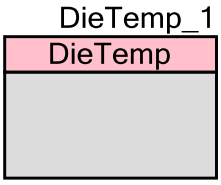
Die Temperature (DieTemp)
1.50

Features



- Accuracy of +/-5° C
- Range -40° C to +140° C (0xffd8 to 0x008c)
- Blocking and non blocking API

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 acquire the die temperature. The API includes blocking and non blocking calls.

When to use a DieTemp

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

Input/Output Connections

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

Parameters and Setup

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 would poll this function until completion of the command.

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.

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.

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 are indicative of expected performance and 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
1.50.c	Minor datasheet edit.	
1.50.b	Minor datasheet edit.	
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 ES1.
	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*.

© Cypress Semiconductor Corporation, 2010-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage (“Unintended Uses”). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.