Preliminary

Thermistor Datasheet Thermistor V 2.00

Copyright © 2012-2013 Cypress Semiconductor Corporation. All Rights Reserved.

This datasheet contains Preliminary information.

MUM		PSoC® Blocks				API Memory (Bytes)		Pins
		Digital	Analog CT	Analog SC	Decimator	Flash	RAM	
Supported Devices: CY8C24x23A, CY8C27x43, CY8C24x94, CY8C29x66, CY8C28x23, CY8C28x33, CY8C28x43, CY8C28x45, CY8C28x52								
Configuration	Routed Reference, RefLo ≠ VSS	2	2	1	1	Code_Size* + LUT_Size**	18	3
	Direct Reference, RefLo = VSS	2	1	1	1	Code_Size* + LUT_Size*	18	3

Notes:

* Code_Size:

Base Device	Code Size, Bytes
CY8C24x23A, CY8C27x43	1056
CY8C24x94, CY8C29x66	1101
CY8C28x23, CY8C28x33, CY8C28x43, CY8C28x45, CY8C28x52	1081

** LUT_Size: The LUT size (bytes) is determined by following equation:

LUT_Size = ((Max Temp - Min Temp) + 1) * 4;

The Min Temp and Max Temp are defined below in "Thermistor Wizard" section.

Features and Overview

The Thermistor User Module measures the temperature from a thermistor by measuring resistance and converting to temperature using a linear spline fit to a table. This table is created using the Steinhart-Hart equation. Following are this user module's important features:

- Provides an interface to measure temperature from a thermistor using the ratio metric method
- Temperature calculation by using a linear spline fit to the Lookup Table
- Automatic clock calculation
- Offset error compensation

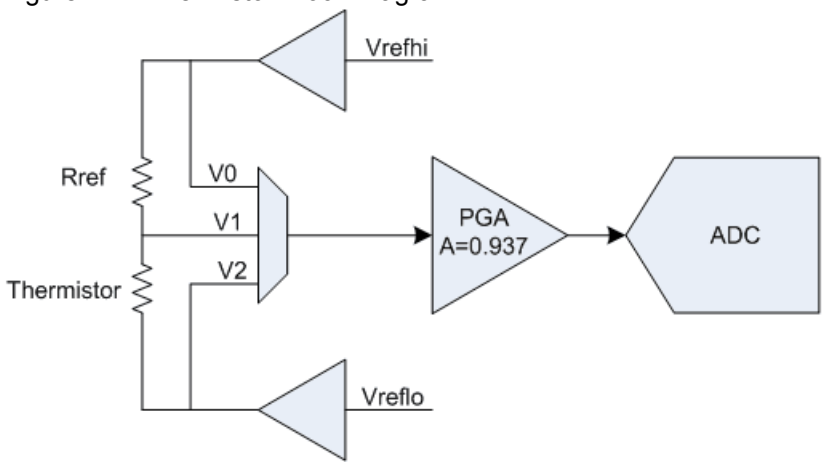
Quick Start

1. Select and place the Thermistor User Module from the temperature category in the user module catalog.
2. In the multi user module (MUM) wizard, select the Routed or Direct Reference configuration.
3. Right-click the Thermistor User Module in **Workspace Explorer** to access the Thermistor wizard (the wizard is explained later in this datasheet).
4. Enter the Reference Resistance parameters used in temperature calculation.
5. Enter other parameters such as minimum, middle, and maximum values for three temperature/resistance points in celsius degrees/ohms respectively.
6. Enter the pin to which the thermistor will be connected. Click **OK.**
7. Enter user module parameters.
8. Generate the application and switch to the **Application Editor**.
9. Adapt the sample code, as required.
10. Program the PSoC on the target board with the hex generated by PSoC Designer™.

Functional Description

The thermistor measurement is based on the Cypress application note, AN2017 - PSoC 1 Temperature Measurement with Thermistor.

Figure 1. Thermistor Block Diagram



In Figure 1, the equation to measure resistance of thermistor is:

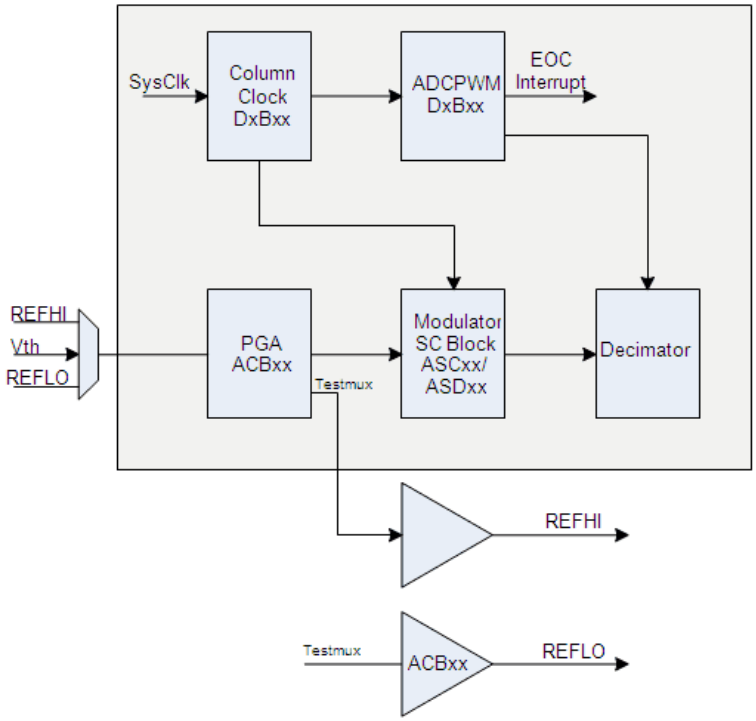
Rthermistor = Rref * (V1-V2) / (V0-V1)

After the thermistor resistance is calculated from measured voltages, you can calculate the temperature by a combination of binary search through the temperature/resistance table to get the integer part of the result, and then linear spline fit to achieve the decimal resolution.

The resistance of thermistor is an (unsigned long) integer value in tenths of an Ohm. The temperature value is an integer value in hundredths of a degree C.

The following diagram shows the PSoC 1 hardware block diagram of the Thermistor User Module.

Figure 2. Hardware Block Diagram



The Thermistor User Module essentially consists of a programmable gain amplifier (PGA), an incremental analog-to-digital converter (ADC), two analog buffers, and a mux.

A CT block is configured as a PGA. This CT block's testmux is configured for REFHI and connected to the analog bus of that column. Another CT block is configured as RefMux to bring out REFLO. The PGA gain is configured as 0.937.

A single SC block is configured as a first-order ADC modulator. The output of the modulator is connected to the input of a decimator.

A counter is used to generate the column clock for the ADC. A second counter is configured to generate the gate pulses for the ADC. The ADC is configured for a 13-bit resolution and can use a maximum column clock of 2.0 MHz. This requires a column clock divider minimum equal to 12 and results in a maximum sample ADC sample rate of 61 Hz. With three samples taken to get the temperature measurement, the maximum update rate is 20.3 temperature values per second.

DC and AC Electrical Characteristics

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

Table 1. Thermistor DC and AC Electrical Characteristics

Parameter	Description	Conditions	Min	Typ	Max	Units
Thermistor Function						
R_{Error}	Resistance measurement error	Reference resistance = 10K, 0.1% Thermistor resistance: 1K 10K 100K		 0.50 0.25 0.50	 – – –	%
I_{oper}	Operating current	Low power Medium power High power	–	272 1380 5452	–	µA
T_{Error}	Temperature Error	10-k thermistor; reference resistor = 10k, 0.1% R_{TH} for -40 °C R_{TH} for 25 °C R_{TH} for 100 °C	 –0.2– 0.15 –0.2	 ±0.1 ±0.1 ±0.1	 0.2 0.15 0.2	°C

Placement

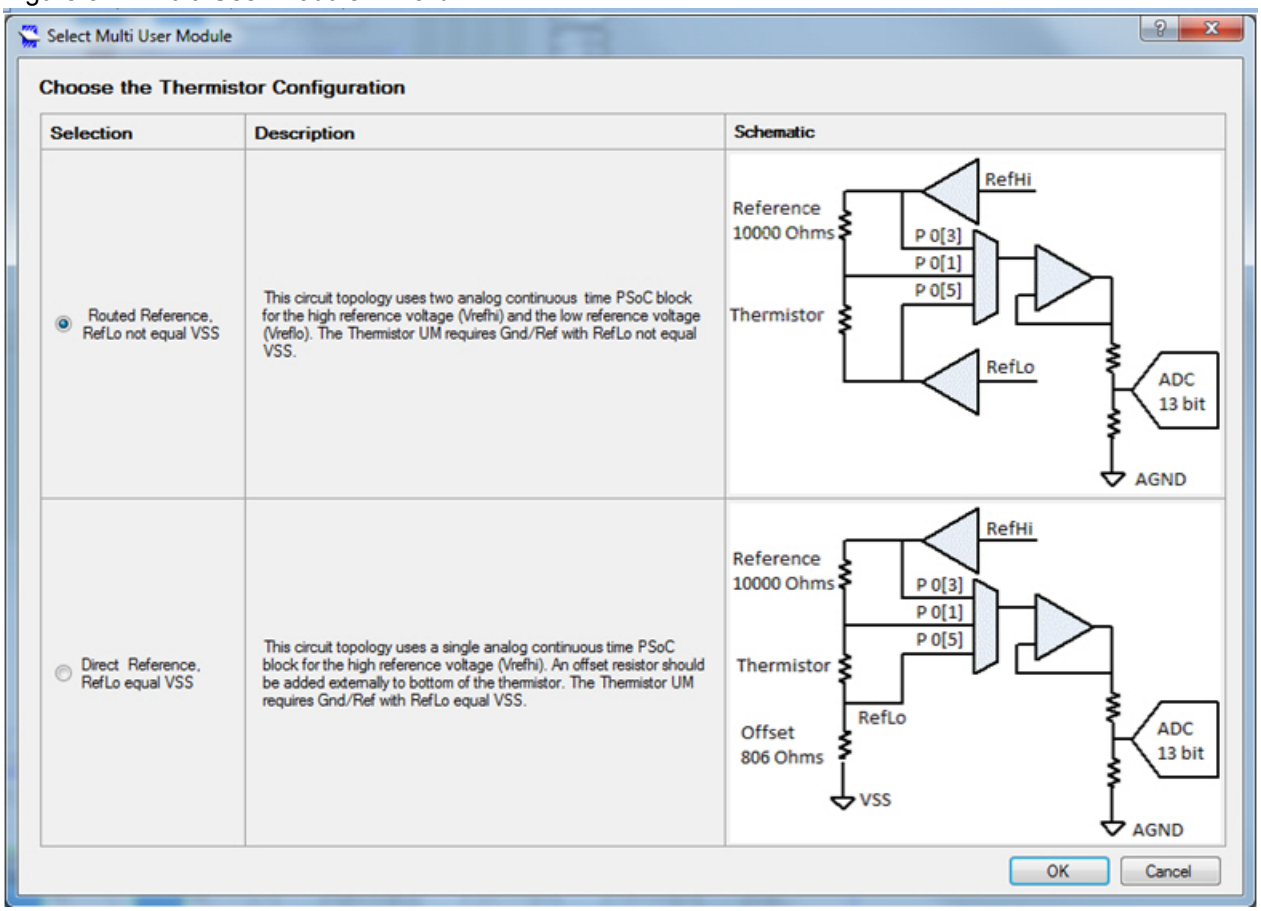
Thermistor is a Multi User Module (MUM). The Thermistor with RefLo ≠ VSS requires two CT blocks, one SC block, and two digital blocks. The thermistor with RefLo = VSS requires one CT block, one SC block, and two digital blocks.

Only one instance of the user module can be placed in the project.

Multi User Module Wizard

Select and place the Thermistor User Module from the Temperature category in the user module catalog. The MUM wizard (see the following screenshot) helps you to choose the configuration.

Figure 3. Multi User Module Wizard



There are two MUM configurations:

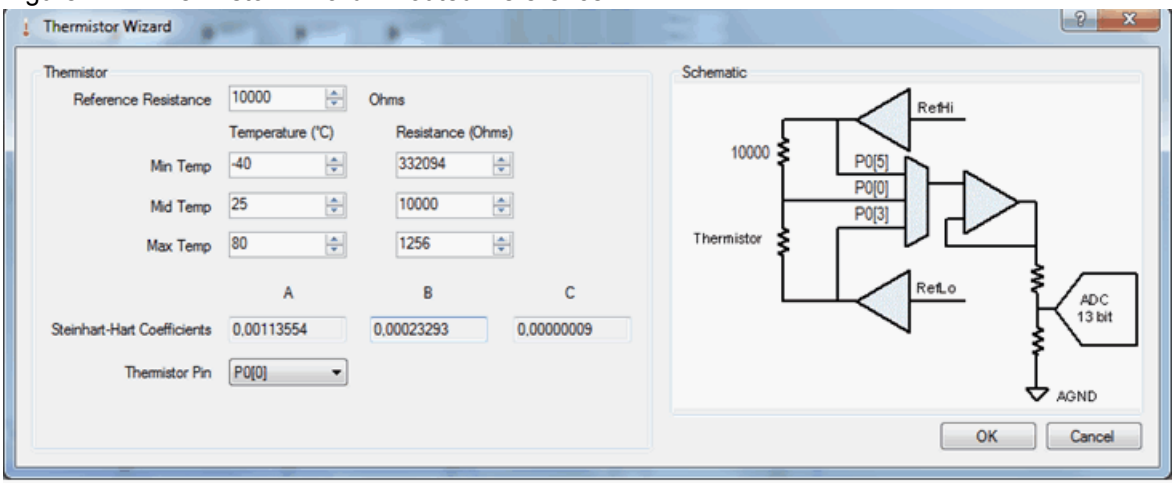
1. Routed Reference, RefLo ≠ VSS: This circuit topology uses two analog continuous time PSoC blocks for the high reference voltage (Vrefhi) and the low reference voltage (Vreflo). The Thermistor User Module requires Gnd/Ref with RefLo ≠ VSS.
2. Direct Reference, RefLo = VSS: This circuit topology uses a single analog continuous time PSoC block for the high reference voltage (Vrefhi). Add an offset resistor externally to the bottom of the thermistor. The Thermistor User Module requires Gnd/Ref with RefLo = VSS.

Click **OK** to place the Thermistor User Module.

Thermistor Wizard

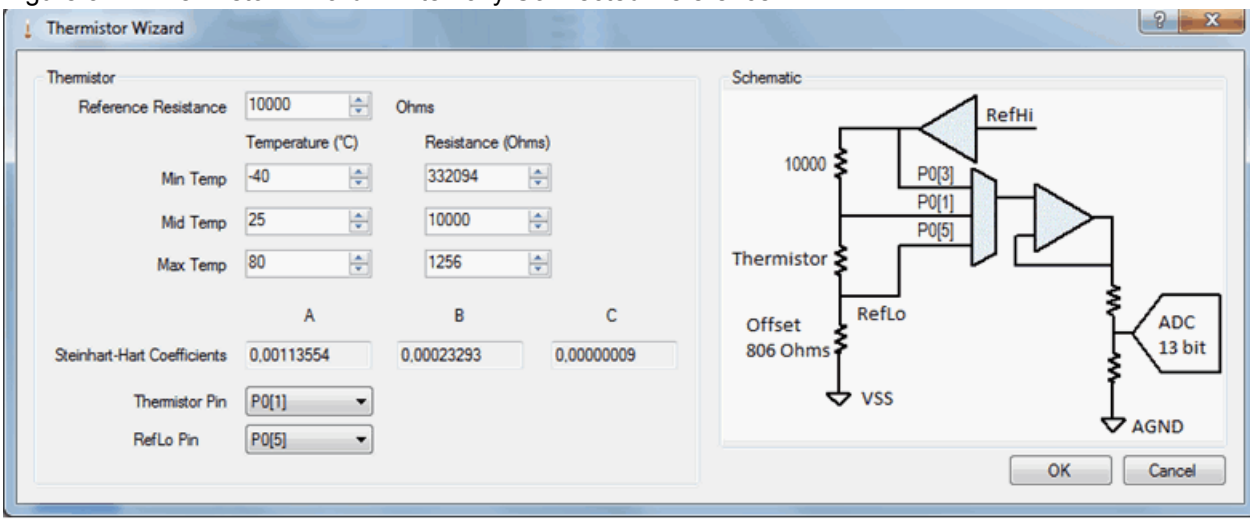
The Thermistor Wizard can also be invoked from the right-click menu of the user module in the **Chip Editor**, **Workspace Explorer,** by selecting the "Thermistor Wizard" item. When the MUM is selected as routed reference, (RefLo is not equal to VSS), the wizard is as follows.

Figure 4. Thermistor Wizard - Routed Reference



When the MUM is selected as an externally connected reference (RefLo = VSS), the wizard is as shown in the following screenshot. The difference is in the image of the block diagram and the available pin selections.

Figure 5. Thermistor Wizard - Externally Connected Reference



The wizard generates a lookup table with an interval of 1.0 degree. The number of points in lookup table depends on "Min Temp" and "Max Temp" parameters but not more than 141. The MUM wizard has the following parameters:

Reference Resistance

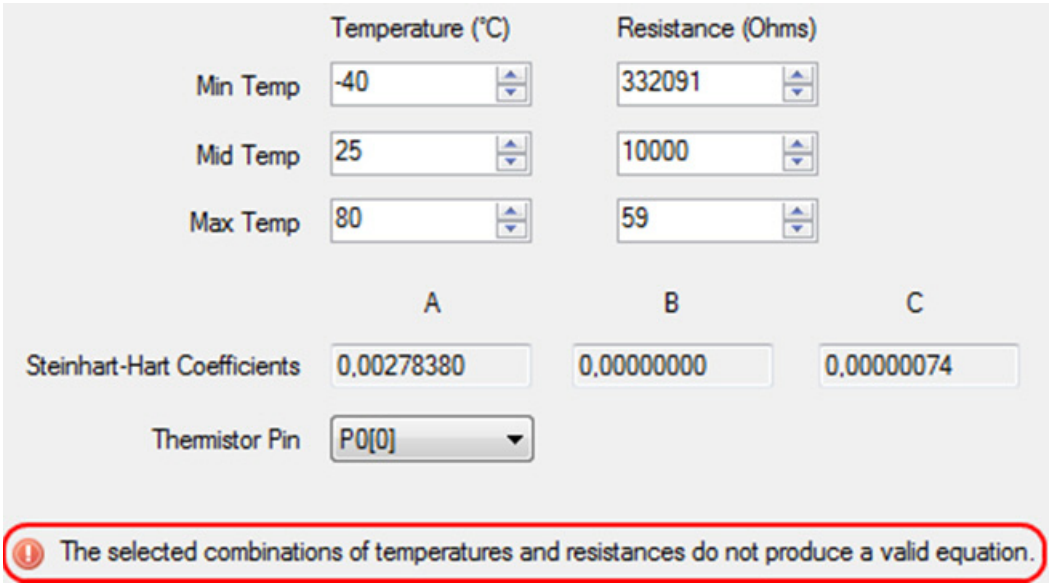
The Reference Resistance parameter sets the reference resistance used in temperature calculation. The resistance range is between 1000 and 30000. The default value is 10000.

Min Temp, Mid Temp, and Max Temp

The Min Temp, Mid Temp, and Max Temp parameters set the minimum, middle, and maximum values for three temperature and resistance points. The temp ranges are from -40 °C to 100 °C. The default

min temp is -40 °C, the default mid temp is 25 °C, and the default max temp is 80 °C. The resistance ranges are from 50 to 5.0E6 Ω. The default resistance for min temp is 332094 Ω, for mid temp 10000 Ω, and for max temp 1256 Ω.

Not all combinations of Temperature and Resistance produce a valid Steinhart-Hart equation. If the entered values produce an invalid equation, the following error is generated on the Wizard:



This should not occur when using reference values from a thermistor datasheet or when using accurately pre-measured values.

Steinhart-Hart Coefficients

The Steinhart-Hart Coefficients are calculated based on the three sets of temperature and resistance values entered for Min Temp, Mid Temp, and Max Temp. The Resistance values and Steinhart-Hart Coefficients are inter-dependent. If you change any of the resistance values, the Steinhart-Hart values get updated. Similarly, if you change the Steinhart-Hart values, the resistance values get updated.

Vref Pins

You can select Vref pins {P0[3] and P0[5]} or {P0[2] and P0[4]} for internally routed reference. This may be updated based on the placement of the user module. This information is provided in the UM wizard to give an idea of the combination of Vref pins and thermistor pins.

Thermistor Pin

The Thermistor Pin sets the pin to which the junction of the thermistor and reference resistor is connected.

RefLo Pin

The RefLo Pin is available to be routed only for the externally routed reference selection (RefLo = VSS).

Schematic

The schematic is updated with the "Reference Resistance", "Vref Pins", and the "Thermistor Pin" values, and the RefLo Pin if applicable.

Wizard Buttons

The user module Wizard exposes four buttons:

- OK - This button checks the wizard parameters settings. If the parameters are correctly defined, then the wizard stores the parameters and closes; if not, the wizard shows the appropriate warning message and does not close.
- Cancel - This button closes the wizard without saving the parameters.
- Close - This button closes the wizard without saving any parameters.
- Help - This button calls the Thermistor User Module Wizard Help. It describes the user module wizard features.

Parameters and Resources

After the user module is placed, these parameters must be configured for proper operation: Thermistor Pin, RefLo Pin, and Temperature Sample Rate.

Thermistor Pin

This parameter selects the pin to which the thermistor is connected. The available selection depends on the "Vref Pins" selection made in the MUM wizard. The range is from P0[0] to P0[7] and the default setting is P0[0].

RefLo Pin

This parameter selects the pin to which the RefLo is connected. The selection is available only when the configuration for the externally routed RefLo is selected. The available selection depends on the "Vref Pins" selection made in the Wizard. The port pins selected in the "Vref Pins" parameter are not listed in this parameter. The range is from P0[0] to P0[7] and the default setting is P0[5].

Temperature Sample Rate

This parameter sets the sample rate of the ADC. The range is from 20.3 (integer value = 20 sps) down to 1.0 sps. The sample rate is one-third of the ADC sample rate (60.1 sps) because three measurements are taken to compute temperature. The maximum column clock at which ADC is guaranteed to meet specs is 2.0 MHz. Optimum performance is expected when the sample rate is a sub-multiple of the local AC line frequency.

The ranges vary from 1 to 20. The default is the sample rate corresponding to three samples of a 13-bit converter with a maximum column clock of 2.0 MHz divided by 12 in PWM.

Application Programming Interface

The Application Programming Interface (API) routines are provided as part of the user module to allow the designer to deal with the module at a higher level. This section specifies the interface to each function together with related constants provided by the “include" files.

Note

In this, as in all user module APIs, the values of the A and X register may be altered by calling an API function. It is the responsibility of the calling function to preserve the values of A and X before the call if those values are required after the call. This “registers are volatile" policy was selected for efficiency reasons and has been in force since version 1.0 of PSoC Designer. The C compiler automatically takes care of this requirement. Assembly language programmers must ensure their code observes the policy, too. Though some user module API function may leave A and X unchanged, there is no guarantee they may do so in the future.

Thermistor_Start

Description:

Starts the Thermistor User Module resources. The Power setting is calculated based on Thermistor Sample Rate parameter in customizer and saved as a default value.

Thermistor Sample Rate	Symbolic Name	Default Value
≤ 5 sps	Thermistor_LOWPOWER	1
> 5 sps and ≤ 10 sps	Thermistor_MEDPOWER	2
> 10 sps	Thermistor_HIGHPOWER	3

C Prototype:

```
void Thermistor_Start (void)
```

Assembly:

```
lcall  Thermistor_Start
```

Parameters:

The power setting is calculated in the customizer and saved as a default value.

Symbolic Name	Value
Thermistor_LOWPOWER (sample rate ≤ 5 sps)	1
Thermistor_MEDPOWER (sample rates > 5 sps and ≤ 10 sps)	2
Thermistor_HIGHPOWER (sample rates > 10 sps)	3

Return Value:

None

Side Effects:

None.

Thermistor_SetPower

Description:

Changes the power setting of the user module from the default value set in the Thermistor_Start API.

C Prototype:

```
void  Thermistor_SetPower(BYTE bPowerSetting)
```

Assembly:

```
mov A, bPowerSetting
lcall  Thermistor_SetPower
```

Parameters:

bPowerSetting

Symbolic Name	Value
Thermistor_LOWPOWER	1
Thermistor_MEDPOWER	2
Thermistor_HIGHPOWER	3

Return Value:

None

Side Effects:

None.

Thermistor_Stop

Description:

Stops the Thermistor User Module resources.

C Prototype:

```
void  Thermistor_Stop (void)
```

Assembly:

```
lcall  Thermistor_Stop
```

Parameters:

None

Return Value:

None

Side Effects:

None.

Thermistor_StartConversion

Description:

This function initiates a thermistor temperature measurement. After it is initiated, the following functions are performed in the ISR: multiplexing and measurement of Vrefhi / Vth / Vreflo voltages. The ISR is implemented as a state machine to perform this sequence of operations.

C Prototype:

```
void Thermistor_StartConversion(void)
```

Assembly:

```
lcall Thermistor_StartConversion
```

Parameters:

None.

Return Value:

None

Side Effects:

None.

Thermistor_fIsDataAvailable

Description:

This function checks for completion of thermistor conversion.

C Prototype:

```
BYTE  Thermistor_fIsDataAvailable(void)
```

Assembly:

```
lcall  Thermistor_fIsDataAvailable
```

Parameters:

None

Return Value:

Returns 0 if conversion is not complete and 1 if conversion is complete.

Side Effects:

None.

Thermistor_iGetTemperature

Description:

Returns the temperature value from the last conversion.

C Prototype:

```
INT Thermistor_iGetTemperature(void)
```

Assembly:

```
lcall Thermistor_iGetTemperature
```

Parameters:

None

Return Value:

iTemperature in 0.01 °C.

Side Effects:

None.

Thermistor_GetResistance

Description:

Reads the resistance value of the thermistor in 0.1 Ohm resolution. This parameter may be used for calibration. Its use is internal and not normally required for delivering the temperature value.

C Prototype:

```
void Thermistor_GetResistance(DWORD * pdwResistance)
```

Assembly:

```
mov   A, >pdwResistance
mov   X, pdwResistance
lcall Thermistor_GetResistance
```

Parameters:

None

Return Value:

pdwResistance: Pointer to a buffer to hold the resistance value of the thermistor. The X register is loaded with the address of the return buffer.

Side Effects:

None.

Sample Firmware Source Code

The following C sample code demonstrates the Thermistor User Module functionality.

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all user modules

INT Temperature;
DWORD Resistance;

void main(void)
{
    M8C_EnableGInt;                          // Enable Global Interruptss
    Thermistor_Start();                      // Apply power to Thermistor
    Thermistor_StartConversion();           // Have Thermistor run

    for(;;)
    {
        while (!(Thermistor_fIsDataAvailable())); /* Loop until value ready */
        Temperature = Thermistor_iGetTemperature();/* get temperature */
        Thermistor_GetResistance(&Resistance); /* get resistance */

        /* Add user code here to use or display result */
    }
}
```

The following is the Assembly sample code:

```
;-------------------------------------------------------------------------
; Assembly main line
;-------------------------------------------------------------------------

include "m8c.inc"       ; part specific constants and macros
include "memory.inc"    ; Constants & macros for SMM/LMM and Compiler
include "PSoCAPI.inc"   ; PSoC API definitions for all User Modules

export _main
```

```
area bss (RAM,REL)
    Resistance:     BLK    4
    Temperature:    BLK    2

area text (ROM, REL)

_main:

    M8C_EnableGInt                    ; Enable Global Interrupts
    lcall Thermistor_Start            ; Apply power to Thermistor
    lcall Thermistor_StartConversion  ; Have Thermistor run continuously

terminate:
wait:
    lcall Thermistor_fIsDataAvailable ; Loop until value ready
    jz wait

    lcall Thermistor_iGetTemperature  ;  Get temperature
    mov [Temperature], X
    mov [Temperature + 1], A

    mov   A, >Resistance
    mov   X, Resistance
    lcall Thermistor_GetResistance    ;  Get resistance

    jmp terminate
```

Backward Compatibility

The Thermistor UM v1.00 remains during opening with PD5.4 without any updating.

The Thermistor UM v2.00 is backward incompatible through the regular UM update mechanism.

Version History

Version	Originator	Description
1.00	DHA	Initial version.
2.00	HPHA	1. Major rework to simplify User Module operation and usage. 2. Eliminated ADC features. 3. Added CY8C24x23A family support.

Note PSoC Designer 5.1 introduces a Version History in all user module datasheets. This section documents high level descriptions of the differences between the current and previous user module versions.

Copyright © 2012-2013 Cypress Semiconductor Corporation. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC Designer™ and Programmable System-on-Chip™ are trademarks and PSoC® is a registered trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.