APPLICATION NOTE

PAC52XX Temperature Sensing and Protection

Power Application Controller^{TM}

Marc Sousa
Senior Manager, Systems and Firmware

TABLE OF CONTENTS

OVERVIEW

The PAC52XX family of devices has integrated temperature sensing and protection features that are available for all applications.

The following describes how to use the integrated temperature sensor as well as how to recover from the over-temperature protection when activated.

TEMPERATURE SENSING

The PAC52XX has an integrated temperature sensor that may be used to measure the temperature of the device. The temperature sensor is calibrated when the device is tested, and calibration parameters may be used during temperature measurement to provide an accurate calculation of temperature.

Temperature Calculation

The integrated temperature sensor on the PAC52XX is available through an ADC channel on the Configurable Analog Front-End (CAFE). The temperature sensor has been calibrated at 25°C with a nominal value of 1.5V.

Each PAC52XX device calibrates the temperature and stores the result in the INFO block. For more information on how to accurately calculate the temperature using these calibration values, see the section below on Temperature Calibration.

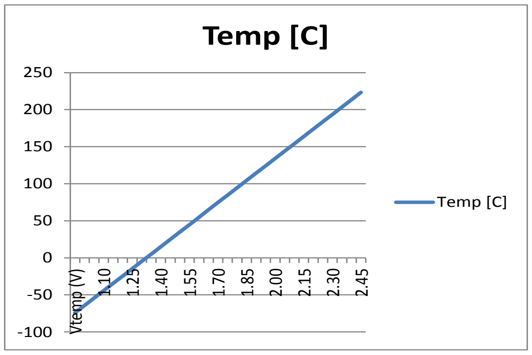
The temperature can be calculated based on sampling the ADC channel for the temperature sensor. This voltage (Vsensor) may be used to calculate the temperature as follows:

```
Temp (°C) = (Vsensor – 1.5V) / 5.04mV/°C + 25°C
```

Where:

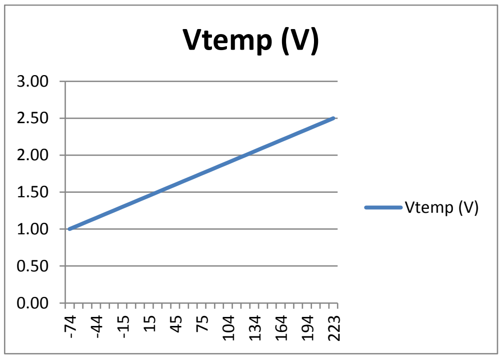
- **Vsensor** – Temperature (V) as measured from the ADC
- **1.5V** – Vsensor calibration value (for 25°C)[1]
- **5.04mV/°C** – Temperature gain[2]
- **25°C** – Temperature calibration value (for 1.5V)

The data below shows the sensor temperature (°C) as a function of Vtemp (V):

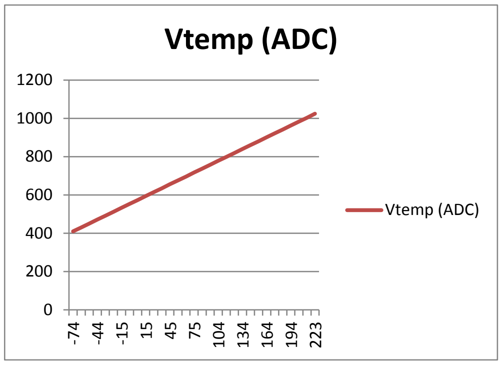


The data below shows Vtemp (V) as a function sensor temperature (°C):

[1] Note that the Vsensor calibration value and Temperature calibration value may be calibrated based on information in the INFO FLASH block. See the section on Temperature Calibration for more information.
[2] The Temperature gain value used here is for the PAC5220 device data sheet. Other devices in the family may have different temperature gains.



The data below shows Vtemp (in ADC counts) as a function of sensor temperature (°C):



Temperature Calibration

Because the temperature sensor on each PAC52XX device may behave differently, each PAC52XX device calibrates the temperature sensor.

The two parameters that are calibrated are shown in the table below.

Name	Address	Nominal Value	Description
FTTEMP	0x0010 0028	25°C	Test temperature for the internal temperature sensor in °C.
TTEMPS	0x0010 002A[3]	614 counts	The number of ADC counts sampled for the temperature sensor at the temperature given in FTTEMP.

In the table above, the following are:

- ***Name*** – The name of the calibration parameter from the PAC52XX User Guide
- ***Address*** – The 32-bit address of the parameter located in INFO FLASH according to the PAC52XX User Guide
- ***Nominal Value*** – The ideal value of the parameter
- ***Description*** – Description of the parameter

With this information, an accurate measurement of temperature can now be done.

[3] Note that access to this data in INFO FLASH must be 32-bit aligned. Therefore, the user should read 32-bits from 0x010028 into a local register or RAM to get both 16-bit values.

Before calibration, the formula for calculating temperature was:

```
Temp (°C) = (Vsensor – 1.5V) / 5.04mV/°C + 25°C
```

In the formula above, we can replace the 1.5V with TTEMPS and 25°C with FTTEMP. We can also convert 5.04 to ADC counts (2, rounded from 2.06). Now the formula can be calculated quickly using integer arithmetic as follows:

```
Temp (°C) = (Vsensor_adc – TTEMPS) / 2 + FTTEMP
```

For more accuracy, do not convert to ADC counts and use a Q-Math library to operate on the voltage values as floating point numbers.

At the end of this document, there is a table that shows the Voltage and ADC counts for the ideal temperature calculation.

Temperature Measurement

To measure the temperature in firmware, the user must configure the ADC to sample the AB10 channel on PC0. The AB10 bus always has the temperature sensor.

The code to sample this bus (and the temperature sensor) using ADC manual mode is[4]:

```
pac5xxx_adc_enable(0);                                    // Disable ADC (if not already)
pac5xxx_adc_config(ADCCTL_ADSTART_SINGLE, ADC_DIV, 0);    // Configure the ADC
PAC5XXX_ADC->ADCINT.ADCINT = 0;                           // Clear ADC interrupt flag (also used for ADC complete check)
pac5xxx_adc_enable(1);                                    // Enable ADC (must be on for following steps

pac5xxx_tile_register_write(ADDR_ADCSCAN, 0x08);          // ENADCBUF=1, ENSCAN=0, so we can write ADDR_ADCIN1
pac5xxx_tile_register_write(ADDR_ADCIN1, SIGMGR_AB10);    // Change ADC MUX to AB10 – temperature sensor

pac5xxx_adc_set_mux_and_start(ADCCTL_ADMUX_AD0);          // Change ADC MUX to PC0/AD0 and start conversion

while (pac5xxx_adc_conversion_complete() == 0);           // Wait for ADC conversion to be complete

temp_counts = pac5xxx_adc_get_result();                   // Get ADC result and store
```

Once the temperature has been sampled in ADC counts, it can be converted to degrees C by the following code:

```
#define INFO_WORD (*((unsigned long*)0x00100028))   // 32-bit value from INFO FLASH at 0x0010028
#define TTEMPS (INFO_WORD >> 16)
#define FTTEMP (INFO_WORD & 0xFFFF)

temp_degC = ((temp_counts – TTEMPS) >> 1) + FTTEMP
```

[4] This code configures the ADC in manual mode. To perform automatic conversions of the temperature, see the PAC52XX User Guide or ADC application notes.

TEMPERATURE PROTECTION

The PAC52XX family of devices supports temperature protection. There are two levels of temperature protection available in hardware:

- Temperature Warning
- Temperature Fault

The sections below describe the features of these two levels of temperature protection, as well as how to implement firmware to use these thresholds during operation.

Of course, the application firmware can always sample the device temperature (via the internal sensor) as described above to use for compensation and software protection as well.

Temperature Warning

The PAC52XX family of devices will declare an over-temperature warning when the die temperature reaches 140°C.[5] When this event occurs, the OTWINT bit (bit 6) in SOC.SYSSTAT will be set to a 1.

The PAC52XX has a mask-able interrupt that may be used to interrupt the MCU if this condition happens. See the section below on temperature warning interrupts for more information.

If interrupts are not enabled for this warning, the user may create firmware to test for this condition by reading the SOC.SYSSTAT register. If desired, this condition may be cleared by writing 1 to the OTWINT bit as shown below:

```
sysstat = pac5xxx_tile_register_read(ADDR, SYSSTAT);

if ((sysstat & 0x40) != 0)
{
  // Temperature warning detected!

  value = 0x40;                                          // OTWINT bit (bit 6)
  pac5xxx_tile_register_write(ADDR_SYSSTAT, value);      // Clear OWINT bit
}
```

NOTE:

Once the OTWINT bit is cleared (by writing a one to it), the over-temperature warning condition will not happen again until the temperature falls below 140°C, and then rises above 140°C again. So, if the temperature remains over 140°C, and the warning bit is cleared it will not be set again until the temperature falls, and then rises again.

Temperature Fault

The PAC52XX family of devices will declare an over-temperature fault when the die temperature reaches 170°C.[6] When this event occurs, if the OTSD bit in SOC.SYSTAT is clear, then the device is shut down (this is default behavior). If this bit is set, the device will not shut down.

[5] Consult the data sheet parameter for the T_{WARN} parameter for the exact threshold.

If the device is configured to shut down during this fault, then all the power supply voltage faults and over-temperature fault need to be manually cleared via firmware to be used again, unless the device is reset.

NOTE:

The MCU may be reset when the analog sub-system is not reset. So, the firmware may begin executing its program without the analog sub-system being reset during a soft reset (like when the debugger from the IDE is attached). The firmware can test for this condition by sampling the bits in the SOC.PWRSTAT register as described below.

When the over-temperature fault happens, the user can tell by the following behaviors:

- The power supply voltage fault bits in SOC.PWRSTAT are set:
    - VCC18 (bit 0)
    - VCC33 (bit 1)
    - VCCIO (bit 2)
    - VSYS (bit 3)
- The TMP (over-temperature fault bit – bit 4) in SOC.PWRSTAT is set

NOTE:

The over-temperature fault condition usually indicates a serious problem has occurred that is worth noting via the user. The firmware application may decide (by sampling the bits above) if the system should be latched off, or if it should be reset.

In order to reset this condition, to re-enable operation of the system the following actions must take place:

- Disable the driver manager
- Disable the signal manager
- Clear all fault bits set in the SOC.PWRSTAT register
- Re-enable the signal manager
- Re-enable the driver manager

After these steps are completed after an over-temperature fault, the device will be able to resume proper operation.

The firmware to clear the over-temperature fault condition is shown below:

```
pac5xxx_tile_register_write(ADDR_ENSIG, 0);         // Disable signal manager (bit 0 = 0)
pac5xxx_tile_register_write(ADDR_ENDRV, 0);         // Disable driver manager (bit 0 = 0)
pac5xxx_tile_register_write(ADDR_PWRSTAT, 0x1F);    // Disable fault bits: VCC18 [0], VCC33[1],
                                                    //   VCCIO [2], VSYS [3], TMP [4]

pwrstat = pac5xxx_tile_register_read(ADDR_PWRSTAT);
If (pwrstat != 0)
{
  ; // Error bits should be cleared at this point!
}

pac5xxx_tile_register_write(ADDR_ENSIG, 1);         // Enable signal manager (bit 0 = 1)
pac5xxx_tile_register_write(ADDR_ENDRV, 1);         // Enable driver manager (bit 0 = 1)
```

[6] Consult the data sheet parameter for the T_{FAULT} parameter for the exact threshold.

Temperature Warning Interrupts

A temperature warning condition has a mask-able interrupt available that may be used by the firmware on the PAC52XX. To enable this interrupt, the user may set the OTWINTEN bit (SOC.SYSSTAT, bit 0) to a 1.

If this bit is set, then the MCU will receive an interrupt on nIRQ1 (PB0 on the MCU) when the over-temperature warning event occurs.

To enable this interrupt, the user may use code similar to that below:

```
sysstat = pac5xxx_tile_register_read(ADDR, SYSSTAT);        // Read SOC.SYSSTAT
sysstat |= 0x40;                                            // Set OTWINT bit
pac5xxx_tile_register_write(ADDR_SYSSTAT, sysstat);         // Update register
```

Once the over-temperature warning interrupt is set as shown above, the user should enable interrupts on PB0 on the MCU to receive these interrupts, as shown below:

```
pac5xxx_gpio_configure_interrupt_b(1, 0);        // PB0 interrupt enable (port_mask[PB0], active_high_mask[active low])

__enable_irq();                                  // Enable global interrupts (if not already)
```

Now, the user will receive interrupts on PB0. The user should define an interrupt handler

```
void GpioB_IRQHandler(void)
{
  // Port B interrupt. Read register to see if this is over-temperature warning

  sysstat = pac5xxx_tile_register_read(ADDR, SYSSTAT);

  if ((sysstat & 0x40) != 0)
  {
    // Temperature warning detected!

    value = 0x40;                                         // OTWINT bit (bit 6)
    pac5xxx_tile_register_write(ADDR_SYSSTAT, value);     // Clear OWINT bit

    // HANDLE OT WARNING here
  }

  // Clear interrupt flag

  pac5xxx_gpio_int_flag_clear_b(0x01);                    // Clear PB0 interrupt flag
}
```

APPENDIX A: IDEAL TEMPERATURE VOLTAGE AND ADC COUNTS

Vtemp (V)	Vtemp (ADC)	Temp [C]
1.00	410	-74
1.05	430	-64
1.10	451	-54
1.15	471	-44
1.20	492	-35
1.25	512	-25
1.30	532	-15
1.35	553	-5
1.40	573	5
1.45	594	15
1.50	614	25
1.55	635	35
1.60	655	45
1.65	676	55
1.70	696	65
1.75	717	75
1.80	737	85
1.85	758	94
1.90	778	104
1.95	799	114
2.00	819	124
2.05	840	134
2.10	860	144
2.15	881	154
2.20	901	164
2.25	922	174
2.30	942	184
2.35	963	194
2.40	983	204
2.45	1004	213
2.50	1024	223

ABOUT ACTIVE-SEMI

Active-Semi, Inc. headquartered in Dallas, TX is a leading innovative semiconductor company with proven power management, analog and mixed-signal products for end-applications that require power conversion (AC/DC, DC/DC, DC/AC, PFC, etc.), motor drivers and control and LED drivers and control along with ARM microcontroller for system development.

Active-Semi's latest family of Power Application Controller (PAC)™ ICs offer high-level of integration with 32-bit ARM Cortex M0, along with configurable power management peripherals, configurable analog front-end with high-precision, high-speed data converters, single-ended and differential PGAs, integrated low-voltage and high-voltage gate drives. PAC IC offers unprecedented flexibility and ease in the systems design of various end-applications such as Wireless Power Transmitters, Motor drives, UPS, Solar Inverters and LED lighting, etc. that require a microcontroller, power conversion, analog sensing, high-voltage gate drives, open-drain outputs, analog & digital general purpose IO, as well as support for wired and wireless communication. More information and samples can be obtained from http://www.active-semi.com or by emailing marketing@active-semi.com

Active-Semi shipped its 1 Billionth IC in 2012, and has over 120 in patents awarded and pending approval.