Programmable System Wake-up Controller Using MSP430™ MCUs

Introduction

The programmable system wake-up controller function of the MSP430FR2000 microcontroller (MCU) offers a simple way to add an external, real-time, and low-power wake-up controller to an existing system. This type of system wake-up controller is useful to applications that need to stay in low-power modes for variable extended periods of time. To get started, download project files and a code example demonstrating this functionality. For a similar application, but with consistent wake-up time, see *Simple RTC-Based System Wake-up Controller Using MSP430™ MCUs*.

Implementation

A low-frequency 32.768-kHz crystal is required for this application. Alternatively, the internal trimmed low-frequency reference oscillator (REFO) can be used at the cost of extra current. See the device-specific data sheet for specifications. The wake-up time is sent to the MSP430FR2000 MCU through SPI in the following order: software counter (increment), RTCMOD high byte, and RTCMOD low byte (see Table 1). Table 2 lists the SPI settings. At this point, the host should go into a low-power or sleep mode as the real-time clock (RTC) starts immediately after reception of the third byte. The line from the MSP430 MCU to the host is used to communicate to the host to wake-up from its low-power or sleep mode. This line should be connected to an interruptible or wake-up capable source pin on the host (see Figure 1).

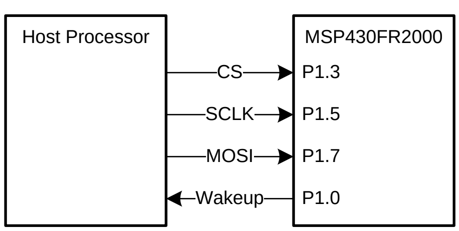

Figure 1. Programmable System Wake-up Controller Block Diagram

The wake-up time is a function of the RTC Counter peripheral and software scaling factors. The RTC Counter module in this application is clocked by XT1 at approximately 32.768 kHz. The largest predivider for the module is 1024. By using this divider value, every 32 counts of the RTC Counter is 1 second. The RTCMOD register holds a count value that gives an interrupt when the RTC Counter counts to it. The RTCMOD register of the RTC Counter is 16 bits wide so the maximum that time the RTC Counter can count before overflow is approximately 34 minutes. For example, if a host wanted to setup a wake-up time of 1 hour, it would send a value of 0x01E0FF over SPI (see Table 1). This means the host has set a counting interval of one, with an RTCMOD value of 0xE0FF, which corresponds to 30 minutes with the clock settings previously discussed. Equation 1 and Equation 2 can be used to calculate wake-up time in seconds in both a general case and for the parameters described above.

$$\frac{RTCMOD}{\left(\frac{ClockSource}{Clock_{Predivider}}\right)} \times (increment) = WakeUpTime_{Seconds} \quad (1)$$

$$\frac{RTCMOD}{32} \times (increment) = WakeUpTime_{Seconds} \quad (2)$$

Figure 2 shows the code flow for the application. The programmable wake-up controller is design to stay in low-power mode 3 (LPM3) to conserve power. When the host controller sends the 3 wake-up time bytes, the RTC starts counting to the time value sent to the device. The RTC interrupt manages the total wake-up time and sends a low-to-high pulse to the host controller after the time value has been reached. The host cannot start a new count or restart the current count until the previous one has ended. To start a new count, the host must send another set of wake-up time bytes to the MSP430FR2000 MCU before going to sleep again.

Performance

Table 3 lists the power consumption of the programmable system wake-up controller. The average current of the application is dominated by the LPM3 current of the device and approaches this level as the wake-up time period is extended. Table 3 lists measured values for 1- and 10-second wake-up intervals and calculated values for longer time periods.

Table 1. SPI Packet for RTC Wakeup

Byte 1	Byte 2	Byte 3
Software counter	RTCMOD high byte	RTCMOD low byte

Table 2. 4-Wire SPI Connections

Slave or Master	CLK Phase	CLK Polarity	CS Polarity
Slave	Data changed on first clock edge and captured on second	Inactive state is low	Active high, idle low

Table 3. Average Power Consumption

Wake-up Time:	1 s	10 s	1 m*	1 h*	24 h*
Average Current:	1.36 µA	1.33 µA	1.32 µA	1.32 µA	1.32 µA

* Estimated

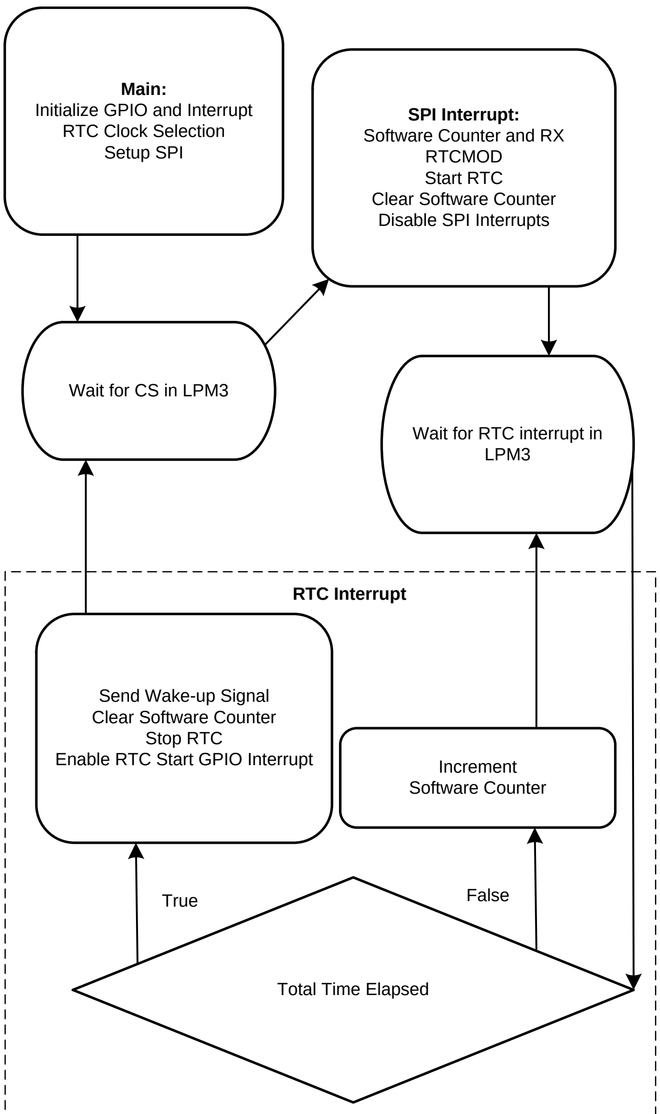

Figure 2. Software Flow Diagram

Device Recommendations

The device used in this example is part of the MSP430 Value Line Sensing portfolio of low-cost MCUs, designed for sensing and measurement applications. This example can be used with the devices shown in Table 4 with minimal code changes. For more information on the entire Value Line Sensing MCU portfolio, visit www.ti.com/MSP430ValueLine.

Table 4. Device Recommendations

Part Number	Key Features
MSP430FR2000	0.5KB FRAM, 0.5KB RAM, eComp
MSP430FR2100	1KB FRAM, 0.5KB RAM, 10-bit ADC, eComp
MSP430FR2110	2KB FRAM, 1KB RAM, 10-bit ADC, eComp
MSP430FR2111	3.75KB FRAM, 1KB RAM, 10-bit ADC, eComp

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265