



APPLICATION NOTE 3839

Controlling the DS3900 Serial Communications Module Using LabView

Abstract: This application note discusses the DS3900 Serial Communications Module and LabView, a graphical programming platform for developing embedded applications. The article serves as a user's guide for a LabView-based interface to the DS3900.

Introduction

LabView is popular as a graphical programming platform for developing embedded applications. The [DS3900](#) Serial Communications Module is a general-purpose interface board to communicate with devices equipped with an I²C interface using the PC's serial port. The DS3900's instruction set allows the application software to communicate directly to the I²C device.

This application note is a user's guide for a LabView-based interface to the DS3900. To use this application, the user needs to install LabView. Standard VIs are provided for the user to load and run the program. The LabView code for this application note is available for [download](#) (ZIP, 321K).

Using the LabView Software

After loading the VI, the user must run the program and perform the following steps. (See **Figure 1.**)

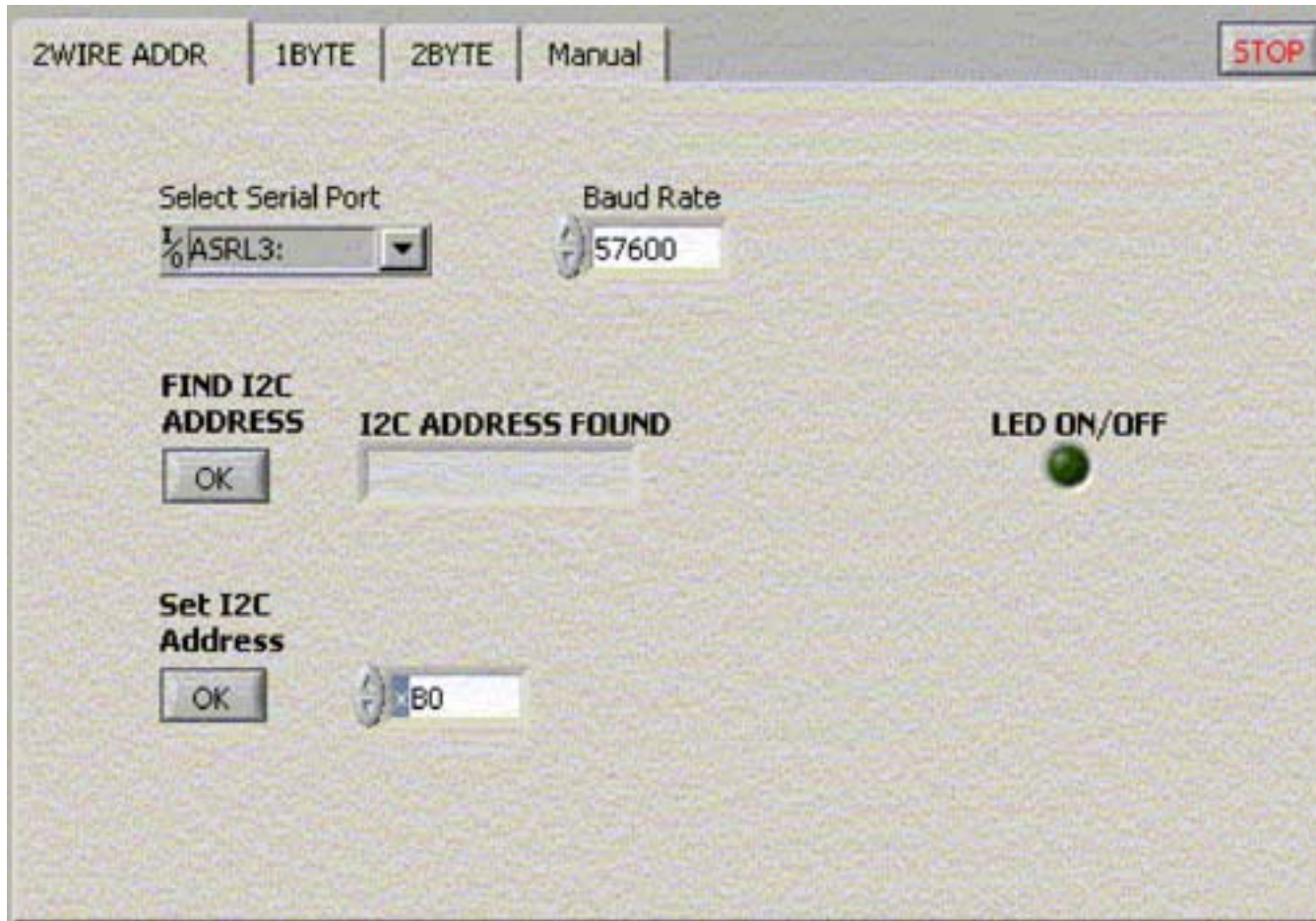


Figure 1. LabView screen from which the user configures a system for operation.

1. The user first selects the serial port that will be used for programming the DS3900. LabView automatically searches for available serial-port resources, and displays them in a drop-down list. Select the serial port to which the DS3900 is connected.
Note: The default baud rate at which the DS3900 communicates is 57600. This application has been tested and functions correctly at this baud rate. Users are advised not to change this rate when using the application.
2. If the serial port is set correctly, the LED on/off function should work. If the end application has an LED connected to the pulse output of the DS3900 (pin P3), then this LED should turn on and off when the LED on/off button is pressed.
3. Next find the slave address for all devices on the I²C bus by clicking the **FIND I2C ADDRESS** button. Use the **Set I2C Address** field to enter the slave address to which the VI will communicate.
4. Tabs at the top determine the specific function that the user wants to use. Various options are available:
 1. 1BYTE: writing or reading a register, 1 byte at a time.
 2. 2BYTE: writing or reading two consecutive memory addresses. This assumes that the device's internal address counter increments automatically each time a read or write is done.
 3. Manual: the control where the user decides how the I²C should operate. Some examples are given in the **Manual Control** section below.

Single-Byte Reads and Writes

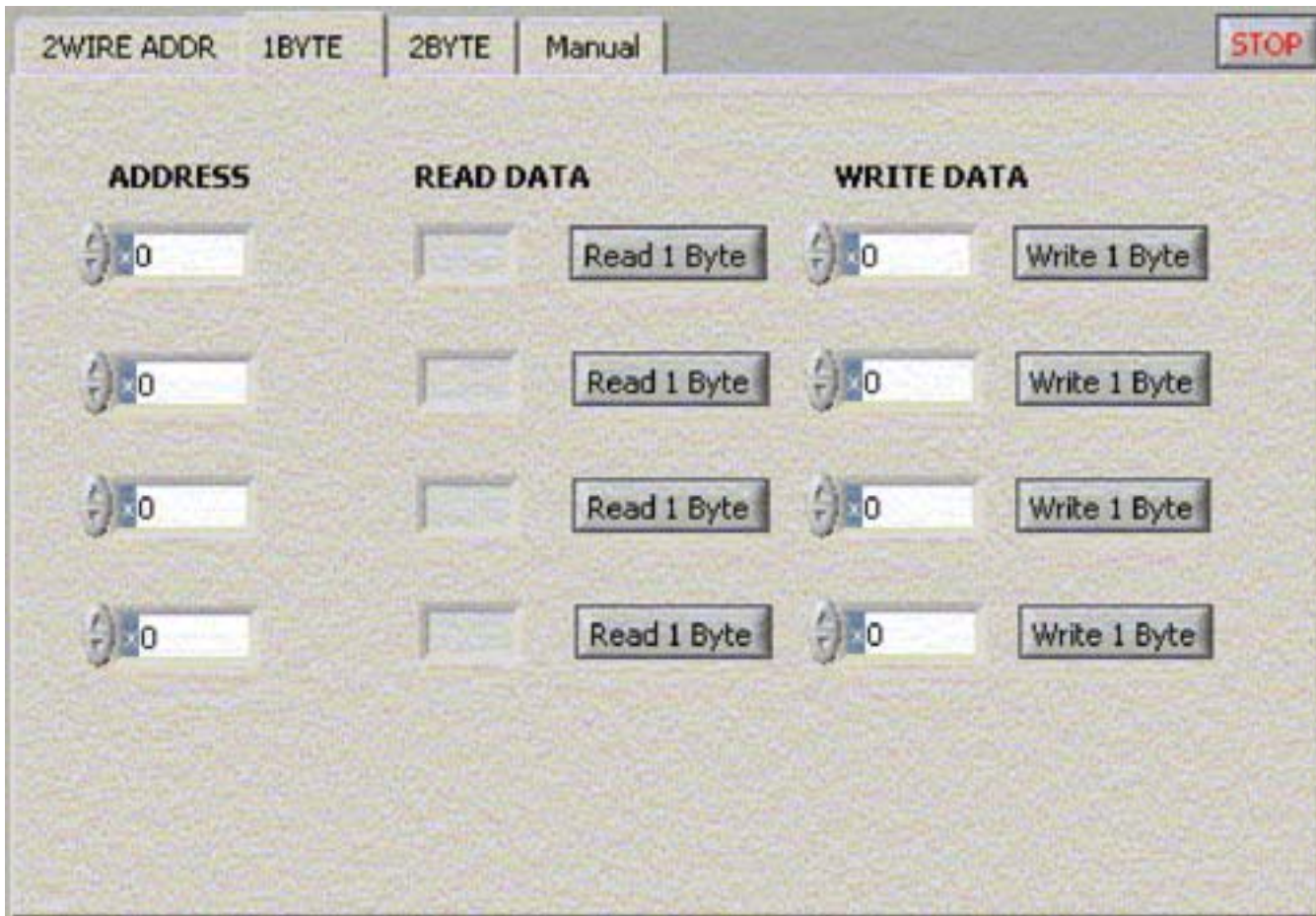


Figure 2. When performing a single-byte operation, the user selects among four addresses for communicating with the I²C device.

Users have the option of selecting four different addresses from which to communicate (**Figure 2**). Any **READ DATA** will be copied into the corresponding **WRITE DATA** section for easy manipulation of individual bits.

Two-Byte Reads and Writes

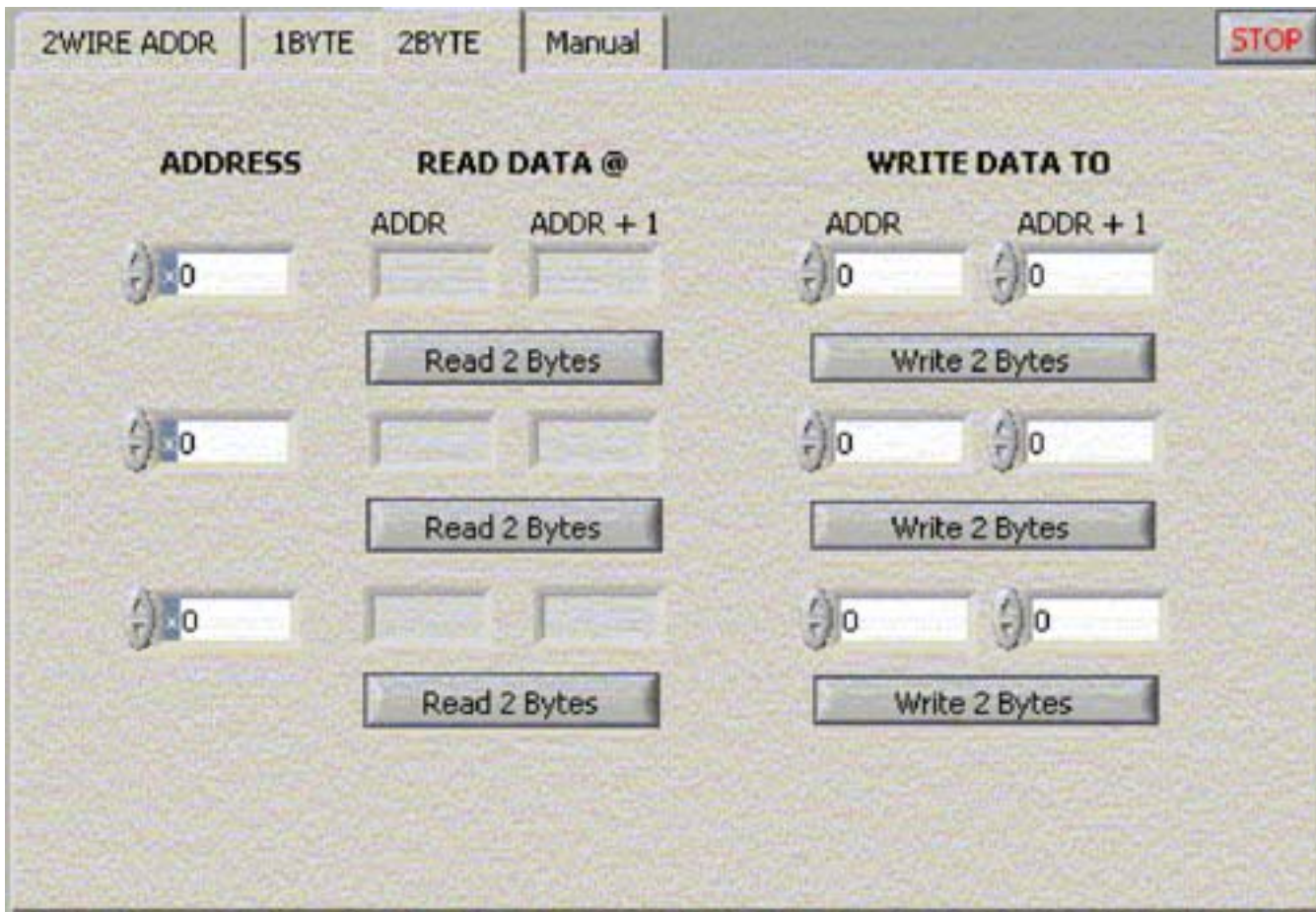


Figure 3. The 2BYTE tab lets the user read data from two registers.

For certain devices, one register might consist of two consecutive bytes in memory. This **2BYTE** tab (**Figure 3**) allows the user to read the value of two consecutive registers, provided that the memory address pointer in the device automatically increments to the next register. The value read is automatically copied into the **WRITE DATA TO** entry section.

Manual Control

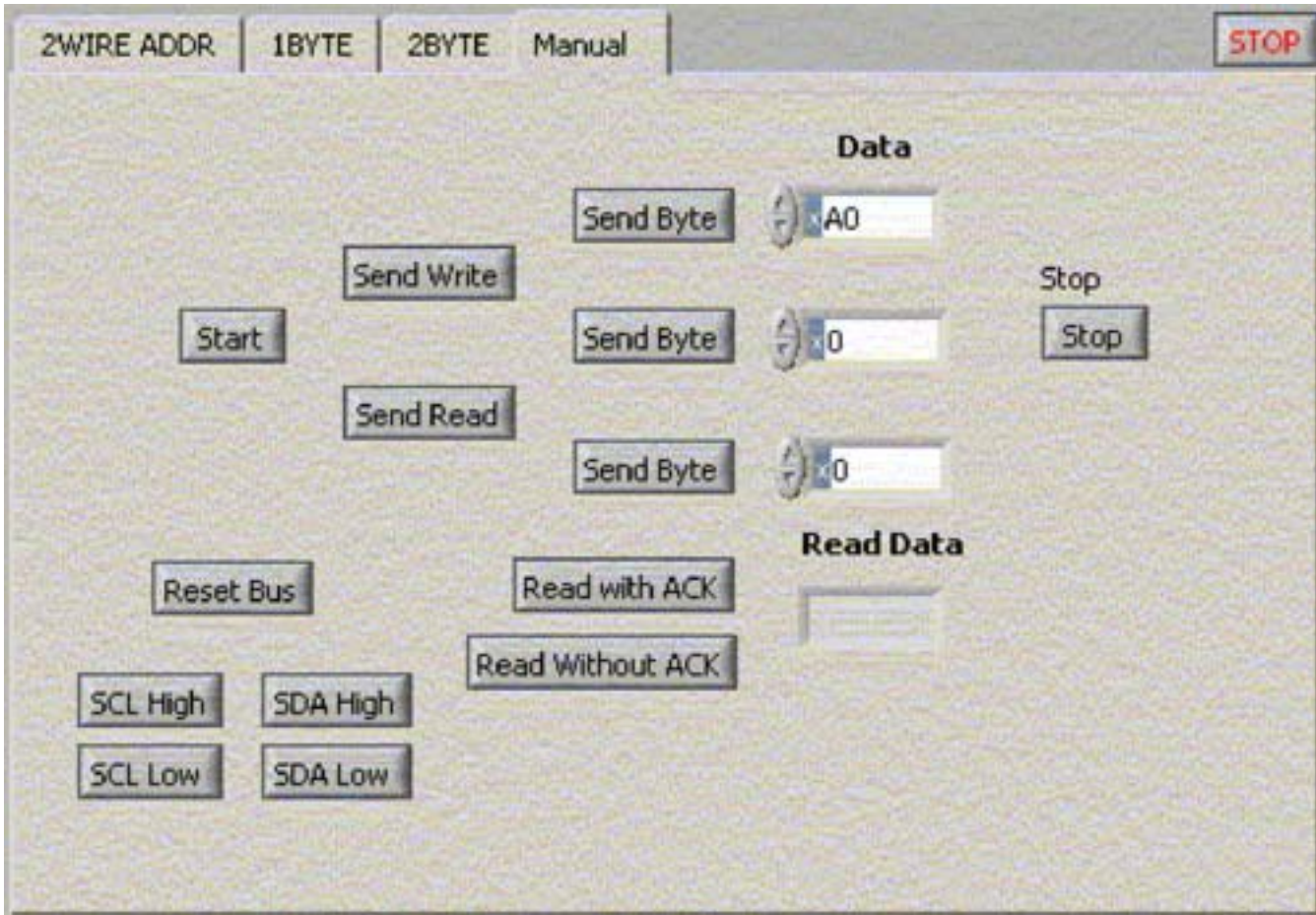


Figure 4. The MANUAL control tab lets the user define the number of bytes and intended read or write operation.

With **MANUAL** control (**Figure 4**), the user decides the number of bytes and the particular operation desired. Examples are provided below to do a standard 1-byte Write and a 2-byte Read.

1-Byte Write

START	SEND WRITE (DEVICE ADDRESS FOR WRITE)	SEND BYTE (MEMORY ADDRESS)	SEND BYTE (DATA TO BE WRITTEN)	STOP
-------	---------------------------------------	----------------------------	--------------------------------	------

2-Byte Read

START	SEND WRITE (DEVICE ADDRESS FOR WRITE)	SEND BYTE (MEMORY ADDRESS)	SEND START (REPEATED START)	SEND READ (DEVICE ADDRESS FOR READ)	READ WITH ACK (READ 1ST BYTE)	READ WITH NACK (READ 2ND BYTE)	STOP
-------	---------------------------------------	----------------------------	-----------------------------	-------------------------------------	-------------------------------	--------------------------------	------

Important note: when reading data, the last byte read should be a 'Read with NACK'. This allows the user to read as many bytes as required. The 'Read with NACK' is required to inform the device that no more data is to be sent.

The user can also reset the I²C bus. This is useful if the user gets lost during manual communication and leaves the bus in an unknown state. **SCL** and **SDA** High/Low buttons let the user force these pins to a desired state.

Conclusion

The LabView VI included with this application note shows how useful LabView can be for communicating with the DS3900. Once these initial concepts are understood, the user can modify the VI or copy certain blocks to accommodate specific application needs.

Application Note 3839: www.maxim-ic.com/an3839

More Information

For technical support: www.maxim-ic.com/support

For samples: www.maxim-ic.com/samples

Other questions and comments: www.maxim-ic.com/contact

Automatic Updates

Would you like to be automatically notified when new application notes are published in your areas of interest? [Sign up for EE-Mail™.](#)

Related Parts

DS3900: [QuickView](#) -- [Full \(PDF\) Data Sheet](#)

AN3839, AN 3839, APP3839, Appnote3839, Appnote 3839

Copyright © by Maxim Integrated Products

Additional legal notices: www.maxim-ic.com/legal