EVALUATION KIT AVAILABLE

*Click here to ask an associate for production status of specific part numbers.*

# MAX31889

# ±0.25°C Accurate I²C Temperature Sensor

## General Description

The MAX31889 operates from a 1.7V to 3.6V supply voltage, and is a low power, high-accuracy digital temperature sensor with ±0.25ºC accuracy from -20ºC to +105ºC and ±0.65ºC accuracy from -40ºC to +125ºC. The MAX31889 has 16-bit resolution (0.005ºC).

The device uses a standard I²C serial interface to communicate with a host controller. Two GPIO pins are available. GPIO1 can be configured to trigger a temperature conversion, while GPIO0 can be configured to generate an interrupt for selectable status bits.

The MAX31889 includes a 32-word FIFO for the temperature data and also includes high and low threshold digital temperature alarms. The device is available in a 2mm x 2mm x 0.8mm, 6-pin µDFN package.

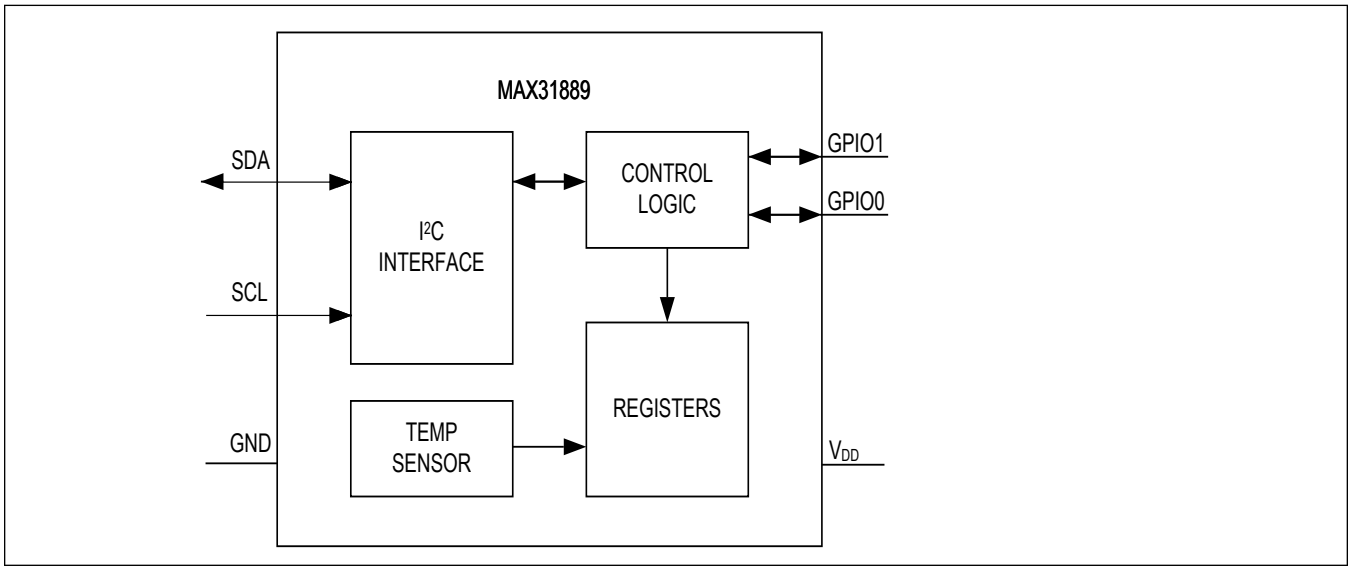
## Applications

- Precision Temperature Monitoring
- RTD Replacement
- Internet of Things (IoT) Sensors

## Benefits and Features

- High Accuracy and Precision
  - ±0.25ºC Accuracy from -20ºC to +105ºC
  - ±0.65ºC Accuracy from -40ºC to +125ºC
- Long Battery Life
  - 1.7V to 3.6V Operating Voltage
  - 68µA Operating Current During Measurement
  - 0.55µA Standby Current
- Small Size
  - 2mm x 2mm x 0.8mm, 6-Pin µDFN
- Safety and Compliance
  - High and Low Temperature Alarms
- I²C Digital Interface
  - Configurable Convert Temperature Input Pin
  - Configurable Interrupt Output Pin
  - 32-Word FIFO for Temperature Data
  - 4 I²C Addresses Available—More Addresses Available by Request
  - Unique ROM IDs Allow Device to Be NIST Traceable

## Simplified Block Diagram




*Ordering Information appears at end of data sheet.*

19-100693; Rev 1; 9/21

© 2022 Analog Devices, Inc. All rights reserved. Trademarks and registered trademarks are the property of their respective owners.

One Analog Way, Wilmington, MA 01887 U.S.A. | Tel: 781.329.4700 | © 2022 Analog Devices, Inc. All rights reserved.

## Absolute Maximum Ratings

GPIOx to GND ..... -0.3V to +6V
SDA,SCL to GND ..... -0.3V to +6V
Continuous Power Dissipation (Multilayer Board µDFN, $T_A$ = +70 °C, derate 5.47mW/°C above 70°C ) ..... 437.25mW/°C
Operating Temperature Range ..... -40°C to 125°C
Junction Temperature ..... +150°C
Storage Temperature Range ..... -65°C to +150°C
Lead Temperature (soldering, 10s) ..... +300°C
Soldering Temperature (reflow) ..... +260°C

*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 in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

## Package Information

### µDFN

| | |
|---|---|
| Package Code | L622+2 |
| Outline Number | **21-100397** |
| Land Pattern Number | **90-100138** |
| **Thermal Resistance, Four-Layer Board:** | |
| Junction to Ambient ($\theta_{JA}$) | 182.96°C/W |
| Junction to Case ($\theta_{JC}$) | 50.75°C/W |

For the latest package outline information and land patterns (footprints), go to ***www.maximintegrated.com/packages***. Note that a "+", "#", or "-" in the package code indicates RoHS status only. Package drawings may show a different suffix character, but the drawing pertains to the package regardless of RoHS status.

Package thermal resistances were obtained using the method described in JEDEC specification JESD51-7, using a four-layer board. For detailed information on package thermal considerations, refer to ***www.maximintegrated.com/thermal-tutorial***.

## Electrical Characteristics

($V_{DD}$ = 1.8V, $T_A$ = +25°C, minimum and maximum limits are from $T_A$ = -40°C to +125°C, unless otherwise noted. (Note 1))

| PARAMETER | SYMBOL | CONDITIONS | | MIN | TYP | MAX | UNITS |
|---|---|---|---|---|---|---|---|
| **TEMPERATURE SENSOR** | | | | | | | |
| Temperature Measurement Error | | $V_{DD}$ = 1.7V to 3.6V | -20°C to +105°C, 3-sigma | -0.25 | | +0.25 | °C |
| | | | -20°C to +105°C, 6-sigma | -0.45 | | +0.45 | |
| | | | -40°C to +125°C, 3-sigma | -0.65 | | +0.65 | |
| | | | -40°C to +125°C, 6-sigma | -0.9 | | +0.9 | |
| Resolution | | 16-Bit | | | 0.005 | | °C |
| Repeatability | | VDD=1.8V, 1sps,120 samples | | | 0.008 | | °C rms |
| Conversion Time | $t_{CONV}$ | 16-bit | | | 16.5 | 17.85 | ms |
| Long-Term Stability | | $T_A$ = +70°C, 0% RH | | | 0.015 | | °C/ 1000hrs |
| Operating Supply Voltage | $V_{DD}$ | Guaranteed by PSRR | | 1.7 | | 3.6 | V |
| DC Power Supply Rejection Ratio | PSRR | $T_A$ = +25°C | | | 0.006 | | °C/V |
| Operating Current | | During Conversion, $V_{DD}$ = 1.7V to 3.6V | $T_A$ = +25ºC | | 68 | 105 | µA |
| | | | $T_A$ = +85ºC | | 71 | 110 | |
| | | | $T_A$ = +125ºC | | 87 | 140 | µA |
| Standby Current | | $V_{DD}$ = 3.6V | $T_A$ = +25ºC | | 0.55 | 3 | µA |
| | | | $T_A$ = +85ºC | | 3.7 | 10 | |
| | | | $T_A$ = +125ºC | | 15 | 38 | |
| **GPIO PINS** | | | | | | | |
| Input Voltage Low | $V_{IL\_GPIO}$ | | | | | 0.4 | V |
| Input Voltage High | $V_{IH\_GPIO}$ | | | 1.4 | | | V |
| Input Hysteresis | $V_{HYS\_GPIO}$ | | | | 320 | | mV |
| Input Leakage Current | $I_{IN\_GPIO}$ | $V_{IN}$ = 0V, $T_A$ = +25°C | | | 0.01 | 1 | µA |
| Input Capacitance | $C_{IN\_GPIO}$ | | | | 10 | | pF |
| Input Low Pulse Width | | | | 5 | | | µs |
| Output Low Voltage | $V_{OL\_GPIO}$ | $I_{SINK}$ = 2mA | | | | 0.4 | V |
| **I2C INTERFACE / DIGITAL I/O CHARACTERISTICS** | | | | | | | |
| Input Voltage Low | $V_{IL}$ | | | | | 0.4 | V |
| Input Voltage High | $V_{IH}$ | | | 1.4 | | | V |
| Input Hysteresis | $V_{HYS}$ | | | | 200 | | mV |
| Input Capacitance | $C_{IN}$ | | | | 10 | | pF |
| Open Drain Output Low Voltage | $V_{OL\_OD}$ | $I_{SINK}$ = 6mA, SDA Pin Only | | | | 0.4 | V |
| **I2C INTERFACE / TIMING CHARACTERISTICS (Note 2)** | | | | | | | |
| I2C Write Address | | GPIO1 = GPIO0 = 0V. See Table 3 | | | A0 | | Hex |
| I2C Read Address | | GPIO1 = GPIO0 = 0V. See Table 3 | | | A1 | | Hex |

## Electrical Characteristics (continued)

($V_{DD}$ = 1.8V, $T_A$ = +25°C, minimum and maximum limits are from $T_A$ = -40°C to +125°C, unless otherwise noted. (Note 1))

| PARAMETER | SYMBOL | CONDITIONS | MIN | TYP | MAX | UNITS |
|---|---|---|---|---|---|---|
| Serial Clock Frequency | $f_{SCL}$ | | 0 | | 400 | kHz |
| Bus Free Time Between STOP and START Conditions | $t_{BUF}$ | | 1.3 | | | µs |
| Hold Time START and Repeat START Condition | $t_{HD\_STA}$ | | 0.6 | | | µs |
| SCL Pulse-Width Low | $t_{LOW}$ | | 1.3 | | | µs |
| SCL Pulse-Width High | $t_{HIGH}$ | | 0.6 | | | µs |
| Setup Time for a Repeated START Condition | $t_{SU\_STA}$ | | 0.6 | | | µs |
| Data Hold Time | $t_{HD\_DAT}$ | | 0 | | 900 | ns |
| Data Setup Time | $t_{SU\_DAT}$ | | 100 | | | ns |
| Setup Time for STOP Condition | $t_{SU\_STO}$ | | 0.6 | | | µs |
| Pulse Width of Suppressed Spike | $t_{SP}$ | | | | 50 | ns |
| Bus Capacitance | $C_B$ | | | | 400 | pF |
| SDA and SCL Receiving Rise Time | $t_R$ | | $20 + 0.1C_B$ | | 300 | ns |
| SDA and SCL Receiving Fall Time | $t_F$ | | $20 + 0.1C_B$ | | 300 | ns |
| SDA Transmitting Fall Time | $t_{TF}$ | | $20 + 0.1C_B$ | | 300 | ns |

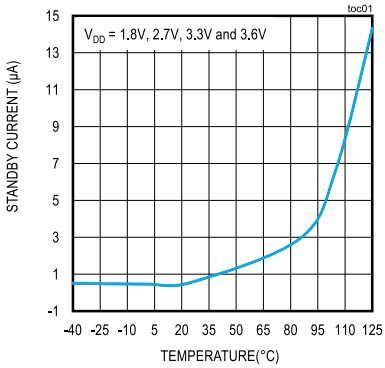
**Note 1:** All devices are 100% production tested at $T_A$ = +25°C. Specifications over temperature limits are guaranteed by design and characterization.

**Note 2:** For design guidance only. Not production tested.

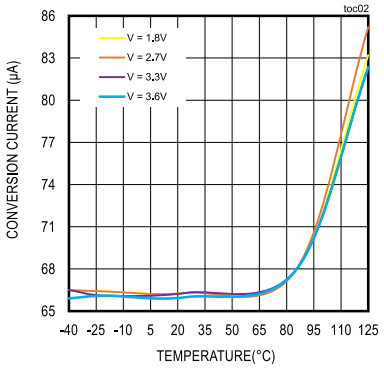
## Typical Operating Characteristics

($V_{DD}$ = +1.8V, $T_A$ = +25°C, unless otherwise noted.)

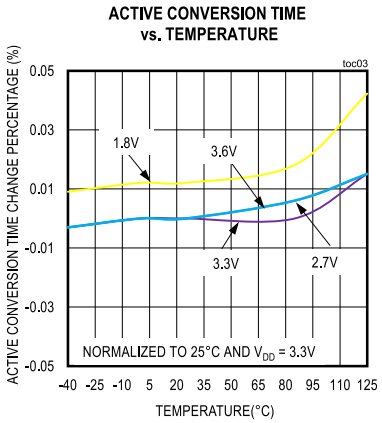
STANDBY CURRENT vs. TEMPERATURE



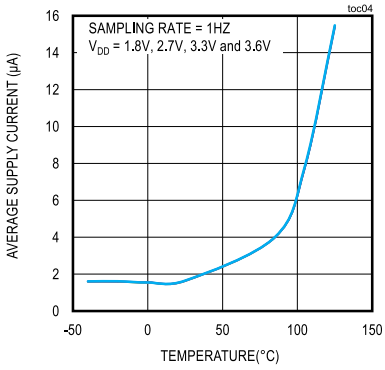

CONVERSION CURRENT vs. TEMPERATURE



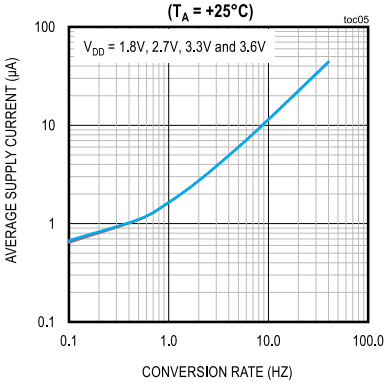

ACTIVE CONVERSION TIME vs. TEMPERATURE



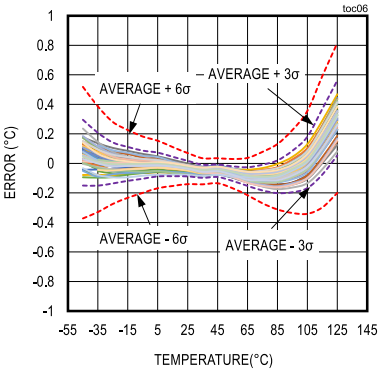

AVERAGE SUPPLY CURRENT vs. TEMPERATURE



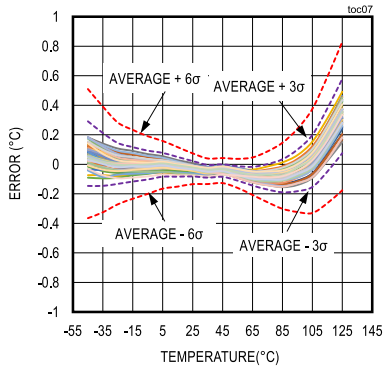

AVERAGE SUPPLY CURRENT vs. CONVERSION RATE ($T_A$ = +25°C)



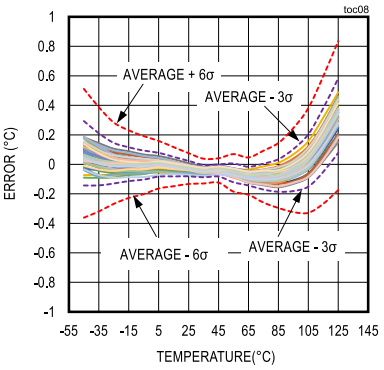

ACCURACY vs. TEMPERATURE ($V_{DD}$ = 1.8V 59UNITS)



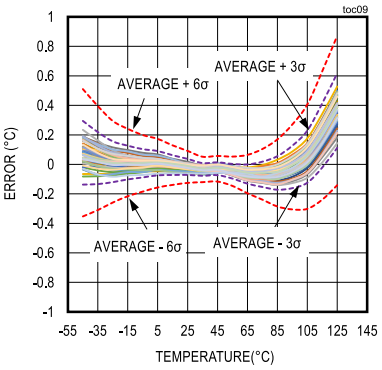

ACCURACY vs. TEMPERATURE ($V_{DD}$ = 3.0V 59UNITS)




ACCURACY vs. TEMPERATURE ($V_{DD}$ = 3.3V 59UNITS)



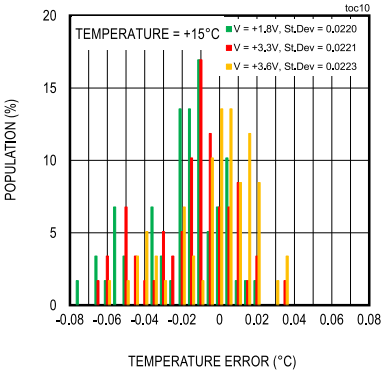

ACCURACY vs. TEMPERATURE ($V_{DD}$ = 3.6V 59UNITS)



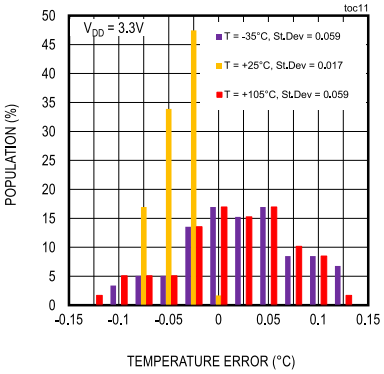
## Typical Operating Characteristics (continued)

($V_{DD}$ = +1.8V, $T_A$ = +25°C, unless otherwise noted.)

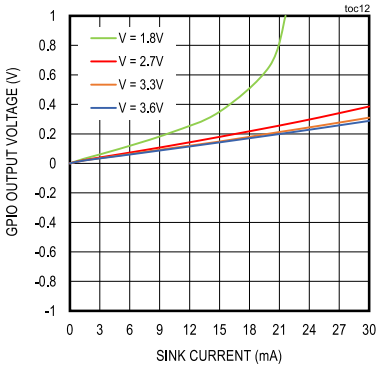
TEMPERATURE ERROR OVER SUPPLY VOLTAGE
(NO AVERAGING, 59UNITS)



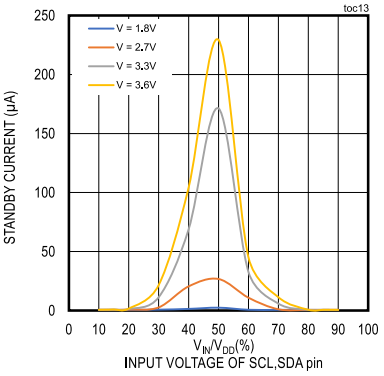

TEMPERATURE ERROR OVER TEMPERATURE
(NO AVERAGING, 59UNITS)




GPIO OUTPUT VOLTAGE
vs. SINK CURRENT

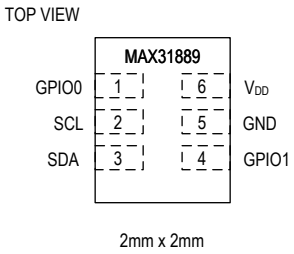



STANDBY CURRENT
vs. INPUT CELL VOLTAGE



## Pin Configuration

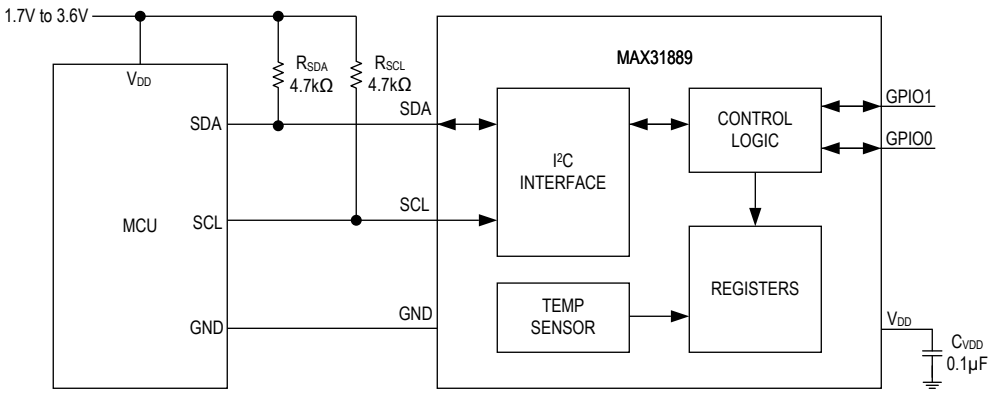
### µDFN




## Pin Description

| PIN | NAME | FUNCTION |
|---|---|---|
| 1 | GPIO0 | General Purpose Input/Output 0. Programmable digital input or output. State at each I²C start condition can be used to configure I²C addresses. See Table 1 and Table 2. Can be configured as an interrupt output. |
| 2 | SCL | SCL Input. I²C Clock. |
| 3 | SDA | SDA Input/Output. I²C Data I/O. |
| 4 | GPIO1 | General Purpose Input/Output 1. State at each I²C start condition can be used to configure I²C addresses (see Table 1 and Table 2). Can be configured to act as an external temperature convert input. |
| 5 | GND | Ground. |
| 6 | $V_{DD}$ | Power Supply Input. A 100nF capacitor connected between $V_{DD}$ and GND is recommended. |

## Functional Diagrams



## Detailed Description

The MAX31889 temperature sensor provides 16-bit Celsius temperature measurements with ±0.25°C accuracy over a -20°C to +105°C temperature range and ±0.65°C accuracy over a -40°C to +125°C temperature range.

A host can communicate with the sensor over an I2C/SMBus-compatible 2-wire interface with serial data (SDA) and serial clock (SCL) lines to read the FIFO, which contains up to 32, 2-byte temperature readings.

In addition to the FIFO, the interface provides access to the 2-byte high alarm trigger and 2-byte low alarm trigger registers (AH and AL) and a temperature sensor setup register. The Alarm High, Alarm Low, and Setup registers are volatile, so they don't retain data when the device is powered down.

The MAX31889 offers two programmable GPIO pins. The default state of the GPIO pins at power-up determines the 2 LSBs in the I2C address of the device. GPIO1 allows for an optional external convert temperature trigger while GPIO0 can be configured as an interrupt for selectable status bits.

### Operation

#### Measuring Temperature

The device's core functionality is its direct-to-digital temperature sensor. The device powers up in a low-power standby state. There are two ways to initiate a temperature measurement; the master can write a "1" to the CONVERT_T bit in the TEMP_SENSOR_SETUP [0x14] register or set GPIO1 as an active low convert trigger to allow temperature conversions to be triggered by an external signal. Following the conversion, the resulting temperature sensor data is stored in the FIFO memory as a 2-byte temperature word and the device returns to the standby state.

The output temperature data is calibrated in degrees Celsius. The temperature data is stored as a left-justified, 16-bit sign-extended two's complement number in the FIFO Data register (see Figure 1). The data is two's complement where the MSB indicates the sign of the temperature, with an MSB of 1 indicating a negative temperature and an MSB of 0 indicating a positive temperature.

To calculate the temperature from the measurement data, convert the two's complement value to the decimal value and use the following equation for 16-bit resolution.

T = Decimal Value x 0.005

For example, if the result is 0x1CE8, convert to decimal to get 7400, then T = 7400 x 0.005, or 37°C. Table 1 gives examples of digital output data and the corresponding temperature reading.

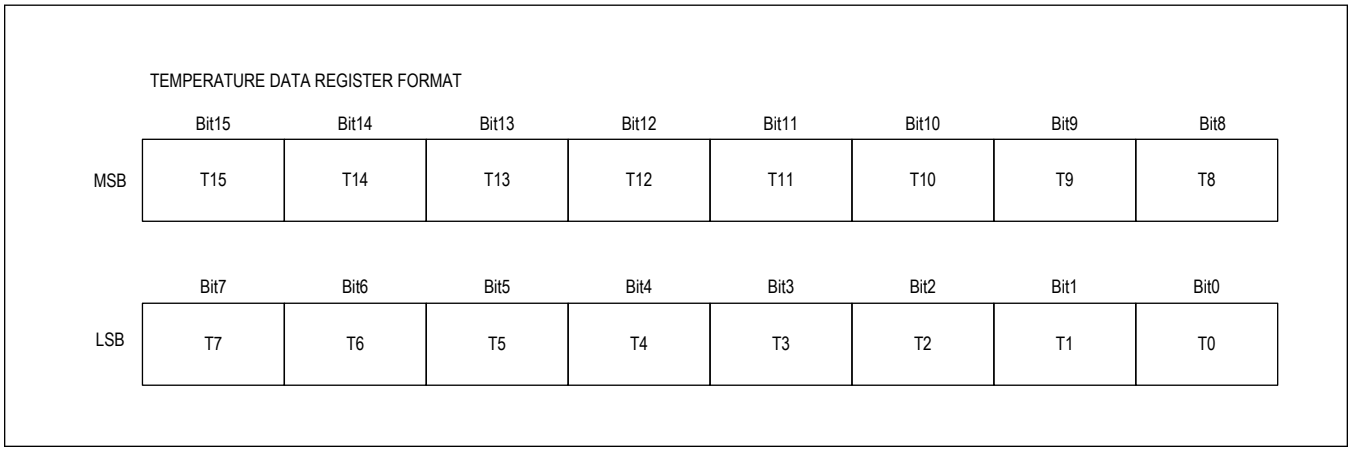



*Figure 1. Temperature Data Register Format*

## Table 1. 16-bit Temperature Data Format

| TEMPERATURE (°C) | DIGITAL OUTPUT (BINARY) | DIGITAL OUTPUT (HEX) | DIGITAL OUTPUT (DEC) |
|---|---|---|---|
| +125 | 0110 0001 1010 1000 | 61A8 | 25000 |
| +100 | 0100 1110 0010 0000 | 4E20 | 20000 |
| +85 | 0100 0010 0110 1000 | 4268 | 17000 |
| +70 | 0011 0110 1011 0000 | 36B0 | 14,000 |
| +50 | 0010 0111 0001 0000 | 2710 | 10,000 |
| +41 | 0010 0000 0000 1000 | 2008 | 8,200 |
| +37 | 0001 1100 1110 1000 | 1CE8 | 7,400 |
| +35.8 | 0001 1011 1111 1000 | 1BF8 | 7,160 |
| +25 | 0001 0011 1000 1000 | 1388 | 5,000 |
| +15 | 0000 1011 1011 1000 | 0BB8 | 3,000 |
| +0.04 | 0000 0000 0000 1000 | 0008 | 8 |
| +0.02 | 0000 0000 0000 0100 | 0004 | 4 |
| +0.01 | 0000 0000 0000 0010 | 0002 | 2 |
| +0.005 | 0000 0000 0000 0001 | 0001 | 1 |
| 0 | 0000 0000 0000 0000 | 0000 | 0 |
| -0.005 | 1111 1111 1111 1111 | FFFF | -1 |
| -0.01 | 1111 1111 1111 1110 | FFFE | -2 |
| -0.02 | 1111 1111 1111 1100 | FFFC | -4 |
| -0.04 | 1111 1111 1111 1000 | FFF8 | -8 |
| -20 | 1111 0000 0110 0000 | F060 | -4000 |
| -40 | 1110 0000 1100 0000 | E0C0 | -8000 |

### Alarm Signaling

After the device performs a temperature conversion, the temperature value is compared with the user-defined two's complement alarm trigger values stored in the 2-byte Alarm High and 2-byte Alarm Low registers (see Figure 2). The default value for AH is 0x7FFF (+163.835°C) and the default value for AL is 0x8000 (-163.840°C). The MSB indicates if the value is positive or negative; for positive numbers the MSB is 0 and for negative numbers the MSB is 1. The Alarm High threshold (AH) is programmed in registers ALARM_HI_MSB [0x10] and ALARM_HI_LSB [0x11]. The alarm low threshold (AL) is programmed in registers ALARM_LO_MSB [0x12] and ALARM_LO_LSB [0x13].

If the measured temperature is lower than AL or higher than AH, an alarm condition exists and the corresponding status bit (TEMP_LO or TEMP_HI) is set in the STATUS [0x00] regsiter. When the alarm condition is detected and the corresponding interrupt enable bit (TEMP_LO_EN or TEMP_HI_EN) is set in the INTERRUPT_ENABLE [0x01] register and if GPIO0_MODE in the GPIO_SETUP [0x20] register is set to 0x3, then a hardware interrupt asserts on the GPIO0 pin. The status bits, the alarm flag and the hardware interrupt stay asserted until the STATUS [0x00] register is read through the serial interface. The alarm flag only clears when STATUS is read. If the alarm flag is set and the next result does not trip the flag, then the flag remains set.

If the resolution or alarm settings change while the device is under an alarm condition, the alarm status must be cleared and another temperature conversion executed to update the alarm condition.

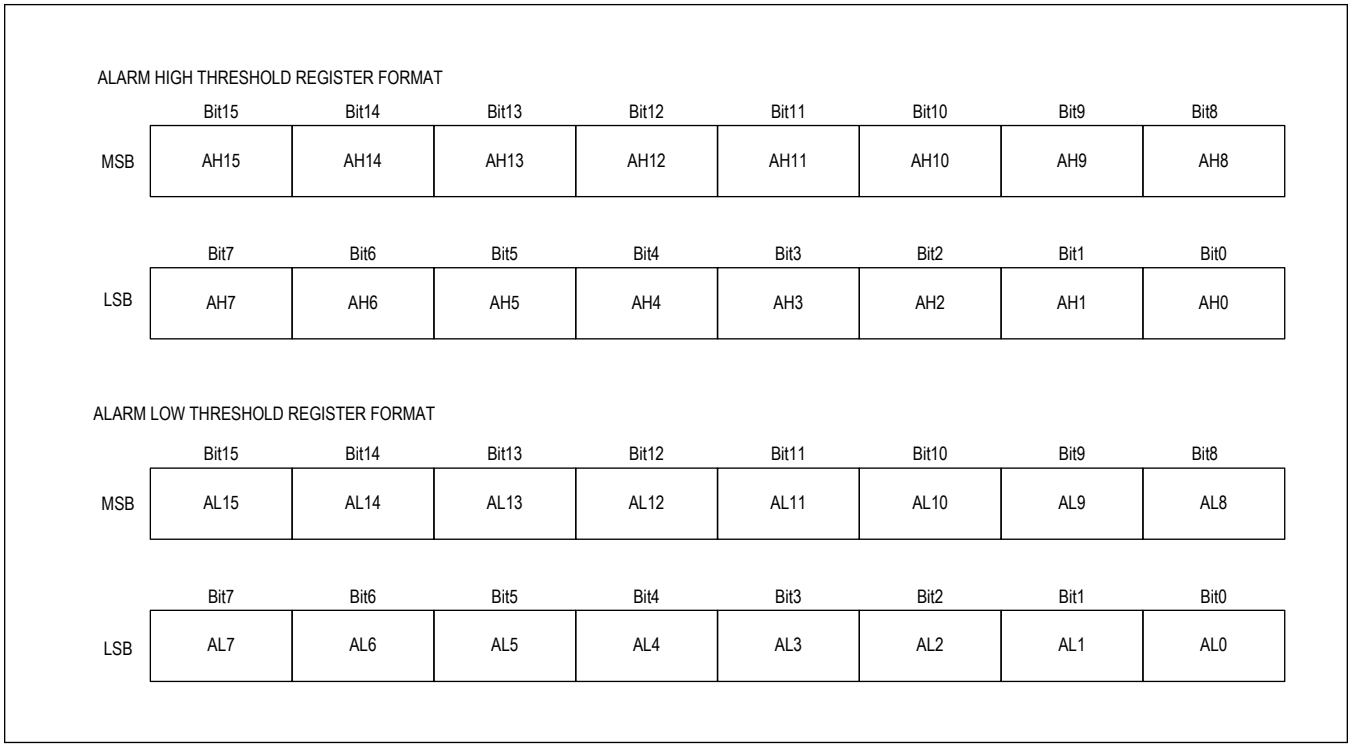



*Figure 2. Alarm Threshold Register Format*

### GPIO

The MAX31889 provides access to two GPIO pins which can be used to provide additional functionality as shown in Table 2. GPIO0 can be configured to output an interrupt while GPIO1 can be configured as an input for a temperature converison. The interrupt on GPIO0 is triggered based on selectable status bits in the INTERRUPT_ENABLE[0X01] register. By writing to one of the available bits in the INTERRUPT_ENABLE register, the flag for an interrupt is raised if GPIO0_MODE[1:0] in the GPIO_SETUP[0x20] register is set to 11. When GPIO1_MODE[7:0] in the GPIO_SETUP register is set to 11, driving the line low initiates an external temperature converison. Table 1 shows a complete list of the functions of the two GPIO Pins.

The state of GPIO pins and the setting of the GPIO_SETUP[0x20] register are used to determine the last two bits of the I2C address at each I2C start condition. This use of the GPIO pins is further detailed in the *I2C Slave Address* section.

## Table 2. GPIO Mode Functions

| GPIOX_MODE[1:0] (X = 0,1) | GPIO0 | GPIO1 |
|---|---|---|
| 00 | HiZ Input | HiZ Input |
| 01 | Output | Output |
| 10 (default) | 1MΩ Internal Pulldown Input | 1MΩ Internal Pulldown Input |
| 11 | INTB | CONV |

# I2C

### I2C Slave Address

The I2C address is determined by a combination of the GPIO Mode and the state of the GPIO pins. I2C Slave Address is shown in Table 3, the LSB of I2C address is 0 for write option and 1 for read option. At power-up, GPIOx_MODE[1:0] bits in the GPIO_SETUP register(0x20) are set to mode 10 as shown in Table 3. GPIO Mode bits 10 configures GPIO0 and GPIO1 pins as input with a 1MΩ internal pulldown. With nothing connected, GPIO0 and GPIO1 pins are considered low, I2C address will be detected as 0xA0 for write option and 0xA1 for read option. If drive GPIO0 and GPIO1 pins to high at power-up, I2C address will be detected as 0xA6 for write option and 0xA7 for read option.

If the state of the GPIO mode bits for a given GPIO input are set to 01 or 11 (see Table 2), then corresponding address state of that bit is 0 for the purposes of determing the I2C address, the actual pin state of GPIO 1 and GPIO 0 are not measured. For example, when set GPIO Mode bits of GPIO0 and GPIO1 to 01 or 11, the I2C address is always 0xA0 for write option and 0xA1 for read option, regardless the voltage level of GPIO0 and GPIO1 pins. If the GPIO Mode bits for either of the GPIO pins are set as input and driven high, change of GPIO Mode bits to 01 or 11 will change I2C address. For example, when GPIO Mode bits of both GPIO0 and GPIO1 are set as 00( HiZ Input) and driven high, the I2C address is 0xA6 for write option and 0xA7 for read option. As soon as we set GPIO Mode bits of GPIO 0 and GPIO1 to 11(GPIO0 as INTB,GPIO1 as CONV), the I2C address is changed to 0xA0 for write option and 0xA1 for read option.

## Table 3. I2C Slave Address

| I2C ADDRESS | | | | | | | W/R | GPIO STATES | | CONDITION |
|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | GPIO1_MODE[1:0] | GPIO0_MODE[1:0] | |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | GPIO0 | 0/1 | 10 | 10 | Default state at powerup |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | GPIO0 | 0/1 | 00 | 00 | Both GPIO1 and GPIO0 are inputs |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | GPIO0 | 0/1 | 10 | 00 | Both GPIO1 and GPIO0 are inputs |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | GPIO0 | 0/1 | 00 | 10 | Both GPIO1 and GPIO0 are inputs |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | 0 | 0/1 | 00 | 01 | GPIO1 is an input,GPIO0 is an output |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | 0 | 0/1 | 00 | 11 | GPIO1 is an input,GPIO0 is an output |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | 0 | 0/1 | 10 | 01 | GPIO1 is an input,GPIO0 is an output |
| 1 | 0 | 1 | 0 | 0 | GPIO1 | 0 | 0/1 | 10 | 11 | GPIO1 is an input,GPIO0 is an output |
| 1 | 0 | 1 | 0 | 0 | 0 | GPIO0 | 0/1 | 01 | 00 | GPIO1 is an output,GPIO0 is an input |
| 1 | 0 | 1 | 0 | 0 | 0 | GPIO0 | 0/1 | 01 | 10 | GPIO1 is an output,GPIO0 is an input |
| 1 | 0 | 1 | 0 | 0 | 0 | GPIO0 | 0/1 | 11 | 00 | GPIO1 is an input,GPIO0 is an input |
| 1 | 0 | 1 | 0 | 0 | 0 | GPIO0 | 0/1 | 11 | 10 | GPIO1 is an input,GPIO0 is an input |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 | 01 | 01 | GPIO1 and GPIO0 are outputs |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 | 01 | 11 | GPIO1 and GPIO0 are outputs |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 | 11 | 01 | GPIO1 is an input,GPIO0 is an output |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 | 11 | 11 | GPIO1 is an input,GPIO0 is an output |

### I2C/SMBus Compatible Serial Interface

The MAX31889 features an I2C/SMBus™ compatible, 2-wire serial interface consisting of a serial data line (SDA) and a serial clock line (SCL). SDA and SCL facilitate communication between the MAX31889 and the master at clock rates up to 400kHz. Figure 3 shows the 2-wire interface timing diagram. The master generates SCL and initiates data transfer on the bus. The master device writes data to the MAX31889 by transmitting the proper slave address followed by the register address and then the data word. Each transmit sequence is framed by a START (S) or REPEATED START (Sr) condition and a STOP (P) condition. Each word transmitted to the MAX31889 is 8 bits long and is followed by an acknowledge clock pulse. A master reading data from the MAX31889 transmits the proper slave address followed by a series of nine SCL pulses. The MAX31889 transmits data on SDA in sync with the master-generated SCL pulses. The master acknowledges receipt of each byte of data. Each read sequence is framed by a START (S) or REPEATED START (Sr) condition, a Not Acknowledge (NACK), and a STOP (P) condition. SDA operates as both an input and an open-drain output. A pullup resistor is required on SDA. SCL operates only as an input. A pullup resistor is required on SCL if there are multiple masters on the bus, or if the single master has an open-drain SCL output. Series resistors in line with SDA and SCL are optional. Series resistors can protect the digital inputs of the MAX31889 from high-voltage spikes on the bus lines, and minimize crosstalk and undershoot of the bus signals.

### Detailed I2C Timing Diagram

The detailed timing diagram of various electrical characteristics is shown in Figure 3.

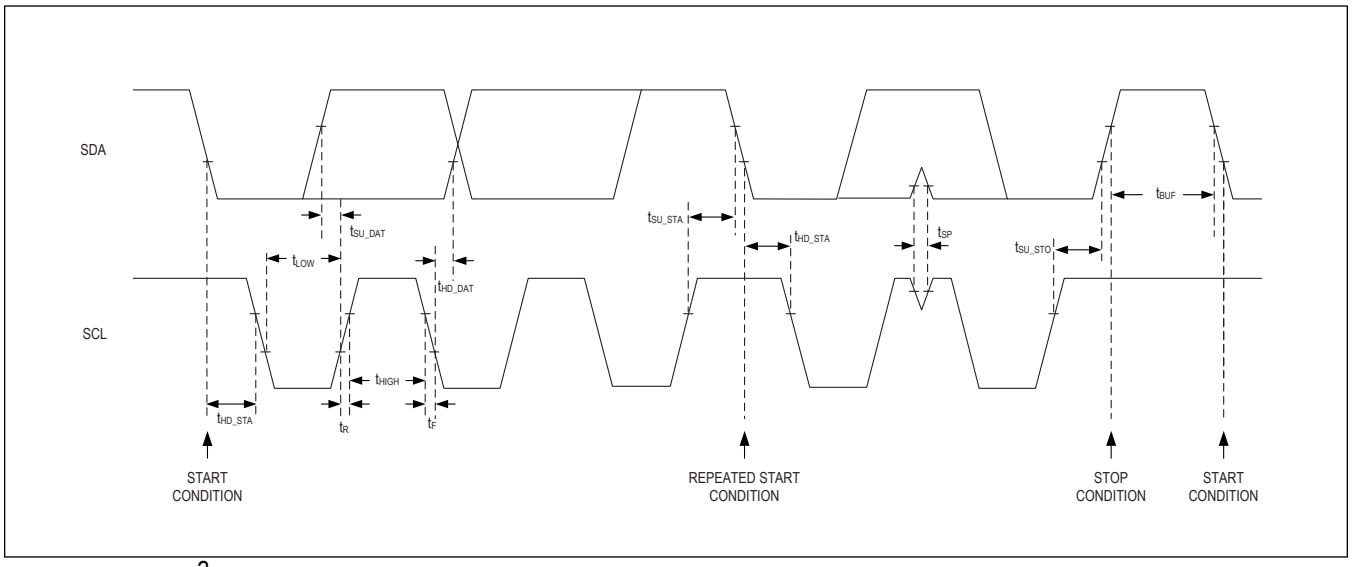



*Figure 3. Detailed I2C Timing Diagram*

### Bit Transfer

One data bit is transferred during each SCL cycle. The data on SDA must remain stable during the high period of the SCL pulse. Changes in SDA while SCL is high are control signals (see the *START and STOP Conditions* section).

### START and STOP Conditions

SDA and SCL idle high when the bus is not in use. A master initiates communication by issuing a START condition. A START condition is a high-to-low transition on SDA with SCL high. A STOP condition is a low-to-high transition on SDA while SCL is high (Figure 4). A START condition from the master signals the beginning of a transmission to the MAX31889. The master terminates transmission, and frees the bus, by issuing a STOP condition. The bus remains active if a REPEATED START condition is generated instead of a STOP condition.

### Early STOP Conditions

The MAX31889 recognizes a STOP condition at any point during data transmission except if the STOP condition occurs in the same high pulse as a START condition. For proper operation, do not send a STOP condition during the same SCL

high pulse as the START condition.

### Acknowledge Bit

The acknowledge bit (ACK) is a clocked 9th bit that the MAX31889 uses to handshake receipt each byte of data when in write mode (Figure 5). The MAX31889 pulls down SDA during the entire master-generated 9th clock pulse if the previous byte is successfully received. Monitoring ACK allows for detection of unsuccessful data transfers. An unsuccessful data transfer occurs if a receiving device is busy or if a system fault has occurred. In the event of an unsuccessful data transfer, the bus master will retry communication. The master pulls down SDA during the 9th clock cycle to acknowledge receipt of data when the MAX31889 is in read mode. An acknowledge is sent by the master after each read byte to allow data transfer to continue. A not-acknowledge is sent when the master reads the final byte of data from the MAX31889, followed by a STOP condition.

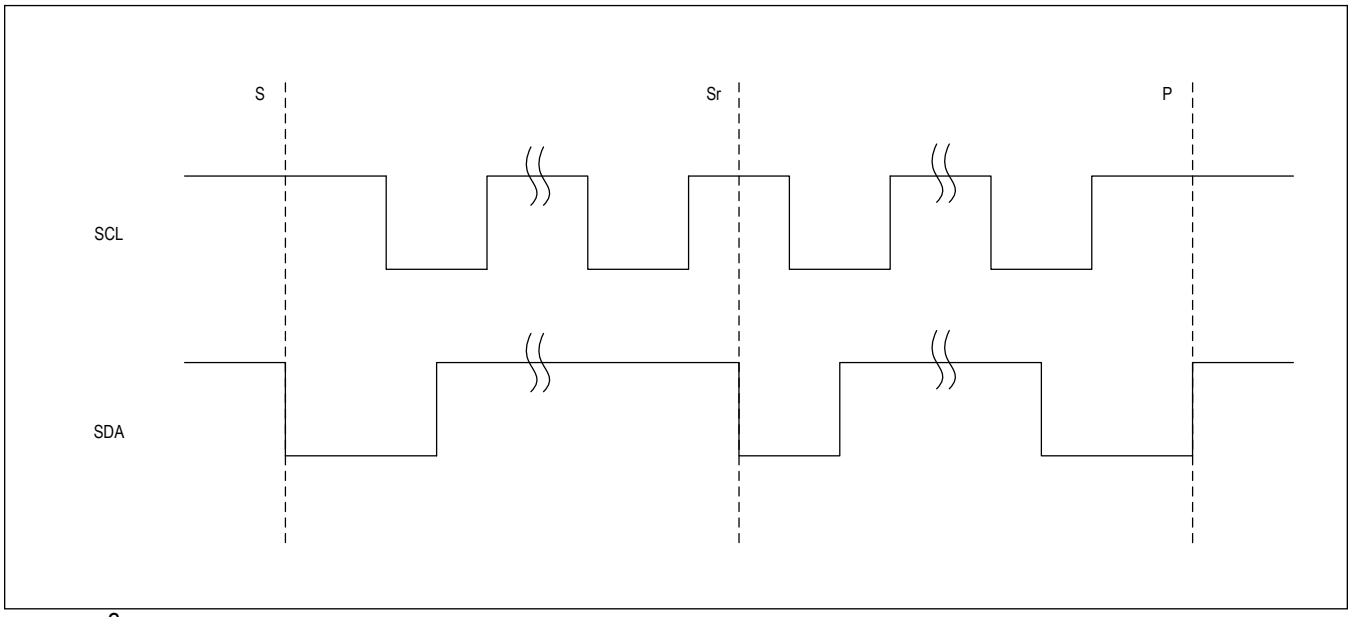



*Figure 4. I2C START, STOP, and REPEATED START Conditions*

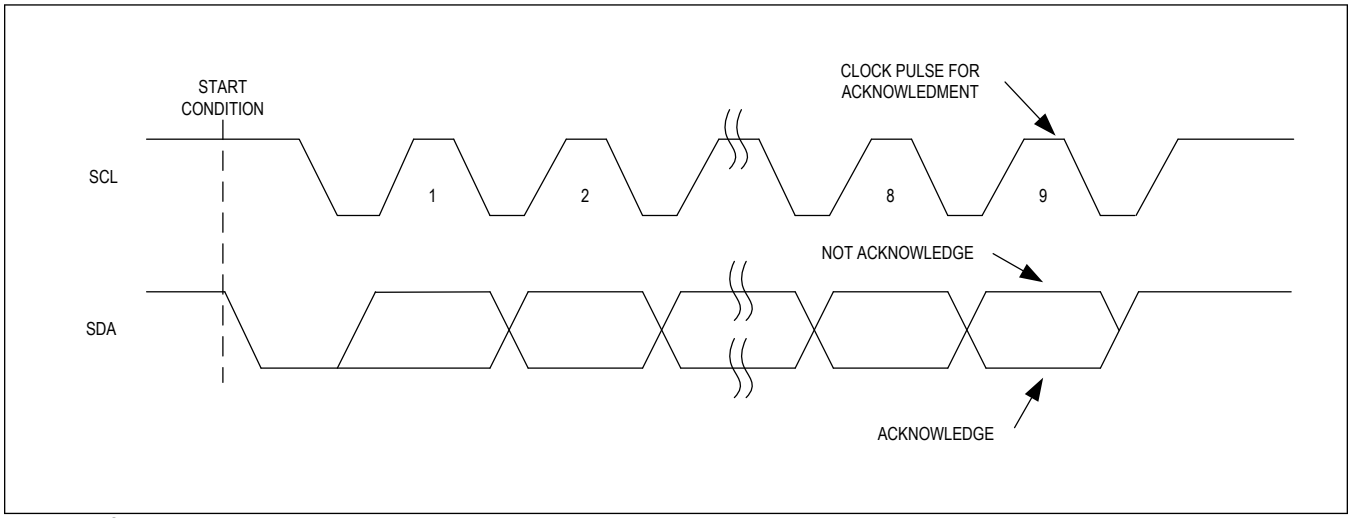



*Figure 5. I2C Acknowledge Bit*

### I2C Write Data Format

A write to the MAX31889 includes transmission of a START condition, the slave address with the R/$\overline{W}$ bit set to 0, one byte of data to configure the internal register address pointer, one or more bytes of data, and a STOP condition. Figure 6 illustrates the proper frame format for writing one byte of data to the MAX31889. Figure 7 illustrates the frame format for writing n-bytes of data to the MAX31889.

The master first sends the slave address with the R/$\overline{W}$ bit set to 0. This indicates that the master intends to write data to the MAX31889. The MAX31889 acknowledges receipt of the address byte during the master-generated 9th SCL pulse.

The second byte transmitted from the master configures the MAX31889's internal register address pointer. The pointer tells the MAX31889 where to write the next byte of data. An acknowledge pulse is sent by the MAX31889 upon receipt of the address pointer data.

The third byte sent to the MAX31889 contains the data that will be written to the chosen register. An acknowledge pulse from the MAX31889 signals receipt of the data byte. The address pointer auto increments to the next register address after each received data byte. This auto-increment feature allows a master to write to sequential registers within one continuous frame. The master signals the end of transmission by issuing a STOP condition. The auto_increment feature is disabled when there is an attempt to write to the FIFO_DATA (0x08) register.

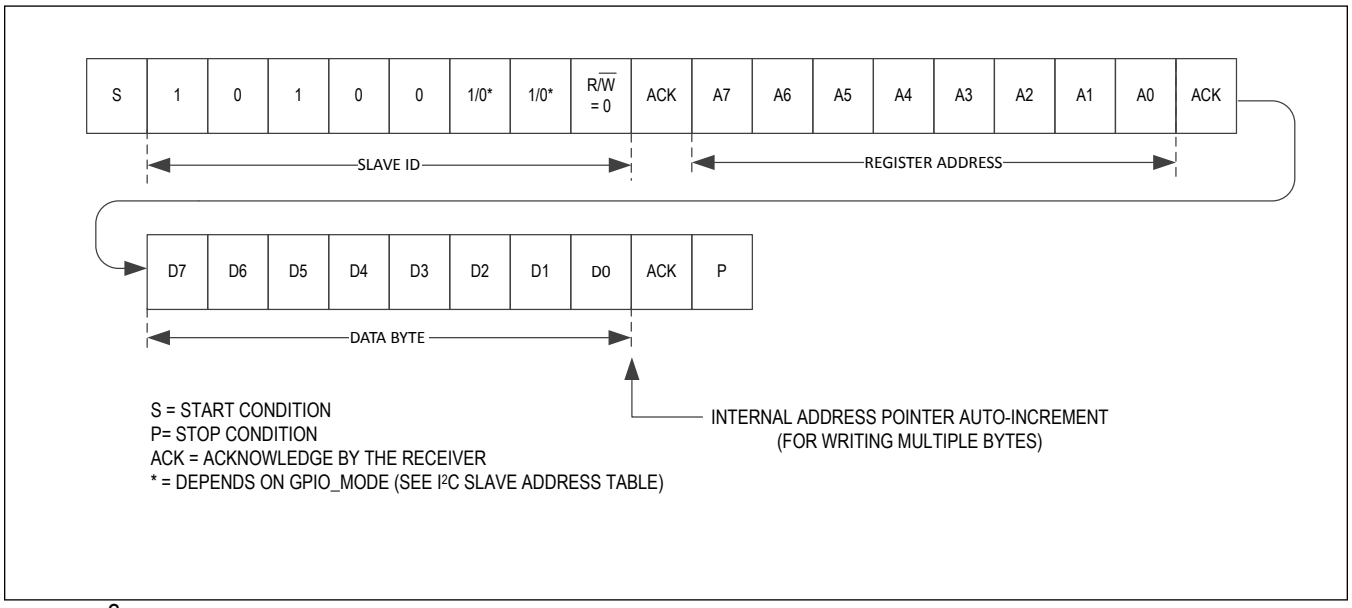



*Figure 6. I2C Single Byte Write Transaction*

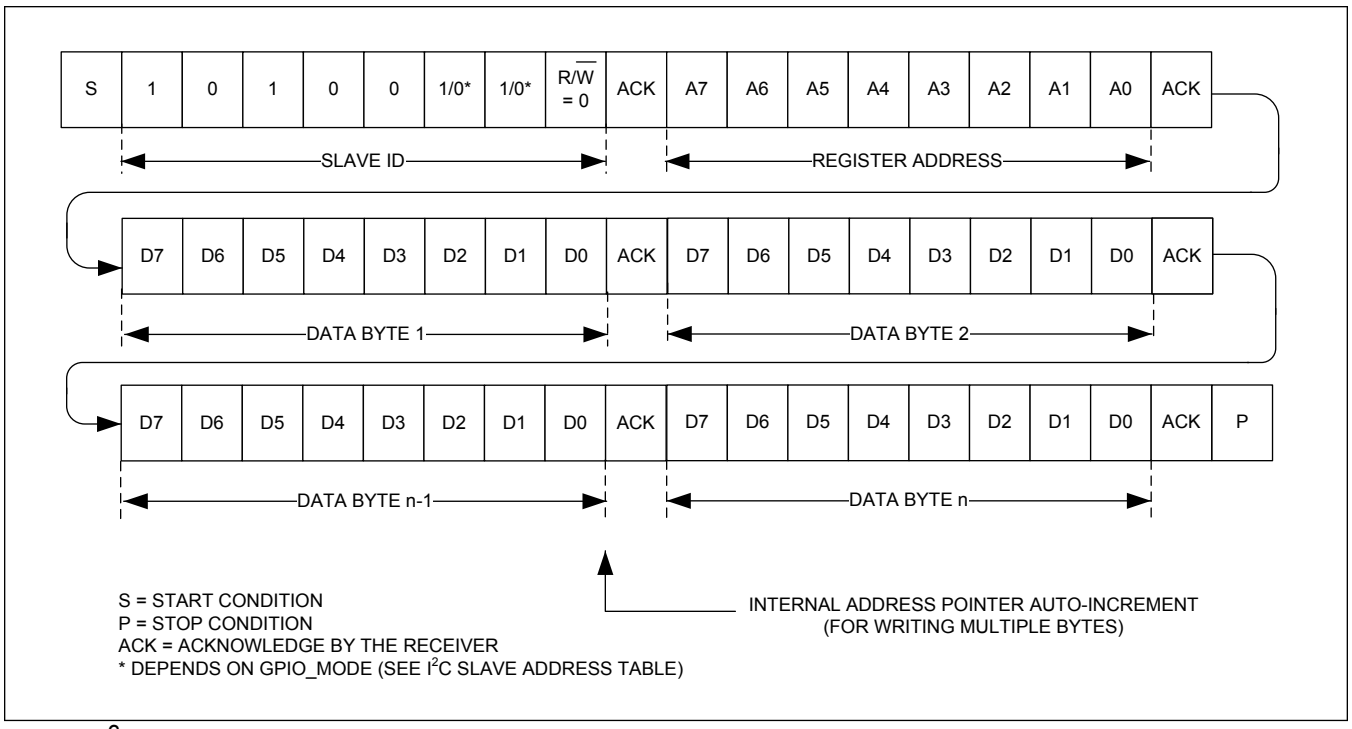



*Figure 7. I2C Multi-Byte Write Transaction*

### I2C Read Data Format

The master sends the slave address with the R/$\overline{W}$ bit set to 1 to initiate a read operation. The MAX31889 acknowledges receipt of its slave address by pulling SDA low during the 9th SCL clock pulse. A START command followed by a read command resets the address pointer to register 0x00.

The first byte transmitted from the MAX31889 will be the contents of register 0x00. Transmitted data is valid on the rising edge of SCL. The address pointer auto-increments after each read data byte. This auto-increment feature allows all registers to be read sequentially within one continuous frame. The auto_increment feature is disabled when there is an attempt to read from the FIFO_DATA register, this allows for burst reading of the FIFO_DATA register. A STOP condition can be issued after any number of read data bytes. If a STOP condition is issued followed by another read operation, the first data byte to be read will be from register 0x00.

The address pointer can be preset to a specific register before a read command is issued. The master presets the address pointer by first sending the MAX31889 slave address with the R/$\overline{W}$ bit set to 0 followed by the register address. A REPEATED START condition is then sent followed by the slave address with the R/$\overline{W}$ bit set to 1. The MAX31889 then transmits the content of the specified register. The address pointer auto-increments after transmitting the first byte.

The master acknowledges receipt of each read byte during the acknowledge clock pulse. The master must acknowledge all correctly received bytes except the last byte. The final byte must be followed by a not acknowledge from the master and then a STOP condition. Figure 8 illustrates the frame format for reading one byte from the MAX31889. Figure 9 illustrates the frame format for reading multiple bytes from the MAX31889.

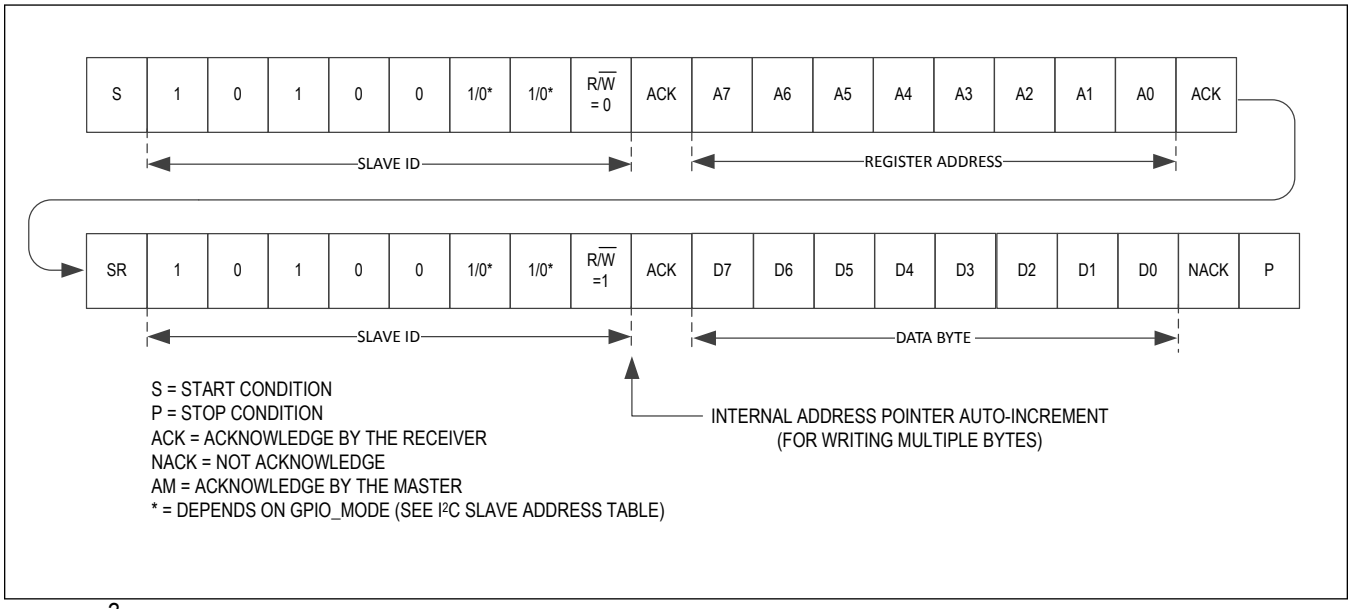



*Figure 8. I2C Single Byte Read Transaction*

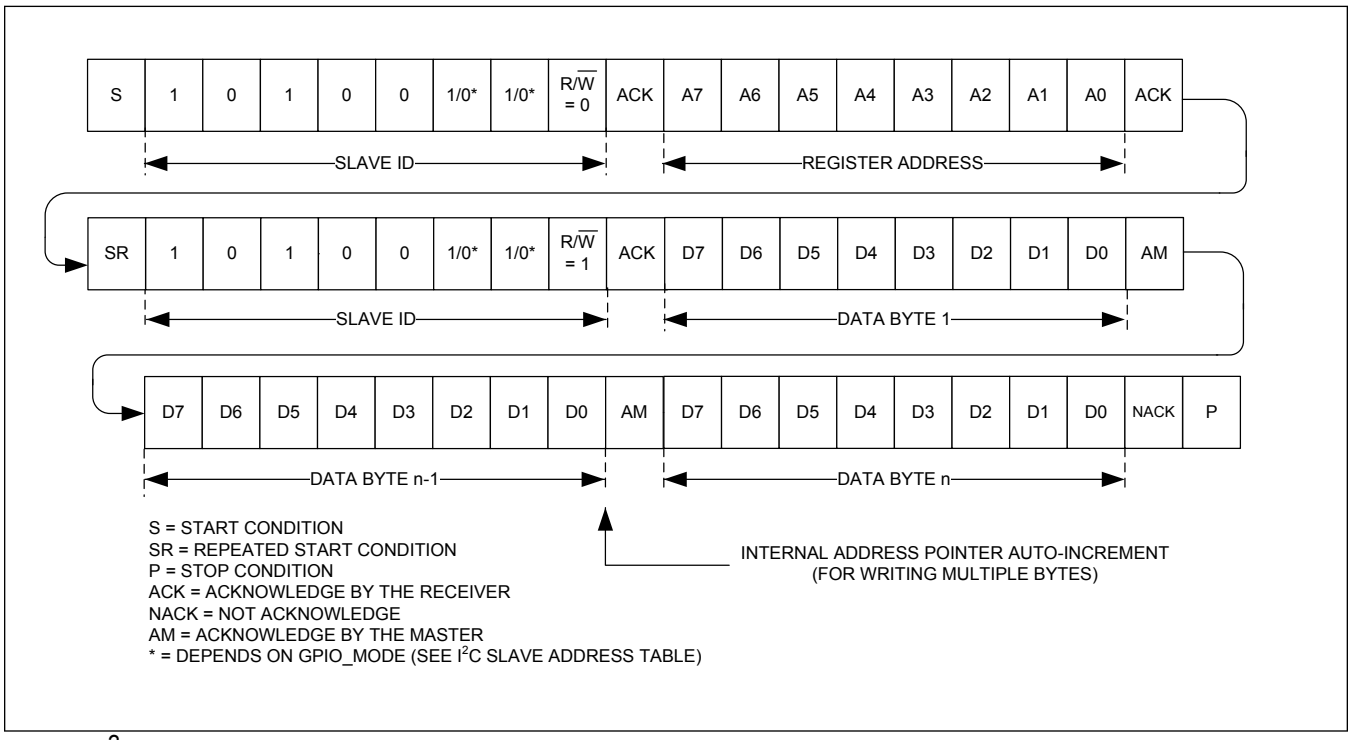



*Figure 9. I2C Multi-Byte Read Transaction*

## FIFO Description

The FIFO is 32 samples long and is designed for 16-bit wide temperature data. The master does a burst read of two

bytes starting at register 0x08 to read one 16-bit temperature sample, referred to as a word, from the FIFO. The master reads 2N bytes from the FIFO to get N samples.

There are seven registers that control how the FIFO is configured and read out. These registers are described in Table 4.

## Table 4. FIFO Register Map

| ADDRESS | REGISTER NAME | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0 |
|---|---|---|---|---|---|---|---|---|---|
| 0x04 | FIFO Write Pointer | - | - | - | FIFO_WR_PTR[4:0] | | | | |
| 0x05 | FIFO Read Pointer | - | - | - | FIFO_RD_PTR[4:0] | | | | |
| 0x06 | FIFO Overflow Counter | - | - | - | OVF_COUNTER[4:0] | | | | |
| 0x07 | FIFO Data Counter | - | - | FIFO_DATA_COUNT[5:0] | | | | | |
| 0x08 | FIFO Data Register | FIFO_DATA[7:0] | | | | | | | |
| 0x09 | FIFO Configuration 1 | - | - | - | FIFO_A_FULL[4:0] | | | | |
| 0x0A | FIFO Configuration 2 | - | - | - | FLUSH_FIFO | FIFO_STAT_CLR | A_FULL_TYPE | FIFO_RO | - |

**FIFO_WR_PTR (address 0x04), Write Pointer**

FIFO_WR_PTR[4:0] points to the FIFO location where the next word is to be written. This pointer advances for each word pushed on to the FIFO by the internal conversion process. The write pointer is updated from 5 bit counter and wraps around to count 0x00 from count 0x1F.

**FIFO_RD_PTR (address 0x05), Read Pointer**

FIFO_RD_PTR[4:0] points to the location from where the next word from the FIFO is to be read through the serial interface. This advances each time a word is read from the FIFO. The read pointer can be both read and written to. This allows a word to be reread from the FIFO if it has not already been overwritten. The read pointer is updated from a 5 bit counter and wraps around to count 0x00 from count 0x1F.

**OVF_COUNTER (address 0x06), Overflow Counter**

OVF_COUNTER[4:0] logs the number of words lost if new words are written after the FIFO is full. This counter saturates at count value 0x1F. Each time a complete word is popped from the FIFO (when the read pointer advances), the OVF_COUNTER is reset to zero. This counter is essentially a debug tool. It should be read immediately before reading the FIFO in order to check if an overflow condition has occurred.

**FIFO_DATA_COUNT (address 0x07), FIFO Data Counter**

FIFO_DATA_COUNT[5:0] is a read-only register which holds the number of words available in the FIFO for the master to read. This increments when a new word is pushed to the FIFO, and decrements when the master reads a word from the FIFO.

**FIFO_DATA (address 0x08), FIFO Data**

FIFO_DATA[7:0] is a read-only register used to retrieve data from the FIFO. It is important to burst read the data from the FIFO. Each word is two bytes. So burst reading two bytes at FIFO_DATA register through the serial interface advances the FIFO_RD_PTR by one. This configuration is best illustrated by the examples below.

Table 5 shows the Temperature Data format in the FIFO.

## Table 5. Temperature FIFO Data Format

| FIFO DATA FORMAT (FIFO_DATA[15:0]) | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit15 | Bit14 | Bit13 | Bit12 | Bit11 | Bit10 | Bit9 | Bit8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |

**FIFO_DATA Read Example**

Number of samples available in the FIFO after the last read can be obtained by reading the OVF_COUNTER[4:0] and FIFO_DATA_COUNT[5:0] registers using the following pseudo-code:

read the OVF_COUNTER register

read the FIFO_DATA_COUNT register

if OVF_COUNTER == 0 //no overflow occurred

NUM_AVAILABLE_SAMPLES = FIFO_DATA_COUNT

else

NUM_AVAILABLE_SAMPLES = 32 // overflow occurred and data has been lost

FIFO_WR_PTR[4:0] and FIFO_RD_PTR[4:0] are available for debug. They may also be used to calculate the number of available samples using the following pseudo-code:

If OVF_COUNTER is zero,

NUM_AVAILABLE_WORDS = FIFO_WR_PTR – FIFO_RD_PTR

(Note: pointer wrap around should be taken into account)

else

NUM_AVAILABLE_WORDS = 32

Table 6 shows the order in which the two bytes of the Temperature Data are read through the serial interface.

## Table 6. FIFO Data Read Format

| SAMPLE NUMBER | BYTE NUMBER | FIFO DATA READ FORMAT | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Sample N | 1 | T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 |
| | 2 | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |
| Sample N+1 | 1 | T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 |
| | 2 | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |
| Sample N+2 | 1 | T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 |
| | 2 | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |
| … | | | | | | | | | |
| Sample N+31 | 1 | T15 | T14 | T13 | T12 | T11 | T10 | T9 | T8 |
| | 2 | T7 | T6 | T5 | T4 | T3 | T2 | T1 | T0 |

**FIFO_A_FULL (address 0x09), FIFO Almost Full**

The FIFO_A_FULL[4:0] field in the FIFO Configuration 1 (0x09) register sets the application custom threshold for the FIFO and determines when the A_FULL bit in the STATUS (0x00) register is asserted. The A_FULL bit is set when the FIFO contains 32 minus FIFO_A_FULL[4:0] words. For example, when FIFO_A_FULL is set to 2, the flag is set when the 30$^{th}$ word is written to the FIFO. When the FIFO almost full condition is met, the A_FULL bit is asserted in the STATUS register. If the A_FULL_EN bit in the INTERRUPT_ENABLE (0x01) register is set and GPIO0_MODE = 0x3 in the GPIO_SETUP (0x20) register, then the interrupt is asserted on the GPIO0 pin. This condition should prompt the applications processor to read samples from the FIFO before it fills.

Before reading the data out of the FIFO, the application processor has to determine the number of available temperature samples. This can be accomplished by reading the OVF_COUNTER and FIFO_DATA_COUNT registers.

**FIFO_RO (address 0x0A), FIFO Rollover**

The FIFO_RO bit in the FIFO Configuration 2 (0x0A) register determines whether samples get pushed on to the FIFO or discarded when it is full. If FIFO_RO is enabled when FIFO is full, old samples are overwritten. If FIFO_RO is not set, the new sample is discarded and the FIFO is not updated.

**A_FULL_TYPE (address 0x0A), Almost Full Type**

The A_FLL_TYPE bit defines the behavior of the A_FULL interrupt. If the A_FIFO_TYPE bit is set low, the A_FULL interrupt gets asserted when the A_FULL condition is detected and cleared by a STATUS register read, then reasserts for every sample if the A_FULL condition persists. If the A_FIFO_TYPE bit is set high, the A_FULL status bit is asserted only when a new A_FULL condition is detected. The status bit is cleared by a STATUS register read and does not reassert

for every sample until a new A_FULL condition is detected.

**FIFO_STAT_CLR (address 0x0A), FIFO Status Clear**

The FIFO_STAT_CLR bit defines whether the A_FULL status bit should clear by a FIFO_DATA register read. If FIFO_STAT_CLR is set low, A_FULL and TEMP_RDY status bits are not get cleared by FIFO_DATA register read but are cleared by STATUS register read. If FIFO_STAT_CLR is set high, A_FULL and TEMP_RDY status bits are cleared by a FIFO_DATA register read or a STATUS register read.

**FLUSH_FIFO (address 0x0A)**

The FLUSH_FIFO bit is used for flushing the FIFO. The FIFO becomes empty and the FIFO_WR_PTR[4:0], FIFO_RD_PTR[4:0], FIFO_DATA_COUNT[5:0] and OVF_COUNTER[4:0] are reset to zero. FLUSH_FIFO is a self-clearing bit.

## Register Map

### Register Map

| ADDRESS | NAME | MSB | | | | | | | LSB |
|---|---|---|---|---|---|---|---|---|---|
| **INTERRUPT AND STATUS** | | | | | | | | | |
| 0x00 | STATUS[7:0] | A_FULL | – | – | – | – | TEMP_LO | TEMP_HI | TEMP_RDY |
| 0x01 | INTERRUPT ENABLE[7:0] | A_FULL_EN | – | – | – | – | TEMP_LO_EN | TEMP_HI_EN | TEMP_RDY_EN |
| **FIFO** | | | | | | | | | |
| 0x04 | FIFO WRITE POINTER[7:0] | – | – | – | FIFO_WR_PTR[4:0] | | | | |
| 0x05 | FIFO READ POINTER[7:0] | – | – | – | FIFO_RD_PTR[4:0] | | | | |
| 0x06 | FIFO OVERFLOW COUNTER[7:0] | – | – | – | OVF_COUNTER[4:0] | | | | |
| 0x07 | FIFO DATA COUNTER[7:0] | – | – | FIFO_DATA_COUNT[5:0] | | | | | |
| 0x08 | FIFO DATA[7:0] | FIFO_DATA[7:0] | | | | | | | |
| 0x09 | FIFO CONFIGURATION 1[7:0] | – | – | – | FIFO_A_FULL[4:0] | | | | |
| 0x0A | FIFO CONFIGURATION 2[7:0] | – | – | – | FLUSH_FIFO | FIFO_STAT_CLR | A_FULL_TYPE | FIFO_RO | – |
| **SYSTEM** | | | | | | | | | |
| 0x0C | SYSTEM CONTROL[7:0] | – | – | – | – | – | – | – | RESET |
| **TEMPERATURE** | | | | | | | | | |
| 0x10 | ALARM HIGH MSB[7:0] | ALARM_HI_MSB[7:0] | | | | | | | |
| 0x11 | ALARM HIGH LSB[7:0] | ALARM_HI_LSB[7:0] | | | | | | | |
| 0x12 | ALARM LOW MSB[7:0] | ALARM_LO_MSB[7:0] | | | | | | | |
| 0x13 | ALARM LOW LSB[7:0] | ALARM_LO_LSB[7:0] | | | | | | | |
| 0x14 | TEMP SENSOR SETUP[7:0] | RFU[1:0] | | – | – | – | – | – | CONVERT_T |
| **GPIO** | | | | | | | | | |
| 0x20 | GPIO SETUP[7:0] | GPIO1_MODE[1:0] | | – | – | – | – | GPIO0_MODE[1:0] | |
| 0x21 | GPIO CONTROL[7:0] | – | – | – | – | GPIO1_LL | – | – | GPIO0_LL |
| **IDENTIFIERS** | | | | | | | | | |
| 0x31 | ROM ID 1[7:0] | ROM_ID1[7:0] | | | | | | | |
| 0x32 | ROM ID 2[7:0] | ROM_ID2[7:0] | | | | | | | |
| 0x33 | ROM ID 3[7:0] | ROM_ID3[7:0] | | | | | | | |
| 0x34 | ROM ID 4[7:0] | ROM_ID4[7:0] | | | | | | | |
| 0x35 | ROM ID 5[7:0] | ROM_ID5[7:0] | | | | | | | |
| 0x36 | ROM ID 6[7:0] | ROM_ID6[7:0] | | | | | | | |

| ADDRESS | NAME | MSB | | | | | | | LSB |
|---|---|---|---|---|---|---|---|---|---|
| 0xFF | PART IDENTIFIER[7:0] | PART_ID[7:0] | | | | | | | |

## Register Details

### STATUS (0x0)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | A_FULL | – | – | – | – | TEMP_LO | TEMP_HI | TEMP_RDY |
| Reset | 0b0 | – | – | – | – | 0b0 | 0b0 | 0b0 |
| Access Type | Read Only | – | – | – | – | Read Only | Read Only | Read Only |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| A_FULL | 7 | This is a read-only bit. This bit is cleared when the STATUS register is read. It is also cleared when FIFO_DATA register is read, if FIFO_STAT_CLR = 1. |
| TEMP_LO | 2 | Temperature Sensor Alarm Low status bit:<br><br>This bit is asserted when the latest temperature sensor measurement is less than what is programmed in the Temperature Sensor Alarm Low register. When this bit is asserted and if the TEMP_LO_EN bit is set to 1, it then asserts the interrupt on the GPIO0 pin when programmed as an interrupt output. The master needs to read the status register to determine if the interrupt was asserted by the TEMP_LO status. This bit is cleared after the STATUS register is read. |
| TEMP_HI | 1 | Temperature Sensor Alarm High status bit:<br><br>This bit is asserted when the latest temperature sensor measurement is greater than what is programmed in the Temperature Sensor Alarm High register. When this bit is asserted and if the TEMP_HI_EN bit is set to 1, it then asserts the interrupt on the GPIO0 pin when programmed as interrupt output. The master needs to read the status register to determine if the interrupt was asserted by the TEMP_HI status. This bit is cleared after the STATUS register is read. |
| TEMP_RDY | 0 | Temperature Sensor Conversion Complete status bit:<br><br>This bit is asserted when a temperature sensor measurement has completed and new data is available to be read by the master. When this bit is asserted and if TEMP_RDY_EN bit is set to 1, it then asserts the interrupt on the GPIO0 pin when programmed as an interrupt output. The master needs to read the status register to determine if the interrupt was asserted by the TEMP_RDY status. This bit is cleared after the STATUS register is read or after the Temperature Data registers are read. |

### INTERRUPT ENABLE (0x1)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | A_FULL_EN | – | – | – | – | TEMP_LO_EN | TEMP_HI_EN | TEMP_RDY_EN |
| Reset | 0b0 | – | – | – | – | 0b0 | 0b0 | 0b0 |
| Access Type | Write, Read | – | – | – | – | Write, Read | Write, Read | Write, Read |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| A_FULL_EN | 7 | Set A_FULL_EN to 1 to enable the A_FULL interrupt on GPIO0 when programmed as an interrupt output. Set A_FULL_EN to 0 to disable the A_FULL interrupt. |
| TEMP_LO_EN | 2 | Temperature Sensor Alarm Low Interrupt Enable :<br><br>Set TEMP_LO_EN to 1 to enable the TEMP_LO interrupt on the GPIO0 pin when programmed as an interrupt output. Set TEMP_LO_EN to 0 to disable the TEMP_LO interrupt. |
| TEMP_HI_EN | 1 | Temperature Sensor Alarm High Interrupt Enable:<br><br>Set TEMP_HI_EN to 1 to enable the TEMP_HI interrupt on the GPIO0 pin when programmed as an interrupt output. Set TEMP_HI_EN to 0 to disable the TEMP_HI interrupt. |
| TEMP_RDY_EN | 0 | Temperature Sensor Conversion Complete Interrupt Enable bit:<br><br>Set TEMP_RDY_EN to 1 to enable the TEMP_RDY interrupt on the GPIO0 pin when programmed as an interrupt output. Set TEMP_RDY_EN to 0 to disable the TEMP_RDY interrupt. |

## FIFO WRITE POINTER (0x04)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | – | – | – | FIFO_WR_PTR[4:0] | | | | |
| **Reset** | – | – | – | 0x00 | | | | |
| **Access Type** | – | – | – | Read Only | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| FIFO_WR_PTR | 4:0 | See the FIFO Description section for details. |

## FIFO READ POINTER (0x05)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | – | – | – | FIFO_RD_PTR[4:0] | | | | |
| **Reset** | – | – | – | 0x00 | | | | |
| **Access Type** | – | – | – | Write, Read, Ext | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| FIFO_RD_PTR | 4:0 | See the FIFO Description section for details. |

## FIFO OVERFLOW COUNTER (0x06)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | – | – | – | OVF_COUNTER[4:0] | | | | |
| **Reset** | – | – | – | 0x00 | | | | |
| **Access Type** | – | – | – | Read Only | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| OVF_COUNTER | 4:0 | See the FIFO Description section for details. |

## FIFO DATA COUNTER (0x07)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | – | – | FIFO_DATA_COUNT[5:0] | | | | | |
| **Reset** | – | – | 0x00 | | | | | |
| **Access Type** | – | – | Read Only | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| FIFO_DATA_COUNT | 5:0 | See the FIFO Description section for details. |

## FIFO DATA (0x08)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | FIFO_DATA[7:0] | | | | | | | |
| **Reset** | 0x00 | | | | | | | |
| **Access Type** | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| FIFO_DATA | 7:0 | See the FIFO Description section for details. |

## FIFO CONFIGURATION 1 (0x09)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | – | – | – | FIFO_A_FULL[4:0] | | | | |
| **Reset** | – | – | – | 0x0F | | | | |
| **Access Type** | – | – | – | Write, Read | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| FIFO_A_FULL | 4:0 | See the FIFO Description section for details. |

## FIFO CONFIGURATION 2 (0x0A)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Field** | – | – | – | FLUSH_FIFO | FIFO_STAT_CLR | A_FULL_TYPE | FIFO_RO | – |
| **Reset** | – | – | – | 0b0 | 0b0 | 0b0 | 0b0 | – |
| **Access Type** | – | – | – | Write, Read | Write, Read | Write, Read | Write, Read | – |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| FLUSH_FIFO | 4 | See the FIFO Description section for details. |
| FIFO_STAT_CLR | 3 | See the FIFO Description section for details. |
| A_FULL_TYPE | 2 | See the FIFO Description section for details. |
| FIFO_RO | 1 | See the FIFO Description section for details. |

## SYSTEM CONTROL (0x0C)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | – | – | – | – | – | – | – | RESET |
| Reset | – | – | – | – | – | – | – | 0b0 |
| Access Type | – | – | – | – | – | – | – | Write Only |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| RESET | 0 | Setting this bit to 1 resets all register settings to default values. This is a self-clearing bit. |

## ALARM HIGH MSB (0x10)

This is the most significant byte of the alarm high threshold.

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ALARM_HI_MSB[7:0] | | | | | | | |
| Reset | 0x7F | | | | | | | |
| Access Type | Write, Read | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ALARM_HI_MSB | 7:0 | Alarm High Threshold Most Significant Byte:<br><br>The ALARM_HI_MSB[7:0] bits are the most significant byte of the 16-bit temperature sensor alarm high bits. The ALARM_HI_MSB[7:0] and the ALARM_HI_LSB[7:0] bits form the full 16-bit temperature sensor alarm high threshold. The default for Alarm High Threshold is 0x7FFF, which is the highest temperature setting and also disables the alarm. |

## ALARM HIGH LSB (0x11)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ALARM_HI_LSB[7:0] | | | | | | | |
| Reset | 0xFF | | | | | | | |
| Access Type | Write, Read | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ALARM_HI_LSB | 7:0 | Alarm High Threshold Least Significant Byte:<br><br>The ALARM_HI_LSB[7:0] bits are the least significant byte of the 16-bit temperature sensor alarm high bits. The ALARM_HI_MSB[7:0] and the ALARM_HI_LSB[7:0] bits form the full 16-bit temperature sensor alarm high threshold. The default for Alarm High Threshold is 0x7FFF, which is the highest temperature setting and also disables the alarm. |

## ALARM LOW MSB (0x12)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ALARM_LO_MSB[7:0] | | | | | | | |
| Reset | 0x80 | | | | | | | |
| Access Type | Write, Read | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ALARM_LO_MSB | 7:0 | Alarm High Threshold Most Significant Byte:<br><br>The ALARM_LO_MSB[7:0] bits are the most significant byte of the 16-bit temperature sensor alarm low bits. The ALARM_LO_MSB[7:0] and the ALARM_LO_LSB[7:0] bits form the full 16-bit temperature sensor Alarm Low Threshold. The default for Alarm Low Threshold is 0x8000, which is the lowest temperature setting and also disables the alarm. |

## ALARM LOW LSB (0x13)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ALARM_LO_LSB[7:0] | | | | | | | |
| Reset | 0x00 | | | | | | | |
| Access Type | Write, Read | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ALARM_LO_LSB | 7:0 | Alarm Low Threshold Least Significant Byte:<br><br>The ALARM_LO_LSB[7:0] bits are the least significant byte of the 16-bit temperature sensor alarm high bits. The ALARM_LO_MSB[7:0] and the ALARM_LO_LSB[7:0] bits form the full 16-bit temperature sensor Alarm High Threshold. The default for Alarm Low Threshold is 0x8000, which is the lowest temperature setting and also disables the alarm. |

## TEMP SENSOR SETUP (0x14)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | RFU[1:0] | | – | – | – | – | – | CONVERT_T |
| Reset | 0b11 | | – | – | – | – | – | 0b0 |
| Access Type | Read Only | | – | – | – | – | – | Write, Read |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| RFU | 7:6 | These bits are reserved for future use. When writing to this register, these bits must always be set to 1. |
| CONVERT_T | 0 | Start Temperature measurement<br><br>Writing '1' to this field starts temperature measurement. This is a self clearing bit, and automatically reset to 0 when the temperature measurement completes. |

## GPIO SETUP (0x20)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | GPIO1_MODE[1:0] | | – | – | – | – | GPIO0_MODE[1:0] | |
| Reset | 0b10 | | – | – | – | – | 0b10 | |
| Access Type | Write, Read | | – | – | – | – | Write, Read | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| GPIO1_MODE | 7:6 | 00 = Digital input (HiZ). GPIO1 logic level read from the GPIO1_LL bit in the GPIO_CONTROL register.<br>01 = Digital output (open-drain). Set GPIO1 logic level by writing to the GPIO1_LL bit in the GPIO_CONTROL register.<br>10 = Digital Input with 1MΩ pulldown.<br>11 = Convert Temperature Input (active low) |
| GPIO0_MODE | 1:0 | 00 = Digital input (HiZ). GPIO0 logic level read from the GPIO0_LL bit in the GPIO_CONTROL register.<br>01 = Digital output (open-drain). Set GPIO0 logic level by writing to the GPIO0_LL bit in the GPIO_CONTROL register.<br>10 = Digital Input with 1MΩ pulldown.<br>11 = INTB (open-drain, active low) |

## GPIO CONTROL (0x21)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | – | – | – | – | GPIO1_LL | – | – | GPIO0_LL |
| Reset | – | – | – | – | 0b0 | – | – | 0b0 |
| Access Type | – | – | – | – | Write, Read | – | – | Write, Read |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| GPIO1_LL | 3 | If GPIO1 is programmed as a digital output then set the GPIO1_LL bit to 0 to make the GPIO1 pin a logic low level or set the GPIO1_LL bit to 1 to make the GPIO1 pin a logic high level. A read of the GPIO1_LL bits returns the logic level on the GPIO1 pin when the register is read, regardless of the GPIO1 mode. |
| GPIO0_LL | 0 | If GPIO0 is programmed as a digital output then set the GPIO0_LL bit to 0 to make the GPIO0 pin a logic low level or set the GPIO0_LL bit to 1 to make the GPIO0 pin a logic high level. A read of the GPIO0_LL bits returns the logic level on the GPIO0 pin when the register is read, regardless of the GPIO0 mode. |

## ROM ID 1 (0x31)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ROM_ID1[7:0] | | | | | | | |
| Reset | | | | | | | | |
| Access Type | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ROM_ID1 | 7:0 | Factory set to unique ID |

## ROM ID 2 (0x32)

Unique ROM_ID2

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ROM_ID2[7:0] | | | | | | | |
| Reset | | | | | | | | |
| Access Type | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ROM_ID2 | 7:0 | Factory set to unique ID |

## ROM ID 3 (0x33)

Unique ROM_ID3

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ROM_ID3[7:0] | | | | | | | |
| Reset | | | | | | | | |
| Access Type | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ROM_ID3 | 7:0 | Factory set to unique ID |

## ROM ID 4 (0x34)

Unique ROM_ID4

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ROM_ID4[7:0] | | | | | | | |
| Reset | | | | | | | | |
| Access Type | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ROM_ID4 | 7:0 | Factory set to unique ID |

## ROM ID 5 (0x35)

Unique ROM_ID5

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ROM_ID5[7:0] | | | | | | | |
| Reset | | | | | | | | |
| Access Type | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ROM_ID5 | 7:0 | Factory set to unique ID |

## ROM ID 6 (0x36)

Unique ROM_ID6

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | ROM_ID6[7:0] | | | | | | | |
| Reset | | | | | | | | |
| Access Type | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| ROM_ID6 | 7:0 | Factory set to unique ID |

## PART IDENTIFIER (0xFF)

| BIT | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Field | PART_ID[7:0] | | | | | | | |
| Reset | 0x30 | | | | | | | |
| Access Type | Read Only | | | | | | | |

| BITFIELD | BITS | DESCRIPTION |
|---|---|---|
| PART_ID | 7:0 | |

## Applications Information

### Measurement Considerations

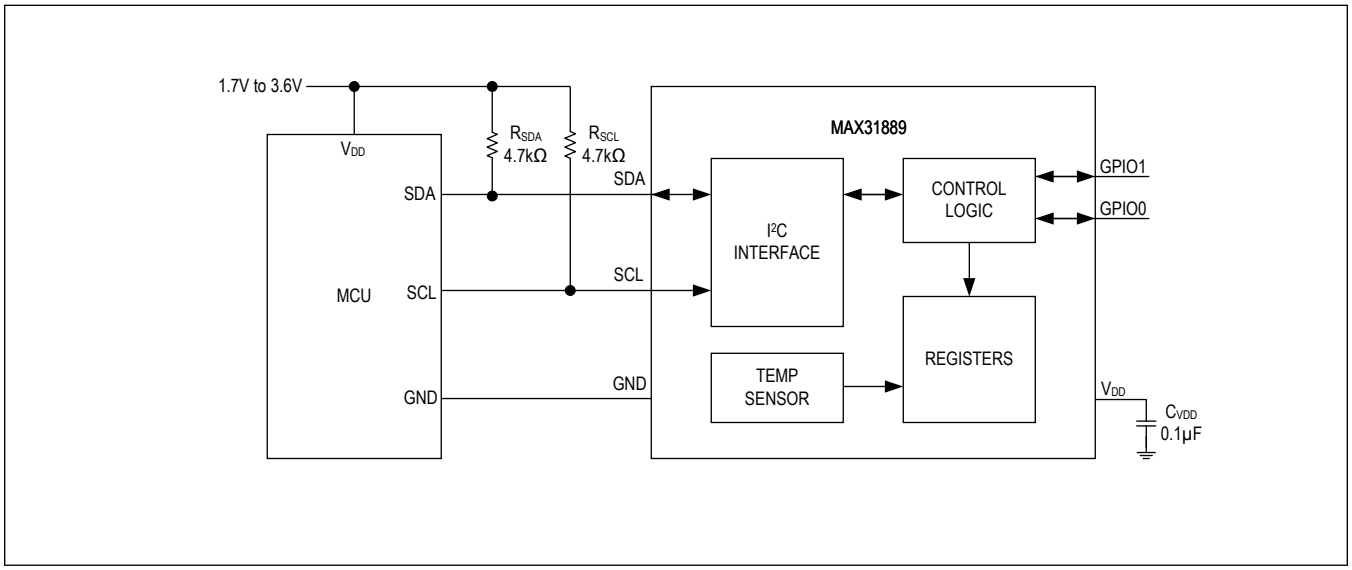
Key parameters affecting the performance of temperature sensors are the thermal conductivity from the IC to the board and from the IC to the air. A conventional surface-mount temperature sensor IC has high thermal conductivity to the circuit board on which it is mounted. Heat travels from the board, through the package leads, to the sensor die. Although air temperature also affects die temperature, the sensor's plastic package does not conduct heat as well as its leads. Therefore, board temperature has a greater influence on the measured temperature.

• Place the sensor as close as possible to the target surface to be measured and create a good thermal contact with the top of the package.

• Use traces that are as thin as possible to minimize the thermal conduction between the sensor and the rest of the PCB.

• If the board contains components that heat or cool it, mount the sensor as far as possible from those components.

## Typical Application Circuit




## Ordering Information

| PART NUMBER | TEMP RANGE | PIN-PACKAGE |
|---|---|---|
| MAX31889ALT+T | -40°C to +125°C | µDFN |

*+ Denotes a lead(Pb)-free/RoHS-compliant package.*

*T = Tape and reel.*

## Revision History

| REVISION NUMBER | REVISION DATE | DESCRIPTION | PAGES CHANGED |
|---|---|---|---|
| 0 | 1/20 | Initial release | — |
| 1 | 9/21 | Updated *Measuring Temperature* section | 8 |

*Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use, nor for any infringements of patents or other rights of third parties that may result from its use. Specifications subject to change without notice. No license is granted by implication or otherwise under any patent or patent rights of Analog Devices. Trademarks and registered trademarks are the property of their respective owners.*

# Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Analog Devices Inc.:

MAX31889ALT+ MAX31889ALT+T