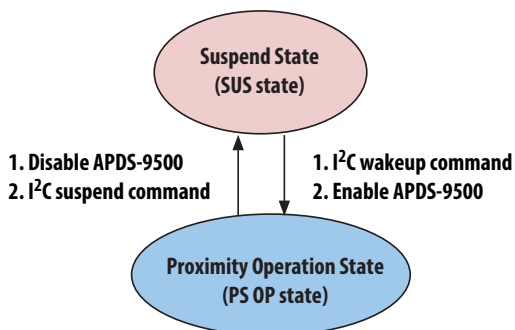


Figure 22. State Machine of Gesture Detection.

Register Bank Select

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0/1	R_RegBankSel	0xEF	0	0x00	R/W	Register bank select 0: Register bank 0 1: Register bank 1

Image Size Setting

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	R_ImageHeight	0xAA	5:0	0x1E	R/W	DSP image vertical size
0	R_ImageWidth	0xAB	5:0	0x1E	R/W	DSP image horizontal size
1	Cmd_HSize	0x00	5:0	0x1E	R/W	Horizontal size
1	Cmd_VSize	0x01	5:0	0x1E	R/W	Vertical size
1	Cmd_HStart	0x02	5:0	0x00	R/W	Horizontal start point
1	Cmd_VStart	0x03	5:0	0x00	R/W	Vertical start point
1	Cmd_ASkip_V	0x04	5	0x01	R/W	Analog vertical skip
1	Cmd_ASkip_H	0x04	4	0x01	R/W	Analog horizontal skip
1	Cmd_DAvg_V	0x04	3	0x00	R/W	Digital vertical average
1	Cmd_VFlip	0x04	1	0x00	R/W	Vertical flip
1	Cmd_HFlip	0x04	0	0x00	R/W	Horizontal flip

Setting 30x30 Pixels Image Output Mode

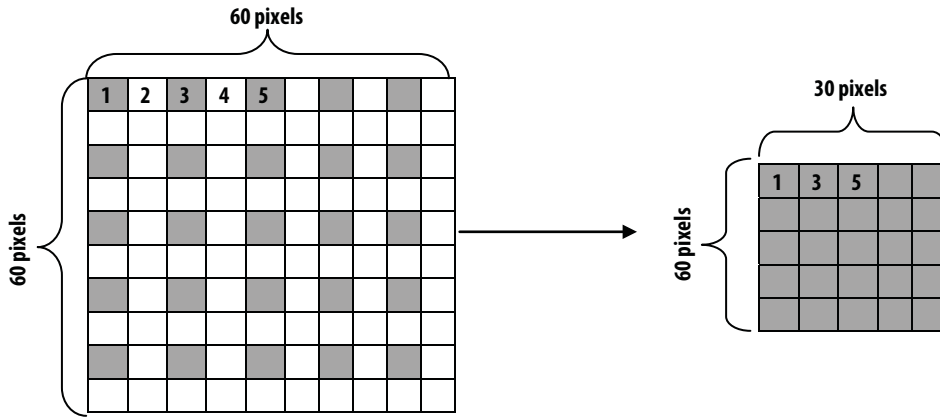
Bank	Register Name	Addr	Bits	2x Skip Mode	2x2 Average Mode	WOI Mode
0	R_ImageHeight	0xAA	5:0	30	30	30
0	R_ImageWidth	0xAB	5:0	30	30	30
1	Cmd_HSize	0x00	5:0	30	30	30
1	Cmd_VSize	0x01	5:0	30	60	30
1	Cmd_HStart	0x02	5:0	0	0	15
1	Cmd_VStart	0x03	5:0	0	0	15
1	R_LS_Comp_DAvg_V	0x04	7	0	1	0
1	R_LS_Comp_DAvg_H	0x04	6	0	0	0
1	Cmd_ASkip_V	0x04	5	1	0	0
1	Cmd_ASkip_H	0x04	4	1	1	0
1	Cmd_DAvg_V	0x04	3	0	1	0
1	Cmd_DAvg_H	0x04	2	0	0	0

Hardware connection

Module pin out #15 (SPI_MCLK) must be supplied with 24 MHz clock. Pin out for #4(SPI_Data), #5(SPI_CLK) and #16(SPI_Csn) are SPI output signal for imaging data.

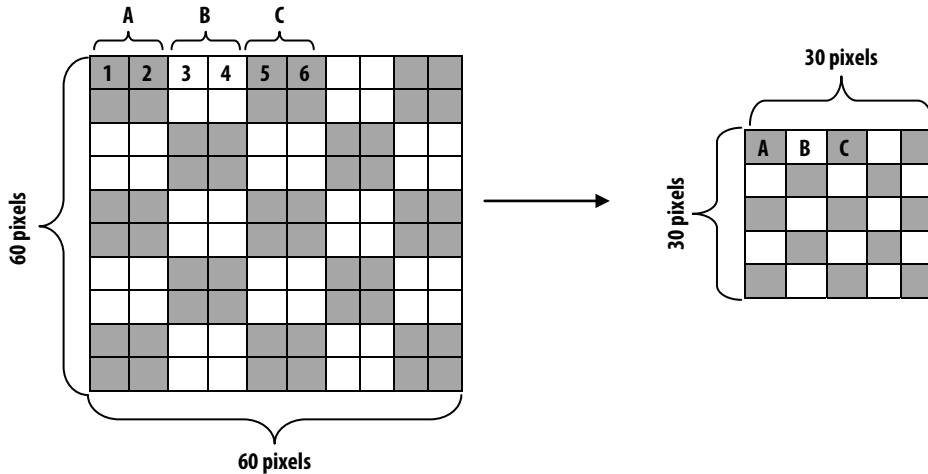
Skip mode

1st, 3rd, 5th....pixels information are used for image formation; the information of 2nd, 4th, 6th....pixel are not used for image formation.



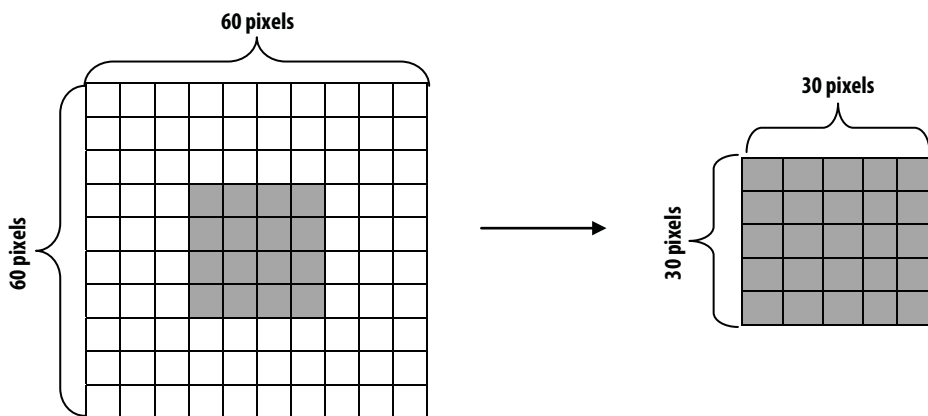
Average Mode

1st and 2nd pixels are averaged as 1-pixel information. Similarly, 3rd and 4th pixels are averaged as 1-pixel information. Hence a total of 60×60 pixels information are averaged to become 30×30 pixels information.



Crop Mode

60×60 pixels are cropped with the center 30×30 pixels being used for image formation.



AE/AG Controls

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	R_AELedOff_UB	0x46	7:0	0x60	R/W	Decrease exposure time, if AE_LED_Off_YAvg (Bank 0, Reg 0x58) > R_AELedOff_UB
0	R_AELedOff_LB	0x47	7:0	0x20	R/W	Increase exposure time, if AE_LED_Off_YAvg (Bank 0, Reg 0x58) < R_AELedOff_LB
0	R_AE_Exposure_UB	0x48	7:0	0x20	R/W	Low byte of auto exposure time up bound, exposure time (μ s) = R_AE_Exposure_UB/4
0	R_AE_Exposure_UB	0x49	7:0	0x03	R/W	High byte of auto exposure time up bound, exposure time (μ s) = R_AE_Exposure_UB/4
0	R_AE_Exposure_LB	0x4A	7:0	0xC8	R/W	Low byte of auto exposure time low bound, R_AE_Exposure_LB = R_AE_Exposure_UB/2
0	R_AE_Exposure_LB	0x4B	7:0	0x00	R/W	High byte of auto exposure time low bound, R_AE_Exposure_LB = R_AE_Exposure_UB / 2
0	R_AE_Gain_UB	0x4C	7:0	0x14	R/W	Auto gain up bound, gain = 1 + R_AE_Gain_UB/16
0	R_AE_Gain_LB	0x4D	7:0	0x00	R/W	Auto gain low bound, gain = 1 + R_AE_Gain_LB/16
0	R_AE_Gain_Step	0x4E	3:0	0x0A	R/W	Gain stage adjust step, new gain stage = current gain stage – R_AE_Gain_Step
0	R_SleepAEAG_AutoDisable	0x4E	4	0x01	R/W	In sleep mode, if gain stage is at UB, disable wakeup AE mode Wake up AE mode : in sleep mode, AE/AG will be set to higher value
0	R_AE_Gain_Default	0x4F	6:0	0x14	R/W	Gain stage default value when AE is turned on
0	R_Exp_Sel	0x50	2:0	0x00	R/W	At R_Manual_Exposure_Default = 1, exp = R_AE_Exposure_UP / 2 ^{R_Exp_Sel}
0	R_Manual_GG	0x51	0	0x01	R/W	1: Gain manual mode, ggh gain = R_ggh, global gain = R_global 0: Gain auto mode
0	R_Manual_Exposure	0x51	1	0x00	R/W	1: Exposure manual mode, exposure time = R_AE_Exposure_UB 0: Auto exposure mode
0	R_Manual_Exposure_Default	0x51	2	0x01	R/W	Exp manual mode, exp = R_AE_Exposure_UP / 2 ^{R_Exp_Sel}
0	R_AE_EnH	0x51	4	0x00	R/W	1: AE Enable 0: AE Disable
0	AG_stage_GG	0x54	7:0	-	R	Gain stage, gain for analog = 2 ^{AG_stage_GG[7:4]}
0	Reg_ExposureNum	0x55	7:0	-	R	Low byte of current exposure time, exposure time (μ s) = Reg_ExposureNum/4
0	Reg_ExposureNum	0x56	7:0	-	R	High byte of current exposure time, exposure time (μ s) = Reg_ExposureNum/4
0	Reg_ggh	0x57	1:0	-	R	ggh gain 0: 1x 1: 2x 3: 4x
0	Reg_global	0x57	7:4	-	R	Global gain 0: 1x 8: 2x Total gain = ggh $\times$ global = 2 ^{AG_stage_GG[7:4]}
0	AE_LED_Off_YAvg	0x58	7:0	-	R	Off frame average brightness
0	AE_Dec	0x59	0	-	R	AE decrease identifier
0	AE_Inc	0x59	1	-	R	AE increase identifier
0	AE_Normal_Factor	0x5A	2:0	-	R	AE normalize factor, DSP data = sensor image data $\times$ 2 ^{AE_Normal_Factor}
1	R_global	0x42	7:4	0x08	R/W	PGA global gain in gain manual mode
1	R_ggh	0x44	7:6	0x00	R/W	PGA ggh gain in gain manual mode

GPIO Setting

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	Im_GPIO0	0x80	0	-	R	Bit[0]: GPIO0 status at input mode
0	Tm_GPIO0_OEL	0x80	1	0x01	R/W	Bit[1]: 0: Set GPIO0 as output
0	Im_GPIO0_IEL	0x80	2	0x01	R/W	Bit[2]: 0: Set GPIO0 as input
0	R_GPIO0	0x80	3	0x01	W	Bit[3]: GPIO0 default value at output mode
To set GPIOX as input, X = 0~3 : Set: Tm_GPIOX_OEL = 1, Tm_GPIOX_IEB = 0 At this time, Im_GPIOX = GPIO status To set GPIOX as Output, X = 0~3 : Set: Tm_GPIOX_OEL = 0, Tm_GPIOX_IEB = 1 At this time, Tm_GPIOX = R_GPIOX						
0	Im_GPIO1	0x80	4	-	R	Bit[4]: GPIO1 status at input mode
0	Tm_GPIO1_OEL	0x80	5	0x01	R/W	Bit[5]: 0: Set GPIO1 as output
0	Im_GPIO1_IEL	0x80	6	0x01	R/W	Bit[6]: 0: Set GPIO1 as input
0	R_GPIO1	0x80	7	0x01	W	Bit[7]: GPIO1 default value at output mode
0	Im_GPIO2	0x81	0	-	R	Bit[0]: GPIO2 status at input mode
0	Tm_GPIO2_OEL	0x81	1	0x01	R/W	Bit[1]: 0: Set GPIO2 as output
0	Im_GPIO2_IEL	0x81	2	0x01	R/W	Bit[2]: 0: Set GPIO2 as input
0	R_GPIO2	0x81	3	0x01	W	Bit[3]: GPIO2 default value at output mode
0	Im_GPIO3	0x81	4	-	R	Bit[4]: GPIO3 status at input mode
0	Tm_GPIO3_OEL	0x81	5	0x01	R/W	Bit[5]: 0: Set GPIO3 as output
0	Im_GPIO3_IEL	0x81	6	0x01	R/W	Bit[6]: 0: Set GPIO3 as input
0	R_GPIO3	0x81	7	0x01	W	Bit[7]: GPIO3 default value at output mode
0	Im_INT	0x82	0	-	R	Bit[0]: INT pin status
0	Tm_INT_OEL	0x82	1	0x00	R/W	Bit[1]: 0: Set INT as output
0	Im_INT_IEL	0x82	2	0x01	R/W	Bit[2]: 0: Set INT as input
0	Tm_INT	0x82	3	0x01	W	Bit[3]: INT default value at output mode
To set INT as Input : Set: Tm_INT_OEL = 1, Tm_INT_IEB = 0 At this time, Im_INT = INT_N Status To set INT as Output : Set: Tm_INT_OEL = 0, Tm_INT_IEB = 1 At this time, Tm_INT = INT_N						

Interrupt Controls

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	R_MCU_IntFlag-GClr	0x40	1	0x01	R/W	1: Auto clean intflag_1 / Intflag_2 after I ² C readout
0	R_MCU_IntFlag-Inv	0x40	4	0x00	R/W	1: INT pin high active 0: INT pin low active
0	R_Int_1_En	0x41	7:0	0xFF	R/W	If the corresponding bit is 1, the corresponding interrupt event is enabled Bit[0] Up Bit[1] Down Bit[2] Left Bit[3] Right Bit[4] Forward Bit[5] Backward Bit[6] Clockwise Bit[7] Counterclockwise
0	R_Int_2_En	0x42	7:0	0xFF	R/W	If the corresponding bit is as follows, the corresponding interrupt event is enabled: Bit[0] Wave, wave mode only Bit[1] Proximity, proximity mode only Bit[2] Has Object, cursor mode only Bit[3] Wake up trigger, trigger mode only Bit[4] N/A Bit[5] N/A Bit[6] N/A Bit[7] No Object, cursor mode only
0	IntFlag_1	0x43	7:0	-	R	When interrupt event happens, the corresponding bit is set to 1: Bit[0] Up Bit[1] Down Bit[2] Left Bit[3] Right Bit[4] Forward Bit[5] Backward Bit[6] Clockwise Bit[7] Counterclockwise
0	IntFlag_2	0x44	7:0	-	R	When interrupt event happens, the corresponding bit is set to 1: Bit[0] Wave, wave mode only Bit[1] Proximity, proximity mode only Bit[2] Has Object, cursor mode only Bit[3] Wake up trigger, trigger mode only Bit[4] N/A Bit[5] N/A Bit[6] N/A Bit[7] No Object, cursor mode only

Gesture Mode Controls

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	R_LightThd	0x83	7:0	0x20	R/W	If pixel > R_LightThd, it is taken as part of the object Example : Pixel array : [10 10 40 40 40 10], R_LightThd = 32 ≥ Object array: [0 0 40 40 40 0]
0	R_GestureStartTh	0x84	7:0	0x20	R/W	Low byte of gesture start threshold, if object size > R_GestureStartTh, state machine goes to "Has Object" state
0	R_GestureStartTh	0x85	1:0	0x00	R/W	High byte of gesture start threshold, if object size > R_GestureStartTh, state machine goes to "Has Object" state
0	R_GestureEndTh	0x86	7:0	0x10	R/W	Low byte of gesture end threshold, if object size < R_GestureEndTh, state machine goes out of "Has Object" state
0	R_GestureEndTh	0x87	1:0	0x00	R/W	High byte of gesture end threshold, if object size < R_GestureEndTh, state machine goes out of "Has Object" state
0	R_ObjectMinZ	0x88	4:0	0x05	R/W	Z direction minimum threshold
0	R_ObjectMaxZ	0x89	5:0	0x18	R/W	Z direction maximum threshold
0	R_ProcessResolution	0x8C	5:4	0x03	R/W	Resolution of the gesture detection object
0	R_TimeDelayNum	0x8D	7:0	0x00	R/W	Detection gap between this gesture and the next gesture
0	R_Disable45Degree	0x8E	0	0x00	R/W	45 degree gesture detection 1: Disable 45 degree gesture detection
0	R_45DegreeRatio	0x8E	7:4	0xF0	R/W	Ratio to define 45 degree
0	R_XtoYGain	0x8F	0	0x01	R/W	X and Y direction gain enable
0	R_XYGainRatio	0x8F	7:4	0x08	R/W	X and Y direction gain ratio
0	R_NoMotionCountThd	0x90	6:0	0x0C	R/W	"No Motion Counter Threshold" to quit "Has Motion" state
0	R_NoObjectCountThd	0x91	6:0	0x06	R/W	"No Object Counter Threshold" to quit "Has Object" state
0	R_NormalizedImageWidth	0x92	4:0	0x1E	R/W	Image normalized factor
0	R_XDirectionThd	0x93	4:0	0x0D	R/W	Gesture detection horizontal threshold
0	R_YDirectionThd	0x94	4:0	0x0A	R/W	Gesture detection vertical threshold
0	R_ZDirectionThd	0x95	4:0	0x0C	R/W	Gesture detection z direction threshold
0	R_ZDirectionXYThd	0x96	4:0	0x0A	R/W	Gesture detection x and y threshold to detect forward or backward
0	R_ZDirectionAngleThd	0x97	3:0	0x04	R/W	Gesture detection angle threshold to detect forward or backward
0	R_RotateAngleThd	0x98	5:0	0x0A	R/W	Gesture detection angle threshold to detect rotation
0	R_RotateContiEnH	0x99	0	0x01	R/W	Continuous rotation gesture detection enable
0	R_RotateContiThd	0x99	5:1	0x00	R/W	Continuous rotation gesture detection angle threshold
0	R_RotateXYThd	0x9A	4:0	0x0A	R/W	Gesture detection x and y threshold to detect rotation
0	R_RotateZThd	0x9B	4:0	0x0A	R/W	Gesture detection z threshold to detect rotation
0	R_FilterWeight	0x9C	1:0	0x03	R/W	IIR filter weight between frame position distance
0	R_FilterDistThd	0x9C	6:2	0x0A	R/W	IIR filter frame position distance threshold
0	R_StartDistThd	0x9D	3:0	0x03	R/W	Object position difference between frames threshold to enter the process state
0	R_EndDistThd	0x9D	6:4	0x03	R/W	Object position difference between frames threshold to quit the process state
0	R_RotateEnH	0x9F	4	0x01	R/W	Rotate gesture detection enable
0	R_ZDirectionEnH	0x9F	5	0x01	R/W	Backward and forward gesture detection enable
0	R_YDirectionEnH	0x9F	6	0x01	R/W	Up and down gesture detection enable
0	R_XDirectionEnH	0x9F	7	0x01	R/W	Left and right gesture detection enable
0	R_FilterImage	0xA5	0	0x01	R/W	Refer to R_FilterAverage_Mode

0	R_FilterAverage_Mode	0xA5	3:2	0x00	R/W	Image filter mode 0: Weak average 1: Strong average 2: 3 out of 9 median average
0	R_UseLightWeight	0xA5	4	0x01	R/W	Use pixel brightness as weight to calculate center enable
0	R_DiffAngleThd	0xA9	3:0	0x04	R/W	Frame angle accumulation threshold
0	ObjectCenterX	0xAC	7:0	-	R	Low byte of horizontal object center
0	ObjectCenterX	0xAD	4:0	-	R	High byte of horizontal object center
0	ObjectCenterY	0xAE	7:0	-	R	Low byte of vertical object center
0	ObjectCenterY	0xAF	4:0	-	R	High byte of vertical object center
0	ObjectAvgY	0xB0	7:0	-	R	Object brightness, maximum = 255
0	ObjectSize	0xB1	7:0	-	R	Low byte of object size, maximum = 900
0	ObjectSize	0xB2	3:0	-	R	High byte of object size, maximum = 900
0	Gx	0xB3	5:0	-	R	Gesture x direction movement
0	Gy	0xB4	5:0	-	R	Gesture y direction movement
0	Gz	0xB5	6:0	-	R	Gesture z direction movement
0	GestureResult	0xB6	3:0	-	R	Gesture result 1: Up 2: Down 3: Left 4: Right 5: Forward 6: Backward 7: Clockwise 8: Counterclockwise 9: Wave 10: N/A
0	State	0xB6	5:4	-	R	DSP FSM state 0: Initial 1: Process 2: End
0	WaveCount	0xB7	3:0	-	R	Wave gesture counter
0	AbortCount	0xB7	6:4	-	R	Abort gesture counter
0	NoObjectCount	0xB8	7:0	-	R	No object counter
0	NoMotionCount	0xB9	7:0	-	R	No motion counter
0	LightCount	0xBA	5:0	-	R	Bright object counter
0	LightAcc	0xBB	7:0	-	R	Low byte of object brightness accumulation
0	LightAcc	0xBC	1:0	-	R	High byte of object brightness accumulation
0	TimeAcc	0xBD	7:0	-	R	Low byte of gesture time period
0	TimeAcc	0xBE	4:0	-	R	High byte of gesture time period
0	AngleAcc	0xC7	7:0	-	R	Low byte of gesture angle accumulation
0	AngleAcc	0xC8	2:0	-	R	High byte of gesture angle accumulation
0	XGainValue	0xCA	7:0	-	R	45 degree gesture detection x direction parameter
0	YGainValue	0xCB	7:0	-	R	45 degree gesture detection y direction parameter
0	R_YtoZSum	0xCC	5:0	0x1A	R/W	Z direction mapping parameter
0	R_YtoZFactor	0xCD	5:0	0x0D	R/W	Z direction mapping parameter
0	R_PositionFilterLength	0xCE	2:0	0x03	R/W	IIR filter length for cursor object center
0	R_ProcessFilterLength	0xCE	6:4	0x00	R/W	IIR filter length for gesture object center
0	R_WaveCountThd	0xCF	3:0	0x03	R/W	Wave gesture counter threshold
0	R_WaveAngleThd	0xCF	7:4	0x06	R/W	Wave gesture angle threshold
0	R_AbortCountThd	0xD0	2:0	0x22	R/W	Abort gesture counter threshold
0	R_AbortXYRatio	0xD0	7:3		R/W	Abort gesture x and y direction ratio
0	R_AbortLength	0xD1	6:0	0x0F	R/W	Abort gesture movement distance threshold
0	R_AbortIntervalCountThd	0xD2	5:0	0x88	R/W	Abort gesture gap duration threshold
0	R_ConfirmMode	0xD2	6		R/W	Confirm mode enable

0	R_WaveEnH	0xD2	7		R/W	Wave gesture detection enable
0	PositionFilterCenterX	0xD3	7:0	-	R	Low byte of horizontal object center after IIR filter for cursor mode
0	PositionFilterCenterX	0xD4	3:4	-	R	High byte of horizontal object center after IIR filter for cursor mode
0	PositionFilterCenterY	0xD4	7:4	-	R	High byte of vertical object center after IIR filter for cursor mode
0	PositionFilterCenterY	0xD5	7:0	-	R	Low byte of vertical object center after IIR filter for cursor mode
0	PositionFilterAvgY	0xD6	7:0	-	R	Low byte of object brightness after IIR filter for cursor mode
0	PositionFilterAvgY	0xD7	0	-	R	High byte of object brightness after IIR filter for cursor mode
0	PositionFilterSize	0xD7	5:4	-	R	High byte of object size after IIR filter for cursor mode
0	PositionFilterSize	0xD8	7:0	-	R	Low byte of object size after IIR filter for cursor mode
0	ProcessFilterCenterX	0xDA	7:0	-	R	Low byte of horizontal object center after IIR filter for gesture detection
0	ProcessFilterCenterX	0xDB	3:0	-	R	High byte of horizontal object center after IIR filter for gesture detection
0	ProcessFilterCenterY	0xDB	7:4	-	R	High byte of vertical object center after IIR filter for gesture detection
0	ProcessFilterCenterY	0xDC	7:0	-	R	Low byte of vertical object center after IIR filter for gesture detection
0	ProcessFilterSize	0xD9	1:0	-	R	High byte of object size after IIR filter for gesture detection
0	ProcessFilterSize	0xDD	7:0	-	R	Low byte of object size after IIR filter for gesture detection
0	ProcessFilterAvgY	0xD9	2	-	R	High byte of object brightness after IIR filter for gesture detection
0	ProcessFilterAvgY	0xDE	7:0	-	R	Low byte of object brightness after IIR filter for gesture detection
0	AbortIntervalCount	0xD9	5:4	-	R	High byte of abort gesture gap duration
0	AbortIntervalCount	0xDF	7:0	-	R	Low byte of abort gesture gap duration

Cursor Mode Controls

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	R_CursorUseTop	0x32	0	0x01	R/W	1: Enable cursor center function If enabled, the cursor object information will involve R_CursorTopRatio for the calculation Refer to R_CursorTopRatio
0	R_CursorUseBGModel	0x32	1	0x00	R/W	1: If cursor center function is enabled, the cursor object information will involve background model for the calculation
0	R_CursorInvertY	0x32	2	0x00	R/W	1: Horizontal cursor center inverse
0	R_CursorInvertX	0x32	3	0x01	R/W	1: Vertical cursor center inverse
0	R_CursorTopRatio	0x32	5:4	0x02	R/W	Use the first R_CursorTopRatio / 8 part of the object to calculate center Example : If the original object size is 100, the cursor object size will be 25 at R_CursorTopRatio = 2, The cursor center will move from the center of the object to the upper 1/4 part
0	R_PositionFilterStartSizeTh	0x33	7:0	0x01	R/W	Low byte As object size > R_PositionFilterStartSizeTh, use the object information after IIR filter for cursor

0	R_PositionFilterStartSizeTh	0x34	0	0x00	R/W	High byte As object size > R_PositionFilterStartSizeTh, use the object information after IIR filter for cursor
0	R_ProcessFilterStartSizeTh	0x35	7:0	0x01	R/W	Low byte As object size > R_ProcessFilterStartSizeTh, use the object information after IIR filter for gesture detection
0	R_PositionFilterStartSizeTh	0x36	0	0x00	R/W	High byte As object size > R_ProcessFilterStartSizeTh, use the object information after IIR filter for gesture detection
0	R_CursorClampLeft	0x37	4:0	0x09	R/W	Minimum cursor horizontal center value, if data < R_CursorClampLeft, clamp at 0 else, clamp at data – (R_CursorClampLeft << R_PositionResolution) Example : R_CursorClampLeft = 9, R_CursorClampRight = 21, R_PositionResolution = 0, $X = 5 \geq X' = 0$ $X = 10 \geq X' = 10 - 9 = 1$ $X = 25 \geq X' = 20 - 1 - 9 = 10$ Example : R_CursorInvertY = 0, R_CursorClampRight = 20, R_PositionResolution = 2, Y = 50, set R_CursorInvertY = 1, R_CursorClampRight = 20, R_PositionResolution = 2, $\geq Y' = 20 \times (2^2) - 50 = 30$
0	R_CursorClampRight	0x38	4:0	0x15	R/W	Maximum cursor horizontal center value, if data is > R_CursorClampRight, clamp at –1
0	R_CursorClampUp	0x39	4:0	0x0A	R/W	Minimum cursor vertical center value, if data < R_CursorClampUp, clamp at 0 else, clamp at data – (R_CursorClampUp << R_PositionResolution)
0	R_CursorClampDown	0x3A	4:0	0x12	R/W	Maximum cursor vertical center value, if data is > R_CursorClampDown, clamp at –1
0	CursorClampCenterX	0x3B	7:0	0x00	R	Low byte of clamping center X
0	CursorClampCenterX	0x3C	3:0	0x7D	R	High byte of clamping center X
0	CursorClampCenterY	0x3D	7:0	0x0F	R	Low byte of clamping center Y
0	CursorClampCenterY	0x3E	3:0	0xA0	R	High byte of clamping center Y
0	R_Cursor_ObjectSizeTh	0x8B	7:0	0x10	R/W	Object size threshold for cursor mode, if cursor object size > R_Cursor_ObjectSizeTh, trigger cursor has object interrupt
0	R_PositionResolution	0x8C	2:0	0x07	R/W	Cursor mode object resolution, Example : (x,y) = (1000,1000) at R_PositionResolution = 7, $\Rightarrow \text{real}(x',y') = (1000 / 2^7, 1000/2^7) = (7.8125, 7.8125)$ which maps to 30x30 scale

Proximity Mode Controls

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	R_Prox_UB	0x69	7:0	0xC8	R/W	Proximity up bound
0	R_Prox_LB	0x6A	7:0	0x40	R/W	Proximity low bound
0	S_State	0x6B	7:0	-	R	PS approach state 1: Approach, ($S_AvgY \geq R_Pox_UB$) 0: Not approach, ($S_AvgY \leq R_Pox_LB$) (Only functional at proximity detection mode)
0	S_AvgY	0x6C	7:0	-	R	Proximity object average brightness

Image Controls

Bank	Address	Register Name	Default Value	R/W	Description
0	0x5B	Trigger	0x00	R/W	Trigger mode use only, as setting to one, trigger IC to report one frame
1	0x77	R_SRAM_Read_EnH	0x00	R/W	SRAM read enable, active high
1	0x7C	R_SPIOUT_PXDNUM [7:0]	0x384	R/W	SPI output pixel amount
1	0x7D	R_SPIOUT_PXDNUM [15:8]		R/W	
1	0x7E	R_SPIOUT_CSN_MODE [1:0]	0x00	R/W	Bit [3:2] 0: Follow TG_VsyncO 1: Follow IDLE time 2: Follow LED 3: Reserved
1	0x7E	R_SPIOUT_EnH	0x00	R/W	Bit[0] SPI output enable, active high

Background Controls

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	R_UseBGModel	0x9F	0	0x01	R/W	Background model enable
0	R_BGUseDiffWeight	0x9F	1	0x00	R/W	When calculating object center, use the weight between background and pixel
0	R_BGUpdateAtProcess	0x9F	2	0x00	R/W	Update background at process state
0	R_BGUpdateMaxIntensity_En	0x9F	3	0x01	R/W	Background up bound threshold enable
0	R_BGUpdateMaxIntensity	0xA0	7:0	0x03	R/W	Background up bound threshold, if pixel data > $R_BGUpdateMaxIntensity \times 2$, update this pixel to BG SRAM
0	R_BGFilterLengthUp	0xA1	1:0	0x01	R/W	IIR filter weight of updating background not at process state, IIR Filter Out (X : previous data, X' : current date) = $X \times 2^{R_BGFilterLength - 1} + X' / 2^{R_BGFilterLength}$
0	R_BGFilterLengthDown	0xA1	5:4	0x10	R/W	IIR filter weight of updating background at process state
0	R_BGDiffThd	0xA2	5:0	0x08	R/W	Only if the difference between BG and pixel > $R_BGDiffThd$, this will be taken as part of object
0	R_BGUpdateFreq	0xA3	7:0	0x00	R/W	Low byte: Update background per this number frames
0	R_BGUpdateFreq	0xA4	1:0	0x03	R/W	High byte: Update background per this number frames
0	R_BGResolution	0xA4	5:4	0x03	R/W	Background resolution, BG data = pixel data << $R_BGResolution$

Lens Shading Compensation

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
1	R_LS_Comp_DAvg_V	0x04	7	0x00	R/W	Lens shading for digital vertical average
1	R_LensShadingComp_EnH	0x25	0	0x00	R/W	Lens shading compensation enable, active high
1	R_OffsetX	0x26	6:0	0x00	R/W	Horizontal offset of lens (s + 6, -63~+63) Initialize register value to 5
1	R_OffsetY	0x27	6:0	0x00	R/W	Vertical offset of lens (s + 6, -63~+63) Initialize register value to 55
1	R_LSC	0x28	6:0	0x40	R/W	A, R2 coefficient, (unsigned, 0~127) Initialize register value to 127
1	R_LSFT	0x29	3:0	0x0A	R/W	Shift amount of A × R2 (unsigned, 0~15) Initialize register value to 10

LED Controls

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
1	R_LED_SoftStart_time	0x30	7:0	0x03	R/W	LED soft start time before TS_ab falling
1	R_LED2_DAC_EnL	0x31	5	0x00	R/W	0: Turn on LED_DAC2 1: Turn off LED_DAC2
1	R_LED1_DAC_EnL	0x31	4	0x00	R/W	0: Turn on LED_DAC1 1: Turn off LED_DAC1
1	R_LED2_DAC_manual	0x31	1	0x00	R/W	LED2 manual enable, active high 0: TS_LED2_EnL / Tm_LEDC2_DAC is controlled automatically by digital 1: TS_LED2_EnL / Tm_LEDC2_DAC is controlled manually by R_LED2_EnL / R_LED2_DAC_UB
1	R_LED1_DAC_manual	0x31	0	0x00	R/W	LED1 manual enable, active high 0: TS_LED1_EnL/Tm_LEDC1_DAC is controlled automatically by digital 1: TS_LED1_EnL / Tm_LEDC1_DAC is controlled manually by R_LED1_EnL / R_LED1_DAC_UB
1	R_LED1_DAC_UB	0x32	4:0	0x14	R/W	LED1 upper bound
1	R_LED2_DAC_UB	0x33	4:0	0x14	R/W	LED2 upper bound
1	R_LEDC_Step_Up	0x34	4:0	0x07	R/W	Step up : B_LEDC_Step_Up[4:0] = 0, let TS_LED1_thermo[30:0] & TS_LED2_thermo[30:0] all current DAC off B_LEDC_Step_Up[4:0] = 1, let TS_LED1_thermo[30:0] & TS_LED2_thermo[30:0] starts from thermo<0> sequentially
1	R_LEDC_Step_Down	0x35	4:0	0x07	R/W	Step down : B_LEDC_Step_Down[4:0] = 0, Let TS_LED1_thermo[30:0] and TS_LED2_thermo[30:0] all current DAC off B_LEDC_Step_down[4:0] = 1, let TS_LED1_thermo[30:0] and TS_LED2_thermo[30:0] starts from thermo<31> sequentially

Sleep Mode

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	SleepMode_Status	0x45	1:0	-	R	Indicate operation status: 0: Normal 1: Weak sleep 2: Deep sleep For normal operation, use R_IDLE_TIME For IDLE weak sleep, use R_IDLE_TIME_SLEEP_1 For IDLE deep sleep, use R_IDLE_TIME_SLEEP_2 × 2 For IDLE, See R_IDLE_TIME
0	R_SleepAE1	0x52	7:6	0x00	R/W	At weak sleep: 1: Exp' = Exp × 2 2: Exp' = Exp × 4 Others : Exp' = Exp Wake up AE mode setting Only active under sleep status
0	R_SleepAE2	0x52	5:4	0x00	R/W	At deep sleep: 1: Exp' = Exp × 2 2: Exp' = Exp × 4 Others : Exp' = Exp Wake up AE mode setting Only active under sleep status
0	R_SleepAG1	0x52	3:2	0x00	R/W	At weak sleep: 1: Gain' = Gain × 2 2: Gain' = Gain × 4 Others : Gain' = Gain Wake up AE mode setting Only active under sleep status
0	R_SleepAG2	0x52	1:0	0x00	R/W	At deep sleep: 1: Gain' = Gain × 2 2: Gain' = Gain × 4 Others : Gain' = Gain Wake up AE mode setting. Only active under sleep status
1	R_IDLE_TIME	0x65	7:0	0xB4	R/W	Low Byte for Idle time for normal operation, unit: T = 256 / System CLK = 32 μs Example : Far Mode : 1 report time = (77 + R_IDLE_TIME) × T Report rate 120 fps : R_IDLE_TIME = 1 / (120 × T) – 77 = 183 Report rate 240 fps : R_IDLE_TIME = 1 / (240 × T) – 77 = 53 Near Mode: 1 report time = (112 + R_IDLE_TIME) × T Report rate 120 fps : R_IDLE_TIME = 1 / (120 × T) – 112 = 148 Report rate 240 fps : R_IDLE_TIME = 1 / (240 × T) – 112 = 18
1	R_IDLE_TIME	0x66	7:0	0x00	R/W	High Byte for Idle time for normal operation
1	R_IDLE_TIME_SLEEP_1	0x67	7:0	0x68	R/W	Low Byte for Idle time for weak sleep, in weak sleep status, using this idle time Report Rate = $120 \times 8.333 / (0.0323 \times R_IDLE_TIME_SLEEP_1 + X)$ X = 2.41 for Far Mode, X = 3.55 for Near Mode
1	R_IDLE_TIME_SLEEP_1	0x68	7:0	0x01	R/W	High Byte for Idle time for weak sleep
1	R_IDLE_TIME_SLEEP_2	0x69	7:0	0xD0	R/W	Low Byte for Idle time for deep sleep 2X in deep sleep status, using this idle time × 2 Report Rate = $120 \times 8.333 / (0.0646 \times R_IDLE_TIME_SLEEP_2 + X)$ X = 2.41 for Far Mode, X = 3.55 for Near Mode

1	R_IDLE_TIME_SLEEP_2	0x6A	7:0	0x02	R/W	High Byte for Idle time for deep sleep
1	R_Obj_TIME_1	0x6B	7:0	0xB0	R/W	Low Byte for weak sleep enter time, unit: one report frame time
1	R_Obj_TIME_1	0x6C	7:0	0x04	R/W	High Byte for weak sleep enter time
1	R_Obj_TIME_2	0x6D	7:0	0x60	R/W	Low Byte for deep sleep enter time, unit: two report frame time
1	R_Obj_TIME_2	0x6E	7:0	0x09	R/W	High Byte for deep sleep enter time
1	R_TG_INIT_TIME	0x6F	7:0	0x32	R/W	Fine tune of idle time for normal operation, unit: 4 μ s
1	R_TG_POWERON_WAKEUP_TIME	0x71	7:0	0x00	R/W	Wakeup time after TG_START for analog CDS power on stable, unit: 4 μ s
1	R_TG_EnH	0x72	7:0	0x00	R/W	TG enable, Enable R_TG_EnH after setting all parameters
1	R_Auto_SLEEP_Mode1	0x73	1:0	0x00	R/W	Weak sleep method selection 0: No motion sleep 1: No object sleep
1	R_Auto_SLEEP_Mode2	0x73	3:2	0x00	R/W	Deep sleep method selection 0: No motion sleep 1: No object sleep
1	R_Auto_WakeUP_Mode	0x73	5:4	0x00	R/W	Wake up method selection bit[4]: 0: Motion wake up at weak sleep 1: Object wake up at weak sleep bit[5]: 0: Motion wake up at deep sleep 1: Object wake up at deep sleep
1	R_WakeUpSig_Sel	0x74	5:4	0x00	R/W	1: GPIO0 output TG_Start signal 2: GPIO0 output TG_Finish signal

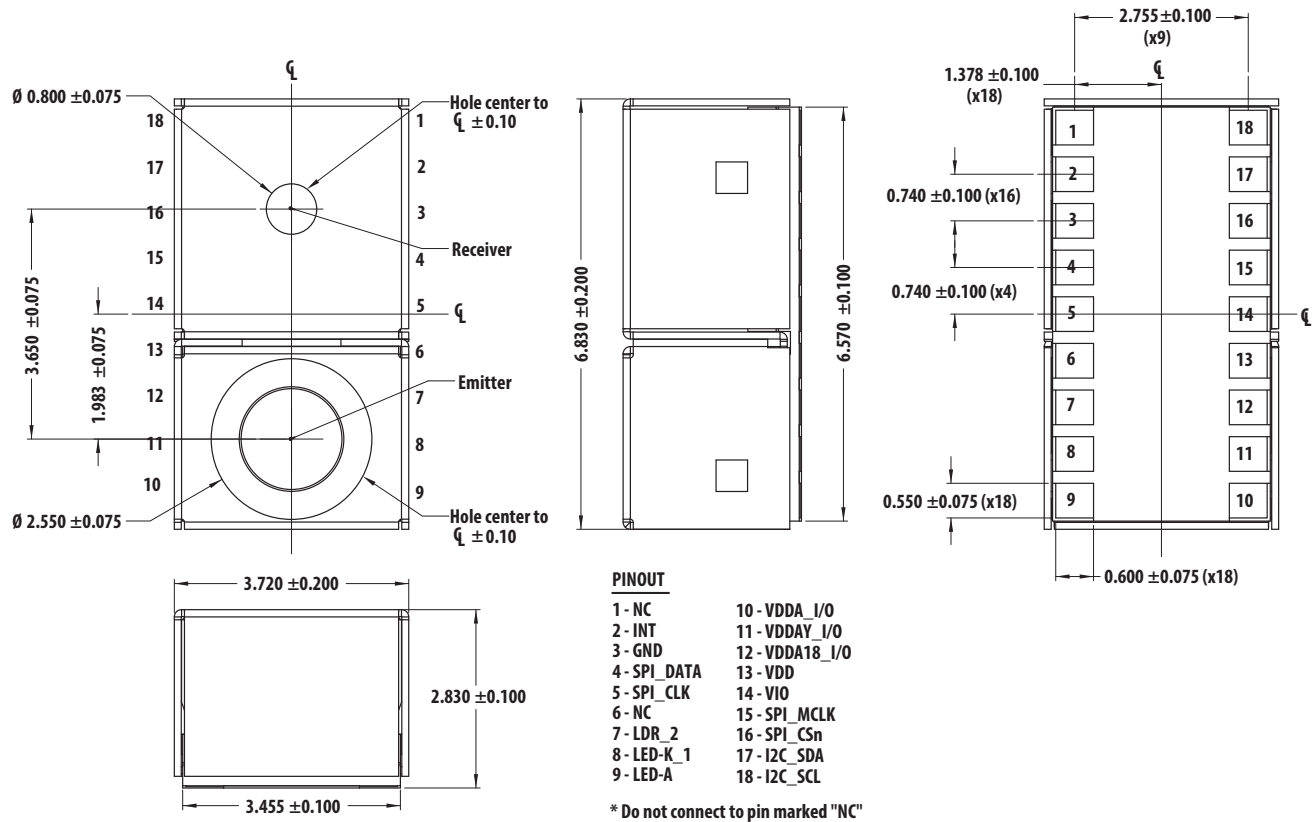
Part ID / Version ID

Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	PartID	0x00	7:0	0x20	R	Low byte of Part ID
0	PartID	0x01	7:0	0x76	R	High byte of Part ID
0	VersionID	0x02	7:0	0x01	R	Version ID

Suspend / Reset

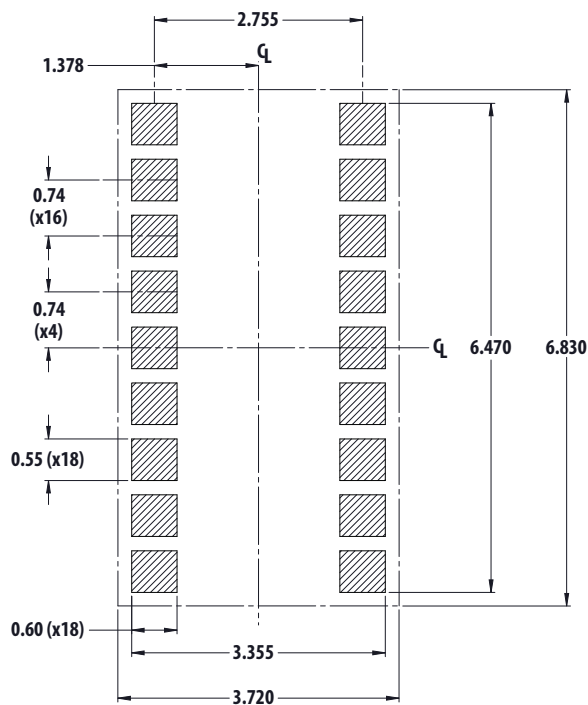
Bank	Register Name	Addr	Bits	Default Value	R/W	Description
0	SW_Suspend_EnL	0x03	7:0	0x01	W	1: Software suspend

Package Outline Dimensions



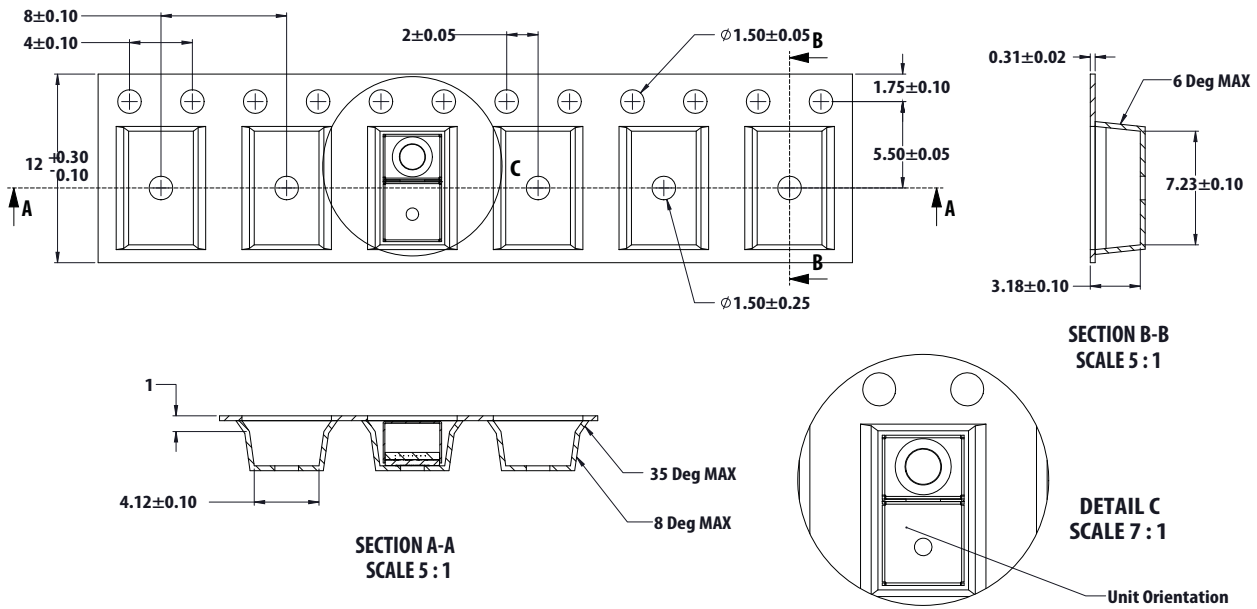
Dimensions are in mm

PCB Pad Layout



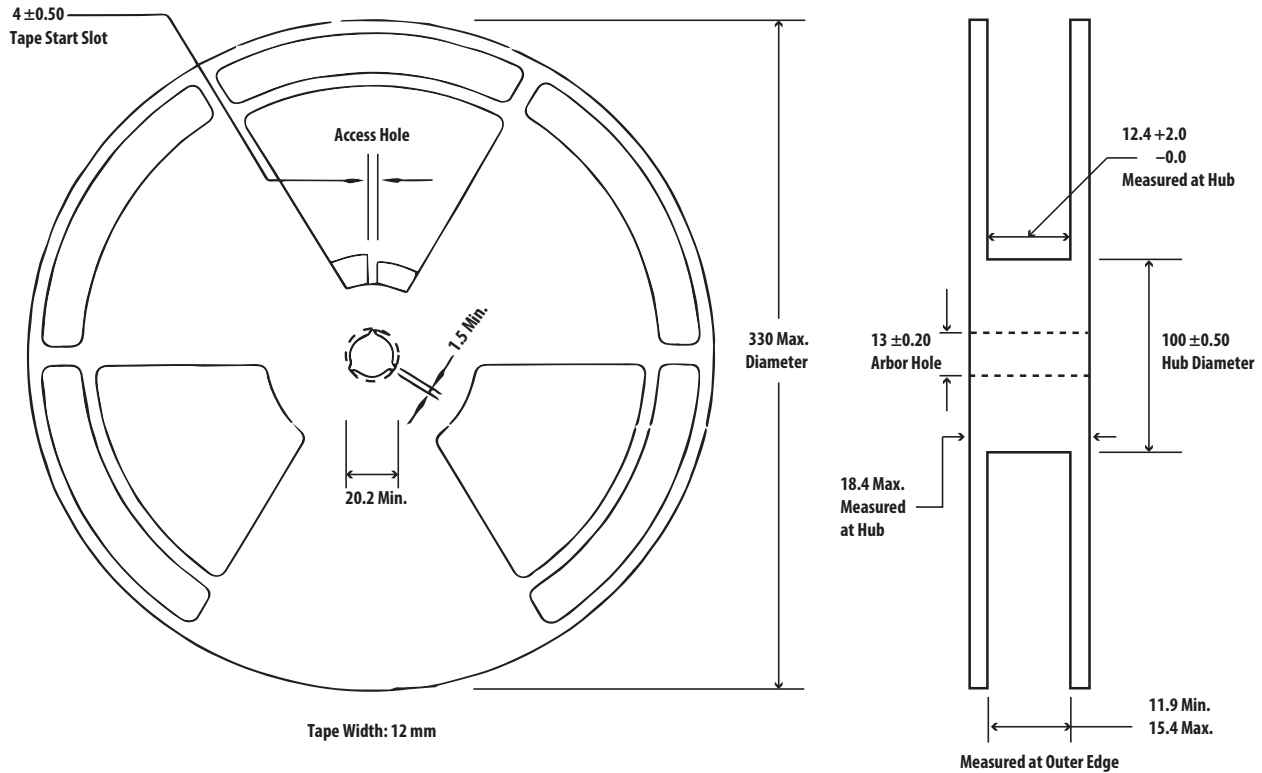
Dimensions are in mm

Tape Dimensions



Dimensions are in mm

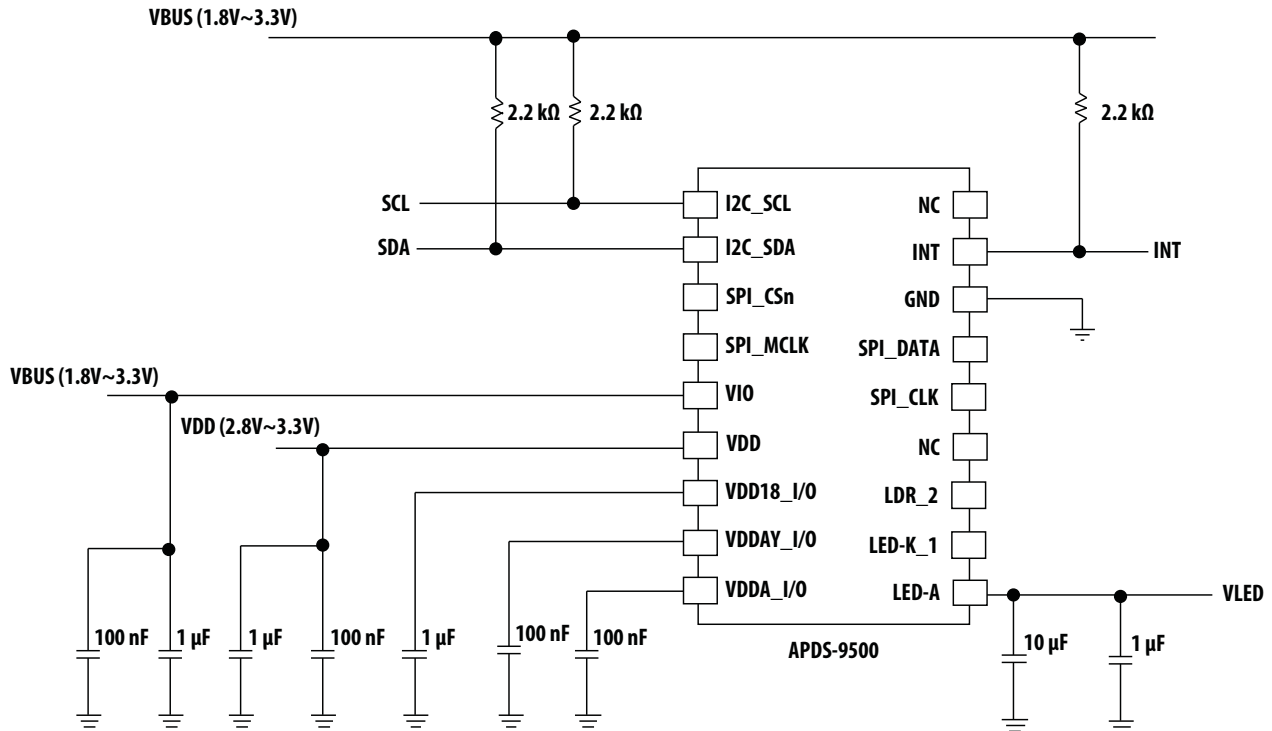
Reel Dimensions



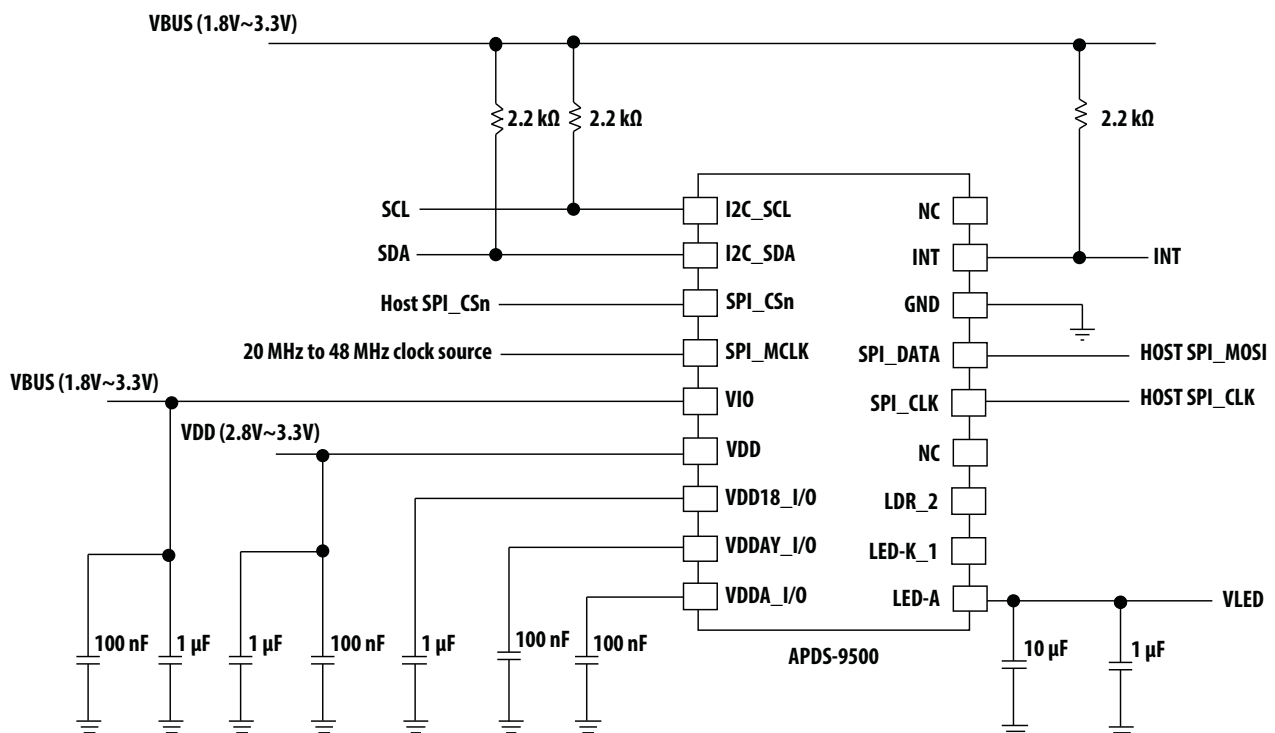
Dimensions are in mm

Application Circuit

To implement the APDS-9500 as a gesture sensor (including PS and cursor modes) use the application circuit below. The LED current is programmable and the maximum peak current may be up to 860 mA. Ensure that the VLED source is able to supply this current. Note that the VLED-A pin may be connected to the system battery if the battery voltage meets recommended operating voltage.

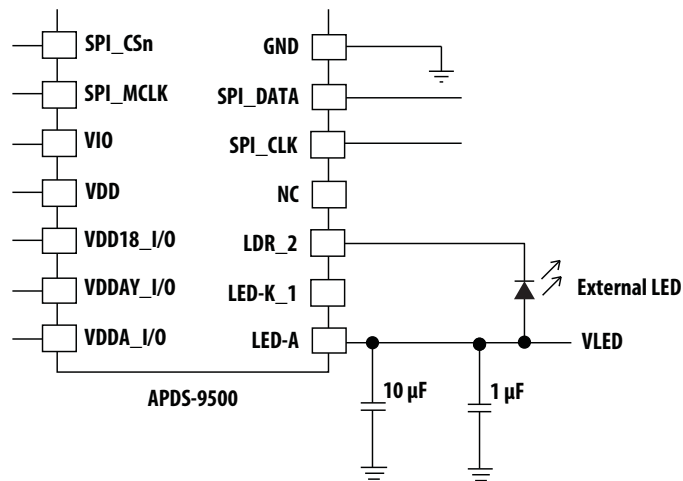


If the APDS-9500 is also used as an imaging device (image mode), refer to the application circuit below. Note the additional connections to the SPI bus necessary to extract the raw image data from the device.

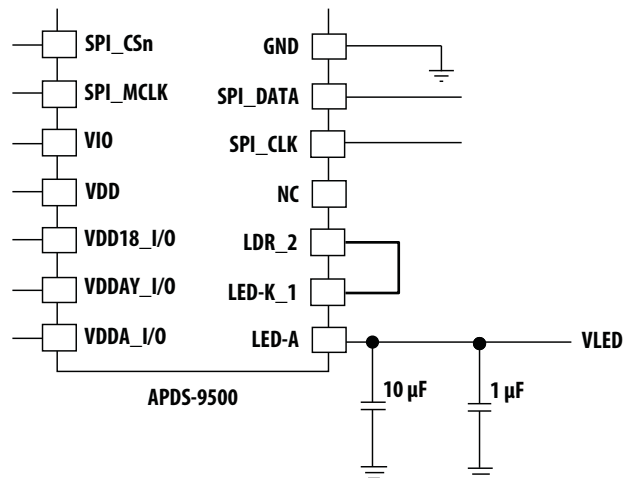


Note that the built-in IR LED cathode is internally connected to the LED-K pin.

The APDS-9500 provides for an additional LED drive with the LDR_2 pin. If an external IR LED is to be used, connect it to the LDR_2 pin as shown below.



It is possible to combine the drive current of the two LED drive pins together. In this case, the connection is shown below.



Programming and Application firmware

This section provides the basic firmware initialization of the recommended register settings for different APDS-9500 modes.

i. Initialization

Step 1: Power on. The VBus must be powered up before VDD.

Step 2: Wait 700 μ s for APDS-9500 to stabilize.

Step 3: Write slave ID or I²C read command to process I²C wake-up.

Read Reg_0x00. It returns 0x20 when system wake-up is complete.

Note that there is no-ack from the APDS-9500 before wake-up is completed.

Step 4: Write initial setting to gesture.

```
unsigned char initial_register_array[][2] = {
{0xEF,0x00},
{0x37,0x07},
{0x38,0x17},
{0x39,0x06},
{0x42,0x01},
{0x46,0x2D},
{0x47,0x0F},
{0x48,0x3C},
{0x49,0x00},
{0x4A,0x1E},
{0x4C,0x20},
{0x51,0x10},
{0x5E,0x10},
{0x60,0x27},
{0x80,0x42},
{0x81,0x44},
{0x82,0x04},
{0x8B,0x01},
{0x90,0x06},
{0x95,0x0A},
{0x96,0x0C},
{0x97,0x05},
{0x9A,0x14},

{0x9C,0x3F},
{0xA5,0x19},
{0xCC,0x19},
{0xCD,0x0B},
{0xCE,0x03},
{0xCF,0x64},
{0xD0,0x21},
{0xEF,0x01},
{0x02,0x0F},
{0x03,0x10},
{0x04,0x02},
{0x25,0x01},
{0x27,0x39},
{0x28,0x7F},
{0x29,0x08},
{0x3E,0xFF},
{0x5E,0x3D},
```



```
{0x65,0x96},
{0x67,0x97},
{0x69,0xCD},
{0x6A,0x01},
{0x6D,0x2C},
{0x6E,0x01},
{0x72,0x01},
{0x73,0x35},
{0x74,0x00},
{0x77,0x01},
};
```

ii. Get Gesture result

Step 1: Set Interrupt or I²C polling timer.

Step 2: Read Bank_0_Reg_0x43/0x44 for gesture result if interrupt or timer event occurs.

Gesture result clears when I²C read is complete.

iii. Change to PS mode

Step 1: Write PS mode setting to gesture.

```
unsigned char change_to_proximity_register_array[][2] = {
{0xEF,0x00},
{0x41,0x00},
{0x42,0x02},
{0x48,0x20},
{0x49,0x00},
{0x51,0x13},
{0x83,0x00},
{0x9F,0xF8},
{0x69,0x96},
{0x6A,0x02},
{0xEF,0x01},
{0x01,0x1E},
{0x02,0x0F},
{0x03,0x10},
{0x04,0x02},
{0x41,0x50},
{0x43,0x34},
{0x65,0xCE},
{0x66,0x0B},
{0x67,0xCE},
{0x68,0x0B},
{0x69,0xE9},
{0x6A,0x05},
{0x6B,0x50},
{0x6C,0xC3},
{0x6D,0x50},
{0x6E,0xC3},
{0x74,0x05},
};
```

iv. Get PS approach status

Step 1: Read Bank_0_Reg_0x6B for PS approach status or read Bank_0_Reg_0x6C for PS raw data.

v. Change to Gesture mode

Step 1: Write Gesture mode setting to gesture.

```
unsigned char change_to_gesture_register_array[][2] = {
{0xEF,0x00},
{0x41,0x00},
{0x42,0x00},
{0xEF,0x00},
{0x48,0x3C},
{0x49,0x00},
{0x51,0x10},
{0x83,0x20},
{0x9f,0xf9},
{0xEF,0x01},
{0x01,0x1E},
{0x02,0x0F},
{0x03,0x10},
{0x04,0x02},
{0x41,0x40},
{0x43,0x30},
{0x65,0x96},
{0x66,0x00},
{0x67,0x97},
{0x68,0x01},
{0x69,0xCD},
{0x6A,0x01},
{0x6b,0xb0},
{0x6c,0x04},
{0x6D,0x2C},
{0x6E,0x01},
{0x74,0x00},
{0xEF,0x00},
{0x41,0xFF},
{0x42,0x01},
};
```

vi. Enter Suspend mode

Step 1: Write Suspend mode setting to gesture.

```
unsigned char suspend_register_array[][2] = {
{0xEF,0x01},
{0x72,0x00},
{0xEF,0x00},
{0x03,0x01},
};
```

vii. Resume Gesture

Step 1: Write slave ID or I²C read command to process I²C wake-up.

Read Reg_0x00. It returns 0x20 when wake-up is complete.

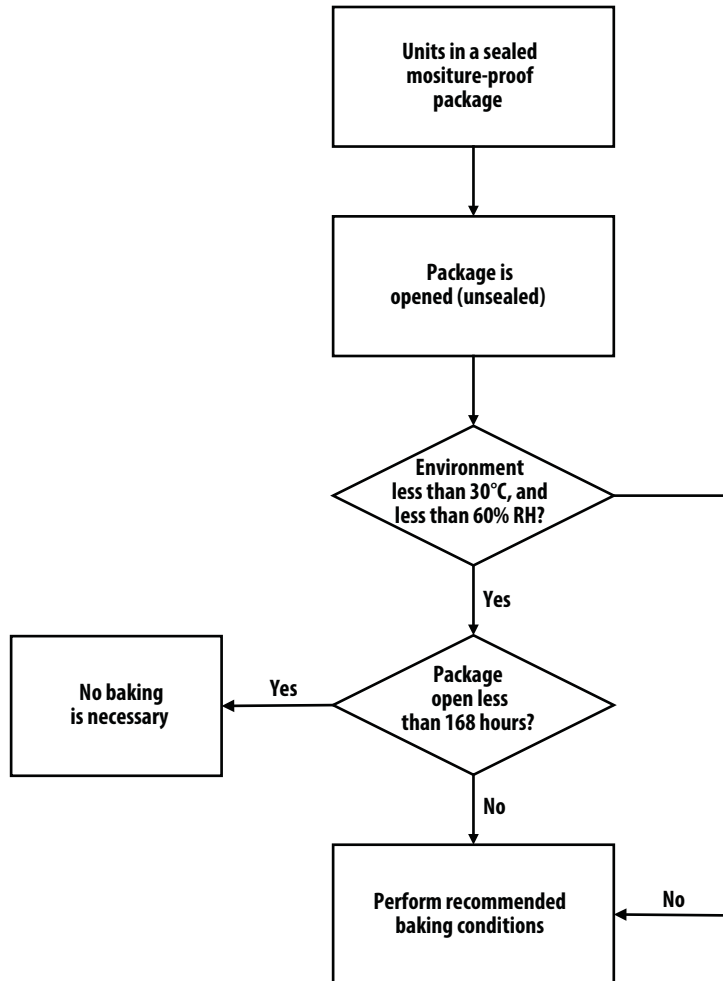
Note that there is no-ack from the APDS-9500 before wake-up is complete.

Step 2: Write Resume setting to gesture.

```
unsigned char resume_register_array[][2] = {
{0xEF,0x01},
{0x72,0x01},
};
```

Moisture Proof Packaging

All APDS-9500 options are shipped in moisture proof package. Once opened, moisture absorption begins. This part is compliant to JEDEC MSL 3.



Baking Conditions

Package	Temperature	Time
In Reel	60°C	48 hours
In Bulk	100°C	4 hours

If the parts are not stored in dry conditions, they must be baked before reflow to prevent damage to the parts.

Baking should only be done once.

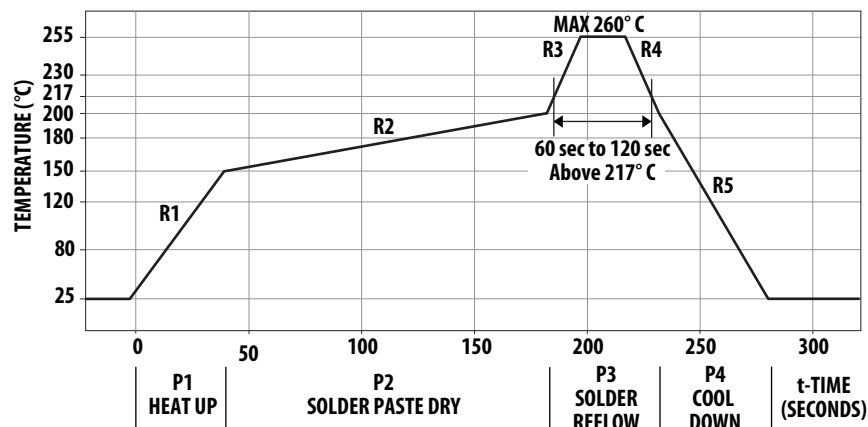
Recommended Storage Conditions

Storage Temperature	10°C to 30°C
Relative Humidity	below 60% RH

Time from unsealing to soldering

After removal from the bag, the parts should be soldered within 168 hours if stored at the recommended storage conditions. If times longer than 168 hours are needed, the parts must be stored in a dry box.

Recommended Reflow Profile



Process Zone	Symbol	ΔT	Maximum $\Delta T/\Delta \text{time}$ or Duration
Heat Up	P1, R1	25°C to 150°C	3°C/s
Solder Paste Dry	P2, R2	150°C to 200°C	100s to 180s
Solder Reflow	P3, R3	200°C to 260°C	3°C/s
	P3, R4	260°C to 200°C	-6°C/s
Cool Down	P4, R5	200°C to 25°C	-6°C/s
Time maintained above liquidus point, 217°C		>217°C	60s to 120s
Peak Temperature		260°C	-
Time within 5°C of actual Peak Temperature		>255°C	20s to 40s
Time 25°C to Peak Temperature		25°C to 260°C	8 minutes

The reflow profile is a straight-line representation of a nominal temperature profile for a convective reflow solder process. The temperature profile is divided into four process zones, each with different $\Delta T/\Delta \text{time}$ temperature change rates or durations. The $\Delta T/\Delta \text{time}$ rates or durations are detailed in the above table. The temperatures are measured at the component to PC connections.

In process zone P1, the PCB and component pins are heated to a temperature of 150°C to activate the flux in the solder paste. The temperature ramp up rate, R1, is limited to 3°C/s to allow for even heating of both the PCB and component pins.

Process zone P2 should be of sufficient time duration (100 to 180 seconds) to dry the solder paste. The temperature is raised to a level just below the liquidus point of the solder.

Process zone P3 is the solder reflow zone. In zone P3, the temperature is quickly raised above the liquidus point

of solder to 260°C (500°F) for optimum results. The dwell time above the liquidus point of solder should be between 60 and 120 seconds. This assures proper coalescing of the solder paste into liquid solder and the formation of good solder connections. Beyond the recommended dwell time, the intermetallic growth within the solder connections becomes excessive, resulting in the formation of weak and unreliable connections. The temperature is then rapidly reduced to a point below the solidus temperature of the solder to allow the solder within the connections to freeze solid.

Process zone P4 is the cool down after solder freeze. The cool down rate, R5, from the liquidus point of the solder to 25°C (77°F) should not exceed 6°C/s max. This limitation is necessary to allow the PC board and component pins to change dimensions evenly, putting minimal stresses on the component.

It is not recommended to perform reflow soldering more than twice.

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

www.avagotech.com

Broadcom, the pulse logo, Connecting everything, Avago Technologies, Avago, and the A logo are among the trademarks of Broadcom and/or its affiliates in the United States, certain other countries and/or the EU.

The term "Broadcom" refers to Broadcom Limited and/or its subsidiaries. For more information, please visit www.broadcom.com.

Data subject to change. Copyright ©2016 by Broadcom. All rights reserved.

AV02-4584EN - October 5, 2016

Avago
TECHNOLOGIES

A Broadcom Limited Company



Lead (Pb) Free
RoHS 6 fully
compliant

RoHS 6 fully compliant options available;
-xxxE denotes a lead-free product