Basic Refrigerator Control Using the MC9RS08KA2

by: Jose Ruiz
RTAC Americas

1 Introduction

Some refrigerators still have a basic electromechanical circuit that controls the temperature. This application shows how to implement a low-cost, basic temperature control for refrigerators using the MC9RS08KA2. This method can be implemented to control the temperature of any device using a thermoresistor, a potentiometer, resistors and a capacitor.

2 Requirements

- MC9RS08KA2 microcontroller (MCU)
- One potentiometer
- One thermoresistor
- One ceramic capacitor
- Two ¼ watt resistors
- CodeWarrior™ 5.1 development tool

Contents

1 Introduction ........................................ 1
2 Requirements ...................................... 1
3 Implementation .................................. 2
  3.1 Control Value ................................ 2
  3.2 Temperature Sensor ......................... 5
  3.3 Temperature Control Application ........ 9
  3.4 Schematic .................................. 12
4 Conclusion ...................................... 12
Appendix A ......................................... 13
  NCP18WB333J03RB Thermistor Range Table ... 13
Appendix B ......................................... 14
  Code Implementation .......................... 14

© Freescale Semiconductor, Inc., 2007. All rights reserved.
3 Implementation

The temperature control is implemented with a single potentiometer and a capacitor connected to one MC9RS08KA2 MCU I/O pin. The temperature sensor is a basic voltage divider formed by a resistor and a thermistor. The output is an I/O pin connected to a relay that switches the supply of the refrigerator.

The flow of the program consists of reading the control wheel value followed by reading the sensor voltage and, finally, switching the output ON or OFF according to the control and sensor values.

3.1 Control Value

The refrigerator temperature control is a basic RC network connected to an I/O pin. By measuring the charging time of the RC network, we can determine the potentiometer resistance, and therefore, the value you entered. The charge curve of the RC network is used to determine the time the curve takes to go from 0 V to the input-high voltage (\(V_{IH}\)). This method is used because the MC9RS08KA2 MCU does not have an integrated analog-to-digital converter (ADC).

![Figure 1. Temperature Control Implementation](image)

The first step is configuring the control pin as output. Set the pin value to 0 to discharge the capacitor. After the capacitor is fully discharged, change the control pin direction to an input. The capacitor starts charging to \(V_{DD}\).

When the voltage of the capacitor gets to \(V_{IH}\), the pin state changes from 0 to 1.

A variable resistor (potentiometer) is used to modify the time the capacitor takes to reach \(V_{IH}\). Adjusting its resistance varies that time.
The capacitor voltage is given by the following equation:

\[ V_c = V_{dd} \left( 1 - e^{-\frac{t}{rc}} \right) \]

Solving for time

\[ t = -\frac{1}{rc} \ln \left( 1 - \frac{V_c}{V_{dd}} \right) \]

- \( V_c \) — Voltage of the capacitor
- \( V_{DD} \) — Supply voltage of RC network
- \( t \) — Time (seconds)
- \( r \) — Resistance
- \( c \) — Capacitor

A 10 kΩ potentiometer and 33 nF capacitor were used in this application note.

From the MC9RS08KA2 datasheet, we know that when \( V_{DD} > 2.3 \) V, the \( V_{IH} \) for the inputs is 0.70 x \( V_{DD} \).
If the MC9RS08 MCU is supplied with 3.3 V then:

\[ V_{IH} = 0.70 \times V_{DD} = (0.70 \times 3.3) = 2.31 \text{ V} \]

Table 1 shows the difference in time using the above with different resistance commercial values.

<table>
<thead>
<tr>
<th>( V_{DD} )</th>
<th>( V_{IH} )</th>
<th>( R )</th>
<th>( C )</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3</td>
<td>2.31</td>
<td>1k</td>
<td>33nF</td>
<td>3.973E-05</td>
</tr>
<tr>
<td>3.3</td>
<td>2.31</td>
<td>3k</td>
<td>33nF</td>
<td>0.0001192</td>
</tr>
<tr>
<td>3.3</td>
<td>2.31</td>
<td>5k</td>
<td>33nF</td>
<td>0.0001987</td>
</tr>
<tr>
<td>3.3</td>
<td>2.31</td>
<td>7k</td>
<td>33nF</td>
<td>0.0002781</td>
</tr>
<tr>
<td>3.3</td>
<td>2.31</td>
<td>10k</td>
<td>33nF</td>
<td>0.0003973</td>
</tr>
</tbody>
</table>

Figure 3. Charge Curve with Different Resistor
3.1.1 Code Implementation

The first step is to configure the control pin as output, and setting a low level on it, (0). Then wait for the RC network to discharge completely;

```
Pin_Measure:
bset control,PTADD ; Set control pin as Output
bcclr control,PTAD ; Discharge RC network
clr ControlValue
lda #$FE
Discharge2:
dbnza Discharge2
```

The following step is to configure the control pin as input and increment a counter while pin state is 0:

```
bclr control,PTADD ; Set Control pin as Input
measure_pin:
inc ControlValue
brclr control,PTAD,measure_pin; Inc value while pin is in low state
rts
```

The `ControlValue` variable represents the time taken for the capacitor to reach $V_{IH}$.

After the pin reaches the high level, we know the approximate position of the potentiometer entered by the user.

3.2 Temperature Sensor

A basic voltage divider with one resistor and one thermoresistor is used to implement the temperature sensor. The thermoresistor resistance depends on the temperature. For each temperature, we have a different voltage in the divider. This value is effectively measured with the ADC implemented by software that uses one resistor, one capacitor, and the analog comparator included in the MC9RS08KA2 MCU.

The voltage divider is composed of the thermoresistor NCP18WB333J03RB and a 82 ohms resistor. It is better to have a big variation in the output voltage of the sensor with a little variation in the temperature.

The supply voltage of the RC network in this application note is 3.3 V and the output voltage of the sensor can be calculated with the next equation.

$$ V_{out} = V_{dd} \left( \frac{NTC}{NTC + R} \right) = 3.3 \left( \frac{NTC}{NTC + 82} \right) $$
According to the thermoresistor specifications, the resistor range is between 89.61 Ω to 116.16 Ω in a range of 4 °C to –0.5 °C (Section Appendix A, “NCP18WB333J03RB Thermistor Range Table”). With those values the following data is calculated:

<table>
<thead>
<tr>
<th>Temperature</th>
<th>NTC Value</th>
<th>Resistor</th>
<th>VDD</th>
<th>Sensor Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>– 1</td>
<td>119.11</td>
<td>82</td>
<td>3.3</td>
<td>1.9544677</td>
</tr>
<tr>
<td>– 0.5</td>
<td>116.16</td>
<td>82</td>
<td>3.3</td>
<td>1.9344368</td>
</tr>
<tr>
<td>0</td>
<td>113.21</td>
<td>82</td>
<td>3.3</td>
<td>1.9138005</td>
</tr>
<tr>
<td>0.5</td>
<td>110.26</td>
<td>82</td>
<td>3.3</td>
<td>1.8925309</td>
</tr>
<tr>
<td>1</td>
<td>107.31</td>
<td>82</td>
<td>3.3</td>
<td>1.8705985</td>
</tr>
<tr>
<td>1.5</td>
<td>104.36</td>
<td>82</td>
<td>3.3</td>
<td>1.8479717</td>
</tr>
<tr>
<td>2</td>
<td>101.41</td>
<td>82</td>
<td>3.3</td>
<td>1.824617</td>
</tr>
<tr>
<td>2.5</td>
<td>98.46</td>
<td>82</td>
<td>3.3</td>
<td>1.8004987</td>
</tr>
<tr>
<td>3</td>
<td>95.51</td>
<td>82</td>
<td>3.3</td>
<td>1.7755788</td>
</tr>
<tr>
<td>3.5</td>
<td>92.56</td>
<td>82</td>
<td>3.3</td>
<td>1.7498167</td>
</tr>
<tr>
<td>4</td>
<td>89.61</td>
<td>82</td>
<td>3.3</td>
<td>1.7231688</td>
</tr>
</tbody>
</table>

Instead of having an ADC module, the MC9RS08KA2 MCU has a basic ADC implemented by software using the analog comparator module. This software ADC is basically composed by a RC network and the analog voltage to be measured. The software measures the time taken by the RC network to reach the sensor input voltage. This ADC by software is fully detailed in the RS08 Quick Reference Guide (RS08QRUG). Download the document at http://www.freescale.com
The formula to calculate the time taken for the capacitor to charge is the same as the temperature control formula:

\[
V_c = V_{dd}\left(1 - e^{-\frac{t}{rc}}\right)
\]

Solving for time:

\[
t = -rc \ln\left(1 - \frac{V_c}{V_{dd}}\right)
\]

But, for the ADC by software the RC network is fixed. In this case, the resistor value is 10 kΩ The capacitor is 0.1 μF.

If the sensor values and the capacitor charging curve are graphed together the result is the time the RC network takes to reach the sensor output voltage.

Figure 5. Capacitor Charge Versus Sensor Output Voltage
Based on the bus speed (8 MHz for this application), it is effective to build a table with the timer value according to the sensor voltage.

To calculate the timer counts of each sensor voltage the next formula must be applied:

\[
\text{TimerCounts} = V_{IH} \times \left( \frac{\text{BusClock}}{\text{prescaler}} \right)
\]

### Table 3. Temperature, Sensor Output, and Microcontroller Counts

<table>
<thead>
<tr>
<th>Temperature</th>
<th>V sensor</th>
<th>(V_{IH}) Time</th>
<th>Timer counts (Bus/32)</th>
<th>Timer counts (Bus/32)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.5°C</td>
<td>1.93444</td>
<td>0.0008824</td>
<td>220.5889</td>
<td>110.2944</td>
</tr>
<tr>
<td>0°C</td>
<td>1.9138</td>
<td>0.0008674</td>
<td>216.8392</td>
<td>108.4196</td>
</tr>
<tr>
<td>0.5°C</td>
<td>1.89253</td>
<td>0.0008521</td>
<td>213.0323</td>
<td>106.5162</td>
</tr>
<tr>
<td>1°C</td>
<td>1.8706</td>
<td>0.0008367</td>
<td>209.1667</td>
<td>104.5833</td>
</tr>
<tr>
<td>1.5°C</td>
<td>1.84797</td>
<td>0.000821</td>
<td>205.2403</td>
<td>102.6201</td>
</tr>
<tr>
<td>2°C</td>
<td>1.82462</td>
<td>0.000805</td>
<td>201.2512</td>
<td>100.6256</td>
</tr>
<tr>
<td>2.5°C</td>
<td>1.8005</td>
<td>0.0007888</td>
<td>197.1975</td>
<td>98.59874</td>
</tr>
<tr>
<td>3°C</td>
<td>1.77558</td>
<td>0.0007723</td>
<td>193.0769</td>
<td>96.53846</td>
</tr>
<tr>
<td>3.5°C</td>
<td>1.74982</td>
<td>0.0007555</td>
<td>188.8873</td>
<td>94.44366</td>
</tr>
<tr>
<td>4°C</td>
<td>1.72317</td>
<td>0.0007385</td>
<td>184.6263</td>
<td>92.31315</td>
</tr>
</tbody>
</table>

### Code Implementation:

```assembly
ADC_Single_conversion:

; Discharge Capacitor

bset 1,PTADD
bclr 1,PTAD
lda #$FE
waste:
dbnz waste
mov #ACMP_ENABLE,ACMPSC ; ACMP Enabled
mov #MTIM_ENABLE,MTIMSC ; Timer Counter Enabled
wait ; Wait for Analog Comparator Interrupt
bset 4,MTIMSC ; Stop MTIM
lda MTIMCNT ; read counter timer value
sta ADCValue ; store counter value
mov #MTIM_STOP_RESET,MTIMSC ; Stop and reset counter

mov #HIGH_6_13(SIP1), PAGESEL
brset 3, MAP_ADDR 6(SIP1), Conv_OK ; branch if ACMP interrupt arrives
bra ADC_Single_conversion

Conv_OK:

mov #ACMP_DISABLED, ACMPSC ; ACMP Disabled, Clear Interrupt flag
rts
```

3.2.1 Code Implementation:
### 3.3 Temperature Control Application

The refrigerator’s temperature control has four positions, the range of each one is:

- Position 4: 0 °C – 1 °C
- Position 3: 1 °C – 2 °C
- Position 2: 2 °C – 3 °C
- Position 1: 3 °C – 4 °C

The control switches on the relay when the temperature is over range. It switches it off when the temperature reaches the window value.

Because of temperature inertia, the window temperature is 1.5 °C. Figure 6 shows the window and the values from it.

![Figure 6. Temperature Control Range](image)
For example, when the temperature position is 1, if the temperature is higher than 4 °C, the relay is closed, and the refrigerator compressor is on. Next, when the temperature reaches 2.5 °C, the application opens the relay and the compressor stops.

This guarantees that the temperature is stable for long periods of time between the ranges and, no matter what; the temperature is never more than 4 °C.

Each temperature limit can be easily changed in the definition part of the main code.

; Variable definitions
; Prescaler /64
TEMP1_ON SET 92
TEMP1_OFF SET 99
TEMP2_ON SET 97
TEMP2_OFF SET 103
TEMP3_ON SET 100
TEMP3_OFF SET 107
TEMP4_ON SET 105
TEMP4_OFF SET 111

The definition_ON is the value that closes the relay, and definition_OFF opens the relay. And the resolution of these values can be adjusted with the timer prescaler.

### 3.3.1 Code Implementation

```
;**************************************************************
;* Comparation (Control vs Temp)                              *
;**************************************************************
comparation:
    lda ControlValue
    cmp #65
    blo Temp1_4
    cmp #130
    blo Temp2_4
    cmp #195
    blo Temp3_4

    mov #04,ControlValue ; selector = 4 (Coldest)
    lda ADCValue
    cmp #TEMP4_ON
    blo Compressor_ON
    cmp #TEMP4_OFF
    bhs Compressor_OFF
    rts

Temp3_4:
    mov #03,ControlValue ; selector = 3 (Mid-Low)
    lda ADCValue
    cmp #TEMP3_ON
    blo Compressor_ON
    cmp #TEMP3_OFF
    bhs Compressor_OFF
    rts

Temp2_4:
```
Implementation

Basic Refrigerator Control Using the MC9RS08KA2, Rev. 0

Freescale Semiconductor
3.4 Schematic

Figure 7. Hardware Schematic

4 Conclusion

This application note shows how to implement a simple on-off control system with a low-end 8-bit microcontroller.
Appendix A  NCP18WB333J03RB Thermistor Range Table

<table>
<thead>
<tr>
<th>Temp (°C)</th>
<th>Resistance (K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-40</td>
<td>1227.263</td>
</tr>
<tr>
<td>-35</td>
<td>874.449</td>
</tr>
<tr>
<td>-30</td>
<td>630.851</td>
</tr>
<tr>
<td>-25</td>
<td>460.457</td>
</tr>
<tr>
<td>-20</td>
<td>339.797</td>
</tr>
<tr>
<td>-15</td>
<td>253.363</td>
</tr>
<tr>
<td>-10</td>
<td>190.766</td>
</tr>
<tr>
<td>-5</td>
<td>144.964</td>
</tr>
<tr>
<td>0</td>
<td>111.087</td>
</tr>
<tr>
<td>5</td>
<td>85.842</td>
</tr>
<tr>
<td>10</td>
<td>66.861</td>
</tr>
<tr>
<td>15</td>
<td>52.470</td>
</tr>
<tr>
<td>20</td>
<td>41.471</td>
</tr>
<tr>
<td>25</td>
<td>33.000</td>
</tr>
<tr>
<td>30</td>
<td>26.430</td>
</tr>
<tr>
<td>35</td>
<td>21.298</td>
</tr>
<tr>
<td>40</td>
<td>17.266</td>
</tr>
<tr>
<td>45</td>
<td>14.076</td>
</tr>
<tr>
<td>50</td>
<td>11.538</td>
</tr>
<tr>
<td>55</td>
<td>9.506</td>
</tr>
<tr>
<td>60</td>
<td>7.870</td>
</tr>
<tr>
<td>65</td>
<td>6.549</td>
</tr>
<tr>
<td>70</td>
<td>5.475</td>
</tr>
<tr>
<td>75</td>
<td>4.595</td>
</tr>
<tr>
<td>80</td>
<td>3.874</td>
</tr>
<tr>
<td>85</td>
<td>3.282</td>
</tr>
<tr>
<td>90</td>
<td>2.789</td>
</tr>
<tr>
<td>95</td>
<td>2.379</td>
</tr>
<tr>
<td>100</td>
<td>2.038</td>
</tr>
<tr>
<td>105</td>
<td>1.751</td>
</tr>
<tr>
<td>110</td>
<td>1.509</td>
</tr>
<tr>
<td>115</td>
<td>1.306</td>
</tr>
<tr>
<td>120</td>
<td>1.134</td>
</tr>
<tr>
<td>125</td>
<td>0.987</td>
</tr>
</tbody>
</table>
Appendix B  Code Implementation

INCLUDE 'derivative.inc' ; Include derivative-specific definitions

; export symbols
XDEF _Startup
ABSENTRY _Startup

; Variable declarations
ACMP_ENABLE SET $92
ACMP_DISABLED SET $20
MTIM_INIT SET $50
MTIM_ENABLE SET $40
MTIM_STOP_RESET SET $30
MTIM_64_DIV SET $06
FREE_RUN SET $00
DEBUG_MODE SET $00
RUN_MODE SET $01

control SET $04
output SET $05

TEMPl_ON SET 94
TEMPl_OFF SET 100
TEMPl2_ON SET 98
TEMPl2_OFF SET 104
TEMPl3_ON SET 102
TEMPl3_OFF SET 108
TEMPl4_ON SET 106
TEMPl4_OFF SET 112

MODE: EQU DEBUG_MODE

; variable/data section
ORG RAMStart

ADCValue: DS.B 1
counter DS.B 1
ControlValue DS.B 1

; code section
ORG ROMStart

;********************************************************************************
;*                               MACRO DECLARATION                                *
;********************************************************************************
TRIM_ICS: MACRO ; Macro used to configure the ICS with TRIM
mov #$FF,PAGESEL ; change to last page
lda #$FA ; load the content which TRIM value is store
tax ; move A content to X
lda ,x ; read D[X]
sta ICSTRM ; Store TRIM value
ENDM

ACK_RTI: MACRO
  mov #HIGH_6_13(SRTISC), PAGESEL
  bset 6, MAP_ADDR_6(SRTISC)
ENDM

;**************************************************************
;*            Comparation (Control vs Temp)                   *
;**************************************************************
comparation:
  lda ControlValue
  cmp #65
  blo Temp1_4
  cmp #130
  blo Temp2_4
  cmp #195
  blo Temp3_4

  mov #04, ControlValue ; selector = 4 (Coldest)
  lda ADCValue
  cmp #TEMP4_ON
  blo Compresor_ON
  cmp #TEMP4_OFF
  bhs Compresor_OFF
  rts

Temp3_4:
  mov #03, ControlValue; selector = 3 (Mid-Low)
  lda ADCValue
  cmp #TEMP3_ON
  blo Compresor_ON
  cmp #TEMP3_OFF
  bhs Compresor_OFF
  rts

Temp2_4:
  mov #02, ControlValue ; selector = 2 (Mid-High)
  lda ADCValue
  cmp #TEMP2_ON
  blo Compresor_ON
  cmp #TEMP2_OFF
  bhs Compresor_OFF
  rts

Temp1_4:
  mov #01, ControlValue; selector = 1 (Hot)
  lda ADCValue
  cmp #TEMP1_ON
  blo Compresor_ON
  cmp #TEMP1_OFF
  bhs Compresor_OFF
  rts
Conclusion

Compresor_ON: ; Compresor ON
   bset output,PTAD
   rts

Compresor_OFF: ; Compresor OFF
   bclr output,PTAD
   rts

;******************************************************************************
/* CONFIGURES SYSTEM CONTROL */
;******************************************************************************
Init_mc:
   mov #HIGH_6_13(SOPT), PAGESEL
   mov #$E3, MAP_ADDR_6(SOPT) ; Enable STOP mode and COP with long timeout period
   clr ICSC1 ; FLL is selected as Bus Clock
   TRIM_ICS
   clr ICSC2
   bset output,PTADD ; Enable PTA5 as output
   rts

;******************************************************************************
/* Modulus Timer Configuration for ADC */
;******************************************************************************
MTIM_ADC_Init:
   mov #MTIM_64_DIV,MTIMCLK ; Select bus clock as reference, Set prescaler with 64
   mov #FREE_RUN,MTIMMOD ; Configure Timer as free running
   mov #MTIM_STOP_RESET,MTIMSC
   rts

;******************************************************************************
/* ADC Single Conversion */
;******************************************************************************
ADC_Single_conversion:
   ; Discharge Capacitor
   bset 1,PTADD
   bclr 1,PTAD
   lda #$FE
   waste:
   dbnza waste

   ; Start Conversion
   mov #ACMP_ENABLE,ACMPSC ; ACMP Enabled, ACMP+ pin active, Interrupt enabled,
   Rising edges detections
   mov #MTIM_ENABLE,MTIMSC ; Timer Counter Enabled
   wait ; Wait for Analog Comparator Interrupt (match signals)
   bset 4,MTIMSC ; Stop MTIM
   lda MTIMCNT ; read counter timer value
   sta ADCValue ; store counter value
   mov #MTIM_STOP_RESET,MTIMSC ; Stop and reset counter

   mov #HIGH_6_13(SIPI1), PAGESEL
   brset 3, MAP_ADDR_6(SIPI1),Conv_OK ; branch if ACMP interrupt arrives
   bra ADC_Single_conversion

   ; Comparator Interrupt OK
Conv_OK:
    mov #ACMP_DISABLED, ACMPSC ; ACMP Disabled, Clear Interrupt flag
    rts

;********************************************************************************
; Control Value
;********************************************************************************
Pin_Measure:
    bset control,PTADD ; Set control pin as Output
    bclr control,PTAD ; Discharge RC network
    clr ControlValue
    lda #$FE
    Discharge2:
        dbnz Discharge2
        bclr control,PTADD ; Set Control pin as Input
    measure_pin:
        inc ControlValue
        bclr control,PTAD,measure_pin; Inc value while pin is in low state
        rts

;********************************************************************************
; RTI Module Configuration
;********************************************************************************
Init_RTI:
    mov #HIGH_6_13(SRTISC), PAGESEL
    mov #$37, MAP_ADDR_6(SRTISC) ; Enable RTI (1 sec period)
    rts

;********************************************************************************
; MAIN
;********************************************************************************
_Startup:
    bsr Init_mc
    bsr Init_RTI
    bsr MTIM_ADC_Init ; Configure MITM for ADC module
    ; Application Loop
    mainLoop:
        feed_watchdog ; Clear COP timer
        bsr ADC_Single_conversion ; ADC Conversion
        bsr Pin_Measure ; Control Measure
        jsr comparation ; Comparation
        stop ; Enter in STOP mode
        ACK_RTI ; Ack for RTI Interrupt
        bra mainLoop

;********************************************************************************
; Startup Vector
;********************************************************************************
ORG $3FFD
    JMP _Startup ; Reset
Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2007. All rights reserved.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to http://www.freescale.com/epp.