Application Note

How to Program I²C for MCx83xx Device Family

Amrit Das, Afraj Shaikh

ABSTRACT

I²C protocol is one of the most preferred communication protocols for a low-cost microcontroller system where on-board communication is required between the controllers and peripherals. This application note provides description of all I²C features supported across the MCx83xx family. Further, the application note explains how the primary device can be programmed to use these I²C features as per the need.

Table of Contents

Trademarks

All trademarks are the property of their respective owners.

1 Introduction

I²C involves two communication lines, serial clock line (SCL) over which the primary device sends clock signal and serial data line (SDA) which primary device uses for either reading a data (from secondary device address) or writing a data (to the secondary device address). Because of high power-efficiency and easy implementation, I²C has been adopted within all the devices of MCx83xx family.

This article includes details on- I²C for the MCx83xx family as follows:

- MCF8315A
- MCF8315C
- MCF8315C-Q1
- MCF8316A
- MCF8316C
- MCF8316C-Q1
- MCF8315D
- MCF8316D
- MCF8329A
- MCT8316A
- MCT8315A
- MCT8329A

2 I²C Related Details for MCx83xx Family

The data word format which the MCx83xx family follows for the I²C communication across all devices is shown in Table 2-1.

Table 2-1. I²C Data Word Format

TARGET_ID	R/W	CONTROL_WORD	DATA	CRC-8
A6 – A0	W0	CW23 - CW0	D15/D31/D63 - D0	C7 – C0

Table 2-2. I²C Control Word Format

OP_R/W	CRC_EN	DLEN	MEM_SEC	MEM_PAGE	MEM_ADDR
CW23	CW22	CW21 – CW20	CW19 – CW16	CW15 – CW12	CW11 – CW0

Table 2-1 shows that the format consists of 7-bit TARGET_ID, followed by a read write bit (R/W bit). After this packet we send a 24-bit control word followed by (16/32/64 bit) DATA, and 8-bit CRC if CRC_EN is set in CONTROL_WORD. The data length can be configured within the CONTROL_WORD.

Detailed description for all fields can be found in the MCx83xx device data sheet under *EEPROM access and I2C interface* section. This section provides some key things to note regarding the usage of these fields.

2.1 TARGET_ID

TARGET_ID is a 7-bit value representing the target address of the MCx83xx device. The TARGET_ID default value is 0x01. The MCx83xx family supports only the 7-bit addressing mode.

1. **TARGET_ID Configuration:** For field oriented control (FOC) devices, TARGET_ID can be configured in DEVICE_CONFIG1 register by setting the 'I2C_TARGET_ADDR'. For trapezoidal control devices (MCT83xx), TARGET_ID can be configured in PIN_CONFIG2 register by setting the 'I2C_TARGET_ADDR'.

   Note

   TARGET_ID change does not take effect at runtime and device continues to communicate with the address burned in EEPROM at the time of power-up. After changing the TARGET_ID, the EEPROM needs to be programmed with the updated TARGET_ID and device needs to be power cycled.

2. **TARGET_ID Detection:** If primary device fails to communicate with default or preprogrammed TARGET_ID, a search can be implemented to find TARGET_ID. Figure 2-1 explains the TARGET_ID detection steps.

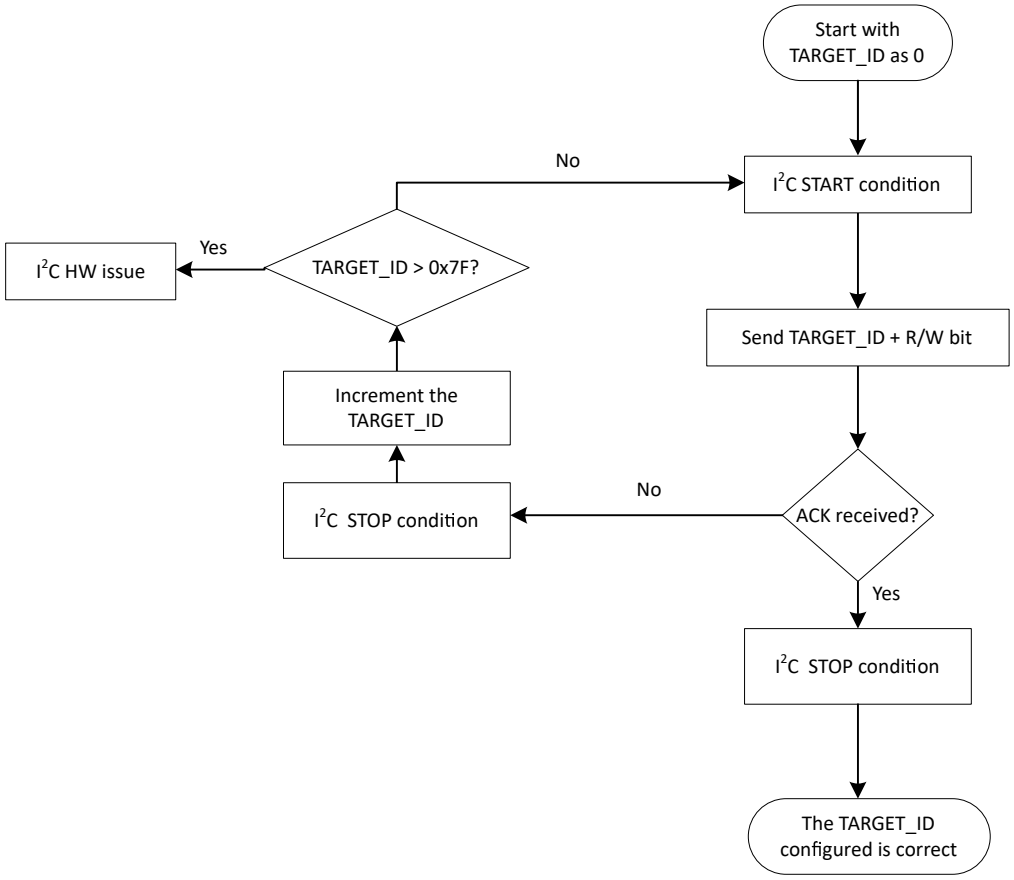

Figure 2-1. TARGET_ID Detection Flow Chart

Address not found can occur due to following possible reasons:

1. SCL, SDA connections are not proper.
   a. Make sure signals from SCL/SDA pins are routed to the device and all connections are made including necessary pull-up resistors for both pins.
   b. Make sure no other communication or other peripheral is using the same pins. The SCL/SDA pins must be permanently configured for I²C communication.
2. MCx83xx is not powered up.
3. Make sure that there is no excessive capacitive loading on SCL and SDA lines. This can be verified by monitoring the SCL and SDA lines on an oscilloscope during communication.

2.2 CRC_EN

If CRC_EN = 1, cyclic redundancy check (CRC) gets enabled otherwise CRC is disabled.

1. **In write operation**
   Primary device is expected to send the CRC byte computed on the data and MCx83xx is expected to use the CRC byte to check the correctness of data received. The data to be included in CRC computation is as follows:
   a. Target ID + write bit.
   b. Control word – 3 bytes
   c. Data bytes – 2/4/8 bytes
2. **In read operation**
   MCx83xx send CRC byte at the end of data bytes. The primary device is expected compute CRC to check for correctness of data sent by MCx83xx by comparing computed CRC against received CRC byte. The data to be included in CRC computation is as follows:

a. Target ID + write bit
b. Control word – 3 bytes
c. Target ID + read bit
d. Data bytes – 2/4/8 bytes

Note

EEPROM parity does not affect the I²C CRC computation. I²C CRC is computed strictly on the bytes specified above.

2.2.1 CRC Computational Details

Following details needs to be taken care while computing CRC on the primary device side:

1. An 8-bit CCITT polynomial is used for CRC computation as shown in Equation 1

$$\left(x^8 + x^2 + x + 1\right) \quad (1)$$

2. The CRC value to be initialized to 0xFF.
3. The CRC computation starts from **Target ID + write bit** byte and continue till end of data bytes.
4. Within a byte the CRC is computed in a bit by bit manner starting with MSB bit of the input byte.
5. Example to verify correctness of CRC computation
   a. For input byte of 0x12, the CRC byte becomes 0x8D from the initial value of 0xFF.

2.3 MEM_SEC, MEM_PAGE, and MEM_ADDR

For the user accessible status and control registers, MEM_SEC = 0x0 and MEM_PAGE = 0x0. The address of the register to be read is the MEM_ADDR. For example, to read 0x80 location data, MEM_ADDR needs to be set to 0x080.

3 I²C Secondary Device Feature Supported by MCx83xx Family

3.1 Clock Stretching

Clock stretching in I²C protocol allows a secondary device to pull down SCL line to slow down communication from the primary device to manage data, store received data, or prepare to transmit another byte of data.

Note

Except for MCF8316A and MCT8316A, all the devices support clock stretching. The clock low timeout is configured as 4.66ms in the devices which support clock stretching.

4 Primary Device Read and Write Expected Flow

4.1 Read Sequence

Upon primary device read command, first configure the primary device with the TARGET_ID and R/W bit as write. Configure the primary device in Tx mode. Prepare the control words to be sent to the secondary device. Generate I²C START condition and send the control words along with TARGET_ID and R/W bit set as 0. If NACK received, retry sending data few times (recommended 5 retries). If still NACK received, generate STOP condition.

Note

Make sure at least 100us delay between each byte of data for reliable communication

After writes are completed, configure the primary device to be in Rx mode. Once all data bytes and CRC packet (if enabled in control words) is received, send I²C STOP condition to complete the transaction.

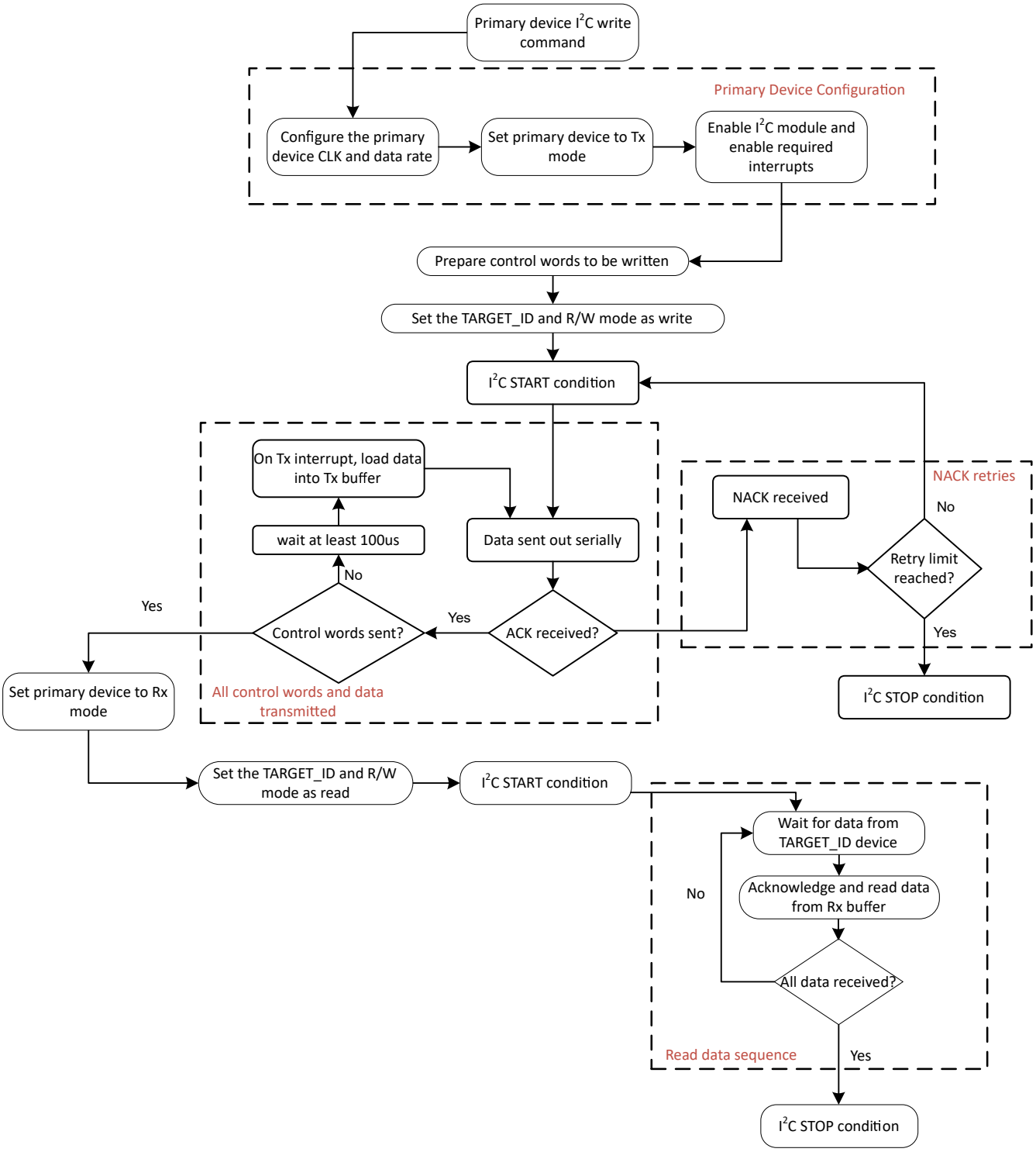

Figure 4-1. Primary Device I²C Read Expected

4.2 Write Sequence

Upon primary device write command, set the primary device to Tx mode. Prepare the data to be transmitted along with CRC bit if enabled. Set the TARGET_ID and set R/W bit as 1b before generating START condition. Transmit the control words and then the data byte by byte till all bytes are sent out and generate stop condition. If NACK received, retry sending data few times (recommended 5 retries). If still NACK received, generate STOP condition.

Note

Make sure at least 100us delay between each byte of data for reliable communication.

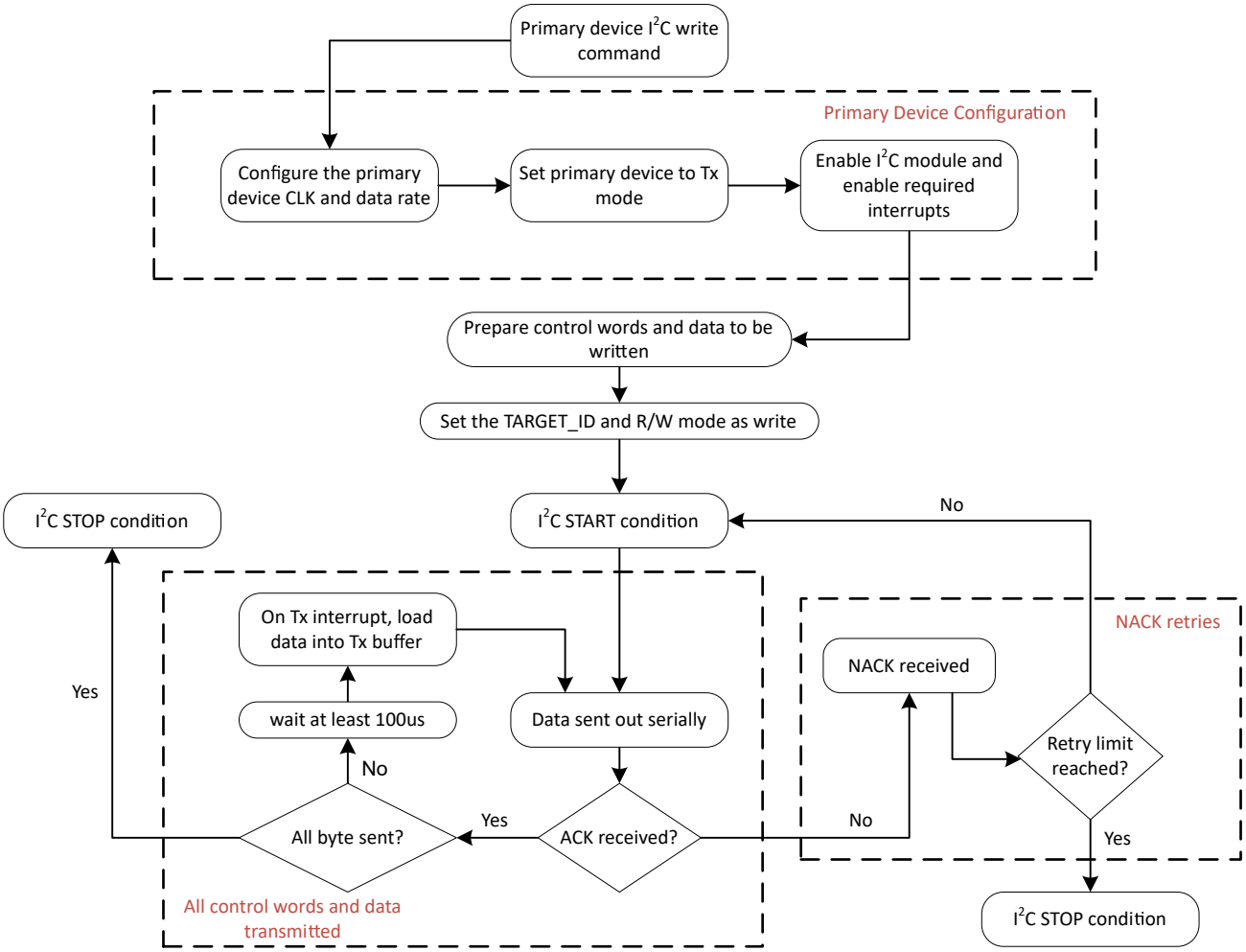

Figure 4-2. Primary Device I²C Write Expected

5 Summary

Reliable I^2C communication with MCx83xx devices is crucial for reading system statuses and to implement desired control for various application. This application note provides the detailed steps to find target device address, steps to debug failure, and implement the read or write transaction to help establish reliable and faster I^2C communication with MCx83xx.

6 References

- Texas Instruments , *MCF8316C-Q1 Sensorless Field Oriented Control (FOC) Integrated FET BLDC Driver*, data sheet.
- Texas Instruments, *MCF8316D Sensorless Field Oriented Control (FOC) Integrated FET BLDC Driver* data sheet.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265