

# **Using Tracii400 to Read From SPI Device**

## **Application Note**

**telos EDV Systementwicklung GmbH  
Schlüterstraße 16  
D-20146 Hamburg**

[info@telos.de](mailto:info@telos.de)  
<http://www.telos.de/>

<b>Version</b>	<b>1.0</b>
<b>Date</b>	<b>2004.08.03</b>

<b>telos EDV Systementwicklung GmbH</b>		
Application Note		Date 2004.08.03
Using Tracii400 to Read From SPI Device	Version 1.0	Page 2

Author(s) RZ

Started by RZ on 2004.08.03

Checked by UH on 2004.08.03

Released by RZ on 2004.08.03

The fields above are not designated for signs. Date format is yyyy.mm.dd.

### Change History

Version	Change Reason	Author
1.0	Document Initiation	RZ

<b>telos EDV Systementwicklung GmbH</b>		
Application Note		Date 2004.08.03
Using Tracii400 to Read From SPI Device	Version 1.0	Page 3

## Table of Contents

1	Abstract.....	4
2	Theory.....	4
3	The Application.....	5

<b>telos EDV Systementwicklung GmbH</b>		
Application Note		Date 2004.08.03
Using Tracii400 to Read From SPI Device	Version 1.0	Page 4

## 1 Abstract

Even though Tracii400 is primarily designed to serve as an I2C interface it can be used to access other bus systems as well.

This application note shows how to use the input and output test pins to read values from an SPI device.

## 2 Theory

Tracii400 is equipped with two input and two output pins. These pins can be read and written independently via designated API functions.

Operating a bus by individually toggling pins through all the involved PC software layers is somewhat slow; however, there are many applications where speed is simply not an issue. The MAX 6628, 6629, 6630 and 6631 ICs are read-only SPI devices which measure the temperature and return it as a decimal signed value.

Two output pins and one input pin are needed in addition to VCC and ground to connect such sensor.

This application note assumes the following connections to a Tracii400 board.

MAX66xx	Tracii400
SO	I1
SCLK	O1
CS	O2
VCC	5V (I2C connector)
GND	GND (I2C connector)

Note that both I2C lines, SCL and SDA are not used. This means that such SPI operation can take place in "pseudo-parallel" to regular I2C master or slave traffic which is an interesting aspect for test bench applications.

SPI lines are always unidirectional and so are the four Tracii400 test pins. If we had chosen an SPI device with read/write operation we would need to implement an additional output pin for MOSI (Master Out, Slave In) which would require some more hardware.

However, since the MAX 6628-31 are read-only we get by with the two output pins available on the Tracii400.

Implementing the SPI protocol is rather simple. Driving the CS signal from high to low discontinues the temperature conversion which is permanently performed while CS is in high state.

With CS held low each bit of the 16 bit data value can be retrieved by pulling the SCLK line to low and reading the value of SO (slave out). After reading the bit SCLK is set to 1 again to prepare for the next bit transfer.

After all 16 bits are captured CS can be set to high again to initiate a new temperature reading.

The temperature sensor needs a power supply. It works over a wide voltage range including 5V. Tracii400 is able to supply 5V regulated power over the center lines of the I2C connector. In order to enable this functionality both the DCPLUG and the I2C jumpers need to be set.

<b>telos EDV Systementwicklung GmbH</b>		
Application Note		Date 2004.08.03
Using Tracii400 to Read From SPI Device	Version 1.0	Page 5

### 3 The Application

The sample application provided with this note is written in MS Visual Basic 6.0 using the TraciiWork API.

It consists of a simple form which allows to initiate a temperature reading and displays the value in numeric format.

The concept of SPI operation can easily be transferred to other applications. The status of the output test pins is modified and kept in a variable rather than written directly. This allows to feed it to I2C master functions which is necessary if the SPI bus is operated together with I2C.

The sensors return their values in signed format. One bit is returned for the sign and 12 bits for the actual value. Signed values are represented in twos complement format, e.g. a –1 is represented as 0x1fff.

The temperature is measured in steps of 1/16°C. The example application converts this to a decimal point notation.