

Interfacing an EEPROM to the MSP430 I²C Module

Christian Hernitscheck

MSP430 Products

ABSTRACT

This report describes the implementation of I²C communication between the MSP430F169 I²C hardware module and an externally connected EEPROM (24xx65). The protocols Byte Write, Current Address Read, Random Address Read, and Acknowledge Polling are covered.

1 Example Schematic

The schematic in Figure 1 shows how an EEPROM device can be connected to the MSP430F169 I²C hardware module.

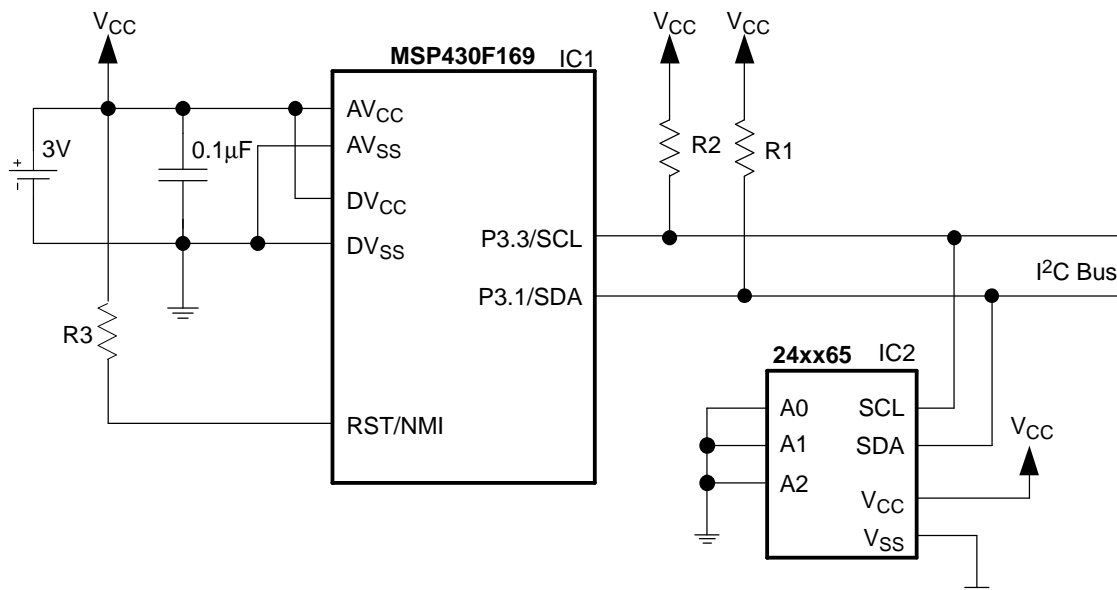


Figure 1. Interfacing an 24xx65-type EEPROM to the MSP430F169 via I²C bus

The user configurable chip select pins (A0, A1, and A2) define the I²C device addresses of the connected EEPROMs. The inputs are used to allow multiple devices to operate on the same bus. The logic levels applied to these pins define the address block occupied by the device in the address map. A particular device is selected by transmitting the corresponding bits (A0, A1, and A2) in the control byte.

2 MSP430 Source Code

The software example shows how to use the MSP430F169 I²C hardware module for communication with EEPROM via I²C bus. Depending on the memory size of the EEPROM the addressing scheme may look different. There are EEPROM versions that only need one byte for addressing (memory size is 256 Byte or less). The example code uses two bytes for addressing. Both C and assembly language versions with identical functionality are provided. A detailed description of the MSP430 I²C module operation can be found in [2].

The files "I2Croutines.c" and "I2Croutines.s43" contain the following functions and can be used as a library:

- **InitI2C()**
The MSP430 I²C hardware module is initialized.
- **EEPROM_ByteWrite(<address>, <data>)**
A Byte Write command is executed. The address and the data that has to be written into the specified address are provided.
- **EEPROM_AckPolling();**
The Acknowledge Polling is used to check if the write cycle of the EEPROM is finished. After writing data into the EEPROM, the Acknowledge Polling function should be called.
- **Result = EEPROM_RandomRead(<address>)**
The Random Read command of the EEPROM allows reading the contents of the specified memory location.
- **Result = EEPROM_CurrentAddressRead()**
The EEPROM internal address pointer is used. After execution of a write or read operation the internal address pointer is automatically incremented.

The following example shows how to use the functions from the files "I2Croutines.c" and "I2Croutines.s43":

```
#include "msp430x16x.h"
...
/*--- external functions of file "I2Croutines.c" -----*/
extern void InitI2C(void);
extern void EEPROM_ByteWrite(unsigned int Address, unsigned char Data);
extern unsigned char EEPROM_RandomRead(unsigned int Address);
extern unsigned char EEPROM_CurrentAddressRead(void);
extern void EEPROM_AckPolling(void);
...
/*--- main program -----*/
void main(void)
{ ... // Misc. initialization
InitI2C(); // Initialize I2C module
_EINT(); // Enable interrupts
...
EEPROM_ByteWrite(0x0000, 0x12);
EEPROM_AckPolling(); // Wait for EEPROM write cycle completion
EEPROM_ByteWrite(0x0001, 0x34);
EEPROM_AckPolling(); // Wait for EEPROM write cycle completion
EEPROM_ByteWrite(0x0002, 0x56);
EEPROM_AckPolling(); // Wait for EEPROM write cycle completion
...
Data[0] = EEPROM_RandomRead(0x0000); // Read from address 0x0000
Data[1] = EEPROM_CurrentAddressRead(); // Read from address 0x0001
Data[2] = EEPROM_CurrentAddressRead(); // Read from address 0x0002
```

2.1 Byte Write

Figure 2 shows the Byte Write protocol. Note that a STOP condition resets the MST bit in the MSP430 U0CTL control register. Before starting the next communication (e.g. Byte Write), the I²C module has to be re-configured into the master mode by setting the MST bit.

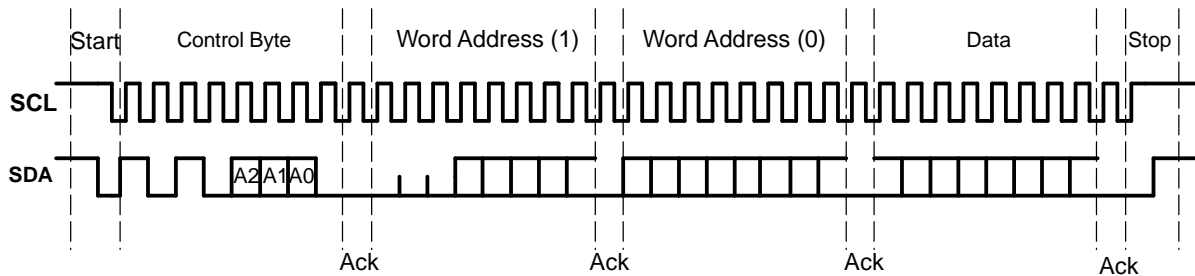


Figure 2. Byte Write

Also, the generation of the I²C Control Byte is done in a special way. The MSP430 I²C module sends out the slave address that is defined in the control register I2CSA. Note that the slave address is located within the control byte in the bit positions 1 to 7. The number that is defined in the register I2CSA has to be shifted one bit position to the left to get the EEPROM Control Byte (Figure 3). For example if the Control Byte should be 0xA0 the slave address (I2CSA) has to be defined as 0x50. The R/W bit, which is also located in the control byte (bit 0), is handled automatically by the selected transmit or receive mode. The mode is selected using the I2CTRX bit in the I2CTCTL control register.

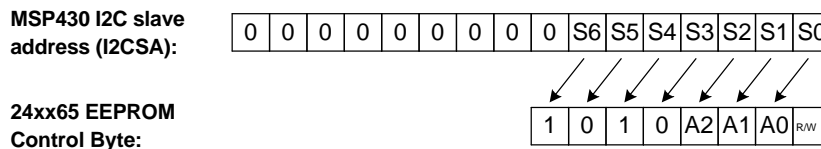


Figure 3. I2CSA Content to I²C Control Byte Mapping

Figure 3 shows how the slave address of the Control Byte is defined. The upper four bits is a fixed number (1010). The lower bits of the slave address are the device select bits. The master can access different EEPROMs connected on the same I²C bus by defining the A0, A1, and A2 bits (EEPROM device address).

2.2 Current Address Read

Figure 4 shows the Current Address Read operation. Before the communication is started the MSP430 I²C module has to be re-configured. The master mode has to be selected and the receive mode has to be enabled.

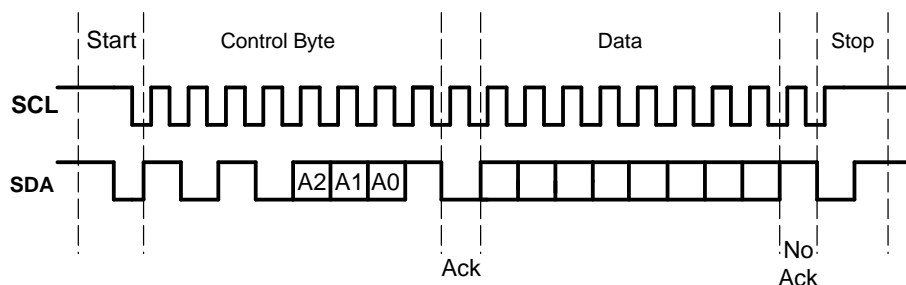


Figure 4. Current Address Read

The control byte generation works in the same way as described in the chapter Byte Write.

References

The example code only receives one byte. Polling is used in the example code to check if the I²C communication is already finished and the data byte was received.

2.3 Random Address Read

Figure 5 shows the Random Address Read protocol. It is a mix between the Byte Write and the Current Address Read operation.

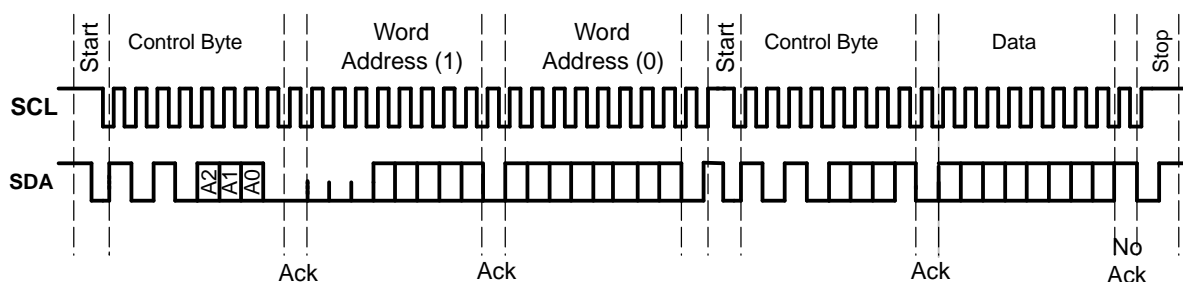


Figure 5. Random Address Read

First the address counter of the EEPROM has to be set. This is done by the Byte Write operation. The new address is transmitted to the EEPROM. After that, a re-start is done and the Current Address Read operation is executed.

2.4 Acknowledge Polling

As soon as a write command was received and the MSP430 generated the stop condition, the EEPROM initiates an internally timed write cycle. The time of this write cycle depends on the used EEPROM. As long as the EEPROM is in a write cycle it will not acknowledge. This can be used to determine when the internal write cycle is completed.

3 References

1. MSP430F169 Data Sheet ([SLAS368](#))
2. MSP430x1xx Family User's Guide ([SLAU049](#))

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

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

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		Applications	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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