Document Number: AN5215
Rev. 1, 03/2017

i.MX 6 Temperature Sensor Module

1. Introduction

All the i.MX6 series application processors use the same temperature sensor module (TempMon). The TempMon implements a temperature sensor conversion function based on a temperature dependent voltage to time conversion. The main purpose of the TempMon is to guard the processor against overheating. The module features an alarm function that can raise an interrupt signal if the temperature exceeds a certain threshold. Software can use this module to monitor the on-die temperature and take appropriate actions such as throttling back the core frequency when a temperature interrupt is set.

This document describes the various calibration procedures used and their associated accuracy. The software implementation of the recommended calibration equation is also discussed.

Contents

2. Calibration

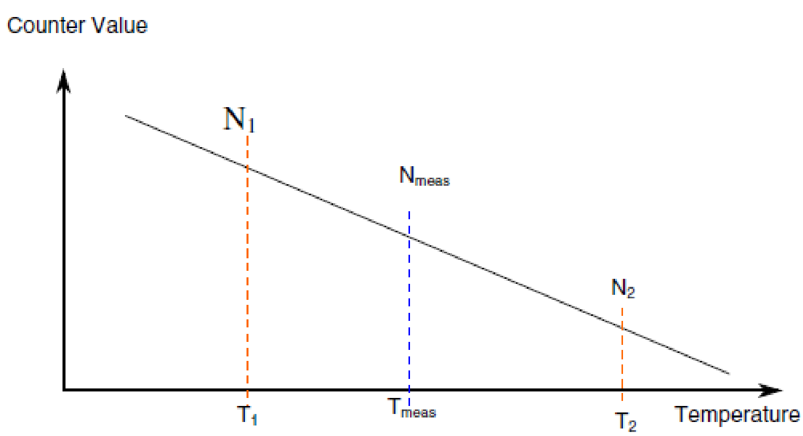
Process variations cause each die to be manufactured slightly different. These variations must be compensated to avoid affecting the accuracy of the TempMon.

The circuit contains a timer and a capacitor that converts a temperature dependent analog voltage from a diode into a digital code based on the ramp speed of the capacitor. Process variation causes this ramp speed to vary across the process corners. Each device needs to be calibrated to remove this process variation effect.

The following sections describe the calibration method in details.

2.1. Original TempMon equation

The output of the TempMon is a temperature code typically ranging from 1,200 to 1,500. Each code corresponds to an exact temperature for a particular device and is linear in response to temperature changes. The key to calibration is identifying the slope of the line, so a linear equation can be used to convert temperature codes into temperatures.



$$T_{meas} = T_2 - (N_{meas} - N_2) * ((T_2 - T_1) / (N_1 - N_2))$$

Figure 1. Original TempMon equation

The most basic way to calculate a slope is to compare two distinct points on the line. The slope is given by the difference in the temperature codes divided by the difference in the temperatures. This slope and a point can then be used in a linear equation to derive the above temperature calculation. This method, known as Two Point Calibration, was originally used to determine the slope and calculate the temperature as documented in Freescale’s reference manual.

The calibration points are measured during the ATE manufacturing testing, and are programmed in the fuse registers defined below.

Table 1. Fuse registers

Fuse offset	Fuse name	Fuse bits description
0x4E0[31:0]	Analog_temp_sense[31:0]	Bits 31-20 **Room Count**: The room temperature from the sensor. Bits 19-8 **Hot Count**: The hot temperature from the sensor. Bits 7-0 **Hot Temp**: The hot test temperature from the test program. Each LSB represents 1C.

2.2. Issues with two point calibration

This original method requires that two codes are measured at two different temperatures and fused into the device for calibration. After the production ramp for the i.MX 6 family started, this method resulted in inaccurate temperature readings from the TempMon. The following issues arose:

a) In a manufacturing environment, exact temperature references do not exist. Each piece of equipment has a set tolerance. Additionally each device has different power consumption leading to varied self heating. Both of these items make it impossible to have a set reference temperature and are sources of error.

b) The hot calibration point and the room temperature calibration point are not taken with the same equipment. The room temperature calibration point is taken at wafer probe. The wafer is on a large metal chuck used to stabilize the temperature. The hot calibration point is taken at final test with an individual die assembled into a package. Final test uses air cooled/heated mechanical handlers to set the reference temperatures. The difference in heating/cooling methods can cause error.

c) Two calibration points double the sources of error. Even if two points could have been taken with the same equipment, the equipment and device tolerance as described in a) is present any time a measurement is taken. Error can be added at each step.

2.3. Single point calibration

In 2013, Freescale published a new temp sense equation based on a single calibration point. The chosen single point is at wafer probe, which is shown to have better thermal control. While this does not eliminate every source of error described in Section 2.2, it minimizes the error as much as possible for the given manufacturing environment. Using this method the temperature equation is changed to the one below.

$$T_{meas} = T_1 + \frac{N_{meas} - N_1}{Slope}$$

Figure 2. Temperature equation

The slope is calculated using a formula derived from empirical data. Analysis shows that the slope of every device is linearly related to the value of the temperature code at room temperature.

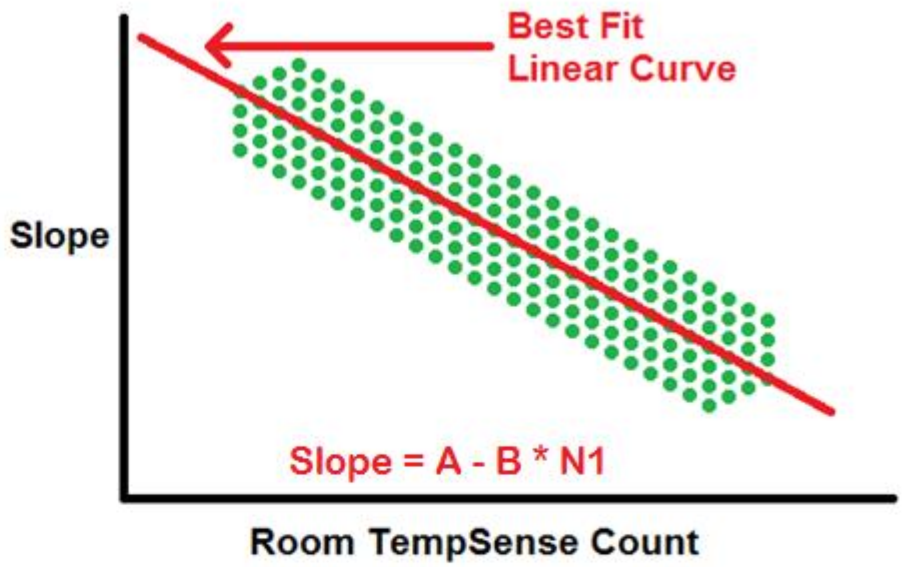


Figure 3. Relationship of the slope and room temperature

3. Accuracy

After the i.MX 6 Family produced in high volumes for a statistically relevant duration, the TempMon equation was revisited to check for accuracy.

3.1. Accuracy study

A sample of parts taken across the process distribution and across the family of i.MX 6 devices was used for the accuracy study. The measurements were taken in a thermally controlled chamber at steady state to ensure an exact calibration (See Appendix B for best measurement techniques). A thermocouple was placed on the top of each device to measure the exact temperature of each device in steady state. The devices are only powered on to take a measurement from the TempMon module to limit self heating.

Each device was manually calibrated across multiple temperatures to determine the exact slope to use in the linear TempMon equation. The codes of the points tested were then put into the TempMon equation from 2.3 and compared against the measured temperature.

3.2. Updated equation

The results show that a slight change to the slope equation is necessary, and an additional offset term needs to be added to the equation. In 2015, Freescale released the following changes to the TempMon equation.

$$\mathbf{T_{meas}} = \mathbf{T_1} + \frac{\mathbf{N_{meas}} - \mathbf{N_1}}{\boldsymbol{Slope}} + \boldsymbol{offset}$$

Where

$$\text{Slope} = 0.4148468 - 0.0015423 * \text{N}_1$$

$$\text{Offset} = 3.580661$$

$$\text{N1} = \text{Room Count (Bits 31-20 of the analog\_temp\_sense efuse register)}$$

$$\text{T1} = 25^\circ\text{C}$$

3.3. Accuracy results

The following table represents the results of the accuracy study after the equation updates are applied.

Table 2. Accuracy results

Temperature (°C)	Estimated accuracy[1]
0	+/-6.31
25	+/-4.72
105	+/-4.61
125	+/-5.84

[1] The Estimated accuracy is the calculated value at +/- 3 standard deviations of the population.

It is worth noting that the TempMon circuit is most accurate at the highest temperature range. This is consistent with the intent of the design of the module. As stated earlier, the TempMon's main purpose is to protect the device when temperatures exceeds a specified threshold. The converter has a granularity of roughly 2 codes per degree Celsius.

Appendix A. Software Revision History

The Freescale Linux BSP has a thermal driver that implements the TempMon equation along with the associated thermal trip points (drivers/thermal/imx_thermal.c). The software thermal driver uses the TempMon module to monitor the on-die temperature and take appropriate actions such as throttling back the core frequency when a temperature interrupt (thermal trip point) is set.

Customers can adapt the thermal trip points to suit their application as long as they ensure that the device T_j max specifications are not exceeded.

Freescale does not recommend modifying the TempMon equation implemented in software as it can lead to incorrect functionality of the thermal driver.

The following list contains the software releases containing updates to the TempMon equation.

- Original release up to imx_3.0.35_12.12.01 - two point calibration
    - Not recommended due to accuracy concerns
- imx_3.0.35_2.1.0 – 2013 version of the one point calibration equation
    - Not recommended due to accuracy concerns
- imx_3.14.38_6ul_ga – 2015 – latest version of the one point calibration equation
    - MLK-10828 thermal: imx: update the temperature calibration data for i.MX 6
    - LINK: *http://git.freescale.com/git/cgit.cgi/imx/linux-2.6-imx.git/commit/?h=imx_3.14.38_6ul_ga&id=c93424a3926c908d811235871cc419c7bfad9b4f*

Appendix B. Die Heating and Process Variation

Usually, two parts running the same code on the same board in the same environment run at different temperatures. The following two sections describe how this is possible and why it should be expected.

B.1. Die heating

When a system is at steady-state and no voltage is applied to the device, all temperatures are maintained at the same value; e.g., the die temperature is equal to the ambient temperature. When the voltage is applied, the die starts generating heat, increasing the junction, case temperatures, and attached heat sinks or spreaders. The die is the hottest component in the system as it generates heat. The package lid, attached heat spreader, or heat sink has a lower T_{case} temperature based on the thermal resistance of the junction. When measuring T_{case}, it always has a lower temperature than $T_{junction}$ unless no voltage is applied to the device.

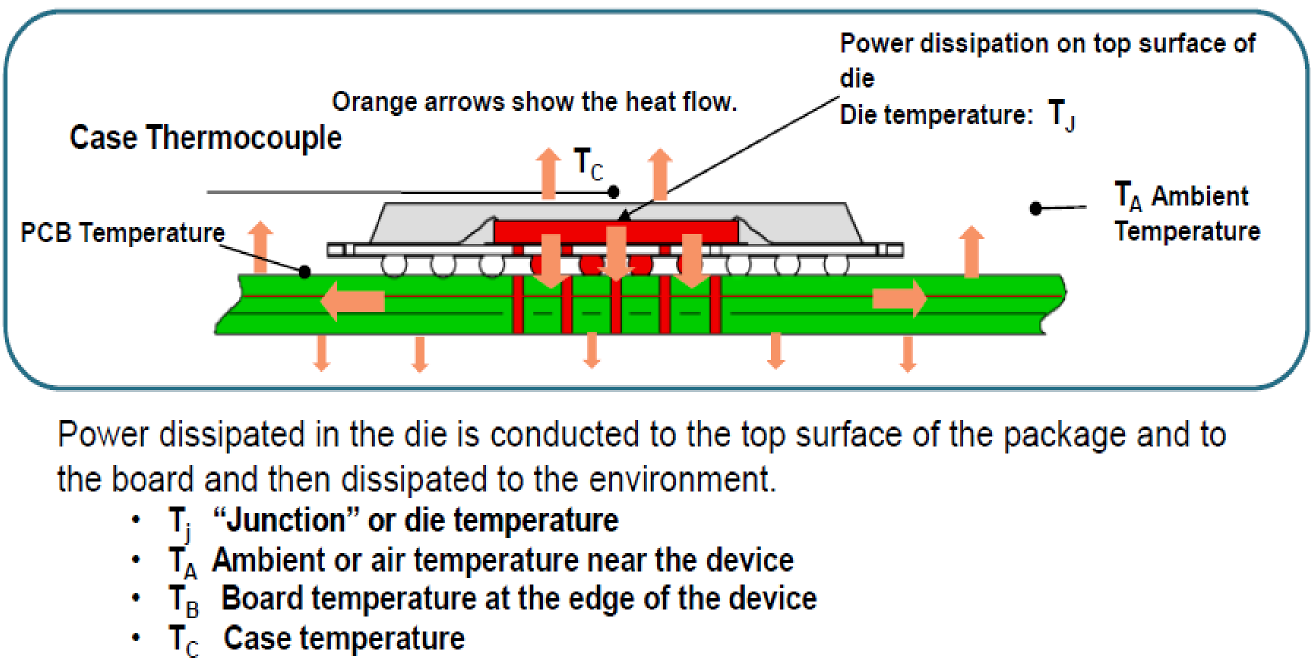


Figure 4. Die heating

B.2. Die process variation

Variations in die-to-die processing results in devices with a range of power consumptions. These variations cause two individual parts that are manufactured identically behave differently. Comparing the upper and lower extremes of the top histogram in Figure 5, devices can be up to 2 W different. Using a thermal resistance of 12 degrees Celsius per Watt, there is a difference of up to 24 ℃, as shown on the bottom histogram in Figure 5.

When using temperature measurements to judge a process, circuit, or software change, a distribution of parts should be used. Looking at two parts may show vast differences, or may show no differences depending on the two devices selected. A distribution of devices always shows the true effect of the change.

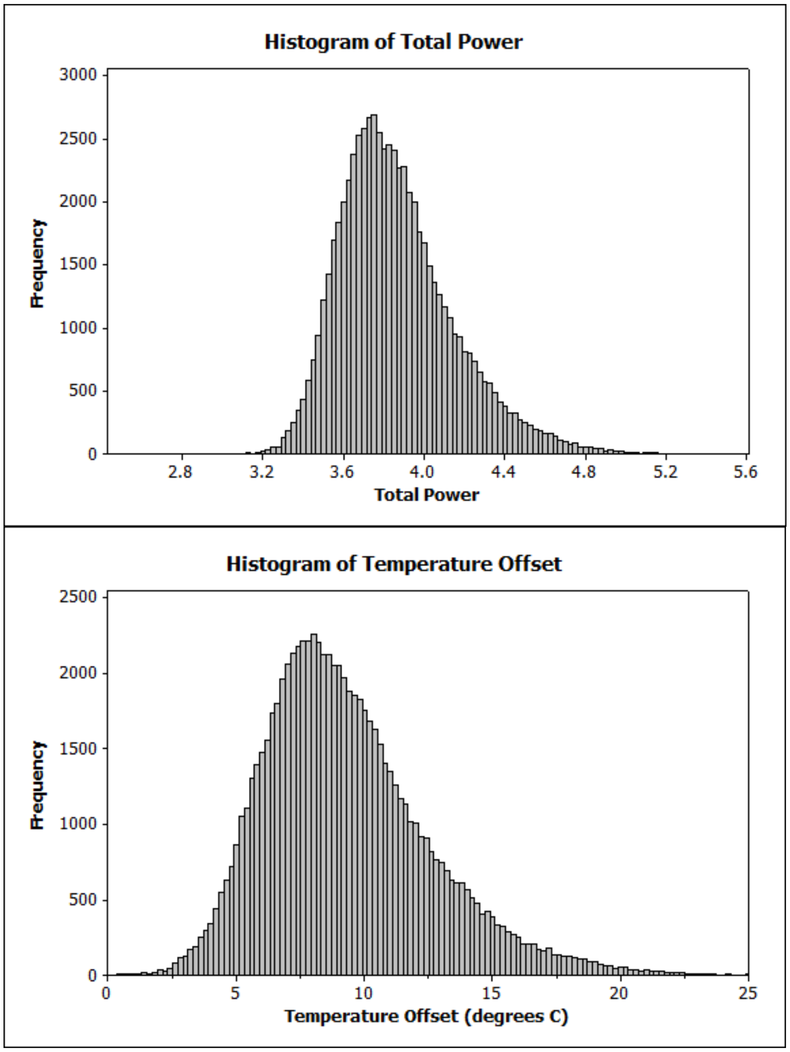


Figure 5. Histogram of total power and temperature offset

Appendix C. Best Practices for Thermal Measurements

The leading causes for temperature measurement error are: not enough soak time, non-controlled ambient environment, and measuring when the system is in a changing state.

A controlled environment is the key to a valid thermal measurement. Variations in the ambient temperature can induce error and confuse any temperature reading the measured device reports.

A steady-state environment is also imperative. If the device's workload is variable, it can cause cyclical localized heating and cooling events. During this cycle, it is nearly impossible to determine what the device is doing when the temperature measurement is taken; making that measurement non-repeatable.

Determining that the device is in steady state can be done by adding more soak time and monitoring the thermocouple on the device. Only when the reading is stable over time has steady-state been achieved.

Table 3. Thermal measurement technique

Number	Thermal measurement technique
1	Always use a temperature controlled ambient environment.
2	Use thermocouples to ensure that the chamber temperature reflects the setpoint.
3	Use ample soak time. A thermocouple on the device measured should indicate how much soak time is enough.
4	Measure in Steady-State events when possible.

Appendix D. References

- *i.MX 6 Series Thermal Management Guidelines* (AN4579)
  http://cache.freescale.com/files/32bit/doc/app_note/AN4579.pdf
- iMX 6 Reference Manuals, Chapter “Temperature Monitor (TEMPMON)”

Appendix E. Revision History

Table 4. Revision history

Revision number	Date	Substantive changes
0	02/2016	Initial release
1	03/2017	Updated the original TempMon equation in Figure 1.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: AN5215
Rev. 1
03/2017