AN14565

Implementation of USB to I2C Demo on MCXN236

Rev. 1.0 — 30 January 2025 Application note

Document information

Information	Content
Keywords	AN14565, MCXNx4x/Nx3x, DCDC, USB, I2C, CDC class
Abstract	This application note describes implementation of USB to I2C bridge function on MCXN236 microcontroller.

1 Introduction

Some applications, such as the sensor test and optical module, require an I2C interface to support information transfer. Also, they require an interface to communicate with a PC to send some test commands and data to the I2C device to test their products.

The USB interface is commonly used in many applications, which can speed up to 480 MHz in HS mode USB2.0 protocol. Therefore, it is a good interface used as a bridge to transfer the data from another interface such as USART, SPI, and I2C to a USB host. This application note provides a demo, which uses the USB interface to bridge the I2C interface. The MCXN236 microcontroller has one HS USB interface and seven I2C interfaces.

1.1 MCXN236 I2C interface features

The I2C interface on MCXN236 supports:

- Controller mode, which can support Standard mode, Fast mode, Fast+ mode, and Ultra-Fast mode
- Target mode, which can support HS mode
- Multi-controller, including synchronization and arbitration, which means that any number of controller nodes can be present. Also, controller and target roles can be changed between messages (after a Stop signal is sent)
- Clock stretching, which is used on the SCL line as an I2C flow control mechanism
- Arbitration for when the system has more than one controller. When used on the SDA line, ensure to have only one I2C transmitter at a time
- General call, 7-bit addressing, and 10-bit addressing
- Software reset, Start byte, and device ID (also require software support)

2 MCXN236 USB to I2C demo introduction

In the USB to I2C demo on MCXN236 device, the USB device uses USB CDC virtual com class to communicate with the PC host. Use the terminal tool to send a serial data to control the I2C interface. The terminal tool used in the following contents is the `pzh-py-com` tool. The customer can download it from here: https://github.com/JayHeng/pzh-py-com.

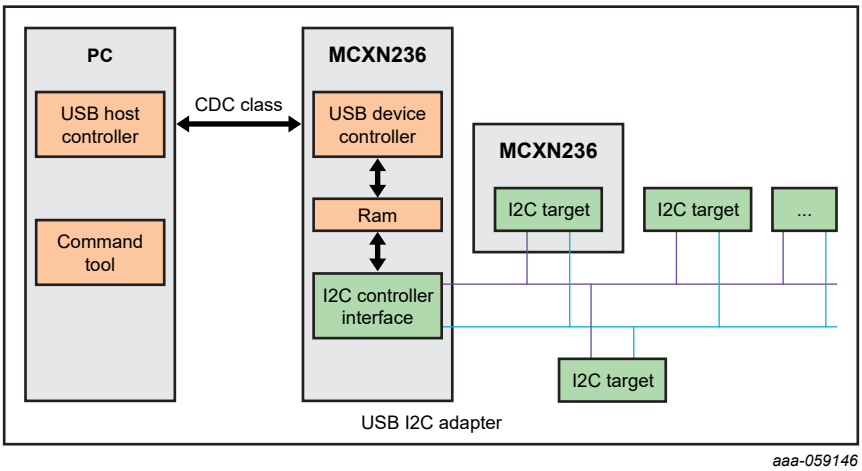

Figure 1. MCXN236 USB to I2C demo hardware block diagram

2.1 Hardware setup

To perform the MCXN236 USB to I2C demo, use two FRDM-MCXN236 boards:

- One board is used as an I2C controller
- Another board is used as an I2C target.

This demo uses P4_0(I2C2_SDA) and P4_1(I2C2_SCL) pins as I2C function.

For hardware connection, see Figure 2. The connecting wires must be kept as short as possible.

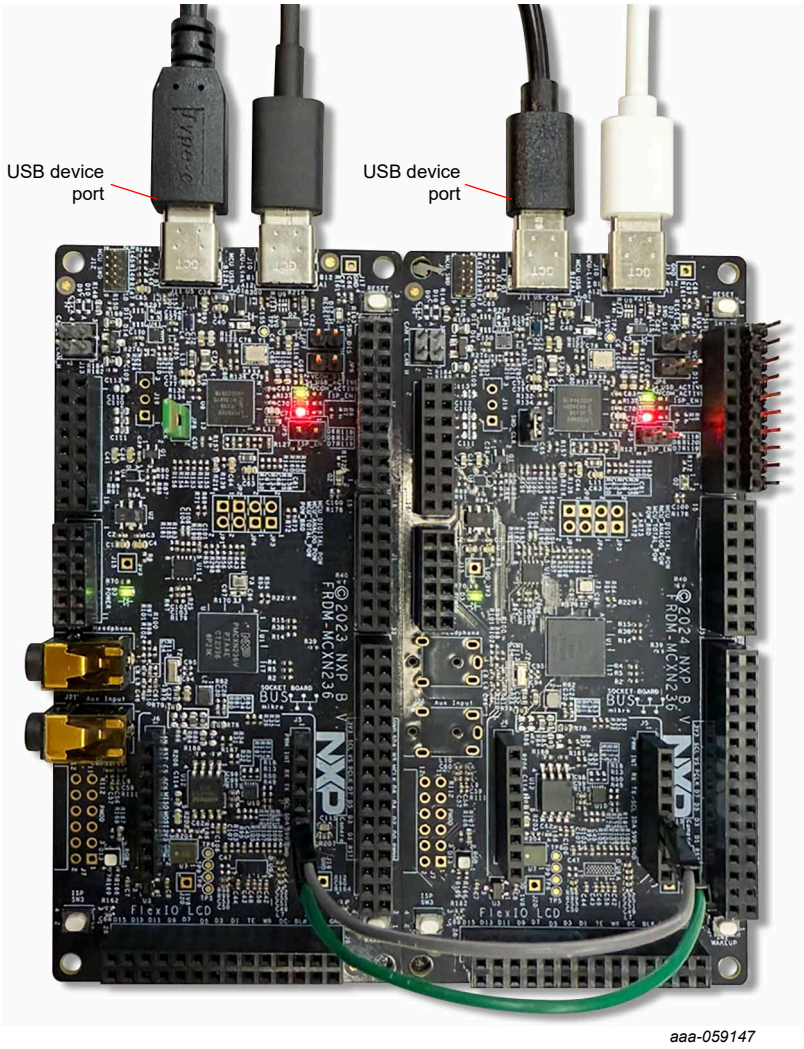

Figure 2. MCXN236 USB to I2C demo hardware setup

2.2 Software introduction

The MCXN236 USB to I2C demo provides the following commands for I2C configuration via USB:

- Section 2.2.1 "TargetBufferFill command"
- Section 2.2.2 "TargetStart command"
- Section 2.2.3 "Write without register address command" (Controller direct write)
- Section 2.2.5 "Read without register address command" (Direct read)
- Section 2.2.4 "Write with register address command"

- Section 2.2.6 "Read with register address command"

To perform the related I2C functions, the customer can use these commands.

To test this demo, use a terminal tool that supports a multi-strings send function.

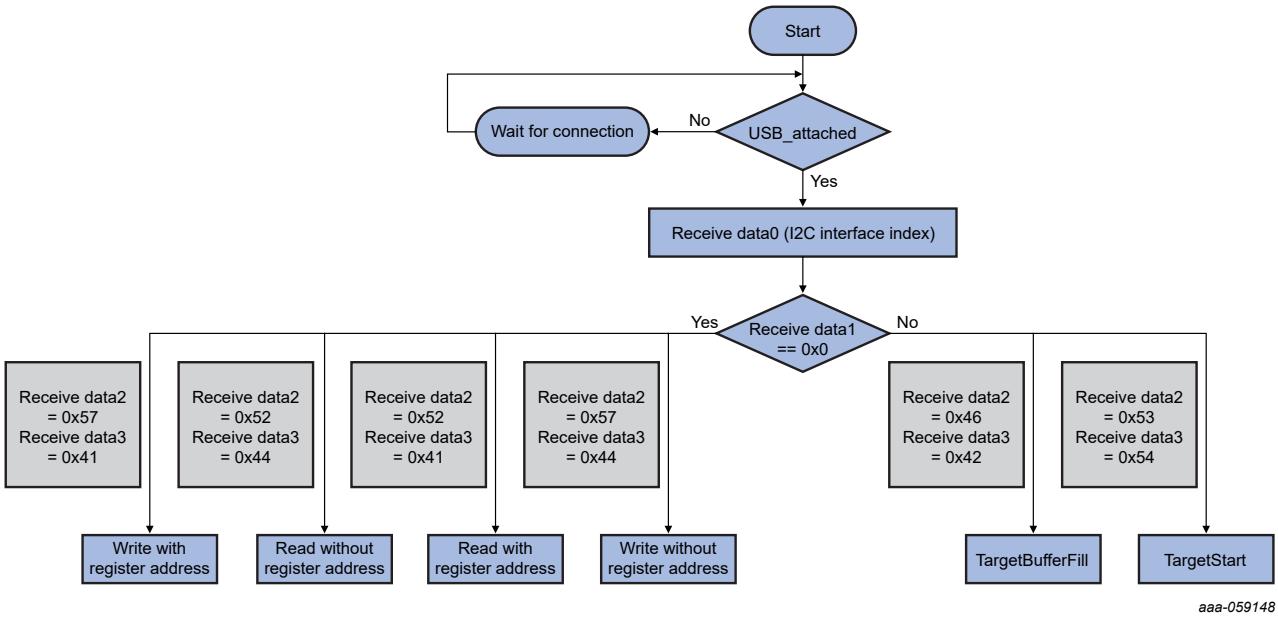

Figure 3. MCXN236 USB to I2C demo software flow

2.2.1 TargetBufferFill command

The `TargetBufferFill` command is used to fill the target send buffer used to store data to transfer to the controller. Figure 4 shows the TargetBufferFill command structure (send ASCII code).

Supported commands							
TargetBufferFill							
USB virtual com send	Interface index	I2C role	F	B	BufferAddress	len	Data(N bytes)
	0x2	0x1	0x46	0x42			
USB virtual com receive							

aaa-059149

Figure 4. TargetBufferFill command structure

The customer can pre-fill the target send buffer with the register address.

In this demo, when the terminal sends the `TargetBufferFill` command, the data is filled to the target send buffer with the register address, see Figure 5.

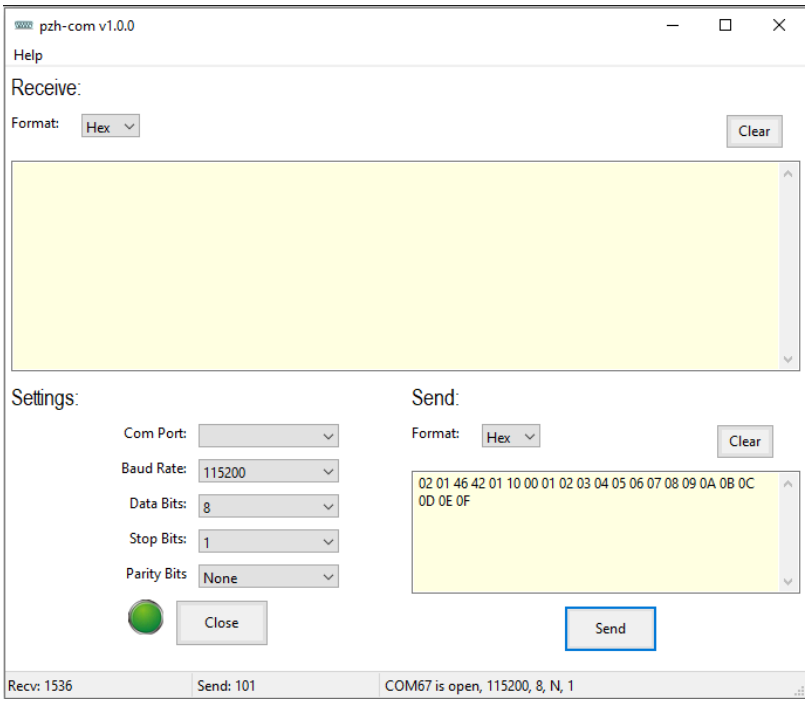

Figure 5. TargetBufferFill command terminal display

2.2.2 TargetStart command

The `TargetStart` command is used to make the target start to transfer. Figure 6 shows the command structure (send ASCII code).

Supported commands						
TargetStart						
USB virtual com send	Interface index	I2C role	T	S	TargetAddress	
	0x2	0x1	0x53	0x54		
USB virtual com receive						

aaa-059150

Figure 6. TargetStart command structure

In this demo, when the terminal sends the `TargetStart` command, the target starts to transfer, see Figure 7.

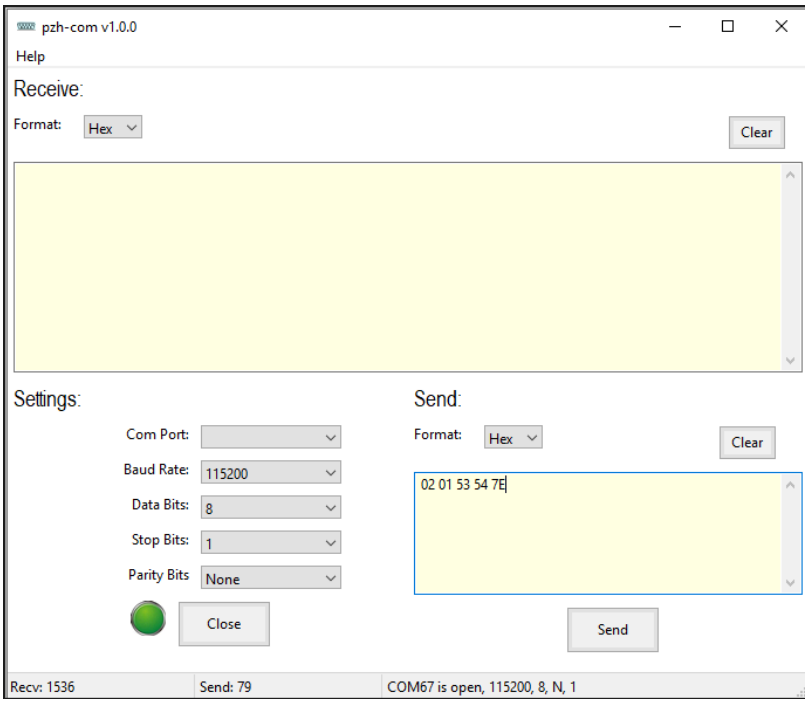

Figure 7. TargetStart command terminal display

2.2.3 Write without register address command

Figure 8 shows `Write without register address` command structure (send ASCII code). When using this command, provide some information that contains the target address/data size and write data.

Supported commands							
Write without register address							
USB virtual com send	Interface index	I2C role	W	N	TargetAddress	len	Data(N bytes)
	0x2	0x0	0x57	0x44			
USB virtual com receive	O				K		
	0x4F				0x4B		

aaa-059151

Figure 8. Write without register address command structure

Figure 9 shows the `Write without register address` command OUT and IN structures in terminal. After the command finishes executing, the terminal receives OK (0x4F, 0x4B) characters.

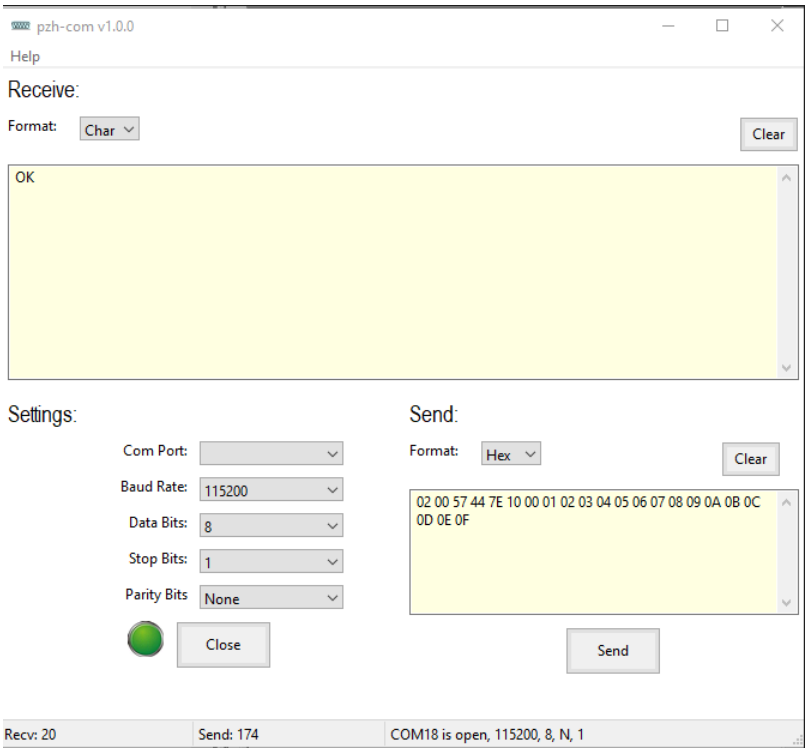

Figure 9. Write without register address command terminal display

Figure 10 shows that the target received data after controller finished write data through I2C interface.

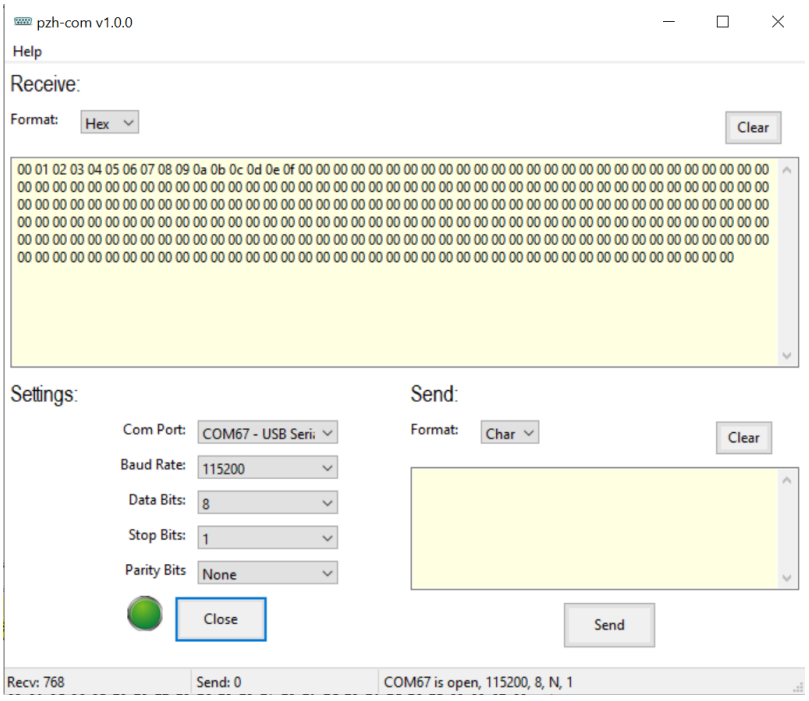

Figure 10. Target received terminal display

Figure 11 shows the `Write without register address` command process I2C timing.

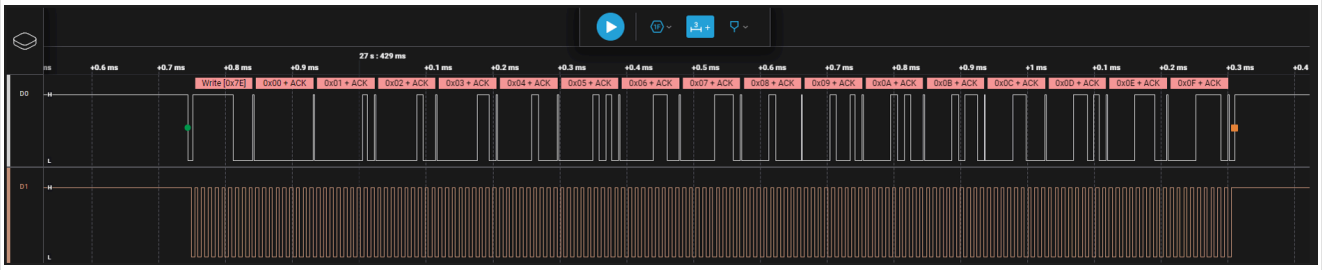

Figure 11. Write without register address command process timing

2.2.4 Write with register address command

Figure 12 shows `Write with register address` command structure (send ASCII code). When using this command, provide some information that contains the target address/data size and write data.

Supported commands								
Write with register address								
USB virtual com send	Interface index	I2C role	W	A	TargetAddress	RegAddress	len	Data(N bytes)
	0x2	0x0	0x57	0x41				
USB virtual com receive	O				K			
	0x4F				0x4B			

aaa-059152

Figure 12. Write with register address command structure

Figure 13 shows the `Write with register address` command OUT and IN structures in terminal. After the command finishes executing, the terminal receives OK (0x4F, 0x4B) characters.

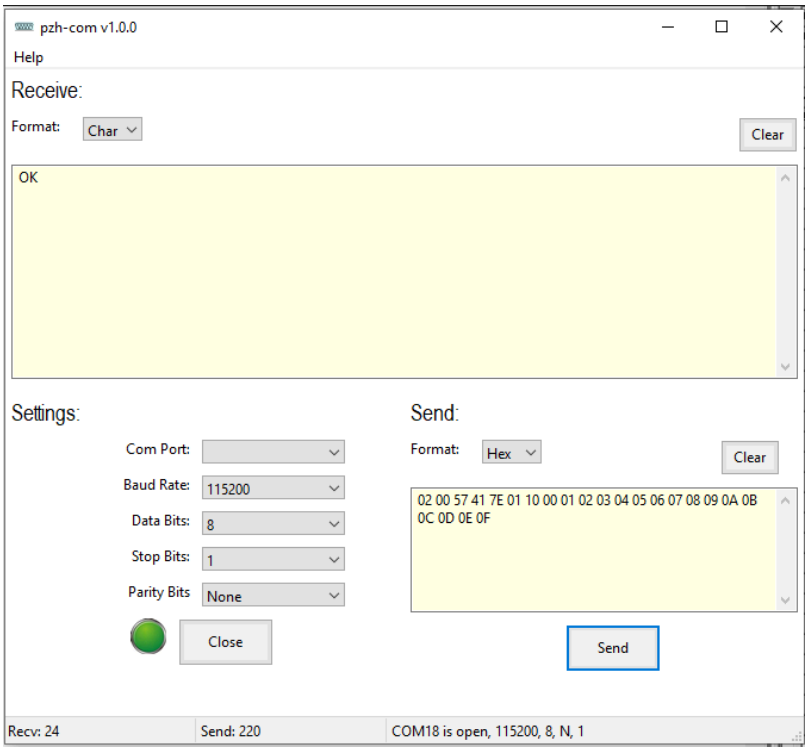

Figure 13. Write with register address command terminal display

Figure 14 shows that the target received data after controller finished write data through I2C interface.

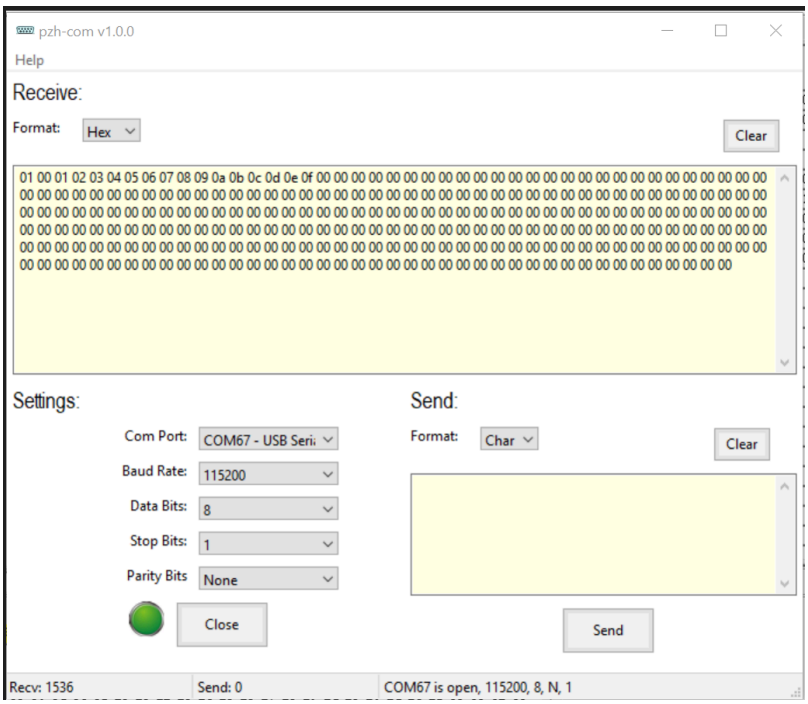

Figure 14. Target received terminal display

Figure 15 shows the `Write with register address` command process I2C timing.

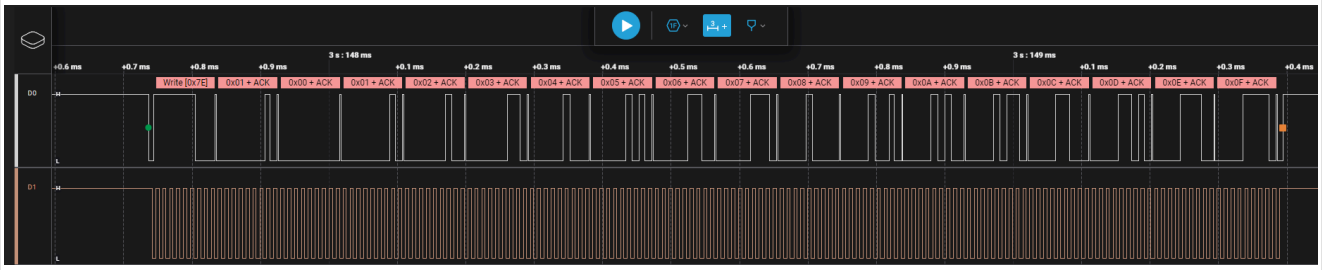

Figure 15. Write with register address command process timing

2.2.5 Read without register address command

Figure 16 shows `Read without register address` command structure (send ASCII code). When using this command, provide some information that contains the target address/data size and write data.

Supported commands						
Read without register address						
USB virtual com send	Interface index	I2C role	R	N	TargetAddress	len
	0x2	0x0	0x52	0x44		
USB virtual com receive	Data(N bytes)					

aaa-059153

Figure 16. Read without register address command structure

Figure 17 shows the `Read without register address` command OUT and IN structures in terminal. After the command finishes executing, the terminal receives register data sent by the target.

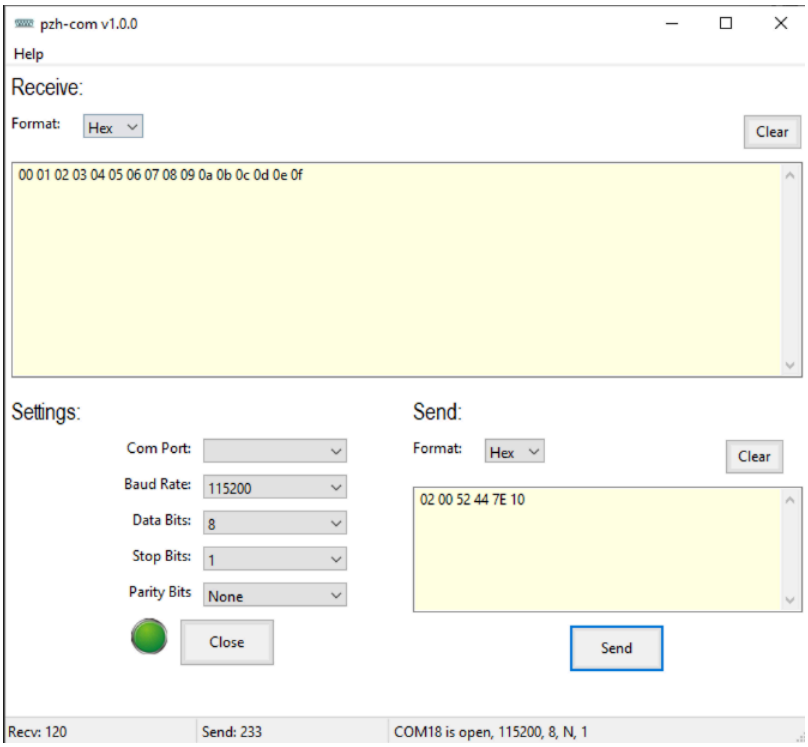

Figure 17. Read without register address command terminal display

Figure 18 shows the `Read without register address` command process I2C timing.

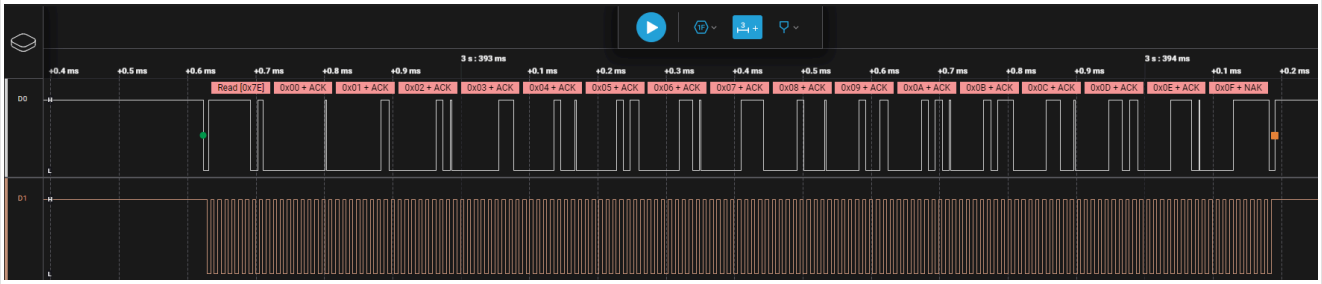

Figure 18. Read without register address command process timing

2.2.6 Read with register address command

Figure 19 shows `Read with register address` command structure (send ASCII code). When using this command, provide some information that contains the target address/data size and write data.

Supported commands							
Read with register address							
USB virtual com send	Interface index	I2C role	R	A	TargetAddress	RegAddress	len
	0x2	0x0	0x52	0x41			
USB virtual com receive	Data(N bytes)						

aaa-059154

Figure 19. Read with register address command structure

Figure 20 shows the `Read with register address` command OUT and IN structures in terminal. After the command finishes executing, the terminal receives register data sent by the target.

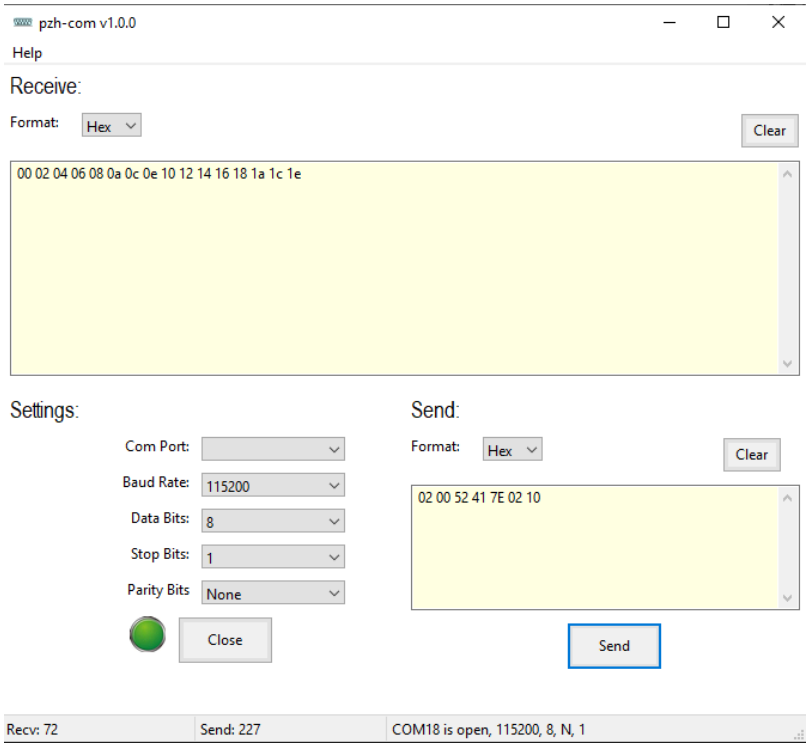

Figure 20. Read with register address command terminal display

Figure 21 shows the `Read with register address` command process I2C timing.

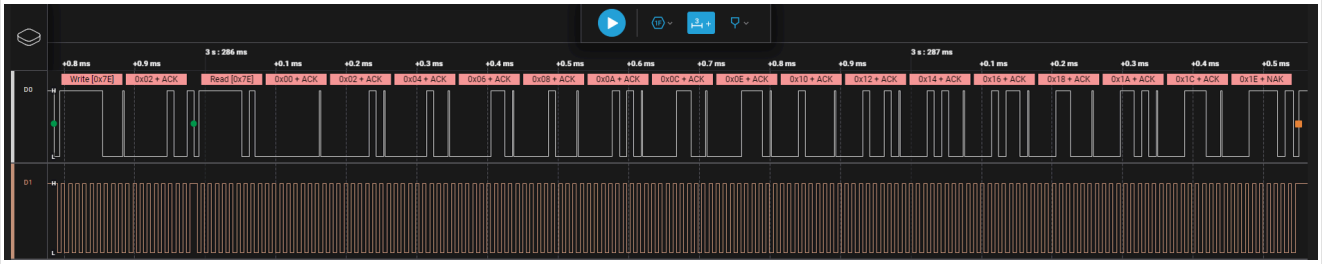

Figure 21. Read with register address command process timing

3 Revision history

Table 1 summarizes the revisions to this document.

Table 1. Revision history

Document ID	Release date	Description
AN14565 v.1.0	30 January 2025	Initial public release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) will cause permanent damage to the device. Limiting values are stress ratings only and (proper) operation of the device at these or any other conditions above those given in the Recommended operating conditions section (if present) or the Characteristics sections of this document is not warranted. Constant or repeated exposure to limiting values will permanently and irreversibly affect the quality and reliability of the device.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

Quick reference data — The Quick reference data is an extract of the product data given in the Limiting values and Characteristics sections of this document, and as such is not complete, exhaustive or legally binding.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Contents