

# Freescaler MQX RTOS Example Guide

## IIC example

This document explains the IIC driver example, what to expect from the example and a brief introduction to the IIC driver API.

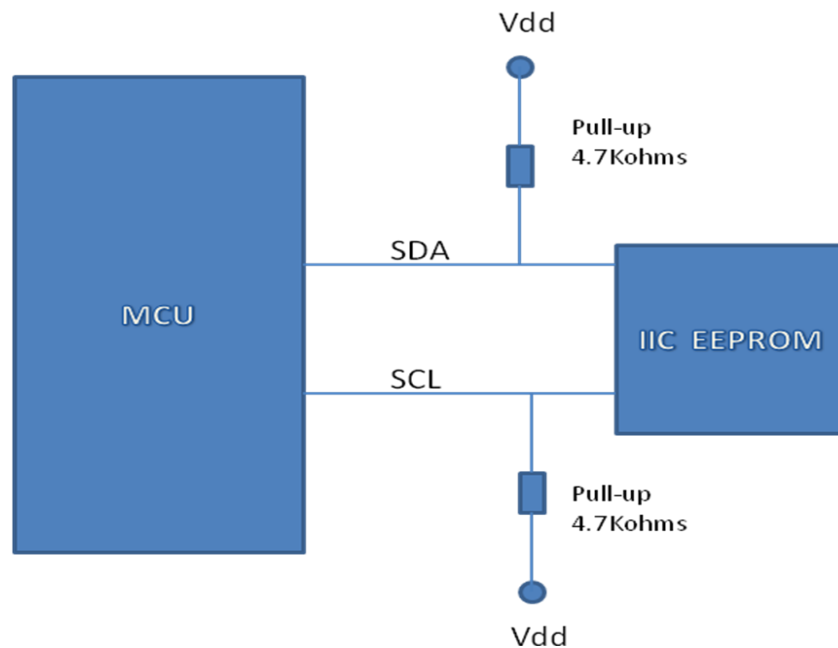
### The Example

The example shows the usage of the IIC driver as a master using both polling or interruption drivers and an EEPROM 24LC16 as slave device. However any 24LCxx device can be implemented. For writing this document, a 24LC04 was used.

### Running the example

The connections needed for running this example are:

- Serial cable connected to the UART used, this may vary between targets. And a terminal set to 115200 baud, no parity, 8 bits.
- Wire SDA and SCL with the corresponding pull-up resistors from your target to the EEPROM device.
- If necessary provide Vdd and GND to the EEPROM from your board.



**Figure 1**  
**Communications lines**

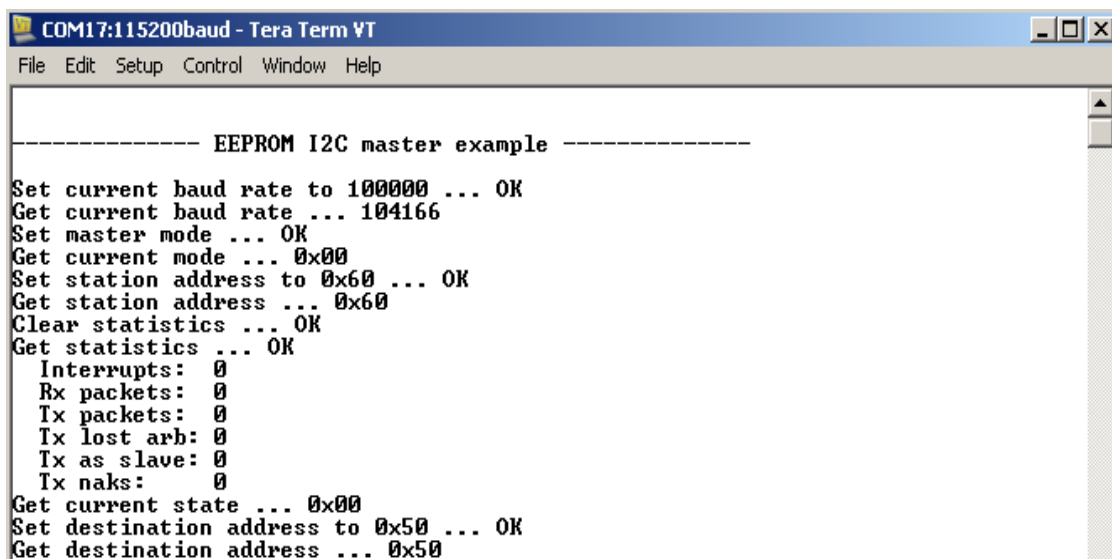
After the connections are set, now the application can be executed. Verify that the target BSP has the IIC driver installed (either polled or interruption), if not, please add the proper macro and rebuild the libraries.

## Explaining the example

In the example, polled mode and interrupt mode work in the same way and only support synchronous API.

The first, driver example will open the IIC driver and will test the different IOCTL commands that are available on the driver, such as:

- IO\_IOCTL\_I2C\_GET\_BAUD
- IO\_IOCTL\_I2C\_SET\_MASTER\_MODE
- IO\_IOCTL\_I2C\_GET\_MODE
- IO\_IOCTL\_I2C\_SET\_STATION\_ADDRESS
- IO\_IOCTL\_I2C\_GET\_STATISTICS



```
COM17:115200baud - Tera Term VT
File Edit Setup Control Window Help

----- EEPROM I2C master example -----
Set current baud rate to 100000 ... OK
Get current baud rate ... 104166
Set master mode ... OK
Get current mode ... 0x00
Set station address to 0x60 ... OK
Get station address ... 0x60
Clear statistics ... OK
Get statistics ... OK
  Interrupts: 0
  Rx packets: 0
  Tx packets: 0
  Tx lost arb: 0
  Tx as slave: 0
  Tx naks: 0
Get current state ... 0x00
Set destination address to 0x50 ... OK
Get destination address ... 0x50
```

**Figure 2**  
**Example output before writing to EEPROM Addresses**

After testing the IOCTL commands, the example will start different variations of read/write operations to the driver like:


```
printf ("Test write 0 bytes ... ");
```

```
result = fwrite (&param, 1, 0, fd);
```

This will write "1" block of "0" bytes in the driver. These operations are just demonstrative of the usage of the driver.

The example implements a couple of functions that performs read/write to the EEPROM with the below functions:

- o i2c\_write\_eeprom
- o i2c\_read\_eeprom



The screenshot shows a terminal window with a black background and white text. The text displays a sequence of I2C operations: reading 53 bytes from address 0x0000001a, setting the I2C bus address to 0x50, writing to address 0x1a, flushing, initiating a repeated start, setting the number of bytes requested to 53, reading 53 bytes, and stopping the transfer. All operations are marked as 'OK'. The final line shows the received data as a string of lowercase and uppercase letters: 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'.

```
Reading 53 bytes from address 0x0000001a ...  
Set I2C bus address to 0x50 ... OK  
Write to address 0x1a ... OK  
Flush ... OK  
Initiate repeated start ... OK  
Set number of bytes requested to 53 ... OK  
Read 53 bytes ... OK  
Stop transfer ... OK  
Received: abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
```

**Figure 3**

**Extract of the example output on EEPROM read/write operations**

After it performs the read/write operations, the example will report the statistics, close the driver and exit MQX.