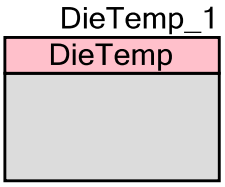
# Die Temperature (DieTemp) 1.60

## Features




- Accuracy of ±5 °C
- Range –40 °C to +140 °C (0xFFD8 to 0x008C)
- Blocking and nonblocking API
- Supports only PSoC3 ES2 silicon with date code 1005 or later and Production PSoC3 silicon

## General Description

The Die Temperature (DieTemp) component provides an API to acquire the temperature of the die. The System Performance Controller (SPC) is used to get the die temperature. The API includes blocking and nonblocking calls.

### When to Use a DieTemp

Use a DieTemp component when you want to measure the die temperature of a device.

## Input/Output Connections

There are no Input/Output Connections on the DieTemp component. It is a software component only.

## Component Parameters

The DieTemp has no configurable parameters other than standard Instance Name and Built-in parameters.

## Resources

| Analog Blocks | Digital Blocks | | | | | API Memory (Bytes) | | Pins (per External I/O) |
|---|---|---|---|---|---|---|---|---|
| | Datapaths | Macro cells | Status Registers | Control Registers | Counter7 | Flash | RAM | |
| N/A | N/A | N/A | N/A | N/A | N/A | 289 | 0 | N/A |

# Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name “DieTemp_1” to the first instance of a component in a given design. You can rename it to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance name used in the following table is “DieTemp.”

| Function | Description |
|---|---|
| DieTemp_Start() | Starts the SPC command to get the die temperature |
| DieTemp_Stop() | Stops the temperature reading |
| DieTemp_Query() | Queries the SPC to see if the temperature command is finished |
| DieTemp_GetTemp() | Sets up the command to get the temperature and blocks until finished |

## cystatus DieTemp_Start(void)

**Description:** Sends the command and parameters to the SPC to start a Die Temperature reading. This function returns before the SPC finishes. If this function is called successfully, the SPC will be locked and DieTemp_Query() will have to be successfully called to unlock it. CySpcUnlock() can also be called if the caller decides not to finish the temperature reading.

**Parameters:** void

**Return Value:** CYRET_STARTED if the SPC command was started successfully.
CYRET_UNKNOWN if the SPC command failed.
CYRET_LOCKED if the SPC was busy.

**Side Effects:** None

## void DieTemp_Stop(void)

**Description:** Stops the temperature reading.

**Parameters:** None

**Return Value:** None

**Side Effects:** None

### cystatus DieTemp_Query(int16 * temperature)

**Description:** Checks to see if the SPC command started by DieTemp_Start() has finished. If the command has not finished, the temperature value is not written. The caller will poll this function until the command is complete.

**Parameters:** int16 * temperature: Address to store the temperature in degrees Celsius.

**Return Value:** CYRET_SUCCESS if the temperature command completed successfully.
CYRET_UNKNOWN if the there was an SPC failure.
CYRET_STARTED if the temperature command has not completed.
CYRET_TIMEOUT if waited too long before reading data.

**Side Effects:** None

### cystatus DieTemp_GetTemp(int16 * temperature)

**Description:** Sends the command and parameters to the SPC to start a Die Temperature reading and waits until it fails or completes. After DieTemp_MAX_WAIT ticks, the function will return even if the SPC has not finished. This is a blocking API.

**Parameters:** int16 * temperature: Address to store the temperature in degree of Celsius.

**Return Value:** CYRET_SUCCESS if the command was completed successfully.
CYRET_TIMEOUT if the command times out.
Status codes from DieTemp_Start() or DieTemp_Query().

**Side Effects:** None

## Sample Firmware Source Code

PSoC Creator provides numerous example projects that include schematics and example code in the Find Example Project dialog. For component-specific examples, open the dialog from the Component Catalog or an instance of the component in a schematic. For general examples, open the dialog from the Start Page or **File** menu. As needed, use the **Filter Options** in the dialog to narrow the list of projects available to select.

Refer to the “Find Example Project” topic in the PSoC Creator Help for more information.

## DC and AC Electrical Characteristics

The following values indicate expected performance and are based on initial characterization data.

| Parameter | Description | Conditions | Min | Typical | Max | Units |
|---|---|---|---|---|---|---|
| | Temp sensor accuracy | Range: –40 °C to +85 °C | -- | ±5 | -- | °C |

# Component Changes

This section lists the major changes in the component from the previous version.

| Version | Description of Changes | Reason for Changes / Impact |
|---|---|---|
| 1.60 | The *DieTemp.c* file GetTemp API was edited to fix the power-on reset error output. | The DieTemp output on power-on reset was erroneous. |
| 1.50.a | Added characterization data to datasheet | |
| | Added information to the component that advertizes its compatibility with silicon revisions. | The tool returns an error/warns if the component is used on incompatible silicon. This component is not compatible with PSoC 3 ES2 or PSoC 5. |
| | Minor datasheet edits and updates | |
| 1.50 | Switch from *cydevice.h* to *cydevice_trm.h*. | The *cydevice.h* file has been made obsolete, so the APIs and generated code provided with PSoC Creator are not included with *cydevice_trm.h*. |