Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

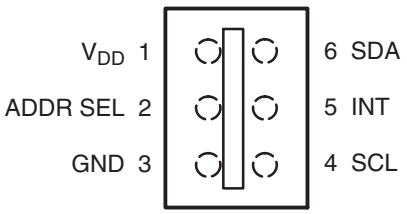
We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

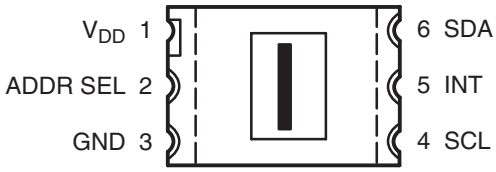
Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

- **Approximates Human Eye Response**
- **Programmable Interrupt Function with User-Defined Upper and Lower Threshold Settings**
- **16-Bit Digital Output with SMBus (TSL2560) at 100 kHz or I²C (TSL2561) Fast-Mode at 400 kHz**
- **Programmable Analog Gain and Integration Time Supporting 1,000,000-to-1 Dynamic Range**
- **Automatically Rejects 50/60-Hz Lighting Ripple**
- **Low Active Power (0.75 mW Typical) with Power Down Mode**
- **RoHS Compliant**

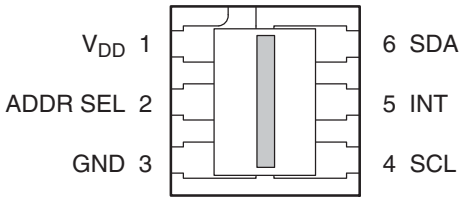
PACKAGE CS
6-LEAD CHIPSCALE
(TOP VIEW)



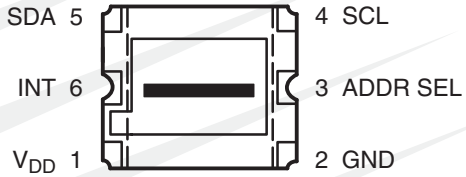
PACKAGE T
6-LEAD TMB
(TOP VIEW)



PACKAGE FN
DUAL FLAT NO-LEAD
(TOP VIEW)



PACKAGE CL
6-LEAD ChipLED
(TOP VIEW)



Package Drawings are Not to Scale

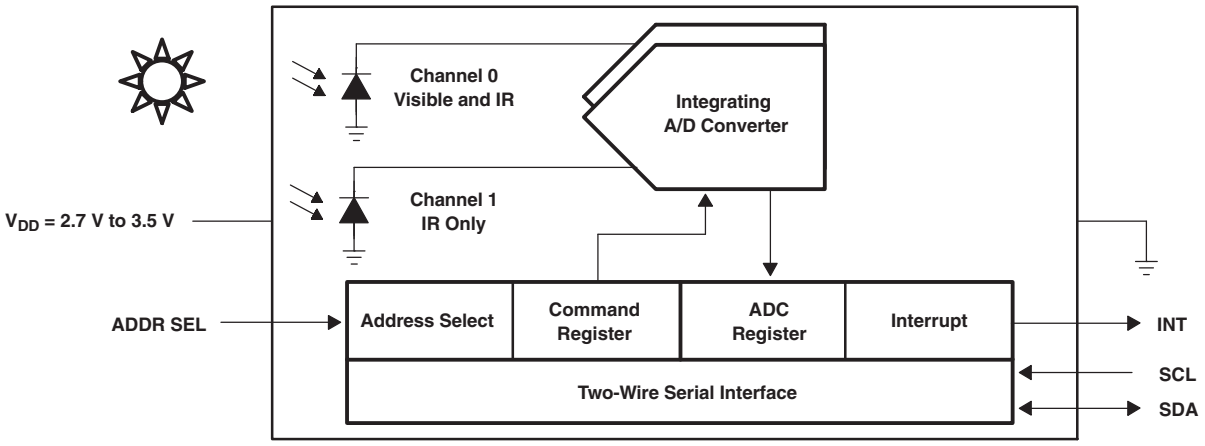
Description

The TSL2560 and TSL2561 are light-to-digital converters that transform light intensity to a digital signal output capable of direct I²C (TSL2561) or SMBus (TSL2560) interface. Each device combines one broadband photodiode (visible plus infrared) and one infrared-responding photodiode on a single CMOS integrated circuit capable of providing a near-photopic response over an effective 20-bit dynamic range (16-bit resolution). Two integrating ADCs convert the photodiode currents to a digital output that represents the irradiance measured on each channel. This digital output can be input to a microprocessor where illuminance (ambient light level) in lux is derived using an empirical formula to approximate the human eye response. The TSL2560 device permits an SMB-Alert style interrupt, and the TSL2561 device supports a traditional level style interrupt that remains asserted until the firmware clears it.

While useful for general purpose light sensing applications, the TSL2560/61 devices are designed particularly for display panels (LCD, OLED, etc.) with the purpose of extending battery life and providing optimum viewing in diverse lighting conditions. Display panel backlighting, which can account for up to 30 to 40 percent of total platform power, can be automatically managed. Both devices are also ideal for controlling keyboard illumination based upon ambient lighting conditions. Illuminance information can further be used to manage exposure control in digital cameras. The TSL2560/61 devices are ideal in notebook/tablet PCs, LCD monitors, flat-panel televisions, cell phones, and digital cameras. In addition, other applications include street light control, security lighting, sunlight harvesting, machine vision, and automotive instrumentation clusters.

Texas Advanced Optoelectronic Solutions Inc.
1001 Klein Road • Suite 300 • Plano, TX 75074 • (972) 673-0759
www.taosinc.com

Functional Block Diagram



Detailed Description

The TSL2560 and TSL2561 are second-generation ambient light sensor devices. Each contains two integrating analog-to-digital converters (ADC) that integrate currents from two photodiodes. Integration of both channels occurs simultaneously. Upon completion of the conversion cycle, the conversion result is transferred to the Channel 0 and Channel 1 data registers, respectively. The transfers are double-buffered to ensure that the integrity of the data is maintained. After the transfer, the device automatically begins the next integration cycle.

Communication to the device is accomplished through a standard, two-wire SMBus or I^2C serial bus. Consequently, the TSL256x device can be easily connected to a microcontroller or embedded controller. No external circuitry is required for signal conditioning, thereby saving PCB real estate as well. Since the output of the TSL256x device is digital, the output is effectively immune to noise when compared to an analog signal.

The TSL256x devices also support an interrupt feature that simplifies and improves system efficiency by eliminating the need to poll a sensor for a light intensity value. The primary purpose of the interrupt function is to detect a meaningful change in light intensity. The concept of a *meaningful change* can be defined by the user both in terms of light intensity and time, or persistence, of that change in intensity. The TSL256x devices have the ability to define a threshold above and below the current light level. An interrupt is generated when the value of a conversion exceeds either of these limits.

Available Options

DEVICE	INTERFACE	PACKAGE – LEADS	PACKAGE DESIGNATOR	ORDERING NUMBER
TSL2560	SMBus	Chipscale	CS	TSL2560CS
TSL2560	SMBus	TMB-6	T	TSL2560T
TSL2560	SMBus	Dual Flat No-Lead – 6	FN	TSL2560FN
TSL2560	SMBus	ChipLED-6	CL	TSL2560CL
TSL2561	I^2C	Chipscale	CS	TSL2561CS
TSL2561	I^2C	TMB-6	T	TSL2561T
TSL2561	I^2C	Dual Flat No-Lead – 6	FN	TSL2561FN
TSL2561	I^2C	ChipLED-6	CL	TSL2561CL

Terminal Functions

TERMINAL NAME	CS, T, FN PKG NO.	CL PKG NO.	TYPE	DESCRIPTION
ADDR SEL	2	3	I	SMBus device select — three-state
GND	3	2		Power supply ground. All voltages are referenced to GND.
INT	5	6	O	Level or SMB Alert interrupt — open drain.
SCL	4	4	I	SMBus serial clock input terminal — clock signal for SMBus serial data.
SDA	6	5	I/O	SMBus serial data I/O terminal — serial data I/O for SMBus.
V_{DD}	1	1		Supply voltage.

Absolute Maximum Ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{DD} (see Note 1) . . . . . 3.8 V
Digital output voltage range, V_O . . . . . –0.5 V to 3.8 V
Digital output current, I_O . . . . . –1 mA to 20 mA
Storage temperature range, T_{stg} . . . . . –40°C to 85°C
ESD tolerance, human body model . . . . . 2000 V

† Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: All voltages are with respect to GND.

Recommended Operating Conditions

	MIN	NOM	MAX	UNIT
Supply voltage, V_{DD}	2.7	3	3.6	V
Operating free-air temperature, T_A	–30		70	°C
SCL, SDA input low voltage, V_{IL}	–0.5		0.8	V
SCL, SDA input high voltage, V_{IH}	2.1		3.6	V

Electrical Characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
I_{DD}	Supply current	Active		0.24	0.6	mA
		Power down		3.2	15	µA
V_{OL}	INT, SDA output low voltage	3 mA sink current	0		0.4	V
		6 mA sink current	0		0.6	V
I_{LEAK}	Leakage current		–5		5	µA

Operating Characteristics, High Gain (16×), V_{DD} = 3 V, T_A = 25°C, (unless otherwise noted) (see Notes 2, 3, 4, 5)

PARAMETER		TEST CONDITIONS	CHANNEL	TSL2560T, FN, & CL TSL2561T, FN & CL			TSL2560CS, TSL2561CS			UNIT
				MIN	TYP	MAX	MIN	TYP	MAX	
f_{osc}	Oscillator frequency			690	735	780	690	735	780	kHz
	Dark ADC count value	E_e = 0, T_{int} = 402 ms	Ch0	0		4	0		4	counts
			Ch1	0		4	0		4	
	Full scale ADC count value (Note 6)	T_{int} > 178 ms	Ch0			65535			65535	counts
			Ch1			65535			65535	
		T_{int} = 101 ms	Ch0			37177			37177	
			Ch1			37177			37177	
		T_{int} = 13.7 ms	Ch0			5047			5047	
			Ch1			5047			5047	
	ADC count value	λ_p = 640 nm, T_{int} = 101 ms, E_e = 36.3 µW/cm^2	Ch0	750	1000	1250				counts
			Ch1		200					
		λ_p = 940 nm, T_{int} = 101 ms, E_e = 119 µW/cm^2	Ch0	700	1000	1300				
			Ch1		820					
		λ_p = 640 nm, T_{int} = 101 ms, E_e = 41 µW/cm^2	Ch0				750	1000	1250	counts
			Ch1					190		
		λ_p = 940 nm, T_{int} = 101 ms, E_e = 135 µW/cm^2	Ch0				700	1000	1300	
			Ch1					850		
	ADC count value ratio: Ch1/Ch0	λ_p = 640 nm, T_{int} = 101 ms		0.15	0.20	0.25	0.14	0.19	0.24	
		λ_p = 940 nm, T_{int} = 101 ms		0.69	0.82	0.95	0.70	0.85	1	
R_e	Irradiance responsivity	λ_p = 640 nm, T_{int} = 101 ms	Ch0		27.5			24.4		counts/(µW/cm^2)
			Ch1		5.5			4.6		
		λ_p = 940 nm, T_{int} = 101 ms	Ch0		8.4			7.4		
			Ch1		6.9			6.3		
R_v	Illuminance responsivity	Fluorescent light source: T_{int} = 402 ms	Ch0		36			35		counts/lux
			Ch1		4			3.8		
		Incandescent light source: T_{int} = 402 ms	Ch0		144			129		
			Ch1		72			67		
	ADC count value ratio: Ch1/Ch0	Fluorescent light source: T_{int} = 402 ms			0.11			0.11		
		Incandescent light source: T_{int} = 402 ms			0.5			0.52		
R_v	Illuminance responsivity, low gain mode (Note 7)	Fluorescent light source: T_{int} = 402 ms	Ch0		2.3			2.2		counts/lux
			Ch1		0.25			0.24		
		Incandescent light source: T_{int} = 402 ms	Ch0		9			8.1		
			Ch1		4.5			4.2		
	(Sensor Lux) / (actual Lux), high gain mode (Note 8)	Fluorescent light source: T_{int} = 402 ms		0.65	1	1.35	0.65	1	1.35	
		Incandescent light source: T_{int} = 402 ms		0.60	1	1.40	0.60	1	1.40	

NOTES:
2. Optical measurements are made using small-angle incident radiation from light-emitting diode optical sources. Visible 640 nm LEDs and infrared 940 nm LEDs are used for final product testing for compatibility with high-volume production.
3. The 640 nm irradiance E_e is supplied by an AlInGaP light-emitting diode with the following characteristics: peak wavelength λp = 640 nm and spectral halfwidth $\Delta\lambda ½$ = 17 nm.
4. The 940 nm irradiance E_e is supplied by a GaAs light-emitting diode with the following characteristics: peak wavelength λp = 940 nm and spectral halfwidth $\Delta\lambda ½$ = 40 nm.
5. Integration time T_{int}, is dependent on internal oscillator frequency (f_{osc}) and on the integration field value in the timing register as described in the *Register Set* section. For nominal f_{osc} = 735 kHz, nominal T_{int} = (number of clock cycles)/f_{osc}.
   Field value 00: $T_{int} = (11 \times 918)/f_{osc}$ = 13.7 ms
   Field value 01: $T_{int} = (81 \times 918)/f_{osc}$ = 101 ms
   Field value 10: $T_{int} = (322 \times 918)/f_{osc}$ = 402 ms
   Scaling between integration times vary proportionally as follows: 11/322 = 0.034 (field value 00), 81/322 = 0.252 (field value 01), and 322/322 = 1 (field value 10).
6. Full scale ADC count value is limited by the fact that there is a maximum of one count per two oscillator frequency periods and also by a 2-count offset.
   Full scale ADC count value = ((number of clock cycles)/2 – 2)
   Field value 00: Full scale ADC count value = $((11 \times 918)/2 - 2)$ = 5047
   Field value 01: Full scale ADC count value = $((81 \times 918)/2 - 2)$ = 37177
   Field value 10: Full scale ADC count value = 65535, which is limited by 16 bit register. This full scale ADC count value is reached for 131074 clock cycles, which occurs for T_{int} = 178 ms for nominal f_{osc} = 735 kHz.
7. Low gain mode has 16× lower gain than high gain mode: (1/16 = 0.0625).
8. The sensor Lux is calculated using the empirical formula shown on p. 22 of this data sheet based on measured Ch0 and Ch1 ADC count values for the light source specified. Actual Lux is obtained with a commercial luxmeter. The range of the (sensor Lux) / (actual Lux) ratio is estimated based on the variation of the 640 nm and 940 nm optical parameters. Devices are not 100% tested with fluorescent or incandescent light sources.

AC Electrical Characteristics, V_{DD} = 3 V, T_A = 25°C (unless otherwise noted)

PARAMETER†		TEST CONDITIONS	MIN	TYP	MAX	UNIT
$t_{(CONV)}$	Conversion time		12	100	400	ms
$f_{(SCL)}$	Clock frequency (I^2C only)		0		400	kHz
	Clock frequency (SMBus only)		10		100	kHz
$t_{(BUF)}$	Bus free time between start and stop condition		1.3			µs
$t_{(HDSTA)}$	Hold time after (repeated) start condition. After this period, the first clock is generated.		0.6			µs
$t_{(SUSTA)}$	Repeated start condition setup time		0.6			µs
$t_{(SUSTO)}$	Stop condition setup time		0.6			µs
$t_{(HDDAT)}$	Data hold time		0		0.9	µs
$t_{(SUDAT)}$	Data setup time		100			ns
$t_{(LOW)}$	SCL clock low period		1.3			µs
$t_{(HIGH)}$	SCL clock high period		0.6			µs
$t_{(TIMEOUT)}$	Detect clock/data low timeout (SMBus only)		25		35	ms
t_F	Clock/data fall time				300	ns
t_R	Clock/data rise time				300	ns
C_i	Input pin capacitance				10	pF

† Specified by design and characterization; not production tested.

PARAMETER MEASUREMENT INFORMATION

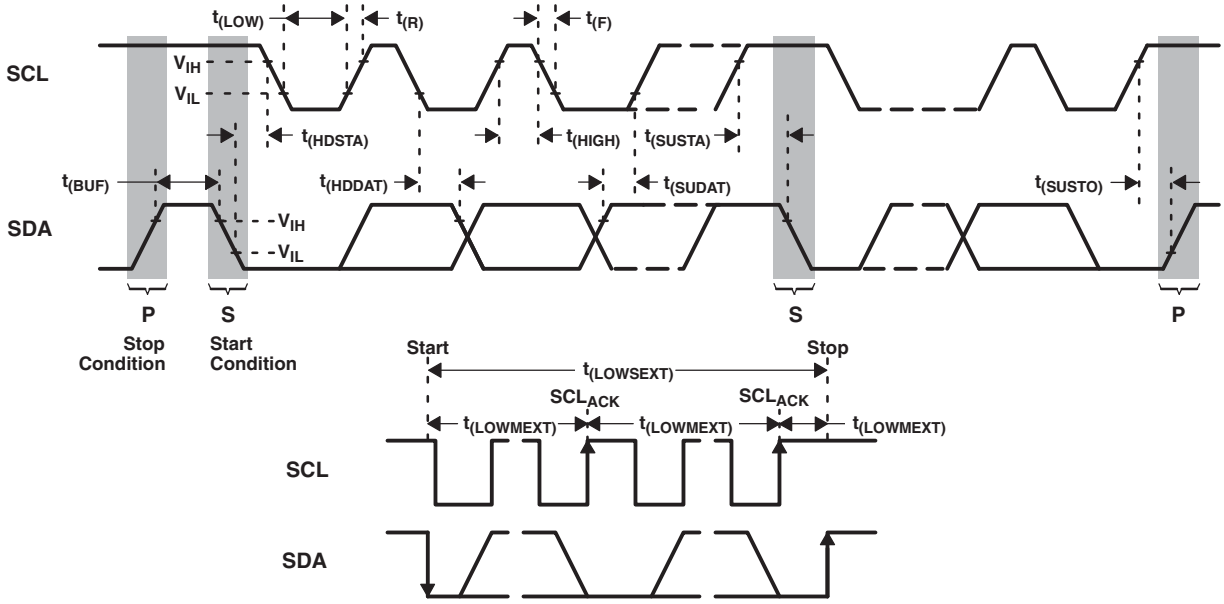


Figure 1. Timing Diagrams

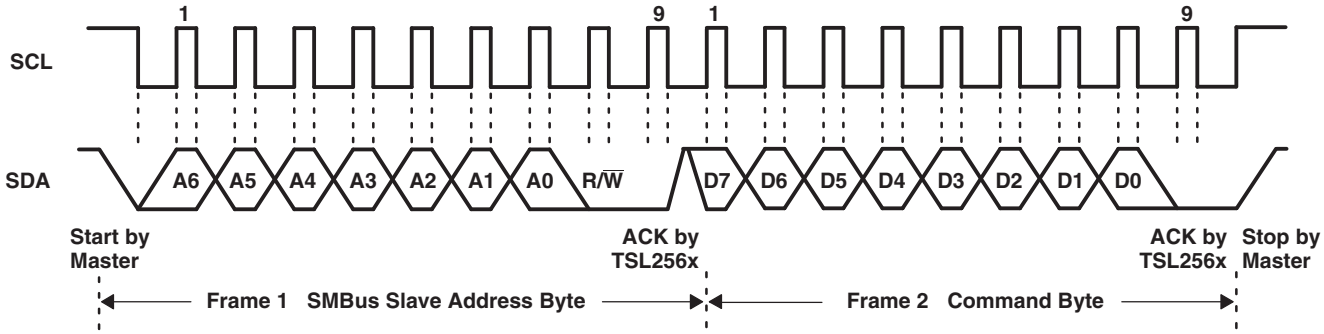


Figure 2. Example Timing Diagram for SMBus Send Byte Format

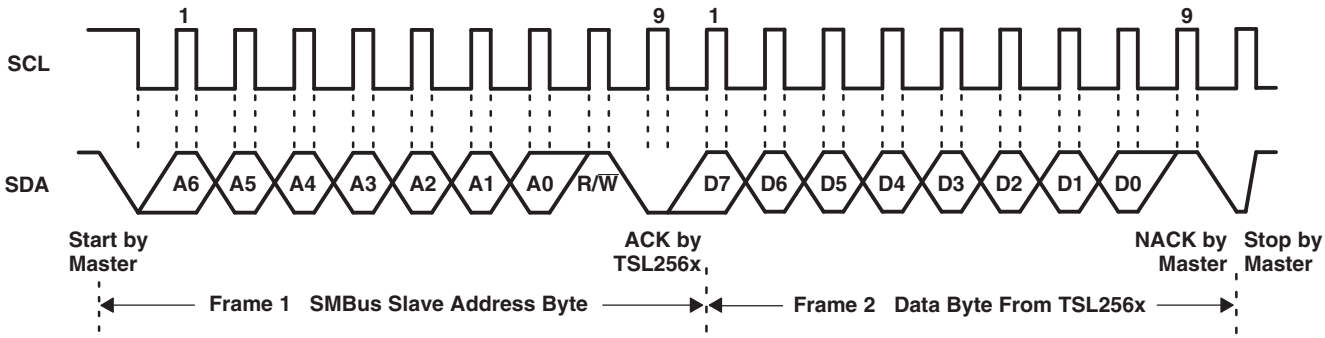


Figure 3. Example Timing Diagram for SMBus Receive Byte Format

TYPICAL CHARACTERISTICS

SPECTRAL RESPONSIVITY

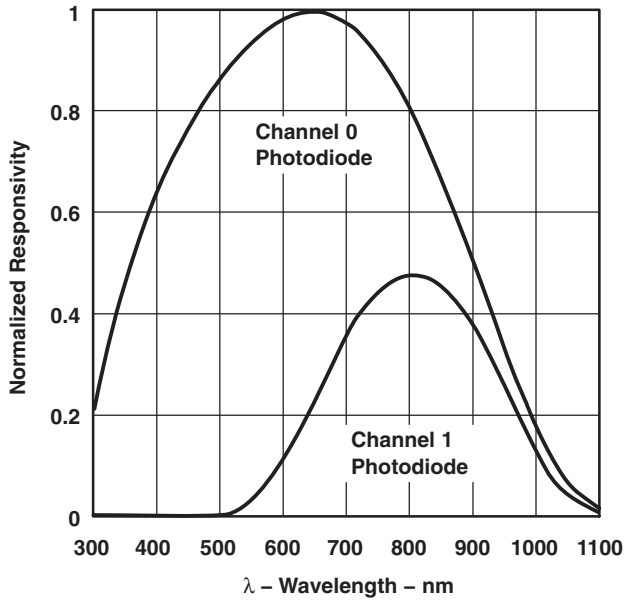


Figure 4

NORMALIZED RESPONSIVITY vs. ANGULAR DISPLACEMENT — CS PACKAGE

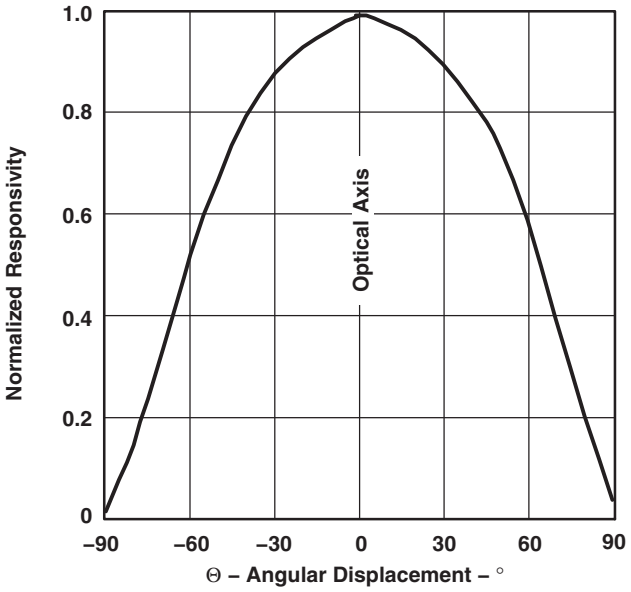


Figure 5

NORMALIZED RESPONSIVITY vs. ANGULAR DISPLACEMENT — T PACKAGE

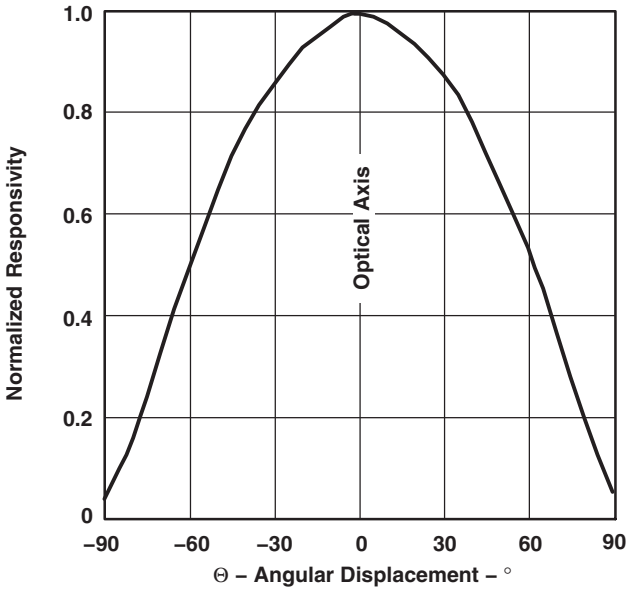


Figure 6

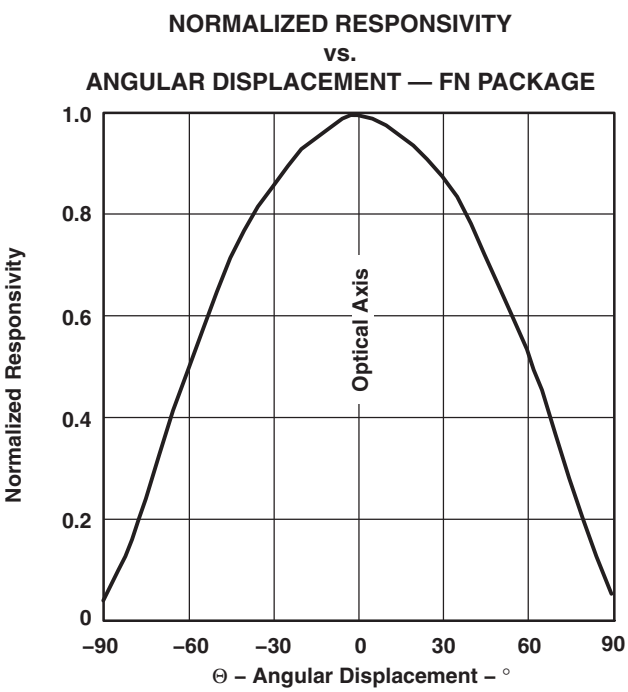


Figure 7

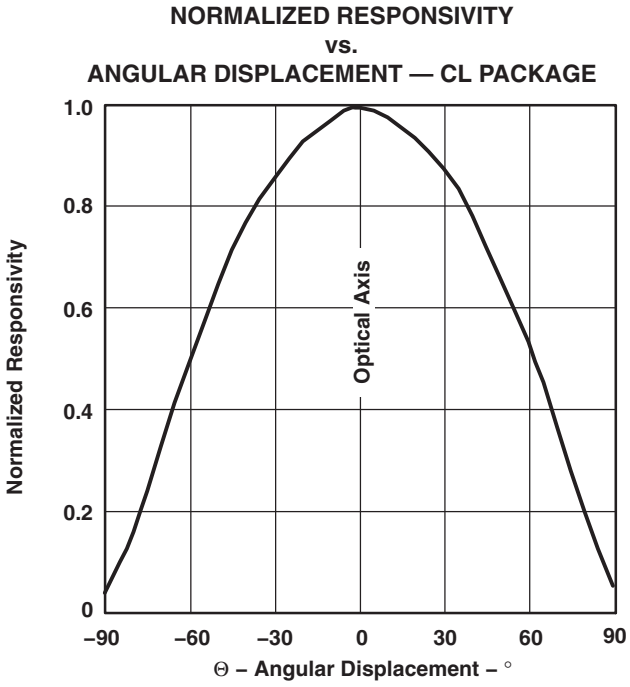


Figure 8

PRINCIPLES OF OPERATION

Analog-to-Digital Converter

The TSL256x contains two integrating analog-to-digital converters (ADC) that integrate the currents from the channel 0 and channel 1 photodiodes. Integration of both channels occurs simultaneously, and upon completion of the conversion cycle the conversion result is transferred to the channel 0 and channel 1 data registers, respectively. The transfers are double buffered to ensure that invalid data is not read during the transfer. After the transfer, the device automatically begins the next integration cycle.

Digital Interface

Interface and control of the TSL256x is accomplished through a two-wire serial interface to a set of registers that provide access to device control functions and output data. The serial interface is compatible with System Management Bus (SMBus) versions 1.1 and 2.0, and I^2C bus Fast-Mode. The TSL256x offers three slave addresses that are selectable via an external pin (ADDR SEL). The slave address options are shown in Table 1.

Table 1. Slave Address Selection

ADDR SEL TERMINAL LEVEL	SLAVE ADDRESS	SMB ALERT ADDRESS
GND	0101001	0001100
Float	0111001	0001100
VDD	1001001	0001100

NOTE: The Slave and SMB Alert Addresses are 7 bits. Please note the SMBus and I^2C protocols on pages 9 through 12. A read/write bit should be appended to the slave address by the master device to properly communicate with the TSL256X device.

SMBus and I^2C Protocols

Each *Send* and *Write* protocol is, essentially, a series of bytes. A byte sent to the TSL256x with the most significant bit (MSB) equal to 1 will be interpreted as a COMMAND byte. The lower four bits of the COMMAND byte form the register select address (see Table 2), which is used to select the destination for the subsequent byte(s) received. The TSL256x responds to any Receive Byte requests with the contents of the register specified by the stored register select address.

The TSL256X implements the following protocols of the SMB 2.0 specification:

- Send Byte Protocol
- Receive Byte Protocol
- Write Byte Protocol
- Write Word Protocol
- Read Word Protocol
- Block Write Protocol
- Block Read Protocol

The TSL256X implements the following protocols of the Philips Semiconductor I^2C specification:

- I^2C Write Protocol
- I^2C Read (Combined Format) Protocol

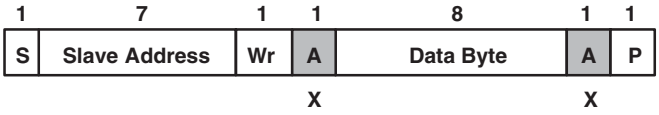
When an SMBus Block Write or Block Read is initiated (see description of COMMAND Register), the byte following the COMMAND byte is ignored but is a requirement of the SMBus specification. This field contains the byte count (i.e. the number of bytes to be transferred). The TSL2560 (SMBus) device ignores this field and extracts this information by counting the actual number of bytes transferred before the Stop condition is detected.

When an I^2C Write or I^2C Read (Combined Format) is initiated, the byte count is also ignored but follows the SMBus protocol specification. Data bytes continue to be transferred from the TSL2561 (I^2C) device to Master until a NACK is sent by the Master.

The data formats supported by the TSL2560 and TSL2561 devices are:

- Master transmitter transmits to slave receiver (SMBus and I^2C):
  - The transfer direction in this case is not changed.
- Master reads slave immediately after the first byte (SMBus only):
  - At the moment of the first acknowledgment (provided by the slave receiver) the master transmitter becomes a master receiver and the slave receiver becomes a slave transmitter.
- Combined format (SMBus and I^2C):
  - During a change of direction within a transfer, the master repeats both a START condition and the slave address but with the R/W bit reversed. In this case, the master receiver terminates the transfer by generating a NACK on the last byte of the transfer and a STOP condition.

For a complete description of SMBus protocols, please review the SMBus Specification at http://www.smbus.org/specs. For a complete description of I^2C protocols, please review the I^2C Specification at http://www.semiconductors.philips.com.



A Acknowledge (this bit position may be 0 for an ACK or 1 for a NACK)

P Stop Condition

Rd Read (bit value of 1)

S Start Condition

Sr Repeated Start Condition

Wr Write (bit value of 0)

X Shown under a field indicates that that field is required to have a value of X

... Continuation of protocol

Master-to-Slave

Slave-to-Master

Figure 9. SMBus and I^2C Packet Protocol Element Key

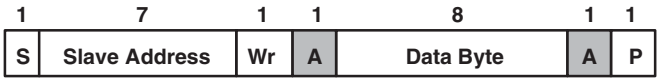


Figure 10. SMBus Send Byte Protocol

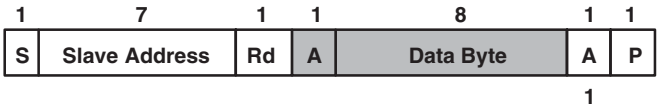


Figure 11. SMBus Receive Byte Protocol

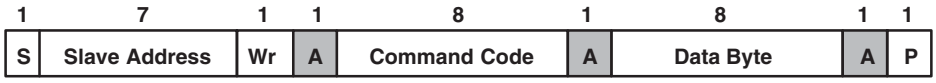


Figure 12. SMBus Write Byte Protocol

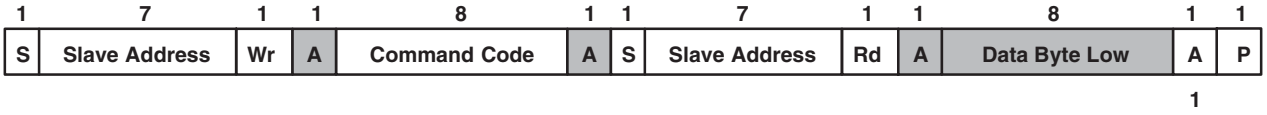


Figure 13. SMBus Read Byte Protocol

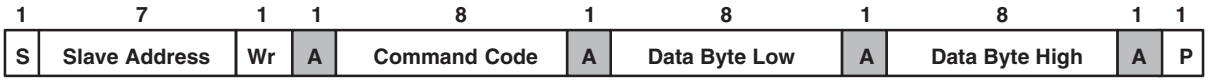


Figure 14. SMBus Write Word Protocol

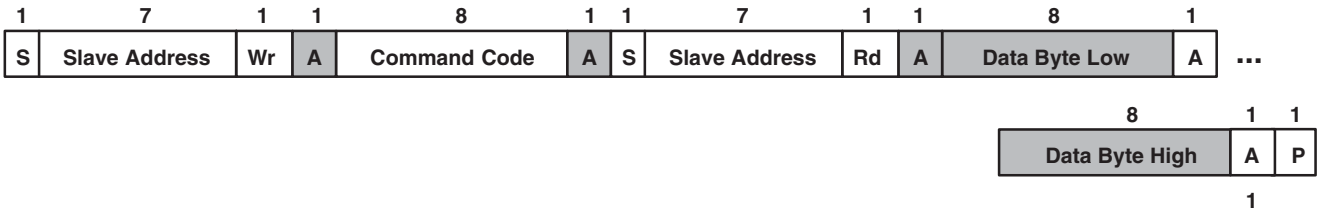


Figure 15. SMBus Read Word Protocol

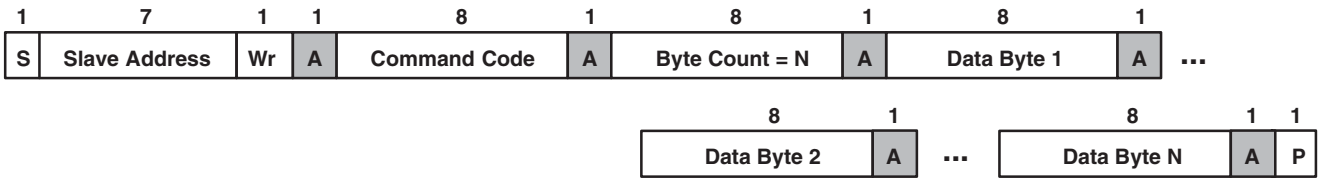


Figure 16. SMBus Block Write or I²C Write Protocols

NOTE: The I²C write protocol does not use the Byte Count packet, and the Master will continue sending Data Bytes until the Master initiates a Stop condition. See the Command Register on page 13 for additional information regarding the Block Read/Write protocol.

Figure 17. SMBus Block Read or I²C Read (Combined Format) Protocols

NOTE: The I²C read protocol does not use the Byte Count packet, and the Master will continue receiving Data Bytes until the Master initiates a Stop Condition. See the Command Register on page 13 for additional information regarding the Block Read/Write protocol.

Register Set

The TSL256x is controlled and monitored by sixteen registers (three are reserved) and a command register accessed through the serial interface. These registers provide for a variety of control functions and can be read to determine results of the ADC conversions. The register set is summarized in Table 2.

Table 2. Register Address

ADDRESS	RESISTER NAME	REGISTER FUNCTION
––	COMMAND	Specifies register address
0h	CONTROL	Control of basic functions
1h	TIMING	Integration time/gain control
2h	THRESHLOWLOW	Low byte of low interrupt threshold
3h	THRESHLOWHIGH	High byte of low interrupt threshold
4h	THRESHHIGHLOW	Low byte of high interrupt threshold
5h	THRESHHIGHHIGH	High byte of high interrupt threshold
6h	INTERRUPT	Interrupt control
7h	––	Reserved
8h	CRC	Factory test — not a user register
9h	––	Reserved
Ah	ID	Part number/ Rev ID
Bh	––	Reserved
Ch	DATA0LOW	Low byte of ADC channel 0
Dh	DATA0HIGH	High byte of ADC channel 0
Eh	DATA1LOW	Low byte of ADC channel 1
Fh	DATA1HIGH	High byte of ADC channel 1

The mechanics of accessing a specific register depends on the specific SMB protocol used. Refer to the section on SMBus protocols. In general, the COMMAND register is written first to specify the specific control/status register for following read/write operations.

Command Register

The command register specifies the address of the target register for subsequent read and write operations. The Send Byte protocol is used to configure the COMMAND register. The command register contains eight bits as described in Table 3. The command register defaults to 00h at power on.

Table 3. Command Register

	7	6	5	4	3	2	1	0	
	CMD	CLEAR	WORD	BLOCK	ADDRESS				COMMAND
Reset Value:	0	0	0	0	0	0	0	0	

FIELD	BIT	DESCRIPTION
CMD	7	Select command register. Must write as 1.
CLEAR	6	Interrupt clear. Clears any pending interrupt. This bit is a write-one-to-clear bit. It is self clearing.
WORD	5	SMB Write/Read Word Protocol. 1 indicates that this SMB transaction is using either the SMB Write Word or Read Word protocol.
BLOCK	4	Block Write/Read Protocol. 1 indicates that this transaction is using either the Block Write or the Block Read protocol. See Note below.
ADDRESS	3:0	Register Address. This field selects the specific control or status register for following write and read commands according to Table 2.

NOTE: An I^2C block transaction will continue until the Master sends a stop condition. See Figure 16 and Figure 17. Unlike the I2C protocol, the SMBus read/write protocol requires a Byte Count. All four ADC Channel Data Registers (Ch through Fh) can be read simultaneously in a single SMBus transaction. This is the only 32-bit data block supported by the TSL2560 SMBus protocol. The BLOCK bit must be set to 1, and a read condition should be initiated with a COMMAND CODE of 9Bh. By using a COMMAND CODE of 9Bh during an SMBus Block Read Protocol, the TSL2560 device will automatically insert the appropriate Byte Count (Byte Count = 4) as illustrated in Figure 17. A write condition should not be used in conjunction with the Bh register.

Control Register (0h)

The CONTROL register contains two bits and is primarily used to power the TSL256x device up and down as shown in Table 4.

Table 4. Control Register

	7	6	5	4	3	2	1	0	
0h	Resv	Resv	Resv	Resv	Resv	Resv	POWER		CONTROL
Reset Value:	0	0	0	0	0	0	0	0	

FIELD	BIT	DESCRIPTION
Resv	7:2	Reserved. Write as 0.
POWER	1:0	Power up/power down. By writing a 03h to this register, the device is powered up. By writing a 00h to this register, the device is powered down. ***NOTE:*** *If a value of 03h is written, the value returned during a read cycle will be 03h. This feature can be used to verify that the device is communicating properly.*

Timing Register (1h)

The TIMING register controls both the integration time and the gain of the ADC channels. A common set of control bits is provided that controls both ADC channels. The TIMING register defaults to 02h at power on.

Table 5. Timing Register

	7	6	5	4	3	2	1	0	
1h	Resv	Resv	Resv	GAIN	Manual	Resv	INTEG		TIMING
Reset Value:	0	0	0	0	0	0	1	0	

FIELD	BIT	DESCRIPTION
Resv	7–5	Reserved. Write as 0.
GAIN	4	Switches gain between low gain and high gain modes. Writing a 0 selects low gain (1×); writing a 1 selects high gain (16×).
Manual	3	Manual timing control. Writing a 1 begins an integration cycle. Writing a 0 stops an integration cycle. ***NOTE:*** *This field only has meaning when INTEG = 11. It is ignored at all other times.*
Resv	2	Reserved. Write as 0.
INTEG	1:0	Integrate time. This field selects the integration time for each conversion.

Integration time is dependent on the INTEG FIELD VALUE and the internal clock frequency. Nominal integration times and respective scaling between integration times scale proportionally as shown in Table 6. See Note 5 and Note 6 on page 5 for detailed information regarding how the scale values were obtained; see page 22 for further information on how to calculate lux.

Table 6. Integration Time

INTEG FIELD VALUE	SCALE	NOMINAL INTEGRATION TIME
00	0.034	13.7 ms
01	0.252	101 ms
10	1	402 ms
11	––	N/A

The manual timing control feature is used to manually start and stop the integration time period. If a particular integration time period is required that is not listed in Table 6, then this feature can be used. For example, the manual timing control can be used to synchronize the TSL256x device with an external light source (e.g. LED). A start command to begin integration can be initiated by writing a 1 to this bit field. Correspondingly, the integration can be stopped by simply writing a 0 to the same bit field.

Interrupt Threshold Register (2h – 5h)

The interrupt threshold registers store the values to be used as the high and low trigger points for the comparison function for interrupt generation. If the value generated by channel 0 crosses below or is equal to the low threshold specified, an interrupt is asserted on the interrupt pin. If the value generated by channel 0 crosses above the high threshold specified, an interrupt is asserted on the interrupt pin. Registers THRESHLOWLOW and THRESHLOWHIGH provide the low byte and high byte, respectively, of the lower interrupt threshold. Registers THRESHHIGHLOW and THRESHHIGHHIGH provide the low and high bytes, respectively, of the upper interrupt threshold. The high and low bytes from each set of registers are combined to form a 16-bit threshold value. The interrupt threshold registers default to 00h on power up.

Table 7. Interrupt Threshold Register

REGISTER	ADDRESS	BITS	DESCRIPTION
THRESHLOWLOW	2h	7:0	ADC channel 0 lower byte of the low threshold
THRESHLOWHIGH	3h	7:0	ADC channel 0 upper byte of the low threshold
THRESHHIGHLOW	4h	7:0	ADC channel 0 lower byte of the high threshold
THRESHHIGHHIGH	5h	7:0	ADC channel 0 upper byte of the high threshold

NOTE: Since two 8-bit values are combined for a single 16-bit value for each of the high and low interrupt thresholds, the Send Byte protocol should not be used to write to these registers. Any values transferred by the Send Byte protocol with the MSB set would be interpreted as the COMMAND field and stored as an address for subsequent read/write operations and not as the interrupt threshold information as desired. The Write Word protocol should be used to write byte-paired registers. For example, the THRESHLOWLOW and THRESHLOWHIGH registers (as well as the THRESHHIGHLOW and THRESHHIGHHIGH registers) can be written together to set the 16-bit ADC value in a single transaction.

Interrupt Control Register (6h)

The INTERRUPT register controls the extensive interrupt capabilities of the TSL256x. The TSL256x permits both SMB-Alert style interrupts as well as traditional level-style interrupts. The interrupt persist bit field (PERSIST) provides control over when interrupts occur. A value of 0 causes an interrupt to occur after every integration cycle regardless of the threshold settings. A value of 1 results in an interrupt after one integration time period outside the threshold window. A value of *N* (where *N* is 2 through15) results in an interrupt only if the value remains outside the threshold window for *N* consecutive integration cycles. For example, if *N* is equal to 10 and the integration time is 402 ms, then the total time is approximately 4 seconds.

When a level Interrupt is selected, an interrupt is generated whenever the last conversion results in a value outside of the programmed threshold window. The interrupt is active-low and remains asserted until cleared by writing the COMMAND register with the CLEAR bit set.

In SMBAlert mode, the interrupt is similar to the traditional level style and the interrupt line is asserted low. To clear the interrupt, the host responds to the SMBAlert by performing a modified Receive Byte operation, in which the Alert Response Address (ARA) is placed in the slave address field, and the TSL256x that generated the interrupt responds by returning its own address in the seven most significant bits of the receive data byte. If more than one device connected on the bus has pulled the SMBAlert line low, the highest priority (lowest address) device will win communication rights via standard arbitration during the slave address transfer. If the device loses this arbitration, the interrupt will not be cleared. The Alert Response Address is 0Ch.

When INTR = 11, the interrupt is generated immediately following the SMBus write operation. Operation then behaves in an SMBAlert mode, and the *software set* interrupt may be cleared by an SMBAlert cycle.

NOTE: Interrupts are based on the value of Channel 0 only.

Table 8. Interrupt Control Register

	7	6	5	4	3	2	1	0	
6h	Resv	Resv	INTR		PERSIST				INTERRUPT
Reset Value:	0	0	0	0	0	0	0	0	

FIELD	BITS	DESCRIPTION
Resv	7:6	Reserved. Write as 0.
INTR	5:4	INTR Control Select. This field determines mode of interrupt logic according to Table 9, below.
PERSIST	3:0	Interrupt persistence. Controls rate of interrupts to the host processor as shown in Table 10, below.

Table 9. Interrupt Control Select

INTR FIELD VALUE	READ VALUE
00	Interrupt output disabled
01	Level Interrupt
10	SMBAlert compliant
11	Test Mode: Sets interrupt and functions as mode 10

NOTE: Field value of 11 may be used to test interrupt connectivity in a system or to assist in debugging interrupt service routine software.

Table 10. Interrupt Persistence Select

PERSIST FIELD VALUE	INTERRUPT PERSIST FUNCTION
0000	Every ADC cycle generates interrupt
0001	Any value outside of threshold range
0010	2 integration time periods out of range
0011	3 integration time periods out of range
0100	4 integration time periods out of range
0101	5 integration time periods out of range
0110	6 integration time periods out of range
0111	7 integration time periods out of range
1000	8 integration time periods out of range
1001	9 integration time periods out of range
1010	10 integration time periods out of range
1011	11 integration time periods out of range
1100	12 integration time periods out of range
1101	13 integration time periods out of range
1110	14 integration time periods out of range
1111	15 integration time periods out of range

ID Register (Ah)

The ID register provides the value for both the part number and silicon revision number for that part number. It is a read-only register, whose value never changes.

Table 11. ID Register

	7	6	5	4	3	2	1	0	
Ah	PARTNO				REVNO				ID
Reset Value:	–	–	–	–	–	–	–	–	

FIELD	BITS	DESCRIPTION
PARTNO	7:4	Part Number Identification: field value 0000 = TSL2560, field value 0001 = TSL2561
REVNO	3:0	Revision number identification

ADC Channel Data Registers (Ch – Fh)

The ADC channel data are expressed as 16-bit values spread across two registers. The ADC channel 0 data registers, DATA0LOW and DATA0HIGH provide the lower and upper bytes, respectively, of the ADC value of channel 0. Registers DATA1LOW and DATA1HIGH provide the lower and upper bytes, respectively, of the ADC value of channel 1. All channel data registers are read-only and default to 00h on power up.

Table 12. ADC Channel Data Registers

REGISTER	ADDRESS	BITS	DESCRIPTION
DATA0LOW	Ch	7:0	ADC channel 0 lower byte
DATA0HIGH	Dh	7:0	ADC channel 0 upper byte
DATA1LOW	Eh	7:0	ADC channel 1 lower byte
DATA1HIGH	Fh	7:0	ADC channel 1 upper byte

The upper byte data registers can only be read following a read to the corresponding lower byte register. When the lower byte register is read, the upper eight bits are strobed into a shadow register, which is read by a subsequent read to the upper byte. The upper register will read the correct value even if additional ADC integration cycles end between the reading of the lower and upper registers.

NOTE: The Read Word protocol can be used to read byte-paired registers. For example, the DATA0LOW and DATA0HIGH registers (as well as the DATA1LOW and DATA1HIGH registers) may be read together to obtain the 16-bit ADC value in a single transaction

APPLICATION INFORMATION: SOFTWARE

Basic Operation

After applying V_{DD}, the device will initially be in the power-down state. To operate the device, issue a command to access the CONTROL register followed by the data value 03h to power up the device. At this point, both ADC channels will begin a conversion at the default integration time of 400 ms. After 400 ms, the conversion results will be available in the DATA0 and DATA1 registers. Use the following pseudo code to read the data registers:

```
// Read ADC Channels Using Read Word Protocol – RECOMMENDED
      Address = 0x39                          //Slave addr – also 0x29 or 0x49

      //Address the Ch0 lower data register and configure for Read Word
      Command = 0xAC                          //Set Command bit and Word bit

      //Reads two bytes from sequential registers 0x0C and 0x0D
      //Results are returned in DataLow and DataHigh variables
      ReadWord (Address, Command, DataLow, DataHigh)
      Channel0 = 256 * DataHigh + DataLow

      //Address the Ch1 lower data register and configure for Read Word
      Command = 0xAE                          //Set bit fields 7 and 5

      //Reads two bytes from sequential registers 0x0E and 0x0F
      //Results are returned in DataLow and DataHigh variables
      ReadWord (Address, Command, DataLow, DataHigh)
      Channel1 = 256 * DataHigh + DataLow     //Shift DataHigh to upper byte


// Read ADC Channels Using Read Byte Protocol
      Address = 0x39                          //Slave addr – also 0x29 or 0x49
      Command = 0x8C                          //Address the Ch0 lower data register
      ReadByte (Address, Command, DataLow)    //Result returned in DataLow
      Command = 0x8D                          //Address the Ch0 upper data register
      ReadByte (Address, Command, DataHigh)   //Result returned in DataHigh
      Channel0 = 256 * DataHigh + DataLow     //Shift DataHigh to upper byte

      Command = 0x8E                          //Address the Ch1 lower data register
      ReadByte (Address, Command, DataLow)    //Result returned in DataLow
      Command = 0x8F                          //Address the Ch1 upper data register
      ReadByte (Address, Command, DataHigh)   //Result returned in DataHigh
      Channel1 = 256 * DataHigh + DataLow     //Shift DataHigh to upper byte
```

APPLICATION INFORMATION: SOFTWARE

Configuring the Timing Register

The command, timing, and control registers are initialized to default values on power up. Setting these registers to the desired values would be part of a normal initialization or setup procedure. In addition, to maximize the performance of the device under various conditions, the integration time and gain may be changed often during operation. The following pseudo code illustrates a procedure for setting up the timing register for various options:

```
// Set up Timing Register
       //Low Gain (1x), integration time of 402ms (default value)
       Address = 0x39
       Command = 0x81
       Data = 0x02
       WriteByte(Address, Command, Data)

       //Low Gain (1x), integration time of 101ms
       Data = 0x01
       WriteByte(Address, Command, Data)

       //Low Gain (1x), integration time of 13.7ms
       Data = 0x00
       WriteByte(Address, Command, Data)

       //High Gain (16x), integration time of 101ms
       Data = 0x11
       WriteByte(Address, Command, Data)

//Read data registers (see Basic Operation example)


//Perform Manual Integration
       //Set up for manual integration with Gain of 1x
       Data = 0x03
       //Set manual integration mode – device stops converting
       WriteByte(Address, Command, Data)

       //Begin integration period
       Data = 0x0B
       WriteByte(Address, Command, Data)

       //Integrate for 50ms
       Sleep (50)                                  //Wait for 50ms

       //Stop integrating
       Data = 0x03
       WriteByte(Address, Command, Data)

//Read data registers (see Basic Operation example)
```

APPLICATION INFORMATION: SOFTWARE

Interrupts

The interrupt feature of the TSL256x device simplifies and improves system efficiency by eliminating the need to poll the sensor for a light intensity value. Interrupt styles are determined by the INTR field in the Interrupt Register. The interrupt feature may be disabled by writing a field value of 00h to the Interrupt Control Register so that polling can be performed.

The versatility of the interrupt feature provides many options for interrupt configuration and usage. The primary purpose of the interrupt function is to provide a meaningful change in light intensity. However, it also be used as an end-of-conversion signal. The concept of a *meaningful change* can be defined by the user both in terms of light intensity and time, or persistence, of that change in intensity. The TSL256x device implements two 16-bit-wide interrupt threshold registers that allow the user to define a threshold above and below the current light level. An interrupt will then be generated when the value of a conversion exceeds either of these limits. For simplicity of programming, the threshold comparison is accomplished only with Channel 0. This simplifies calculation of thresholds that are based, for example, on a percent of the current light level. It is adequate to use only one channel when calculating light intensity differences since, for a given light source, the channel 0 and channel 1 values are linearly proportional to each other and thus both values scale linearly with light intensity.

To further control when an interrupt occurs, the TSL256x device provides an interrupt persistence feature. This feature allows the user to specify a number of conversion cycles for which a light intensity exceeding either interrupt threshold must persist before actually generating an interrupt. This can be used to prevent transient changes in light intensity from generating an unwanted interrupt. With a value of 1, an interrupt occurs immediately whenever either threshold is exceeded. With values of *N*, where *N* can range from 2 to 15, *N* consecutive conversions must result in values outside the interrupt window for an interrupt to be generated. For example, if *N* is equal to 10 and the integration time is 402 ms, then an interrupt will not be generated unless the light level persists for more than 4 seconds outside the threshold.

Two different interrupt styles are available: Level and SMBus Alert. The difference between these two interrupt styles is how they are cleared. Both result in the interrupt line going active low and remaining low until the interrupt is cleared. A level style interrupt is cleared by setting the CLEAR bit (bit 6) in the COMMAND register. The SMBus Alert style interrupt is cleared by an Alert Response as described in the Interrupt Control Register section and SMBus specification.

To configure the interrupt as an end-of-conversion signal, the interrupt PERSIST field is set to 0. Either Level or SMBus Alert style can be used. An interrupt will be generated upon completion of each conversion. The interrupt threshold registers are ignored. The following example illustrates the configuration of a level interrupt:

```
// Set up end-of-conversion interrupt, Level style
      Address = 0x39                          //Slave addr also 0x29 or 0x49
      Command = 0x86                          //Address Interrupt Register
      Data = 0x10                             //Level style, every ADC cycle
      WriteByte(Address, Command, Data)
```

APPLICATION INFORMATION: SOFTWARE

The following example pseudo code illustrates the configuration of an SMB Alert style interrupt when the light intensity changes 20% from the current value, and persists for 3 conversion cycles:

```
// Read current light level
       Address = 0x39                              //Slave addr also 0x29 or 0x49
       Command = 0xAC                              //Set Command bit and Word bit
       ReadWord (Address, Command, DataLow, DataHigh)
       Channel0 = (256 * DataHigh) + DataLow

       //Calculate upper and lower thresholds
       T_Upper = Channel0 + (0.2 * Channel0)
       T_Lower = Channel0 - (0.2 * Channel0)

       //Write the lower threshold register
       Command = 0xA2                              //Addr lower threshold reg, set Word Bit
       WriteWord (Address, Command, T_Lower.LoByte, T_Lower.HiByte)

       //Write the upper threshold register
       Command = 0xA4                              //Addr upper threshold reg, set Word bit
       WriteWord (Address, Command, T_Upper.LoByte, T_Upper.HiByte)

       //Enable interrupt
       Command = 0x86                              //Address interrupt register
       Data = 0x23                                 //SMBAlert style, PERSIST = 3
       WriteByte(Address, Command, Data)
```

In order to generate an interrupt on demand during system test or debug, a test mode (INTR = 11) can be used. The following example illustrates how to generate an interrupt on demand:

```
// Generate an interrupt
       Address = 0x39                              //Slave addr also 0x29 or 0x49
       Command = 0x86                              //Address Interrupt register
       Data = 0x30                                 //Test interrupt
       WriteByte(Address, Command, Data)

       //Interrupt line should now be low
```

APPLICATION INFORMATION: SOFTWARE

Calculating Lux

The TSL256x is intended for use in ambient light detection applications such as display backlight control, where adjustments are made to display brightness or contrast based on the brightness of the ambient light, as perceived by the human eye. Conventional silicon detectors respond strongly to infrared light, which the human eye does not see. This can lead to significant error when the infrared content of the ambient light is high, such as with incandescent lighting, due to the difference between the silicon detector response and the brightness perceived by the human eye.

This problem is overcome in the TSL256x through the use of two photodiodes. One of the photodiodes (channel 0) is sensitive to both visible and infrared light, while the second photodiode (channel 1) is sensitive primarily to infrared light. An integrating ADC converts the photodiode currents to digital outputs. Channel 1 digital output is used to compensate for the effect of the infrared component of light on the channel 0 digital output. The ADC digital outputs from the two channels are used in a formula to obtain a value that approximates the human eye response in the commonly used Illuminance unit of Lux:

CS Package

For $0 < CH1/CH0 \leq 0.52$	$Lux = 0.0315 \times CH0 - 0.0593 \times CH0 \times ((CH1/CH0)^{1.4})$
For $0.52 < CH1/CH0 \leq 0.65$	$Lux = 0.0229 \times CH0 - 0.0291 \times CH1$
For $0.65 < CH1/CH0 \leq 0.80$	$Lux = 0.0157 \times CH0 - 0.0180 \times CH1$
For $0.80 < CH1/CH0 \leq 1.30$	$Lux = 0.00338 \times CH0 - 0.00260 \times CH1$
For $CH1/CH0 > 1.30$	$Lux = 0$

T, FN, and CL Package

For $0 < CH1/CH0 \leq 0.50$	$Lux = 0.0304 \times CH0 - 0.062 \times CH0 \times ((CH1/CH0)^{1.4})$
For $0.50 < CH1/CH0 \leq 0.61$	$Lux = 0.0224 \times CH0 - 0.031 \times CH1$
For $0.61 < CH1/CH0 \leq 0.80$	$Lux = 0.0128 \times CH0 - 0.0153 \times CH1$
For $0.80 < CH1/CH0 \leq 1.30$	$Lux = 0.00146 \times CH0 - 0.00112 \times CH1$
For $CH1/CH0 > 1.30$	$Lux = 0$

The formulas shown above were obtained by optical testing with fluorescent and incandescent light sources, and apply only to open-air applications. Optical apertures (e.g. light pipes) will affect the incident light on the device.

Simplified Lux Calculation

Below is the argument and return value including source code (shown on following page) for calculating lux. The source code is intended for embedded and/or microcontroller applications. Two individual code sets are provided, one for the T, FN, and CL packages, and one for the CS package. All floating point arithmetic operations have been eliminated since embedded controllers and microcontrollers generally do not support these types of operations. Since floating point has been removed, scaling must be performed prior to calculating illuminance if the integration time is not 402 ms and/or if the gain is not 16× as denoted in the source code on the following pages. This sequence scales first to mitigate rounding errors induced by decimal math.

```
extern unsigned int CalculateLux(unsigned int iGain, unsigned int tInt, unsigned int
    ch0, unsigned int ch1, int iType)
```

```
//*****************************************************************************
//
//  Copyright © 2004-2005 TAOS, Inc.
//
//  THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
//  KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
//  IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
//  PURPOSE.
//
//     Module Name:
//           lux.cpp
//
//*****************************************************************************

#define LUX_SCALE   14     // scale by 2^14
#define RATIO_SCALE 9      // scale ratio by 2^9

//--------------------------------------------------
// Integration time scaling factors
//--------------------------------------------------

#define CH_SCALE           10     // scale channel values by 2^10
#define CHSCALE_TINT0      0x7517 // 322/11 * 2^CH_SCALE
#define CHSCALE_TINT1      0x0fe7 // 322/81 * 2^CH_SCALE

//--------------------------------------------------
// T, FN, and CL Package coefficients
//--------------------------------------------------
// For Ch1/Ch0=0.00 to 0.50
//           Lux/Ch0=0.0304-0.062*((Ch1/Ch0)^1.4)
//           piecewise approximation
//                 For Ch1/Ch0=0.00 to 0.125:
//                       Lux/Ch0=0.0304-0.0272*(Ch1/Ch0)
//
//                 For Ch1/Ch0=0.125 to 0.250:
//                       Lux/Ch0=0.0325-0.0440*(Ch1/Ch0)
//
//                 For Ch1/Ch0=0.250 to 0.375:
//                       Lux/Ch0=0.0351-0.0544*(Ch1/Ch0)
//
//                 For Ch1/Ch0=0.375 to 0.50:
//                       Lux/Ch0=0.0381-0.0624*(Ch1/Ch0)
//
// For Ch1/Ch0=0.50 to 0.61:
//           Lux/Ch0=0.0224-0.031*(Ch1/Ch0)
//
// For Ch1/Ch0=0.61 to 0.80:
//           Lux/Ch0=0.0128-0.0153*(Ch1/Ch0)
//
// For Ch1/Ch0=0.80 to 1.30:
//           Lux/Ch0=0.00146-0.00112*(Ch1/Ch0)
//
// For Ch1/Ch0>1.3:
//           Lux/Ch0=0
//--------------------------------------------------
#define K1T   0x0040       // 0.125 * 2^RATIO_SCALE
#define B1T   0x01f2       // 0.0304 * 2^LUX_SCALE
#define M1T   0x01be       // 0.0272 * 2^LUX_SCALE

#define K2T   0x0080       // 0.250 * 2^RATIO_SCALE
```

```
#define B2T   0x0214        // 0.0325 * 2^LUX_SCALE
#define M2T   0x02d1        // 0.0440 * 2^LUX_SCALE

#define K3T   0x00c0        // 0.375 * 2^RATIO_SCALE
#define B3T   0x023f        // 0.0351 * 2^LUX_SCALE
#define M3T   0x037b        // 0.0544 * 2^LUX_SCALE

#define K4T   0x0100        // 0.50 * 2^RATIO_SCALE
#define B4T   0x0270        // 0.0381 * 2^LUX_SCALE
#define M4T   0x03fe        // 0.0624 * 2^LUX_SCALE
#define K5T   0x0138        // 0.61 * 2^RATIO_SCALE
#define B5T   0x016f        // 0.0224 * 2^LUX_SCALE
#define M5T   0x01fc        // 0.0310 * 2^LUX_SCALE

#define K6T   0x019a        // 0.80 * 2^RATIO_SCALE
#define B6T   0x00d2        // 0.0128 * 2^LUX_SCALE
#define M6T   0x00fb        // 0.0153 * 2^LUX_SCALE

#define K7T   0x029a        // 1.3 * 2^RATIO_SCALE
#define B7T   0x0018        // 0.00146 * 2^LUX_SCALE
#define M7T   0x0012        // 0.00112 * 2^LUX_SCALE

#define K8T   0x029a        // 1.3 * 2^RATIO_SCALE
#define B8T   0x0000        // 0.000 * 2^LUX_SCALE
#define M8T   0x0000        // 0.000 * 2^LUX_SCALE

//--------------------------------------------------
// CS package coefficients
//--------------------------------------------------
// For 0 <= Ch1/Ch0 <= 0.52
//           Lux/Ch0 = 0.0315-0.0593*((Ch1/Ch0)^1.4)
//           piecewise approximation
//                 For 0 <= Ch1/Ch0 <= 0.13
//                       Lux/Ch0 = 0.0315-0.0262*(Ch1/Ch0)
//                 For 0.13 <= Ch1/Ch0 <= 0.26
//                       Lux/Ch0 = 0.0337-0.0430*(Ch1/Ch0)
//                 For 0.26 <= Ch1/Ch0 <= 0.39
//                       Lux/Ch0 = 0.0363-0.0529*(Ch1/Ch0)
//                 For 0.39 <= Ch1/Ch0 <= 0.52
//                       Lux/Ch0 = 0.0392-0.0605*(Ch1/Ch0)
// For 0.52 < Ch1/Ch0 <= 0.65
//     Lux/Ch0 = 0.0229-0.0291*(Ch1/Ch0)
// For 0.65 < Ch1/Ch0 <= 0.80
//     Lux/Ch0 = 0.00157-0.00180*(Ch1/Ch0)
// For 0.80 < Ch1/Ch0 <= 1.30
//     Lux/Ch0 = 0.00338-0.00260*(Ch1/Ch0)
// For Ch1/Ch0 > 1.30
//     Lux = 0
//--------------------------------------------------
#define K1C   0x0043 // 0.130 * 2^RATIO_SCALE
#define B1C   0x0204 // 0.0315 * 2^LUX_SCALE
#define M1C   0x01ad // 0.0262 * 2^LUX_SCALE

#define K2C   0x0085 // 0.260 * 2^RATIO_SCALE
#define B2C   0x0228 // 0.0337 * 2^LUX_SCALE
#define M2C   0x02c1 // 0.0430 * 2^LUX_SCALE

#define K3C   0x00c8 // 0.390 * 2^RATIO_SCALE
#define B3C   0x0253 // 0.0363 * 2^LUX_SCALE
#define M3C   0x0363 // 0.0529 * 2^LUX_SCALE
```