NATIONAL CONTROL DEVICES

Fusion I²C Quick Start Guide

Fusion I²C Quick Start Guide

Table of Contents

Chapter

1

Introduction

Fusion series controllers are capable of communicating to I²C devices such as temperature sensors, gyros, accelerometers, light sensors, humidity sensors, non-volatile EEPROM, SRAM, and many other types of I²C devices. The I²C capabilities of a Fusion controller provide users with a pathway to access these devices using any of our popular interface technologies including Ethernet, Wi-Fi, USB, RS-232, Bluetooth, 802.15.4 Wireless, and ZigBee Mesh.

Select Fusion I²C capable devices support at least two I²C communication ports (Port 0 and Port 1). Port 0 has a fixed communication rate of 100 KHz and may be used by NCD devices that contain integrated sensors, I/O Expanders, or other types of I²C devices. Though we may allow access to Port 0 on select controllers, Port 0 should generally be considered for use by the NCD products and peripheral devices. For instance, we may offer expansions that offer additional features by utilizing Port 0. Port 0 is typically accessed using the FXR Relay Expansion Port. I²C Port 1 is for general use by our customers. Port 1 is available for on AD8 Port 1 (shared with UXP Port 1) using I/O Lines 1 (SDA) and 2 (SCL). I²C Port 1 can be tuned for faster or slower speeds. Slower speeds allow Fusion controllers to communicate I²C data over longer distances while faster speeds offer greater compatibility. The Up/Down Jumper for AD8 Port 1 should be set to the "UP" position when using the I²C communication features.

Fusion I²C Port 1 supports 500 Hz, 1 KHz, 10 KHz, 50 KHz, 100 KHz, and 130 KHz communication speeds (which has all been precisely timed with an oscilloscope), and supports I²C device clock stretching, as required by the I²C communication protocol (a feature often skipped by most I²C implementations). Development of the I²C communication routines were centered on a 24LC515 Non-Volatile EEPROM, and has been tested with various I²C sensors to prove functionality.

Use of I²C communication functions on Fusion series controllers require prior knowledge of I²C, as it is outside the scope of our documentation to go into a detailed explanation of the protocol. Here we will focus on the data exchange between a computer and the Fusion controller. The Fusion controller will manage the data exchange with the I²C device, greatly simplifying the amount of information that must be studied.

When working with I²C for the first time, it is helpful to use the 24LC515 EERPOM connected to a Fusion controller. The 24LC515 demonstrates 2-way communications with the device, showing you how to read and write data as shown in the sample provided in Base Station software.

To date, all I²C devices we have tested are compatible with Fusion I²C communication routines.

Overview of I²C

For complete details on the I²C protocol, there are many resources available on the internet, some of which have some very good descriptions of the protocol. Most of the detailed protocol information may be skipped when working with a Fusion controller as the difficult part of communications is completely handled by our firmware. Even so, there are a few things you need to understand before you can successfully implement communications to an I²C device:

1) I²C devices need a power supply of 3.3V or 5V. Fusion controllers ONLY work with 5V I²C devices. There are many I²C level converters on the market that translate 5V data to 3.3V for lower voltage I²C devices. Do not attempt to connect a 3.3V I²C device to a Fusion controller without a level translator.

2) I²C is an Open Collector Protocol. As such, all data is transferred using open collector communications, not standard TTL signals. In order to speak to an Open Collector device, the Pull Up/Down Jumper on the AD8 Port 1 MUST be set to the UP position. This will pull up the SDA and SCL lines using a 10K resistor, which is within I²C specifications. Note that communication errors will result unless this jumper is properly set.

3) I²C devices NEVER send data to the controller spontaneously. You should never expect an I²C device to function in this way. Instead, Fusion controllers send a command to the I²C device and the I²C device replies with data. Put simply, an I²C device only speak when spoken to.

4) Carefully study the datasheet of your I²C device. You can skip most of the information on timing requirements, as this is handled by the Fusion controller. Specifically, you need to know the following about your I²C device:

   a. Make sure the device supports 100 KHz communications, almost all I²C devices do.

   b. You will need to know the address of the device. Every I²C device has a unique address that is required to initiate communications.

   c. Learn the command set for your I²C device. Many I²C devices support both write and write/read operations (writing a command, reading a response). Familiarizing yourself with these commands is essential.

5) Fusion controllers are limited to writing 75 characters on the I²C bus. It is not possible to read or write data to an I²C bus that exceeds the 75 character limit. This is a limitation of the I²C buffer within the Fusion controller. Most I²C exchanges are 16 bytes or less, so this should not be a problem for most applications. Base Station Software is limited to 16 bytes of communication to and from an I²C device, but this does not represent the limit of the Fusion controller.

6) Fusion controllers do not support 10-Bit extended addressing.

7) The wire length between a Fusion controller and I²C devices is an important consideration. The maximum capacitance on an I²C bus is 500pf. A high quality twisted pair cable should be used to maximize distance when required. Network cable is well shielded and pairs of cable are twisted together, reducing noise potential on the cable. If you choose to use network cable, always use a pair of colors that are twisted together. CAT6 Cable has a capacitance of 13.5pf per foot. CAT5 Cable has a capacitance of 17pf per foot. Stranded wire can also be used for shorter distance applications. Most I²C devices are capable of operating at lower clock frequencies below 100 KHz. Reducing the clock frequency will significantly improve communication distance. For customers who need longer distance, use the Device Configuration button in Base Station Software to set the timing of I²C Port 1 to a slower speed. Some users have reported 100 meters of distance when working with 500 Hz communication speeds. We have not attempted to verify this capability, but we believe this is possible. 500 Hz is 200 times slower than 100 KHz, but it's still fast enough for most applications.

8) I²C supports multiple devices on the same communication bus, so it is possible to have many different types I²C devices working together using the I²C bus on the Fusion controller.

Physical Implementation

NCD Fusion controllers have adopted the same loose standard as Arduino and other microcontrollers that support a 4-pin latching header with pins 1-4 offering the following interface connections:

I²C Port 1 Pin 1 on NCD Device Carries the I²C SCL Signal to Pin 4 on an I²C sensor.
I²C Port 1 Pin 2 on NCD Device Carries the I²C SDA Signal to Pin 3 on an I²C sensor.
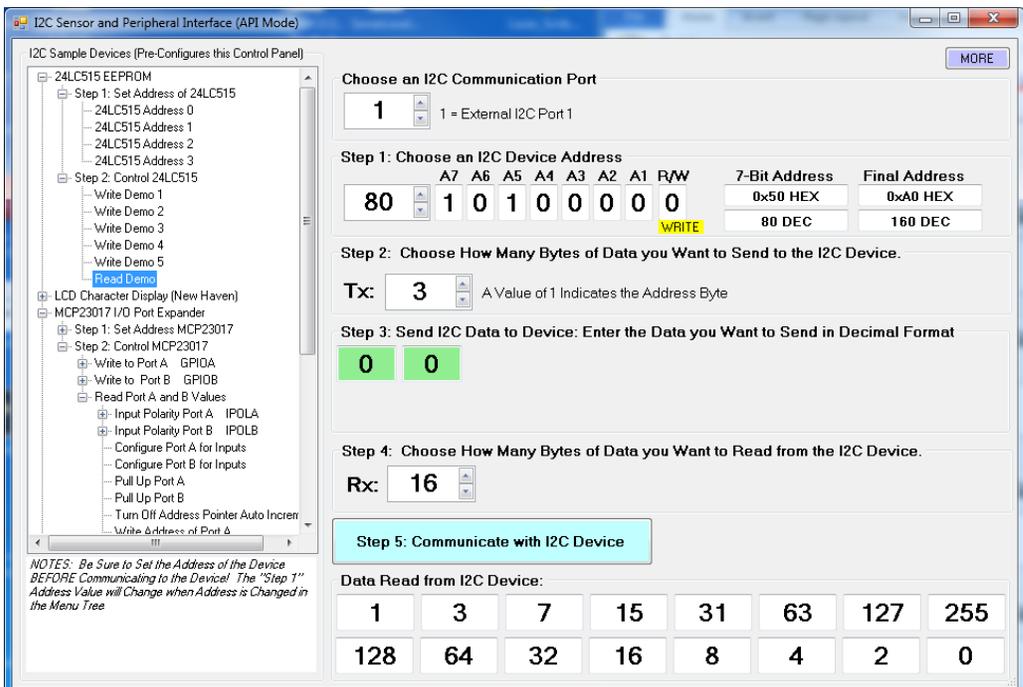I²C Port 1 Pin 3 on NCD Device Carries +5 Volts to Pin 2 on an I²C sensor.
I²C Port 1 Pin 4 on NCD Device Carries Ground to Pin 1 on an I²C sensor.

Port 0 is physically connected to the FXR Expansion Relay port, and is typically used by NCD devices that include integrated I²C peripheral devices. Port 0 is not generally available for direct use, but is available for direct use on select models of Fusion controllers. Integrated 10K Pull-Up resistors are wired to the I²C SCL and SDA Lines of Port 0.

Port 1 is physically connected to AD8 Port 1 (Shared with UXP Port 1). Port 1 is for customer use and may be tuned to 500 Hz to 130 KHz with a default communication speed of 100 KHz.

Software Implementation

NCD Base Station Software includes an I²C control panel used to demonstration communications to various popular I²C devices. The control panel shown below may look a little intimidating at first, but it is about as easy as we can make I²C communications.

Choosing an I²C Communication Port:

This option lets you choose the I²C communication port you will be using. Most of the time, you will be using Port 1 as shown, Port 0 is typically reserved for use by NCD devices that include integrated I²C peripheral devices. Port 0 may at times be available to the user depending on controller model.

Step 1: Choosing an I²C Address

The I²C device address is the most important byte of the I²C communications protocol. This address is always indicated in the datasheet for your I²C device. In an effort to assist customers, the I²C address of know devices is shown in the left menu tree. Simply choose the I²C device you would like to communicate with, and select the address choice indicated in Step 1 of the menu tree.

If you are setting up communications with an I²C device not shown in the left menu tree, click on the A7-A1 boxes, as well as the R/W box to set the bits. The 7-Bit Address and Final address are indicated (data sheets tend to show one or the other type of address which may or may not include the R/W bit).

Step 2: Choose How Many Bytes of Data you want to Send to the I2C Device

Keep in mind, when communicating with an I²C device, you have to send at least 1 byte of data (which includes the address shown in step 1). For this reason, the minimum number of bytes you can send to an I²C device is 1. Additional bytes are often used to indicate memory locations and data that is to be written to an I²C device.

Step 3: Sending Data

Each box should be filled with a decimal value from 0 to 255 (hex is not yet supported). The datasheet of your I²C will indicate the data that should be sent. Most I²C devices use the next few bytes to indicate register addresses inside the I²C device, and sometimes include data write to the selected address registers.

Step 4: Reading Data

This parameter is used to indicate the number of bytes to read from an I²C device. Reading data from an I²C device is optional, but this parameter must be included as part of the command, even if the number of bytes to read is zero. If data is received from an I²C device, it will appear in the "Data Read from I²C Device" frame.

Fusion I²C Command Set

There is only one Fusion command to learn for complete implementing I²C communications. There can be many variations on the use of this command, but the format is always the same. Because of the complex structure, the command will not be shown in our usual way. Instead, we will list every byte with a description of its purpose and implementation.

Fusion I²C Send Command Structure:

Byte 1   188   External Device Communications Command

Byte 2   50   Communicate to Internal I²C Port 0 (NCD Hardware I²C Port)

               51   Communicate to External I²C Port 1 (User I²C Port)

               52   Communicate to External I²C Port 2 (User I²C Port, Not Available on All Models)

               53   Communicate to External I²C Port 3 (User I²C Port, Not Available on All Models)

               54   Communicate to External I²C Port 4 (User I²C Port, Not Available on All Models)

Byte 3   1-75   Number of Bytes to Send

Byte 4   0-255   Byte 4 is the beginning of all data that is sent to the I²C device. All data to be sent to the I²C must be included here.

Last Byte   0-75   The last byte of the command must contain the number of bytes to read from the I²C Device.

Response from a Fusion Controller:

It is possible to receive many types of responses from the Fusion controller when speaking to an I²C device. For instance, 85 will be received if a Fusion controller has successfully transmitted data to an I²C device, but this response is only valid if you are NOT reading any data from an I²C device (Last byte of the transmission is Zero). If you are reading data from an I²C device, then the response should contain the data bytes you are expecting from an I²C device. If an error occurred during I²C communications, an Error Code will be returned.

NOTE: All Fusion Commands MUST BE API Encoded.
Please review the API Codec Quick Start Guide.

Error Codes:

Fusion firmware tries to provide you with clues as to why your command was not accepted by an I²C device. However, it should be stated that not having a device connected, using the wrong address or port number may also trigger these error codes. Fusion firmware will return the following error codes if there is a problem communicating with an I²C device:

88 90 165 67   Error 90: Acknowledgement Not Received from I²C Device
88 91 164 67   Error 91: Device Took Too Long to Respond
88 92 163 67   Error 92: Could Not Set Address of Device

Error 90:

This error is generated when a I2C Device does not acknowledge a command. Typically, this error occurs later in the serial stream, and usually indicates command parameters or memory locations are not properly set.

Error 91:

Fusion I²C firmware supports clock stretching, which allows I²C devices to hold the clock line during the time it takes to measure sensor data. This error code is generated of an I²C device takes too long to respond (1 or 2 seconds is too long for an I²C device).

Error 92:

The first byte of data sent to an I²C device is always its address. This error code is generated if an I²C device fails to respond to this address. This error typically occurs early in the serial stream, and indicates the address byte is wrong or the device is not properly connected.

Command Samples:

The following command examples will demonstrate proper reading and writing to I²C devices by explaining the command structure in a format that is a little easier to understand.

188 50 4 80 254 83 1 0

This command accesses the I²C command set (188) on Port 0 (50). This command will send (4) bytes of data out the I²C port. The following bytes of data will be sent (80 254 83 1). This will tell a New Haven LCD Display to turn off the backlight. Zero (0) bytes will be read from the I²C device. Upon successful execution of this command, the Fusion controller will reply with 85 after successfully communicating to the display.

188 51 19 160 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 0

This command accesses the I²C command set (188) on Port 1 (51). This command will send (19) bytes of data out the I²C port. The following bytes of data will be sent (160 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16). This will tell a 24LC515 EEPROM (160) to move to the specified address location (0 0). Sixteen bytes of data (1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16) will be written to the EEPROM. Zero (0) bytes will be read from the I²C device. Upon successful execution of this command, the Fusion controller will reply with 85 after successfully communicating to the display. Upon successful execution of this command, the Fusion controller will reply with 85.

188 51 3 160 0 0 16

This command accesses the I²C command set (188) on Port 1 (51). This command will send (3) bytes of data out the I²C port. The following bytes of data will be sent (160 0 0). This will tell a 24LC515 EEPROM (160) to move to the specified address location (0 0). Sixteen (16) bytes will be read from the I²C device. Upon successful execution of this command, the Fusion controller will reply with 16 bytes of data.

Special Notes:

Fusion controllers send I²C data to a 16x2 New Haven LCD Display connected to Port 0 on power up (Other display sizes will NOT display correctly). The New Haven display will show the current firmware version of the CPU along with our web address (IORelay.com). This feature cannot be disabled on a Fusion controller. However, if you do not want this information to appear on your display during power up, simply change the address of the display and the I²C data will be ignored. Please review the data sheet for your New Haven LCD Display to see how to edit the data address.

Chapter

4

Technical Support

Technical support is available through our website, <u>controlanything.com</u>. **Support** is the way we connect NCD engineers to our customers.
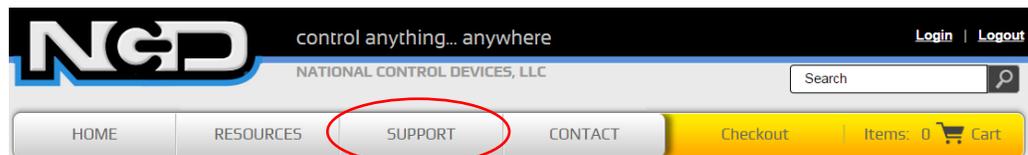
*Click on the **Support** tab at the top of any page on our website to be taken to the **Forum** page. Here you can publicly post or review problems that customers have had, and learn about our recommended solutions.*

Our engineers monitor questions and respond continually throughout the day. Before requesting telephone technical support, we ask that customers please try to resolve their problems through **Support** first. However, for persistent problems, NCD technical support engineers will schedule a phone consultation.

Contact Information

National Control Devices, LLC
PO Box 455
Osceola, MO 64776
417-646-5644 phone
866-562-0406 fax
Open 9 a.m. - 4 p.m. CST

Like "National Control Devices" on Facebook, and follow us on Twitter @ControlAnything.

All orders *must* be placed online at our website, www.controlanything.com

Notice:

The only authorized resellers of NCD products are

- www.controlanything.com
- www.relaycontrollers.com
- www.relaypros.com

All other websites are not authorized dealers; we have noticed some retailers offering our products fraudulently.