

I²C Interfacing of the MSP430 to a 24xx Series EEPROM

Brian Merritt

Mixed Signal Products

ABSTRACT

This application report and the associated software demonstrate how to interface an MSP430 microcontroller to an external EEPROM memory using the I²C bus standard. The circuit and software presented are adaptable to any MSP430 device, EEPROMs with other memory sizes, or other external devices that also utilize an I²C interface.

Introduction

Many of the MSP430 microcontrollers either have Flash memory, which the application software can erase and program in-circuit, or they have one time programmable (OTP) memory that external sources can program in-circuit. Even so, there are applications where additional external EEPROM memory is needed. In the case of Flash parts an EEPROM may be desired if most of the MSP430's memory is required to store the program, or if a large amount of memory is needed for data logging applications. When using OTP or ROM parts, EEPROM memory may be needed to enable the application software to update calibration constants, change serial numbers, or datalog sensor readings. The use of the I²C bus is a common method of interfacing external EEPROM memories, and other devices, to microcontrollers.

The MSP430 microcontrollers are able to emulate the open collector outputs often used for I²C interfacing. This application report demonstrates how to accomplish this by using an MSP430x11x1 device to interface to, and communicate with, a 512 byte, 24LC04 EEPROM memory device via I²C. No specific hardware modules of the MSP430 are required to implement I²C, so this software can be easily adapted to any member of the product line. The example circuit and associated software read and write to random memory locations of the external EEPROM memory.

The software associated with this application report is designed to communicate to an EEPROM memory. Since all I²C communications work in the same general way, the program could be readily modified to communicate with other devices that also utilize the I²C bus interface. The I²C portion of the code only requires about 420 bytes of program memory. The remainder of the program supports a user interface via a PC. That portion of the code requires about 520 bytes of program memory. The user interface is not required for I²C communications. It is included in the program to facilitate the demonstration of the I²C code and hardware.

Theory of Operation

Using any two available I/O pins, the MSP430 is capable of emulating the open collector operation of the SDA and SCL signals required for I²C communications. This is accomplished by setting the output data for these two I/O pins low and using the directions setting of the pins to determine the logic level of the signal lines. Whenever a logic high is required, the direction bit for the corresponding pin is set to configure that pin as an input. The pullup resistor on the signal line then pulls it to a logic high state. If a logic low is required, the direction bit is set to switch the pin to an output. Since the data for the pins was previously set low, the signal line will be pulled low by the pin being switched to an output.

The MSP430 initiates a write to a random byte of the EEPROM memory, by first sending a start condition. Then the control byte is sent. The control byte consists of the device code for the specific device that is being addressed, the address of the memory block being accessed, and a bit that indicates whether a read or write operation is being performed (see Figure 1). The EEPROM then acknowledges receipt of the byte. The MSP430 then sends the specific memory address of the byte being written to. The EEPROM acknowledges again. The MSP430 then sends the data to be written into the memory byte. After the EEPROM acknowledges the data, the MSP430 sends a stop condition.

The reading of random bytes of the EEPROM memory is somewhat different than writing to them. The main difference is that the control byte is sent twice, once before the MSP430 sends the address byte and a second time before the EEPROM returns the requested data byte. The read/write bit of the control word is also set to a logic high to indicate a read operation.

The associated software was written specifically to address the 512 bytes of memory in a 24LC04 EEPROM, but smaller memories would also work since the address bits in the control byte would be ignored. Larger memories that utilize more of the address bits in the control byte could also be supported if the software were modified to set the address bits as required.

The user interface is accomplished by communicating with a terminal program running on a PC. Memory and data bytes are exchanged between the MSP430 and PC in the form of ASCII characters. The terminal program runs at 9600 baud using an 8N1 protocol.

1	0	1	0	X (A2)	X (A1)	A0	R/ \overline{W}
4-bit code for 24LC04				address of memory block		read / write	

Note: X = don't care for 24LC04

Figure 1. Control Byte for the 24LC04 EEPROM

Example Circuit

The example circuit in figure 2 is powered directly from the RS232 serial port of a PC. The interface to the serial port is accomplished using two TI SN74AH1G04 inverters. If a fully compliant RS232 solution is required, the TI MAX3221 (3V) could be substituted. Two pullup resistors are needed on the I²C signal lines to pull the lines high when the MSP430 I/Os are switched to inputs. A TI TPS77033 low drop out regulator is also added to regulate the voltage level from the PC serial port.

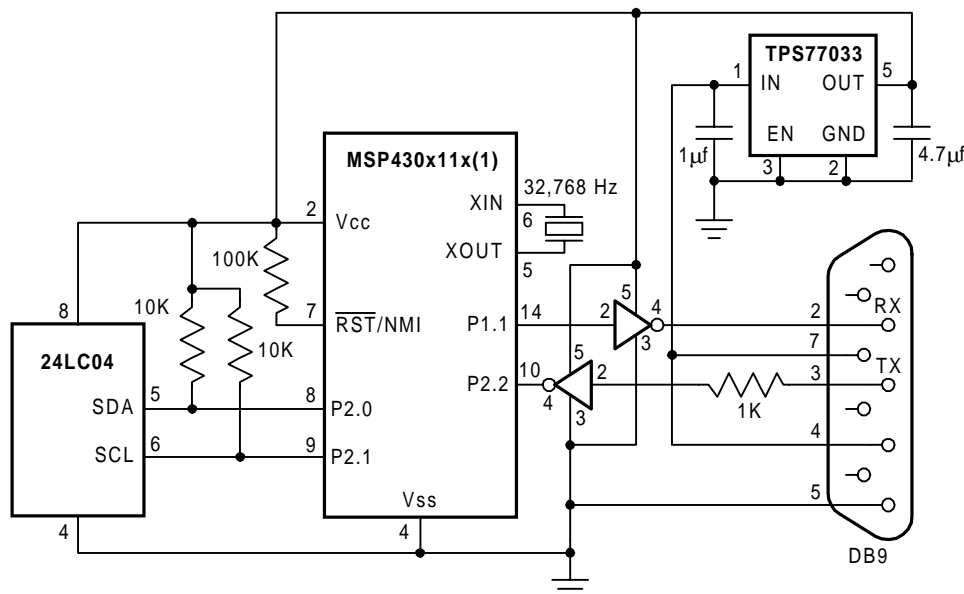


Figure 2. Schematic of the I²C Bus and PC Serial Interfaces to the MSP430

Example Code

The example code for this application is available from the MSP430 web site, <http://www.ti.com/sc/msp430>, as a .zip file under this application note listing. The title of the program is *I2C_EEmem.S43*. The code is written in assembly language using the IAR Kickstart integrated development environment.

References

1. MSP430x11x1 Data sheet (SLAS241C)
2. MSP430x1xx User's Guide (SLAU049)
3. 24xx Data sheets (various companies)

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.