Thermometer using a Thermistor with TWR-MCF51JF

by: Julio Ortega
Applications Engineer, Guadalajara

Contents

1 Introduction

This application note describes temperature control using any thermistor with the low-power feature of the MCF51JF128 ColdFire+ family of Freescale microcontrollers.

The ColdFire+ embedded controller ensures an optimal flow current and temperature measurement that maximizes the thermometer’s life. Allowing an efficient temperature measurement avoids the environmental changes that affect hardware components.

2 Objective

The purpose of this application note is to present a way of controlling temperature by using a thermistor as a sensor for low power consumption, and to introduce the features of the new MCF51Jx device family, including: ADC, LPTMR, GPIO modules, and STOP management.

3 Features

This application note assumes the following:

- The MCF51JF128 ColdFire+ is used as the core system
- Measures temperature from –25 °C to +40 °C in 0.1 °C
- Conversion and average of temperature
- Low-power solution
- Thermistor is used as a temperature sensor
- System connection

4 Thermistor description

Thermistors are thermally sensitive semiconductors whose resistance varies with temperature. Because thermistors are sensitive, they require an excitation source and later read the voltage through its terminals. This source should be constant and accurate. Figure 1 is an illustration of the symbol typically used to represent a thermistor.

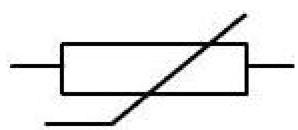

Figure 1. Common symbol for thermistors

The way to connect to a thermistor is shown in Figure 2. It explains the use of a voltage divider. The voltage difference in the resistor is read as the temperature.

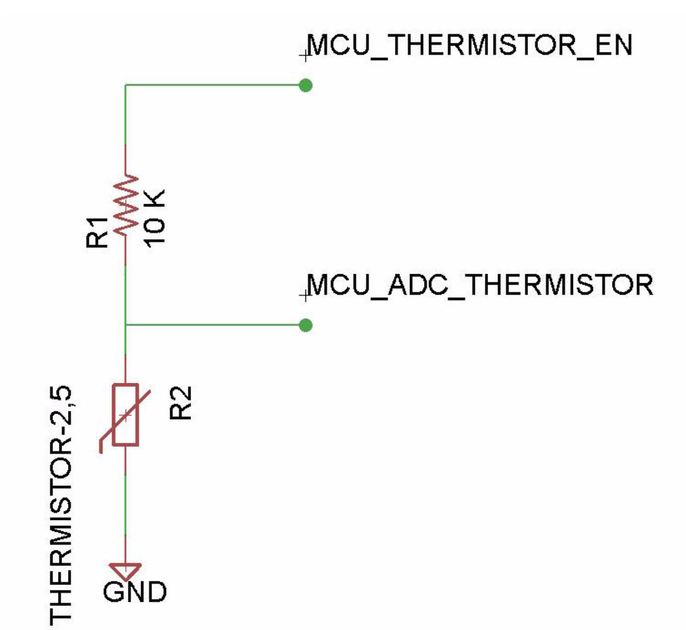

Figure 2. Thermistor connection diagram

The relationship between voltage in a resistor and temperature is not perfectly linear. Each thermistor has a list of coefficients provided by the temperature sensor manufacturer that includes the resistance value with respect to the temperature. But sometimes these values are not known; in these cases, it is possible to calculate coefficient values for a thermistor, that approximate the manufacturer's values.

The calculation of coefficients to the thermistor used for this application note is based on the theory of Scarr and Setterington (1960), who proposed the following equation:

$$R(T) = R_0 \cdot e^{\beta \cdot \left(\frac{1}{T} - \frac{1}{T_0} \right)} = R_0 \cdot e^{\beta \cdot \left(\frac{T_0 - T}{T \cdot T_0} \right)}$$

The values obtained with the equation above can be found in Appendix A.

5 Thermometer description

5.1 Block diagram

The temperature control shown in Figure 3 and the processor use the internal CLK to 10 MHz. The MCU is configured to work in Low-Power mode. For more detailed information on configuration modes, consult document number MCF51JF128RM, *MCF51JF128 Reference Manual*.

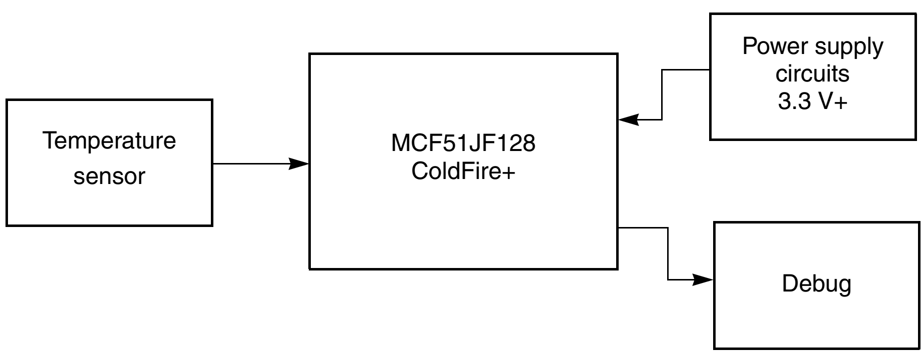

Figure 3. Thermometer block diagram

5.2 Firmware

The thermometer is initialized by configuring all peripherals and variables that will be used for the application: GPIO, ADC, MCG, and LPTMR, which is used for low power and delay for ADC conversions.

To measure temperature, the thermometer uses the circuit thermistor described above and the ADC. The Temp_Read.c module is the hardware abstraction layer (HAL) designed to configure the ADC. It provides functions to initialize the ADC, start conversion for any channel, and stop the ADC. The read data is stored in a buffer to later perform the temperature conversion in the hardware independent layer (HIL) block.

The thermometer application is designed in such a way that the MCU peripherals are independent of HIL modules.

The temperature measurement in the HIL block has the following functions:

- Enable thermistor circuit
- Read the ADC channel connected to the temperature sensor and convert this to a temperature value
- Perform the conversion and obtain the average temperature value and set the MCU in low-power Stop mode
- Wake the MCU from low-power and restart the temperature measurement

The flow chart in Figure 4 explains the function for the thermometer application process.

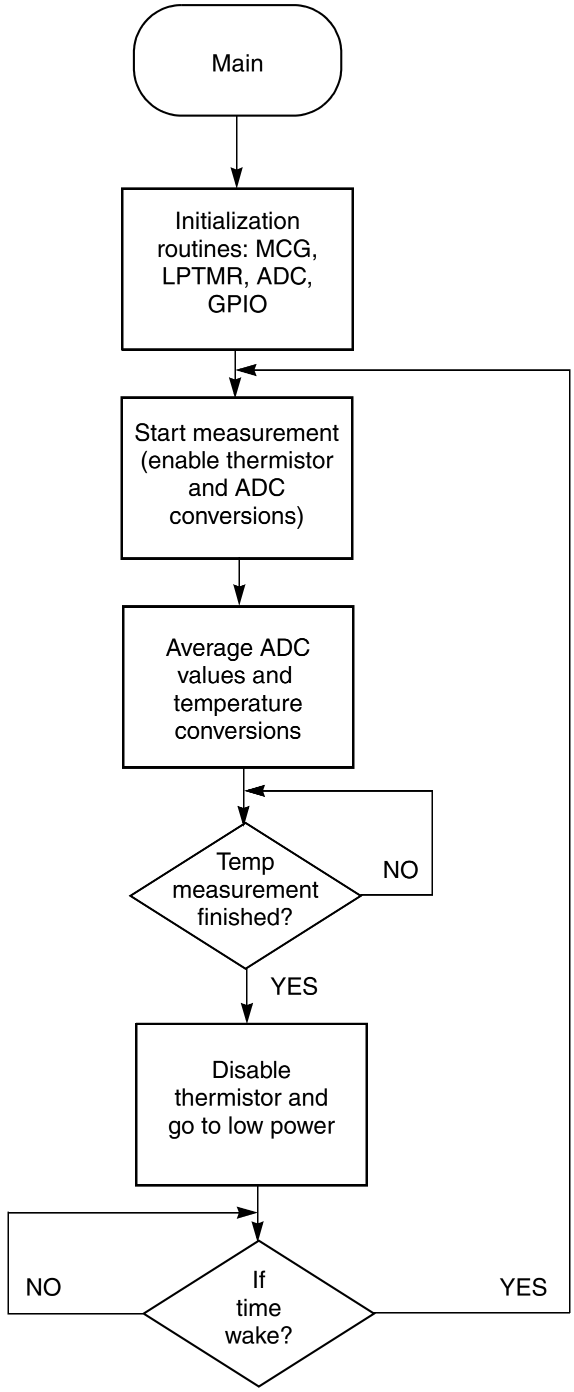

Figure 4. Thermometer software flowchart

5.3 Thermometer functions

- **void main(void){…}** ;

  This function contains the application startup. It configures MCU peripherals and initializes the state machine to the temperature measurement.
- **void LPT0_Init(int count){…}**;

  Low-power timer (LPTMR) registers are configured to operate as timer counter to 1 ms.
- **interrupt void lpt0_isr(void){...}**;

  Interruption used to wake the MCU out of low power once the LPTMR0_CNR reaches the module value.
- **void vfnRead_Temp(void){…}**;

  Called function that starts the temperature measurement each time, enables the thermistor, and performs the call for the temperature conversions.
- **unsigned int Read_Voltage_Thermistor(void){…}**;

  When a measurement is started, this function waits for ADC conversion and returns the value obtained.
- **void Init_ADC(void){…};**

  Enables the ADC module for the conversion process and configures the ADC channel.
- **void End_ADC(void){…};**

  Disables the ADC module and stops ADC conversions to save energy.
- **void Sensor_Read_Process(unsigned int u16VoltageThermistor){…};**

  This function makes an average process which helps to obtain a stable voltage measurement to later calculate the temperature.
- **void vfnCalculate_Temperature(void){…};**

  After determining the voltage across the resistor, this function controls the voltage-temperature conversion. The data obtained in the conversion is stored in a buffer that is averaged, resulting in the temperature value.
- **int i16fnConversionProcess(void){…};**

  This function compares the value of the thermistor voltage with a coefficients table and obtains the temperature through the mathematical method of interpolation. The calculated value is returned as temperature in °C.
- **void enter_stop(void){…};**

  It puts the processor into normal Stop mode. In this mode, core, bus, and peripheral clocks are disabled. Stop mode is exited using any enabled interrupt or RESET, so no exit_stop routine is needed.

5.4 Macros and variables

```
#define PD_Thermistor_OUT  PTA_DD |= 0x40
#define P_Thermistor_ON    PTA_D  |= 0x40
#define P_Thermistor_OFF   PTA_D  &= ~(0x40)

#define PD_ADC_OUT         PTD_DD |= 0x20
#define P_ADC_OFF          PTD_D  &= ~(0x20)

#define ADC_CHANNEL                  12

#define AVERAGE_DATA_MAX   16       /* average number data*/

#define ON     1
#define OFF    0

extern unsigned char  u8Data_ready;  /* variable used to know if thermometer process finished*/


#define LED1_ON    PTA_D &= 0xFE
#define LED2_ON    PTC_D &= 0xDF

#define LED1_OFF   PTA_D |= 0x01
#define LED2_OFF   PTC_D |= 0x20

#define LED1_TOG   if(PTA_D & 0x01) PTA_D &= 0xFE; else PTA_D |= 0x01
#define LED2_TOG   if(PTC_D & 0x20) PTC_D &= 0xDF; else PTC_D |= 0x20
```

6 Conclusions

On any electronic device, it is important to save the electrical current consumption. This application note uses normal Stop mode for low power. The power consumption was reduced to 50%, which is very favorable:

- Power consumption not using Stop mode: 20 mA
- Power consumption using Stop mode: 10 mA

If used, the different power modes (VLPS, VLLS3, LLS, and so on) give better energy-saving results.

Appendix A

Table 1. Thermistor-to-temperature conversion values

Temperature (°C)	Res Thermistor Ohms	Voltage (V)	MCU Values
45.0	2199	0.59	185
42.0	2460	0.65	202
40.0	2653	0.69	215
37.0	2976	0.76	235
35.0	3216	0.80	249
32.0	3619	0.88	272
30.0	3919	0.93	288
27.0	4423	1.01	314
25.0	4800	1.07	332
22.0	5435	1.16	361
20.0	5910	1.23	380
17.0	6714	1.33	411
15.0	7318	1.39	433
12.0	8343	1.50	466
10.0	9115	1.57	488
7.0	10428	1.68	523
5.0	11420	1.76	546
2.0	13115	1.87	581
0.0	14400	1.95	604
–3.0	16601	2.06	639
–5.0	18276	2.13	662
–8.0	21156	2.24	695
–10.0	23356	2.31	717
–13.0	27151	2.41	748
–15.0	30061	2.48	768
–18.0	35101	2.57	797
–20.0	38981	2.63	815
–23.0	45728	2.71	840
–25.0	50944	2.76	856

THIS PAGE IS INTENTIONALLY BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4456
Rev. 0
01/2012