MC9RS08KA-Based Latch-Up Recovery Circuit

by: Martin Mienkina
Rožnov pod Radhoštěm
Czech Republic

1 Introduction

This application note describes the hardware, firmware, and basic functionality of the Latch-up Recovery Circuit based on the MC9RS08KA 8-bit microcontroller. Such circuits are required in many microcontroller applications where the internal watchdog timer is subject to failure, or where it is especially important to protect an application's microcontroller against performance degradations. The MC9RS08KA1CSC processor has been selected because of its low cost, wide supply range (1.8–5.5 V), small package size (6.2 × 5 mm), and sufficient computational and on-chip peripheral performance when clocked from the internal 31.25 kHz clock source.

Microcontroller failure can take many forms, with either a temporary or permanent impact on device performance. The failure modes for integrated circuits are classified into one of five categories as specified in IEC 62132-1.¹ The classification is determined by the performance of the integrated circuit in

1. For further information on this topic, see AN2764,
   Improving the Transient Immunity Performance of
   Microcontroller-Based Applications.
the presence of the ESD or EFT signal. Freescale’s interpretation of the IEC failure modes, and those performance classes specific to microcontrollers, is shown in Table 1.

Table 1. Freescale classification of microcontroller performance degradation

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Normal performance within the specification limits during application of the transient.</td>
</tr>
<tr>
<td>B</td>
<td>Temporary degradation or loss of function or performance that is self-recoverable after the transient is removed. Device returns to normal performance.</td>
</tr>
<tr>
<td>C</td>
<td>Temporary degradation or loss of function or performance that requires an external reset after the transient is removed. Device returns to normal performance.</td>
</tr>
<tr>
<td>D</td>
<td>Temporary degradation or loss of function or performance that requires that power be cycled after the transient is removed. Device returns to normal performance.</td>
</tr>
<tr>
<td>E</td>
<td>Permanent degradation or loss of function that is not recoverable due to damage or loss of data.</td>
</tr>
</tbody>
</table>

Common forms of microcontroller temporary degradation include, but are not limited to: internal reset, latch-up, memory corruption, and code runaway. Microcontrollers with internal reset circuits can generally resume operation without operator involvement if the fault is a software failure, an unexpected reset, or a code runaway that is caught by the watchdog timer. A transient event such as a Powered ESD, EFT, or overshoot or undershoot on a pad, may cause failures of the state machines and nodes of the microcontroller critical to reset recovery or CMOS latch-up. These particular failures are only reset by a Power-on Reset (POR), so if state machines and nodes critical to reset recovery fail, or if the device is in a CMOS latch-up state, characterized by a high sustained current, the device’s only recovery is to reduce the voltage below the POR threshold, usually by removing power from the device. The Latch-up Recovery Circuit has been designed to guarantee the successful recovery of the monitored processor from a latch-up state. In general, it should be used in any safety-critical application where achieving a higher reliability of microcontroller operation is necessary.

2 Safety-critical applications

Figure 1 shows a typical signal interconnection of the Monitored Processor with the Latch-up Recovery Circuit in a safety-critical application. The Latch-up Recovery Circuit is intended to recover the Monitored Processor from the failure modes Classes B, C, and D by reducing the supply voltage below the POR threshold (power cycling). In addition, it improves the reliability of the application by its external reset and supply monitoring functions.

![Figure 1. Typical safety-critical application](image)

The Latch-up Recovery Circuit is based on the MC9RS08KA1 microcontroller. This standalone circuit enhances the reliability of the microcontroller-based application by providing external monitoring and recovery functions. The circuit is comprised of two input control signals which are supposed to be handled deterministically by the Monitored Processor. The first signal, the watchdog input (WDI), must be toggled periodically and any failure to change the state of the signal prior to a

2. For more information on this chip, see MC9RS08KA2, MC9RS08KA2, MC9RS08KA1 Data Sheet.

MC9RS08KA-Based Latch-Up Recovery Circuit, Rev. 0, 08/2012

Freescale Semiconductor, Inc.
watchdog timeout, "tWD," will always result in a processor POR. The second signal, the manual reset input (MR), can be directly attached to a push-button or a processor open-drain output pin. Whenever the MR pin is pulled low, the Monitored Processor becomes unpowered.

Besides the mentioned recovery functions, the Latch-up Recovery Circuit continuously monitors the supply voltage (VDD) and keeps the Monitored Processor unpowered until that voltage reaches the MC9RS08KA1 threshold.

3 Description

The figure below shows a block diagram and the signal interconnections of the MC9RS08KA1 Based Latch-up Recovery Circuit. The Freescale MC9RS08KA1CSC microcontroller (U1), P-channel MOSFET (Q1) and N-channel MOSFET (Q2) are the main components of the circuit (for a complete component list, refer to the schematic diagram in Board Schematics and the bill of materials in Board Bill of Materials). The functions of the pins are listed in Table 2.

![Block diagram](image)

**Figure 2. Block diagram of the MC9RS08-based Latch-up Recovery Circuit**

**Table 2. Latch-up Recovery Circuit pin functions**

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESET/VPP</td>
<td>Reserved for device programming.</td>
</tr>
<tr>
<td>BKGD/MS</td>
<td>Reserved for device programming.</td>
</tr>
<tr>
<td>VDD</td>
<td>Power supply from 1.8 to 5.5 V. Integrates a 10μF bulk capacitor (C1) to supply instantaneous energy to the external watchdog circuit, and a 0.1μF bypass ceramic capacitor, located as close to the MC9RS08KA1 power pins as practical, to suppress high-frequency noise. The integrated 390Ω resistor (R6) improves the immunity of the microcontroller device against voltage transients.</td>
</tr>
<tr>
<td>VSS</td>
<td>Ground</td>
</tr>
<tr>
<td>PWR</td>
<td>Power supply for the monitored processor.</td>
</tr>
<tr>
<td>MR</td>
<td>Manual Reset Input. Pulled high by the MC9RS08KA1 internal pullup resistor (~45kΩ). This pin is designed to be driven low externally either by a mechanical push-button or an open drain output.</td>
</tr>
<tr>
<td>WDI</td>
<td>Watchdog Input to be driven by a Monitored Processor output. The monitoring function may be disabled (watchdog function blocked) by tristating or leaving this pin open. The watchdog timeout is defined by the “tWD” parameter and equals 500 ms by default.</td>
</tr>
</tbody>
</table>

Table 3 below describes the logic dependencies of the internal signals on the control inputs. Note that pin PTA5 of the MC9RS08KA1 is configured either in input mode (with an internal ~45kΩ pullup resistor) or in push-pull output mode. The input mode is active whenever a Manual Reset Input (MR) and Watchdog Input (WDI) are detected. The latter, push-pull output mode is applied to detect the tristate mode of the Watchdog Input (WDI). Note that whenever the WDI pin is in...
tristate mode, and the “WDI pullup” jumper is also opened, the watchdog timer logic is disabled. It is recommended that you close this jumper during normal operation in order to keep the watchdog logic permanently enabled and running. On the other hand, leaving this jumper opened during software upload, debugging, and low power mode testing might be advantageous in preventing power cycles when either the WDI pin is in tristate mode or not handled by the Monitored Processor.

Table 3. Truth table

<table>
<thead>
<tr>
<th>State</th>
<th>Control inputs</th>
<th>MC9RS08KA1 I/O signals</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WDI</td>
<td>MR</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>tristate</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>open</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>close</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>x</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>tristate</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>open</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>close</td>
</tr>
</tbody>
</table>

Watchdog and manual reset logic—PTA5 configured to input mode with an internal pullup

Watchdog disable (WDI tristate detection)—PTA5 configured in push-pull mode

Figure 3 shows the logic tasks executed by the Latch-up Recovery Circuit after a Power-on Reset. The firmware execution starts with a power-up sequence that executes after a POR or COP reset. Following the initial power-up sequence, the main firmware loop starts to execute. Two logic functions, Manual Reset logic and watchdog timer logic, run inside the main loop on a periodic basis. A detailed description of the firmware initial sequence and logic functions is given in the following sections.
### 3.1 Power-up sequence

When the supply voltage (VDD) is initially applied to the Latch-up Recovery Circuit (MC9RS08KA1), or when the supply voltage (VDD) drops below the VPOR, typically 1.4 V, the internal power-on reset circuit will cause a reset condition. As the supply voltage (VDD) rises, the internal low-voltage detection circuitry still holds the device in reset until the supply has risen above the $V_{LVD}$, typically 1.95 V. Until the supply voltage (VDD) rises to the $V_{LVD}$ level, typically 1.95 V, the microcontroller pins are in tristate mode. Whenever PTA0 and PTA1 are tristated, the external pullup resistor (R1) and external pulldown resistor (R2) cause the P-channel (Q1) and N-channel (Q2) MOSFETs to be turned off; thus, the supply voltage (PWR) for the Monitored Processor is not applied (see Figure 4).

When the supply voltage (VDD) rises above the $V_{LVD}$ trip point, typically 1.95 V, the system reset terminates and the CPU starts to execute the initial power-up sequence. Firstly, the startup code executes and all necessary on-chip peripherals are initialized (see On-chip peripheral initialization (MCU_Init.c) for additional information). Secondly, firmware processes the Manual Reset Input (MR). If MR input is pulled high, the software execution stalls until the Modulo Timer (MTIM) reaches a user defined power-off timeout, “tPO.” Finally, the supply voltage (PWR) is applied to the Monitored Processor by turning on the P-channel (Q1) and turning off the N-channel (Q1) MOSFETs.

---

3. For further information, see MC9RS08KA2, MC9RS08KA2, MC9RS08KA1 Data Sheet and AN3394, Resetting MC9RS08KA During Power Transitions.
3.2 Manual Reset logic

The Latch-up Recovery Circuit has a Manual Reset (MR) input to allow for alternative control of the P-channel and N-channel MOSFETs. The MR input is designed for direct connection to a push-button (see Figure 1). The MR input is internally pulled up by a ~45kΩ resistor and must be pulled low to cause the P-channel MOSFET to turn off. Internally, this input is timed such that the power supply for the Monitored Processor (PWR) is not applied for a minimum power-off timeout, “tPO.” This power-off timeout, “tPO,” is user-programmable and may be set from 16 to 2000 ms (8 ms resolution). The circuit interaction when the MR control input is pulled low by pressing a push-button is shown in the figure below.

![Figure 5. Manual reset](image)

3.3 Watchdog timer logic

The watchdog timer logic forces the P-channel MOSFET to turn off whenever the WDI input does not have a transition from low-to-high, or high-to-low, within the watchdog timeout, “tWD.” The watchdog timer logic starts to operate immediately after the Power-up sequence (see Power-up sequence). If a transition occurs on the WDI input pin prior to the watchdog timeout, “tWD,” the watchdog timer resets and a new counting begins. When the watchdog timer overflows, then the P-channel MOSFET is turned off for a minimum power-off timeout, “tPO.” The watchdog timeout, “tWD,” is user-programmable and may be set from 16 to 2000 ms (8 ms resolution).

Figure 6 shows the WDI control signal transitions in a typical safety critical application (see Figure 1). Any processor signal that repeats dependant on the normal operation of the processor, or that is directed by the software operating on the processor, can be used to strobe the watchdog input (WDI). The most reliable is a dedicated I/O output transitioned by a specific software instruction.
The watchdog timer logic can be disabled by placing the WDI input in tristate mode while leaving the “WDI pullup” jumper opened. The WDI pin is periodically pulled high/low via resistor R3 by the PTA5 pin, which is reconfigured by the firmware to the push-pull output mode for that specific purpose. If the Monitored Processor places the WDI signal into tristate mode, then the WDI pin tightly follows the states of the PTA5 pin, which is pulled low, then pulled high and low again by the firmware. Such a tight coupling of the PTA4 (WDI) input states with the PTA5 forced state transitions shows that the WDI pin is in tristate mode and, thus, the watchdog timer logic may be disabled. This sequence runs periodically but the tristate mode is not detected when the WDI input is driven high or low by the Monitored Processor and/or the “WDI pullup” jumper is closed (see Table 3, states 8–12).

4 Firmware application

The application firmware was developed using the CodeWarrior Development Studio for Microcontrollers V6.3. The high level language (C-language) and also the Device Initialization Tool were used intentionally for firmware development to make the source code simple, understandable, and easy to modify. The source code for initialization of all on-chip peripherals is in the MCU_Init.c module (see On-chip peripheral initialization (MCU_Init.c)). The source code of all the logic functions is placed in the main.c module (see Main module (main.c)).

As already indicated, you can modify the power-off, “tPO,” and watchdog “tWD” timeout constants directly in the source code (main.c module). Any update should be made by modifying the corresponding timeout definition and by rebuilding the project.

```c
#define tPO 1.0 /* Power-off timeout */
#define tWD 0.5 /* Watchdog timeout */
```

The simple watchdog logic, efficient coding, and the basic startup routine help to achieve a highly optimized code that runs from the internal relaxation oscillator (IRC), consuming just 481μA@3.3V. The IRC oscillator is trimmed at the factory to 31.25 kHz. The whole application firmware occupies 210 bytes of P-Flash and 3 bytes of data RAM.

5 Operating characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply voltage (limited by MOSFET threshold voltage)</td>
<td>(V_{DD})</td>
<td>2.3</td>
<td>5.0</td>
<td>5.5</td>
<td>V</td>
</tr>
<tr>
<td>Supply current ((V_{DD} = 3.3V, TA = 25°C, P-channel MOSFET off))</td>
<td>(I_{CC})</td>
<td>—</td>
<td>—</td>
<td>481</td>
<td>μA</td>
</tr>
<tr>
<td>Continuous drain current</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>— BSS 84 MOSFET</td>
<td>(I_{PWR})</td>
<td>—</td>
<td>—</td>
<td>0.13</td>
<td>A</td>
</tr>
<tr>
<td>— NTR0202PL MOSFET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power-on reset (POR) voltage</td>
<td>(V_{POR})</td>
<td>0.9</td>
<td>1.4</td>
<td>1.7</td>
<td>V</td>
</tr>
</tbody>
</table>

*Table continues on the next page...*
### Conclusion

This application note describes the Latch-up Recovery Circuit, which was designed based on the Freescale MC9RS08KA1CSC 8-bit microcontroller. The standalone Latch-up Recovery Circuit provides additional safety for a microcontroller application along with the ability to recover from latch-up states by disconnecting the power. This particular design should be considered complementary to the standard external watchdogs and voltage reset circuits provided by other vendors that are capable of monitoring microcontroller operation but that don’t allow for flexible timeout settings. Note that standalone watchdog circuits don’t integrate power MOSFETs for power cycling, which is necessary to recover the monitored processor from a latch-up condition.

The application firmware, hardware design files, and bill of materials are delivered together with this application note. In addition to the binary output code, the application source files and all necessary project and configuration files are also provided. With the help of the complete project, you can easily modify the configuration parameters in the source files, rebuild the project, and generate output code in the preferred binary output format. In this way, the application firmware can be tailored to specific application needs. Moreover, the Latch-up Recovery Circuit can be enhanced via firmware in many directions. With firmware modification, one could achieve a better resolution of the timeouts and watchdog update period, different control signal polarities, or even a different logic and timing function, such as a windowed watchdog, and so on.

### References

<table>
<thead>
<tr>
<th>Document number</th>
<th>Title</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN2764</td>
<td>Improving the Transient Immunity Performance of Microcontroller-Based Applications</td>
<td><a href="http://www.freescale.com">www.freescale.com</a></td>
</tr>
<tr>
<td>AN3394</td>
<td>Resetting MC9RS08KA During Power Transitions</td>
<td></td>
</tr>
<tr>
<td>MC9RS08KA2</td>
<td>MC9RS08KA2, MC9RS08KA1 Data Sheet</td>
<td></td>
</tr>
</tbody>
</table>
Appendix A Board Photo

MC9RS08KA1CSC

Debug Interface

P-channel MOSFET
BSS84

N-channel MOSFET
BSS138

Monitored Processor
Power

Supply voltage
(2.3-5.5 V)

Manual reset
push-button

Control signals (MR, WDI)

Figure A-1. MC9RS08KA1 Latch-up Recovery Circuit board photo

Appendix B Board Schematics

Figure B-1. MC9RS08KA1 Latch-up Recovery Circuit board schematics
### Appendix C Board Bill of Materials

**Table C-1. MC9RS08KA1 Latch-up Recovery Circuit bill of materials**

<table>
<thead>
<tr>
<th>Item</th>
<th>Quant.</th>
<th>Ref.</th>
<th>Part</th>
<th>PCB footprint</th>
<th>Farnell number</th>
<th>Part number</th>
<th>Manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>C1</td>
<td>10UF/10V</td>
<td>TANT_A</td>
<td>1794696</td>
<td>T491A106K010AT</td>
<td>KEMET</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>C2</td>
<td>MLCC, 0805, Y5V, 50V, 100NF</td>
<td>C_0805</td>
<td>1759266RL</td>
<td>MCCAO00387</td>
<td>MULTICOMP</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>JP1</td>
<td>Power supply connector</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>JP2</td>
<td>MCU supply connector</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>J1</td>
<td>Debug header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>J2</td>
<td>MCU control header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>J3</td>
<td>WDI pullup header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>Q1</td>
<td>MOSFET, P CH, 50V, 0.18A, BSS84</td>
<td>SOT-23</td>
<td>1972673</td>
<td>BSS84AK</td>
<td>NXP</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>Q2</td>
<td>RESISTOR, THICK FILM, 390OHM, 125mW, 5%</td>
<td>R_0805</td>
<td>2053833</td>
<td>BSS138BK</td>
<td>NXP</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>R1, R2, R4, R5</td>
<td>RESISTOR, RC11 0805</td>
<td>SOT-23</td>
<td>1566303</td>
<td>CRCW080510K0JN</td>
<td>VISHAY DALE</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>R3</td>
<td>RESISTOR, RC11 0805</td>
<td>R_0805</td>
<td>9234322</td>
<td>RC0805JR-07390KL</td>
<td>YAGEO (PHYCOMP)</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>R6</td>
<td>RESISTOR, THICK FILM, 390OHM, 125mW, 5%</td>
<td>R_0805</td>
<td>1514861</td>
<td>CRCW0805390RJN</td>
<td>VISHAY DALE</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>SW1</td>
<td>SWITCH, SMD, PUSH, 5MM T/R</td>
<td></td>
<td>2082886</td>
<td>1301.9315.24</td>
<td>SCHURTER</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>U1</td>
<td>MC9RS08KA1CSC - IC, 8BIT MCU, 1K FLASH, ACMP, SOIC8</td>
<td>8-pin NB-SOIC</td>
<td>1296221</td>
<td>MC9RS08KA1CSC</td>
<td>FREESCALE</td>
</tr>
</tbody>
</table>
Appendix D Board Layout

Figure D-1. MC9RS08KA1 Latch-up Recovery Circuit top view

Figure D-2. MC9RS08KA1 Latch-up Recovery Circuit bottom view
Figure D-3. MC9RS08KA1 Latch-up Recovery Circuit top layer

Figure D-4. MC9RS08KA1 Latch-up Recovery Circuit bottom layer
Appendix E On-chip peripheral initialization (MCU_Init.c)

/*
** ###########################################################################
**     This code is generated by the Device Initialization Tool
**     It is overwritten during code generation.
**     USER MODIFICATION ARE PRESERVED ONLY INSIDE INTERRUPT SERVICE ROUTINES
**     OR EXPLICITLY MARKED SECTIONS
**
**     Project   : RS08_WatchDog
**     Processor : MC9RS08KA1CSC
**     Version   : Component 01.069, Driver 01.03, CPU db: 3.00.020
**     Datasheet : MC9RS08KA2 Rev. 3 9/2007
**     Date/Time : 6/2/2012, 8:01 AM
**     Abstract  :
**         This module contains device initialization code
**         for selected on-chip peripherals.
**     Contents :
**     Copyright : 1997 - 2009 Freescale Semiconductor, Inc. All Rights Reserved.
**     http      : www.freescale.com
**     mail      : support@freescale.com
** ###########################################################################
*/

#include <MC9RS08KA2.h>                /* I/O map for MC9RS08KA1CSC */
#include "MCU_Init.h"

void MCU_init(void)
{
    /* ** Common initialization of the write once registers */
    /* SOPT: COPE=1,COPT=0,STOPE=0,BKGDPE=1,RSTPE=1 */
    SOPT = 131;
    /* SPMSC1: LVDF=0,LVDACK=0,LVDIE=0,LVDRE=1,LVDSE=1,LVDE=1,BGBE=0 */
    SPMSC1 = 28;
    /* System clock initialization */
    if (*(unsigned char* __paged)CONVERT_TO_PAGED(16378) != 255) { /* Test device trim*/
        ICSTRM = *(unsigned char* __paged)CONVERT_TO_PAGED(16378);   /* Initialize ICSTRM 
register from a non volatile memory */
        ICSSC = *(unsigned char* __paged)CONVERT_TO_PAGED(16379);    /* Initialize ICSSC 
register from a non volatile memory */
    }
    /* ICSC1: CLKS=1,IREFSTEN=0 */
    ICSC1 = 64;                          /* Initialization of the ICS control register 1 */
    /* ICSC2: BDIV=0,LP=1 */
    ICSC2 = 8;                           /* Initialization of the ICS control register 2 */
    /* Common initialization of the CPU registers */
    /* PTASE: PTASE5=1,PTASE4=1,PTASE3=1,PTASE1=1,PTASE0=1 */
    PTASE |= (unsigned char)59;
}

Freescale Semiconductor, Inc.
/* ### Init_GPIO init code */
/* PTAD: PTAD5=0,PTAD4=0,PTAD1=1,PTAD0=0 */
PTAD = (PTAD & (unsigned char)~49) | (unsigned char)2;
/* PTAPUD: PTAPUD5=0,PTAPUD4=0,PTAPUD1=0,PTAPUD0=0 */
PTAPUD &= (unsigned char)~51;
/* PTAPPE: PTAPPE5=1,PTAPPE4=0,PTAPPE1=0,PTAPPE0=0 */
PTAPPE = (PTAPPE & (unsigned char)~19) | (unsigned char)32;
/* PTADD: PTADD5=0,PTADD4=0,PTADD1=0,PTADD0=0 */
PTADD &= (unsigned char)~51;
/* ### Init_COP init code */
SRS = 255;                           /* Clear WatchDog counter */
/* ### Init_MTIM init code */
/* MTIMMOD: MOD7=1,MOD6=1,MOD5=1,MOD4=1,MOD3=1,MOD2=1,MOD1=1,MOD0=1 */
MTIMMOD = 255;
/* MTIMCLK: CLKS1=0,CLKS0=0,PS3=0,PS2=1,PS1=1,PS0=1 */
MTIMCLK = 7;
(void)(MTIMSC == 0);                 /* Overflow int. flag clearing (first part) */
/* MTIMSC: TOF=0,TOIE=0,TRST=1,TSTP=0 */
MTIMSC = 32;                         /* Int. flag clearing (2nd part) and timer setting */
/* ### */
} /*MCU_init*/

/* Initialization of the reset vector */
extern void _Startup(void);

static const unsigned char JMPOpcode             @16381 = 188; /* Opcode of JMP extended */
static void far (*const ResetVector)(void)       @16382 = _Startup;

/* Initialization of the CPU registers in FLASH */
/* NVOPT: SECD=1 */
const unsigned char NVOPT_INIT @0x00003FFC = 255;

/* END MCU_Init */
/

全明星 Main module (main.c)

.TODO- Copyright 2012, Freescale Semiconductor Inc.
GRAM RIGHTS RESERVED.

******************************************************************************/

* @file   main.c
* @author R55013
* @version 1.0.1.0
* @date    Jun-4-2012
* @brief   MC9RS08KA1 Safety Watchdog.

#include <hidef.h> /* for EnableInterrupts macro */
#include "derivative.h" /* include peripheral declarations */
#pragma MESSAGE DISABLE C4000

******************************************************************************/

#define tPO 1.0 /* Power-off timeout */
#define tWD 0.5 /* Watchdog timeout */

/************ Module macros/constants definitions  ************/
#define MTIM_CLKSRC (float)31.25e3 /* factory trimmed */
#define MTIM_DIVIDE 128.0
#define MTIM_MOD(sec) ((float)sec/(2.0/MTIM_CLKSRC*MTIM_DIVIDE))
#define MTIM_RST() { MTIMSC_TRST = 1; }

/************************ WDI & /MR signal macros *****************************/
#define GET_WDI_STATE() PTAD_PTAD4
#define GET_MR_STATE() PTAD_PTAD5
#define SET_MR_AS_INP() { PTADD_PTADD5=0; }
#define SET_MR_AS_OUT(state) { PTAD_PTAD5=state; PTADD_PTADD5=1; }

/************************** Macros for enable/disable output power *****************************/
#define POWER_EN() { PTADD_PTADD1=0; PTADD_PTADD0=1; }
#define POWER_DI() { PTADD_PTADD0=0; PTADD_PTADD1=1; }

/************************** External function declarations *****************************/
extern void MCU_init(void); /* Device initialization function declaration */

/************************** Main function definition *****************************/
void main(void)
{
    register unsigned char tmp = TRUE;

    /******************************************************************************************
    * Power-up sequence  ***********************************************************************
    */
    MCU_init(); /* function generated by Device Initialization tool */
    POWER_DI(); /* power off external MCU */

    /************************************************************************* An external MCU will be left powered off until /MR signal is pulled high */
    while (!GET_MR_STATE())
    { __RESET_WATCHDOG(); }

    /************************************************************************* Power up sequence - count down tPO (Power off Timeout Period) with */
    while (MTIMCNT < MTIM_MOD(tPO)) { __RESET_WATCHDOG(); }
    POWER_EN(); /* power up an external MCU */
    MTIM_RST(); /* initialize timer */

    /************************************************************************ Periodic sequence - ~5.12 ms period ***************************************************************************/
    while (TRUE)
    {
        /************************************************************************* WDI tristated mode check - the routine toggles /MR pin and response */
        /************************************************************************* of the WDI signal state is measured. If WDI signal state follows /MR */
        /************************************************************************* signal states the WDI signal is tristated and timer is initialized */
        SET_MR_AS_OUT (FALSE);
        if (!GET_WDI_STATE ())
        {
            SET_MR_AS_OUT (TRUE);
            if (!GET_WDI_STATE ()
            {
                SET_MR_AS_OUT (FALSE);
                if (!GET_WDI_STATE ()
                {
                    MTIM_RST ();
                }
            }
        }
    }
SET_MR_AS_INP();

/* /MR signal processing - power off external MCU and reset device if */
/* signal pulled low                                                      */
if (!GET_MR_STATE()) { POWER_DI(); while (TRUE); }

/* WDI signal processing - reinitialize timer if signal change detected */
if (GET_WDI_STATE() != tmp) { tmp = GET_WDI_STATE(); MTIM_RST(); }

/* Timer processing - if timer >= tWD then power off an external MCU and */
/* perform power up sequence to recover external MCU from latch-up state */
if (MTIMCNT > MTIM_MOD(tWD)) { POWER_DI(); while (TRUE); }

/* Watchdog timer must be refreshed within 32 ms period */
__RESET_WATCHDOG();

**********************************************************************************
* End of module
**********************************************************************************
Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductors 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 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 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 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 claims alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

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.

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

© 2012 Freescale Semiconductor, Inc.