

# AN14266

## I2C Monitor Mode Using i.MX RT500

Rev. 1.0 — 1 April 2024

Application note

### Document information

Information	Content
Keywords	AN14266, I2C, Monitor mode, i.MX RT500
Abstract	This application note introduces how to enable the Monitor mode of the I2C to facilitate debugging of the I2C during application development.



## 1 Introduction

The i.MX RT500 is a family of dual-core microcontrollers for embedded applications. It features an Arm Cortex-M33 CPU combined with a Cadence Xtensa Fusion F1 Audio Digital Signal Processor CPU. The Cortex-M33 includes two hardware coprocessors providing enhanced performance for an array of complex algorithms. The family offers a rich set of peripherals and a low-power consumption.

The device consists of the following:

- 5 MB SRAM
- Two FlexSPIs (octal/quad SPI Interfaces) each with 32 kB cache, one with dynamic decryption, high-speed USB device/host + PHY, 12-bit 1 MS/s ADC
- Analog comparator
- Audio subsystems supporting up to eight DMIC channels
- 2.5 D vector GPU and LCD controller with MIPI DSI PHY
- Two SDIO/eMMC
- FlexIO
- AES/SHA/Crypto M33 coprocessor
- PUF key generation

The i.MX RT500 provides connectivity interfaces such as UART, SPI, I2C, and I2S. This application note introduces how to enable the Monitor mode of the I2C. It also provides information about events on the I2C-bus to facilitate debugging of the I2C during application development.

To achieve the I2C monitor feature, perform the following steps:

- Enable the Monitor mode in the CFG register
- Enable I2C status flags to generate the interrupt when I2C even occurs
- To discover the event that triggered the interrupt and information about the event, read the INTSTAT/STAT registers

## 2 I2C monitor mode

The Monitor mode provides information about events on the I2C-bus, which includes data movement, data acknowledgment, start/stop events, and so on.

The time-out function provides information about I2C time-out events as follows:

- SCL time-out: Indicates when SCL has remained low for a longer time than specified by the TIMEOUT register.
- Event time-out: Indicates when the time between events has remained longer than the time specified by the TIMEOUT.

The Monitor mode and time-out function are complementary features that can be enabled independently.

When functions are enabled, the user must specify the status flags that generate an interrupt. Once done, each time an event occurs, an interrupt gets generated and can be processed in the proper interrupt handler by reading the INTSTAT register.

### 2.1 Registers modification

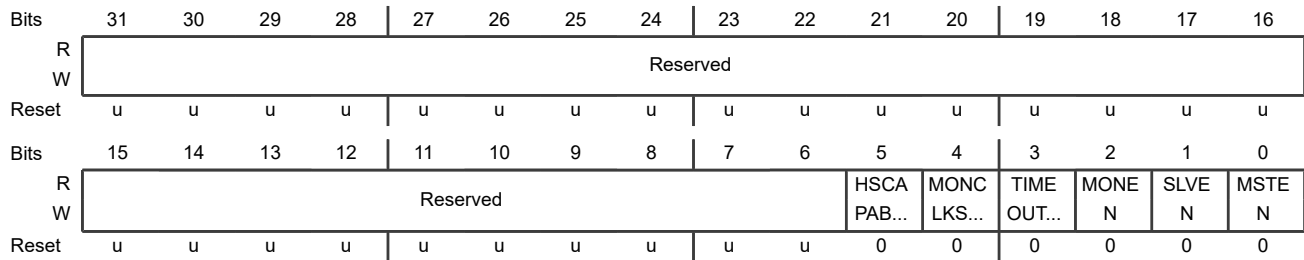
This section describes how to program the registers to enable the I2C functions required for the Monitor mode as follows:

- To enable the desired functions, one must modify the CFG register of the I2C module by enabling the corresponding bit.

Table 1. Offset

Register	Offset
CFG	800h

Table 2. Diagram



- Therefore, to enable the monitor and time-out function, enable the bit position 2 and 3.

Table 3. Fields

Field	Function
3 TIMEOUTEN	I2C bus Time-out Enable 0b - Disabled. The time-out function is disabled. When disabled, the time-out function is internally reset. 1b - Enabled. The time-out function is enabled. If those flags are enabled, both types of time-out flags are generated and causes interrupts. Typically, only one time-out flag is used in a system.
2 MONEN	Monitor Enable 0b - Disabled. The I2C Monitor function is disabled. When disabled, the Monitor function configuration settings are not changed, but the Monitor function is internally reset. 1b - Enabled. The I2C Monitor function is enabled.

- To enable status flags that generate an interrupt, modify the INTENSET register of the I2C module by writing "1" to the corresponding bit.
- The user can overwrite the `FLEXCOMMx_IRQHandler` function to process the event each time an interrupt is generated.
- The INTSTAT register of the I2C module provides information on the active flag, that is, the source of the interrupt and the STAT register for state information. For instance, if `INSTAT` and `0x10000 = 1`, then the `MONRDY` generates the interrupt.

Table 4. Offset

Register	Offset
INTSTAT	818h

Table 5. Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
R	Reserved								SCLT IME...	EVEN TTI...	Reserved				MONI DLE	Rese rved	MONO V	MONR DY
W																		
Reset	u	u	u	u	u	u	0	0	u	u	u	u	0	u	0	0		
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R	SLVD ESEL	Reserved				SLVN OTS...	Reserved		SLVP END...	Rese rved	MSTS TST...	Rese rved	MSTA RBL...	Reserved			MSTP END...	
W																		
Reset	0	u	u	u	1	u	u	0	u	0	u	0	u	u	u	1		

Table 6. Offset

Register	Offset
STAT	804h

Table 7. Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
R	Reserved								SCLT IME...	EVEN TTI...	Reserved				MONI DLE	MONA CTI...	MONO V	MONR DY
W																		
Reset	u	u	u	u	u	u	0	0	u	u	u	u	0	0	0	0		
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R	SLVD ESEL	SLVS EL	SLVIDX	SLVN OTS...	SLVSTATE	SLVP END...	Rese rved	MSTS TST...	Rese rved	MSTA RBL...	MSTSTATE				MSTP END...			
W																		
Reset	0	0	0	0	1	0	0	0	u	0	u	0	0	0	0	1		

- For more information on MONRDY, read the corresponding bit (position 16) in the STAT register. Therefore, the Monitor mode generates the interrupt because data is waiting to be read in the MONRXDAT register.

Table 8. Fields

Field	Function
16 MONRDY	Monitor Ready The MONRDY flag is cleared when the MONRXDAT register is read. 0b - No data. The Monitor function does not currently have data available. 1b - Data waiting. The Monitor function has data waiting to be read.

Table 9. Offset

Register	Offset
MONRXDAT	880h

Table 10. Diagram

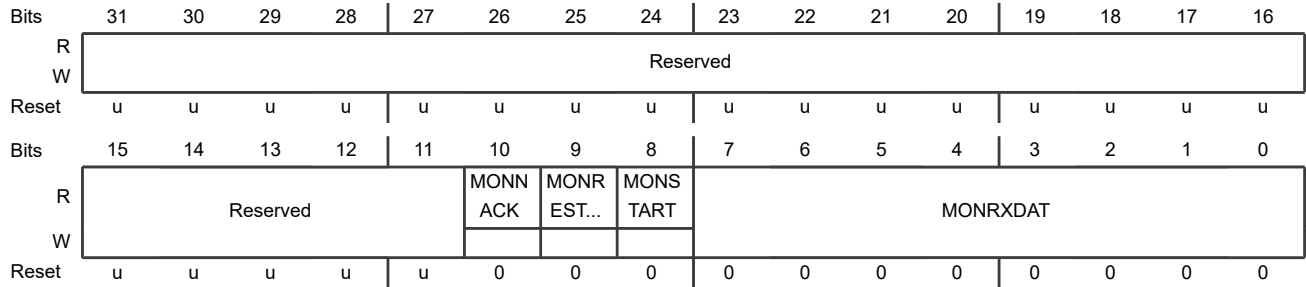


Table 11. Fields

Field	Function
31-11 —	Reserved The read value is undefined; only zero must be written.
10 MONNACK	Monitor Received NACK 0b - Acknowledged. At least one master or slave receiver acknowledges the data currently being provided by the Monitor function. 1b - Not acknowledged. None of the receivers has acknowledged the data currently being provided by the Monitor function.
9 MONRESTART	Monitor Received Repeated Start 0b - No repeated start detected. The Monitor function has not detected a Repeated Start event on the I2C bus. 1b - Repeated start detected. The Monitor function has detected a Repeated Start event on the I2C bus.
8 MONSTART	Monitor Received Start 0b - No start detected. The Monitor function has not detected a Start event on the I2C bus. 1b - Start detected. The Monitor function has detected a Start event on the I2C bus.
7-0 MONRXDAT	Monitor function Receiver Data It reflects every data byte that passes on the I2C pins.

- The user can easily know the type of data waiting to be read. If MONRXDAT and 0x200 = 1, a repeated start has been detected in the I2C bus.
- The flag that generates the interrupt must be cleared after being processed. In the case of MONRDY interrupt, the flag is automatically cleared by reading the MONRXDAT register. For other cases, writing "1" to itself in the STAT register clears the flag. For instance, writing 0x1000000 clears the flag at bit position 24, that is, EVENTTIMEOUT.

For more information, refer to *i.MX RT500 Low-Power Crossover MCU Reference Manual* (document [IMXRT500RM](#)).

### 3 Software example

This example is based on the i.MX RT500 SDK demo `evkmimxrt595_i2c_accel_event_trigger`, which uses SDK 2.13.1 version and MCUXpresso IDE 11.7.

Modification is added to the project to enable the Monitor mode with time-out function, without impacting the behavior of the standard demo. Therefore, only related modifications are highlighted in this application note. Also, this software example introduces how to enable and apply the different I2C features.

The standard demo demonstrates how to wake up the main device in Low-power mode with the accelerometer trigger event. The accelerometer can keep working while the main device is in Low-power mode (or Deep-sleep mode). Only when the configured event is captured, the accelerometer trigger the interrupt to wake up the main device. This example uses I2C to configure the accelerometer to work in 800 Hz data rate with Low-noise mode. The main device wakes up when the tap event is triggered and 32 samples around the trigger event are captured.

### 3.1 Code modification

In this example, the FLEXCOMM4 is programmed as I2C to communicate with the onboard accelerometer.

Monitor and time-out features are enabled by modifying the CFG register of the I2C module as mentioned in [Section 2.1 "Registers modification"](#). This example uses I2C4 (0x40122000).

```
void APP_MonitorInit(I2C_Type *base)
{
    /* set Monitor enable */
    base->CFG = (base->CFG & (uint32_t)I2C_CFG_MASK) | I2C_CFG_MONEN_MASK;
}
void APP_EnableTimeOut(I2C_Type *base)
{
    /* set Timeout enable */
    base->CFG = (base->CFG & (uint32_t)I2C_CFG_MASK) | I2C_CFG_TIMEOUTEN_MASK;
}
```

Different flags can be set to generate an interrupt. The following flags are set related to I2C master, monitor, and time-out functions:

- Master-related flags: Master arbitration loss enabled (MSTARBLOSSEN) and Master start/stop error interrupt enabled (MSTSTSTPERREN).
- Monitor-related flags: Monitor data ready interrupt enabled (MONRDYEN).
- Time-out related flags: Event time-out interrupt enabled (EVENTTIMEOUTEN) and SCL time-out interrupt enabled (SCLTIMEOUTEN).

**Note:** Depending on the application, other flags can be set/unset.

To catch and process the interrupt, the IRQ related to the FLEXCOMM4 must be enabled:

```
void APP_EnableInterrupts(I2C_Type *base)
{
    uint32_t all_interrupts;
    all_interrupts |= I2C_INTENSET_MONRDYEN_MASK;
    all_interrupts |= I2C_INTENSET_MSTARBLOSSEN_MASK | I2C_INTENSET_MSTSTSTPERREN_MASK;
    all_interrupts |= I2C_INTENSET_EVENTTIMEOUTEN_MASK | I2C_INTENSET_SCLTIMEOUTEN_MASK;
    I2C_EnableInterrupts(base, all_interrupts);
    DisableIRQ(FLEXCOMM4_IRQn);
    IRQ_ClearPendingIRQ(FLEXCOMM4_IRQn);
    EnableIRQ(FLEXCOMM4_IRQn);
}
```

The FLEXCOMM4 IRQ handler has been modified to print flags that have generated the interrupt and any available information about data content.

### 3.2 Results

Table 12 shows the debug console during the initialization of the accelerometer. The figure shows the correlation between the events captured by the interrupt handler, and the I2C sequence in the code mentioned in Section 3.1 "Code modification".

Table 12. Debug console

<pre>I2C example -- Accelerometer Event Trigg -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = d -- [ACK detected] -- -- [REPEATED START detected] -- READ @ addr = 1e -- [ACK detected] -- data = c7 -- [NACK detected] -- -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 26 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 5 -- [ACK detected] -- data = d4 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = a -- [ACK detected] -- data = 8 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 21 -- [ACK detected] -- data = 15 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 23 -- [ACK detected] -- data = 19 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 24 -- [ACK detected] -- data = 19 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 25 -- [ACK detected] -- data = 28 -- [ACK detected] -- -- [START event detected] --</pre>	<pre>WRITE @ addr = 1e -- [ACK detected] -- data = 26 -- [ACK detected] -- data = 50 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 27 -- [ACK detected] -- data = f0 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 2d -- [ACK detected] -- data = 8 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 2e -- [ACK detected] -- data = 8 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = e -- [ACK detected] -- data = 1 -- [ACK detected] -- -- [START event detected] -- WRITE @ addr = 1e -- [ACK detected] -- data = 2a -- [ACK detected] -- data = 5 -- [ACK detected] -- Press any key to enter low power mode</pre>
---	--

## 4 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2024 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR

BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## 5 Revision history

[Table 13](#) summarizes the revisions to this document.

**Table 13. Revision history**

Document ID	Release date	Description
AN14266 v.1.0	1 April 2024	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.

**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.

**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.

**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](mailto: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.

**AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro,  $\mu$ Vision, Versatile** — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

**Cadence** — the Cadence logo, and the other Cadence marks found at [www.cadence.com/go/trademarks](http://www.cadence.com/go/trademarks) are trademarks or registered trademarks of Cadence Design Systems, Inc. All rights reserved worldwide.

**i.MX** — is a trademark of NXP B.V.

**I2C-bus** — logo is a trademark of NXP B.V.

---

## Contents

---

<b>1</b>	<b>Introduction .....</b>	<b>2</b>
<b>2</b>	<b>I2C monitor mode .....</b>	<b>2</b>
2.1	Registers modification .....	2
<b>3</b>	<b>Software example .....</b>	<b>5</b>
3.1	Code modification .....	6
3.2	Results .....	7
<b>4</b>	<b>Note about the source code in the</b>	
	<b>document .....</b>	<b>7</b>
<b>5</b>	<b>Revision history .....</b>	<b>8</b>
	<b>Legal information .....</b>	<b>9</b>

---

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

---