AN12355

Using MSCAN on KE16Z MCU

Rev. 0 — 25 Febuary 2019 Application Note

Contents

1 Introduction

NXP's Scalable Controller Area Network (MSCAN) is a communication controller defined in the Bosch specification. For a full understanding in MSCAN specification, users are recommended to read Bosch specification first. MSCAN module on KE16Z MCU has standard and extended data frames, zero-to eight-byte data length, programmable bit rate up to 1 Mbps, support for remote frames and so on. For more features, see *KE1xZP48M48SF0RM*.

This application note explains bit rate calculation and the identifier arbitration about MSCAN. The document provides a use case of the communication of two MSCAN nodes to help understand how to use MSCAN module. Users can refer to and modify the MSCAN configuration to meet the specific needs, to implement MSCAN node communication. The implementation of the example is based on *IAR Embedded Workbench 8.30.1* development environment, *SDK 2.4.0 software*, FRDM-KE16Z board.

2 Features usage

2.1 MSCAN module

2.1.1 Overview

MSCAN is a communication controller implementing the CAN 2.0A/B protocol as defined in the Bosch specification. MSCAN uses an advanced buffer arrangement resulting in predictable real-time behavior and simplified application software.

MSCAN has three transmit buffers with internal prioritization using a **local priority** concept which allows multiple messages to be set up in advance and achieve an optimized performance. It has five receive buffers with FIFO storage scheme to store receive messages.

MSCAN module has flexible identifier filters which can be applied to selecting the incoming message through the identifier (ID) arbitration process. A successful transmission or a message reception with a matching identifier will be flagged and generate an interrupt request to the CPU.

The basic features of MSCAN are as follows:

- Supporting CAN protocol specification version 2.0A/B.
- Five receive buffers in FIFO storage scheme.
- Three transmit buffers with a local priority concept.
- Flexible maskable identifier acceptance filter.
- Internal time stamp support.
- Wake-up functionality with internal low pass filter.
- Loopback mode for self-testing.
- Listen-only mode for monitoring the bus.

2.1.2 CAN node

CAN is a multi-master serial bus standard for connecting Electronic Control Units (ECU) which is also known as nodes. When CAN nodes communicate, they must be connected to the dual-wire CAN bus through a transceiver device. The wires are 120 Ω nominal twisted pair. The transceiver is capable of driving the large current needed for the CAN bus and has current protection against defective CAN or defective stations. Figure 1. on page 2 shows the connection of CAN nodes to CAN bus .

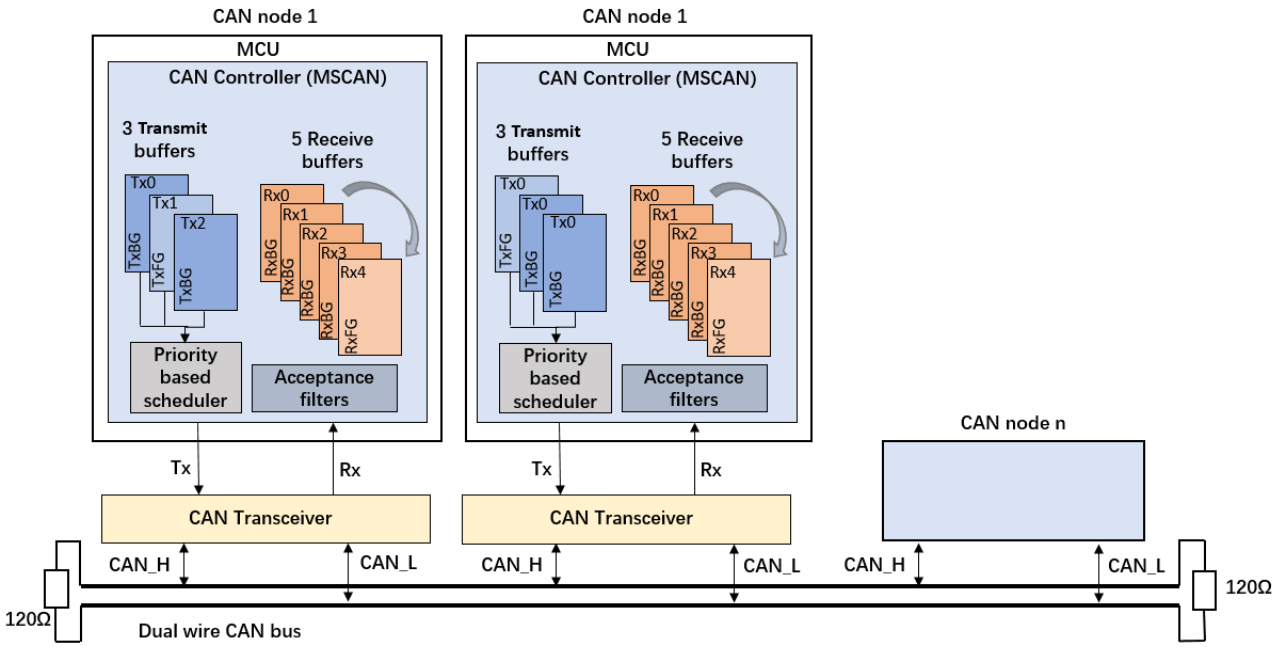

Figure 1. MSCAN system

2.1.3 CAN frame

The message transferring between MSCAN nodes is manifested and controlled by four frame types: data frame, remote frame, error frame and overload frame. This application note is based on the use of data frame.

There are two frame formats for message transmission, and the difference is the length of the identifier field. A frame with an 11-bit identifier is called a standard frame and with a 29-bit identifier is called an extended frame. The data frame contains both standard frame format and extended frame format.

The data frame is used to transmit user data with maximum of 8 bytes. A data frame is a 7-bit field, where the CAN communication protocol is operating. Figure 2. on page 3 shows the structures for standard data frame and extended data frame.

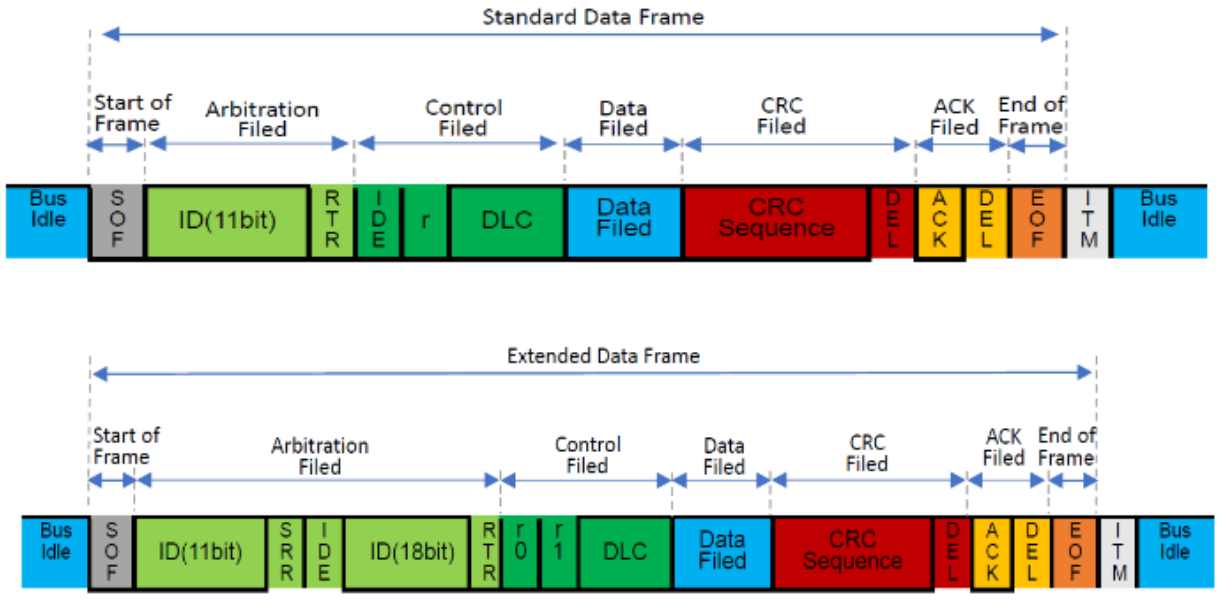

Figure 2. Standard data frame and extended data frame

2.2 MSCAN bit rate

2.2.1 Clock selection

MSCAN module has a programmable clock source and a bus clock or an external oscillator clock (SOSCDIV2_CLK). Figure 3. on page 4 shows the MSCAN block diagram, which can be the reference of the bit rate calculation process.

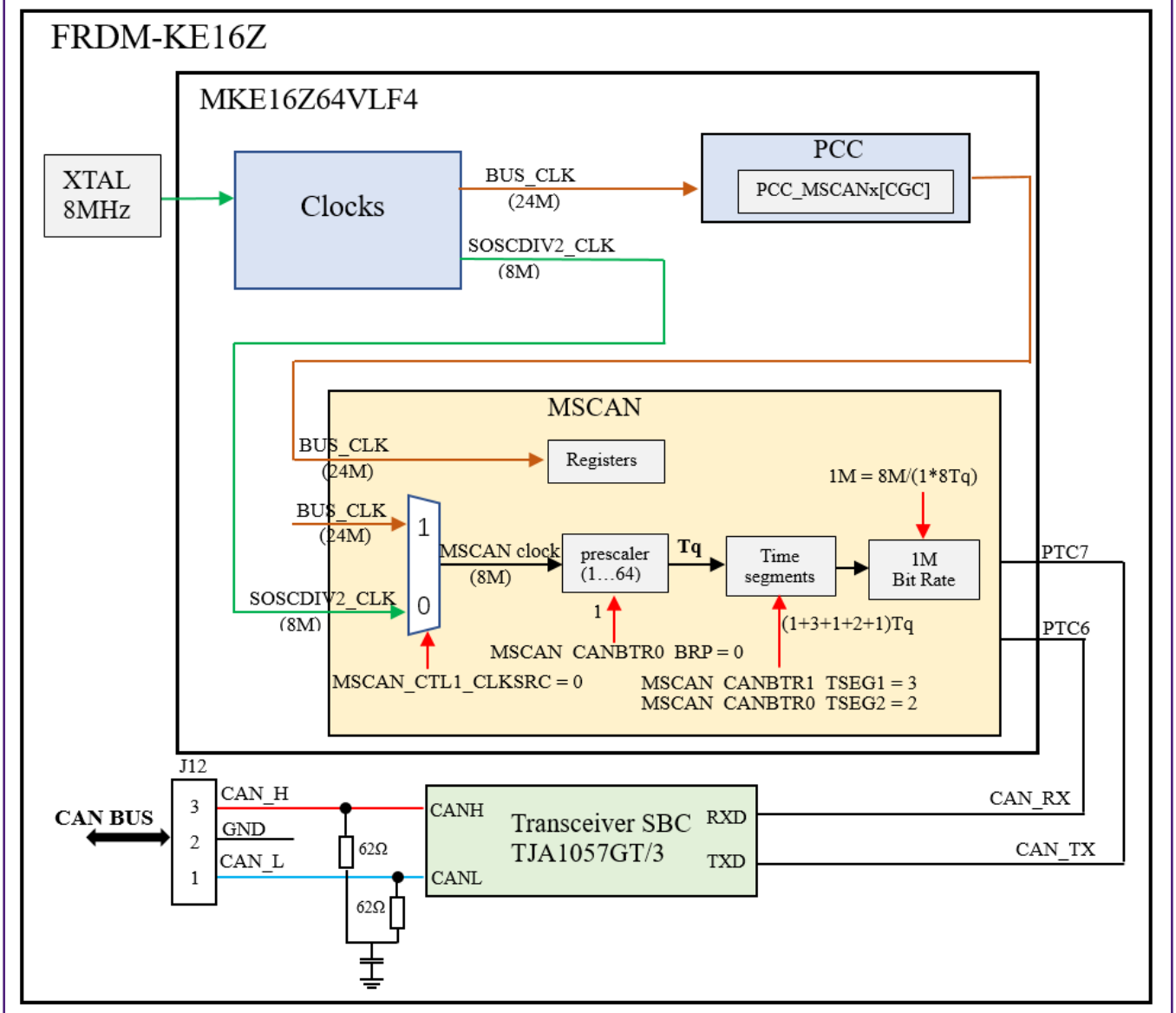

Figure 3. MSCAN block diagram

As shown in Figure 3. on page 4, users can configure MSCAN_CANCTL1_CLKSRC bits to select MSCAN clock (f_{CANCLK}). After the clock source is selected, configure MSCAN_CANBTR0_BRP bits to select the bit rate prescaler. The time quanta (Tq) is the atomic unit of time handled by MSCAN. It can be calculated by selected clock and prescaler value, as shown in Equation 1. on page 4.

Equation 1. Tq calculation

$$Tq = f_{CANCLK} / (\text{prescaler value})$$

2.2.2 Bit time

MSCAN bit time is obtained by configuring bus timing registers MSCAN_CANBTR0 and MSCAN_CANBTR1. A bit time is subdivided into three segments: SYNC_SEG, Time segment1, and Time segment2, as described in CAN specification. These

segments can be represented by time quanta (Tq). The bit time is determined and changed by configuring the number of Tq in each segment.

Figure 4. on page 5 shows the detailed segments within a bit time.

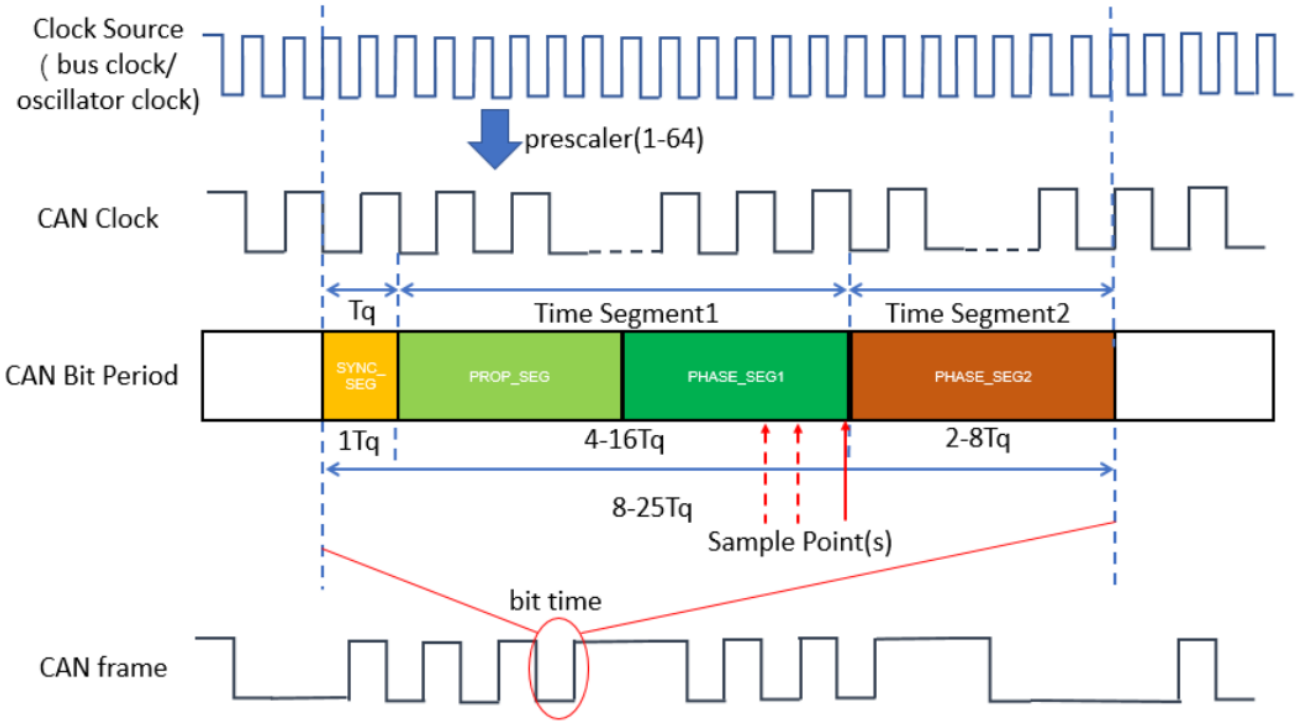

Figure 4. Segments within a bit time

Equation 2. on page 5 shows the method to calculate the bit time.

Equation 2. Bit time calculation

$$\text{bit time} = (\text{SYNC\_SEG} + \text{Time Segment1} + \text{Time Segment2}) * (\text{prescaler value}) / f_{\text{CANCLK}}$$

Where:

- SYNC_SEG has a fixed length of 1 time quanta. Signal edges are expected to happen within this section.
- Time segment 1 includes the PROP_SEG (Propagation Time Segment) and the PHASE_SEG1 (Phase Buffer Segment1) according to the CAN standard. It can be programmed by setting the MSCAN_CANBTR1_TSEG1 to be 4 to 16 time quanta.
- Time segment 2 represents the PHASE_SEG2 (Phase Buffer Segment2) according to the CAN standard. It can be programmed by setting the MSCAN_CANBTR1_TSEG2 to be 2 to 8 time quanta.

Equation 3. on page 5 shows the method to calculate the bit rate.

Equation 3. Bit rate calculation

$$\text{bit rate} = 1/\text{bit time}$$

$$\text{bit rate} = f_{\text{CANCLK}} / ((\text{prescaler value}) * (1 + \text{Time Segment1} + \text{Time Segment2}))$$

According to Equation 3. on page 5, there is an example in Figure 4. on page 5 to show how to get 1 M bit rate by selecting the external oscillator clock as MSCAN clock. Therefore, when selecting the appropriate parameters to configure

MSCAN_CANCTL1_CLKSRC, MSCAN_CANBTR0_BRP, and MSCAN_CANBTR1_TSEG1, MSCAN_CANBTR1_TSEG2 can get the desired bit rate.

2.2.3 Synchronization jump width

The Synchronization Jump Width (SJW) defines the maximum number of Tq clock cycles of one bit can be shortened or lengthened to achieve resynchronization to data transitions on the CAN bus.

Since each unit operates on its own clock, minute clock errors can accumulate, PHASE_SEG1 and PHASE_SEG2 can be used to absorb them. According to the SJW value, by lengthening the PHASE_SEG1 or shortening the PHASE_SEG2, errors are absorbed to adjust synchronization. After SJW is increased, the allowable error is increased, but the communication speed is reduced.

The SJW (see the Bosch CAN 2.0A/B specification for details) is programmed in a range of 1 to 4 time quanta by setting the SJW parameter. Table 1. Bosch CAN 2.0A/B compliant bit time segment settings on page 6 gives an overview of the Bosch CAN 2.0A/B specification compliant segment settings and the related parameter values.

Table 1. Bosch CAN 2.0A/B compliant bit time segment settings

Time segment 1	TSEG1	Time segment 2	TSEG2	Synchronization jump width	SJW
5..10	4..9	2	1	1..2	0..1
4..11	3..10	3	2	1..3	0..2
5..12	4..11	4	3	1..4	0..3
6..13	5..12	5	4	1..4	0..3
7..14	6..13	6	5	1..4	0..3
8..15	7..14	7	6	1..4	0..3
9..16	8..15	8	7	1..4	0..3

2.3 MSCAN identifier acceptance filter and mask filter

When one MSCAN node receives messages from other nodes, each received message is written into the background receive buffer (RxBG). If the message shifts into the foreground receive buffer (RxFG) and passes the identifier arbitration by setting the identifier acceptance and identifier mask registers, it is read by CPU and accepted. Otherwise, the message is overwritten by the next message.

2.3.1 Flexible size of MSCAN filter

The flexible programmable identifier filters in MSCAN module can reduce the evaluation time for the received messages and prevent a certain unwanted message from being handled by the CPU.

The MSCAN identifier acceptance registers define the acceptable patterns of the standard or extended identifier (ID[10:0]or ID[28:0]). The identifier mask registers define which bits in the identifier acceptance registers are **Care** or **Don't care**.

If the acceptance mask bit is set to 0 in identifier mask registers (MSCAN_CANIDMRn), it means that in the corresponding bit, the expected identifier must be same with acceptance code bit in identifier acceptance registers (MSCAN_CANIDARn). If the acceptance mask bit is set to 1 in identifier mask registers (MSCAN_CANIDMRn), the expected identifier may not match the acceptance code bit in identifier acceptance registers (MSCAN_CANIDARn).

The MSCAN module has eight pairs of filter registers. Each pair consists of an 8-bit mask register and an 8-bit acceptance register. Before the ID arbitration process, the MSCAN_CANIDAC_IDAM bits can be configured to change the number of filter registers.

The filter registers are programmable to operate in four modes:

- Two 32-bit identifier filters.
- Four 16-bit identifier filters.
- Eight 8-bit identifier filters.
- Closed filter.

2.3.2 Arbitration process

This section describes arbitration process for the message with 11-bit ID or 29-bit ID by the different mode of identifier filter register.

MSCAN module sets identifier hit flags (IDHIT[2:0]) to indicate an identifier acceptance hit. The IDHIT indicators are always related to the message in the foreground buffer (RxFG). When a message gets shifted into the foreground buffer of the receiver FIFO the indicators are updated as well.

2.3.2.1 In the mode of two 32-bit identifier filters

The two 32-bit filters check for the extended message which consists of the 29 bits of the ID, RTR bit, IDE bit and SRR bit. Any of these bits can be marked as **Don't care** by CANIDMR0 – CANIDMR3 registers first. Then these bits are compare with the contents of CANIDAR0 – CANIDAR3 registers, filter 0 hit will occur if they match. Otherwise, these bits continue to compare with CANIDMR4 – CANIDMR7 and CANIDAR4 – CANIDAR7, filter 1 hit will occur if they match. If it is not hit, the received extended message will be discarded.

Figure 5. on page 7 shows the first bank of acceptance and mask filter arbitration process. The arbitration process in the second bank is similar.

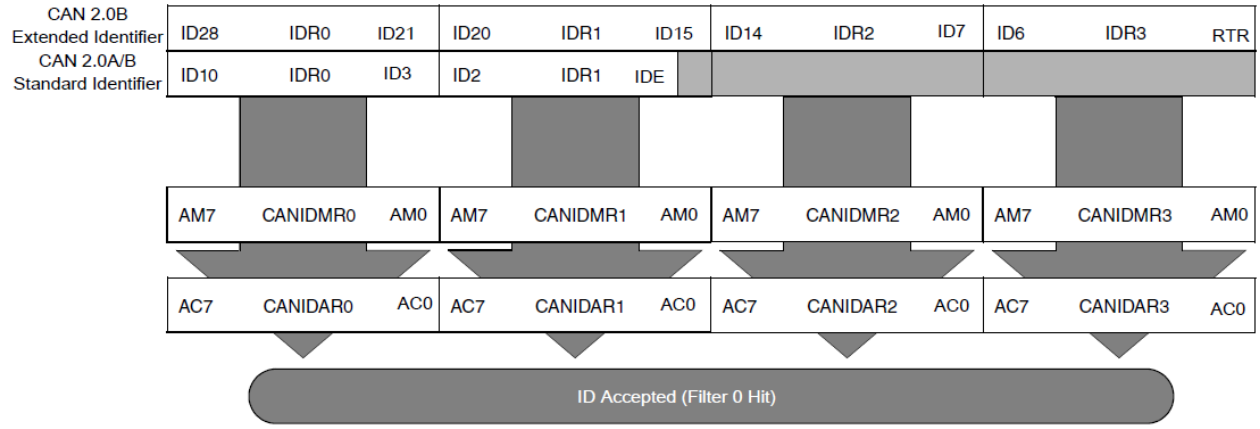

Figure 5. Two 32-bit identifier filters

2.3.2.2 In the mode of four 16-bit identifier filters

The four 16-bit filters check for the extended message which consists of the high 14 bits of the ID, IDE bit and SRR bit. Any of these bits can be marked as **Don't care** by CANIDMR0 – CANIDMR1 registers first. Then these bits are compare with the contents of CANIDAR0 – CANIDAR1 registers, filter 0 hit will occur if they match. Otherwise, these bits continue to compare with CANIDMR2 – CANIDMR3 and CANIDAR2 –CANIDAR3, filter 1 hit will occur if they match. If filter 1 hit does not occur, these bits continue to compare with CANIDMR4 – CANIDMR5 and CANIDAR4 – CANIDAR5, filter 2 hit will occur if they match. If filter 3 hit does not occur, these bits continue to compare with CANIDMR6 – CANIDMR7 and CANIDAR6 – CANIDAR7, filter 3 hit will occur if they match. If it does not hit still, the received extended message will be discarded.

If a standard message has been received then the 11 bits of the ID, RTR bit, IDE bit are checked. The standard message has the same arbitration process with the extended message in four 16-bit identifier mode.

Figure 6. on page 8 shows the first bank of acceptance and mask filter arbitration process. The arbitration process in the second bank is similar.

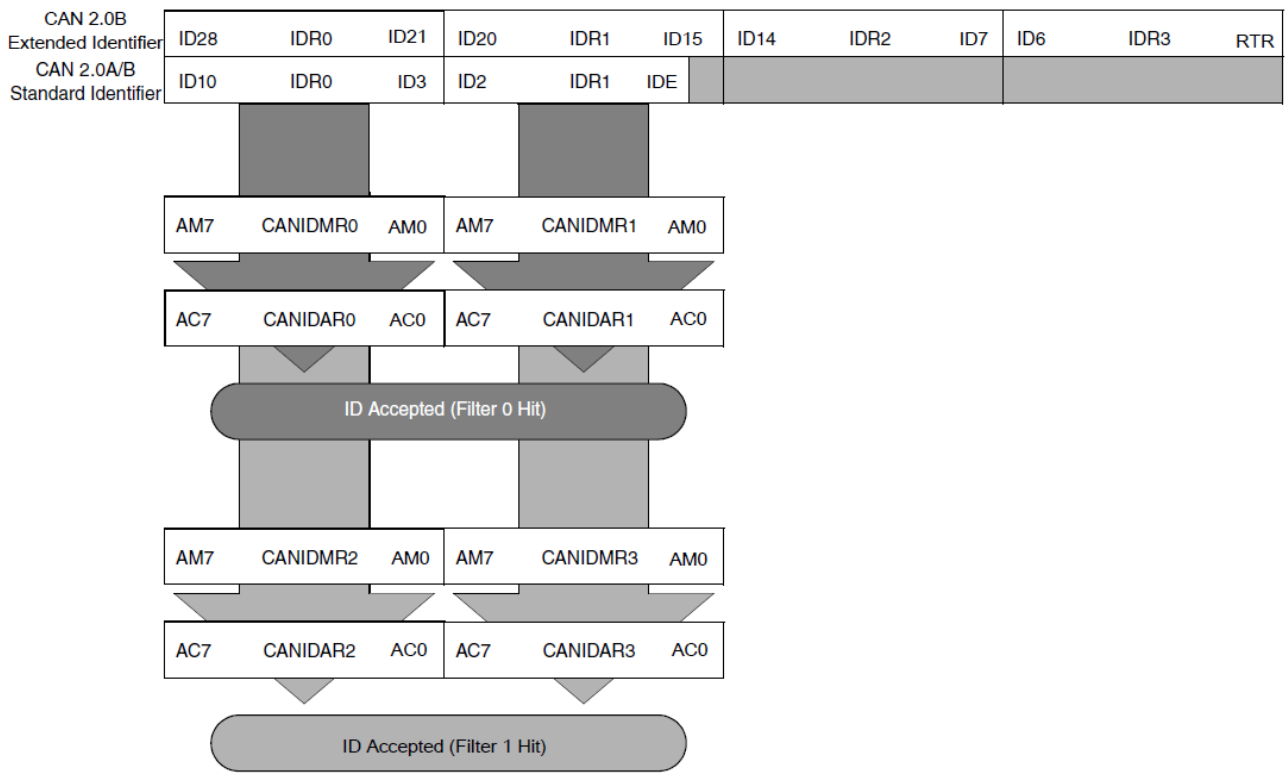

Figure 6. Four 16-bit identifier filters

2.3.2.3 In the mode of eight 8-bit identifier filters

The eight 8-bit filters check the 8 most significant bits of the ID of the standard message and extended message. And the standard message and extended message have the same arbitration process in eight 8-bit identifier mode.

The 8 most significant bits compare with eight pairs of mask registers and acceptance registers (CANIDMRn, CANIDARn) to produce filter 0 to filter 7 hits. The data will be received by one of the filter hits. Otherwise, the data is lost.

Figure 7. on page 9 shows the first bank of acceptance and mask filter arbitration process. The arbitration process in the second bank is similar.

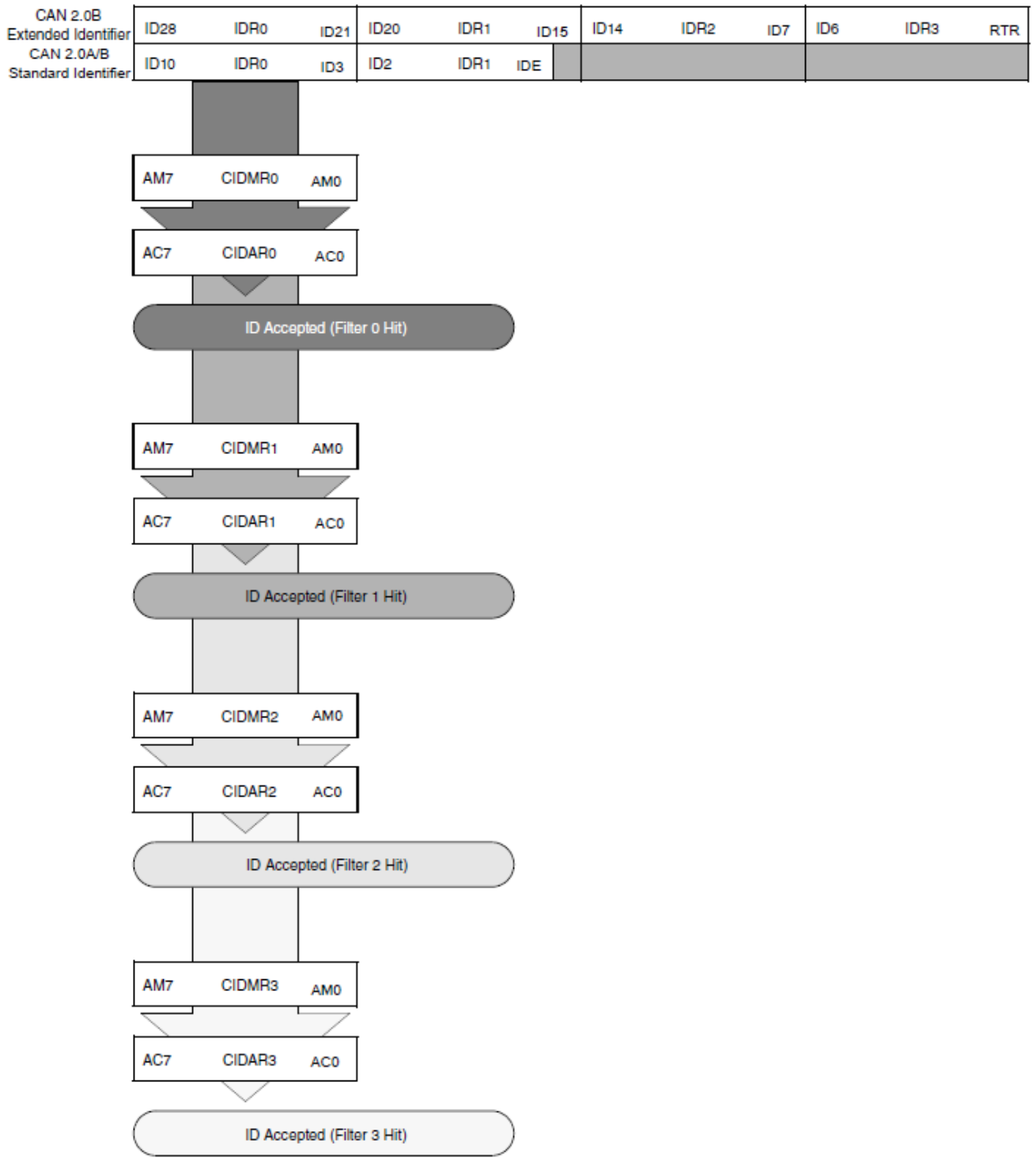

Figure 7. Eight 8-bit identifier filters

2.3.3 Identifier registers configuration

During the MSCAN nodes communication, it is important to set proper identifier for each node. MSCAN module has four extended identifier registers and two standard identifier registers to write received identifier. There is an example to describe how to configure appropriate identifiers for MSCAN nodes communication in the two 32-bit identifier filters mode.

2.3.3.1 Standard data frame with a specific identifier

If a MSCAN node intents to receive message from one of standard data frame with identifier 0x320 to 0x323, now it can implement the function by setting the first bank of identifier filters. Figure 8. on page 10 shows the standard identifier register structure and the process of identifier filters configuration.

	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RSRTR	RSIDE	Reserved		
0x321	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0
0x322	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
0x323	1	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0
Filter Value	1	1	0	0	0	1	0	0	0	X	X	0	0	0	0	0
CANIDMR (0-1)	AM7				CANIDMR0			AM0	AM7			CANIDMR1				AM0
	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
CANIDAR (0-1)	AC7				CANIDAR0			AC0	AC7			CANIDAR0				AC0
	1	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0

Figure 8. MSCAN identifier filter registers configuration for standard data frame

As shown in Figure 8. on page 10, to obtain the expected standard data identifier, the particular bit in CANIDMR0 – CANIDMR1 registers is set to 0, which indicates that the corresponding bit in CANIDAR0 – CANIDAR1 registers can be the same as its identifier bit. In this section, the CANIDAR0 – CANIDAR1 registers can be set as 0x321, 0x322 or 0x323. The bit in CANIDMR0 – CANIDMR1 registers is set to 1, which defines which bit is not used for comparing.

When the configuration of CANIDMR0 – CANIDMR1 registers and CANIDAR0 – CANIDAR1 registers is completed, the ID that can be received is shown as **Filter Value**. In **Filter Value**, **X** means the bit can be 0 or 1. Then the bit 6 (ID0) and bit 7 (ID1) which are **Don't care** can be 1 or 0 in the CANIDAR1 register, as they are meaningless. When a message from MSCAN frame with identifier 0x321, 0x322 or 0x323 produces filter 0 hit, the message will be accepted.

2.3.3.2 Extended data frame with a specific identifier

If a MSCAN node intents to receive messages from an extended data frame with identifier 0x801 or 0x802, it can implement the function by setting the first bank of identifier filters. Figure 9. on page 11 shows the extended identifier register structure and the process of identifier filters configuration.

0x801	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18	RSRR	REIDE	ID17	ID16	ID15
	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RERTR
	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0
0x802	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18	RSRR	REIDE	ID17	ID16	ID15
	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RERTR
	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0
Filter Value	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18	RSRR	REIDE	ID17	ID16	ID15
	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RERTR
	0	0	0	1	0	0	0	0	0	0	0	0	0	X	X	0
CANIDMR (0-3)	AM7			CANIDMR0				AM0	AM7			CANIDMR1				AM0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	AM7			CANIDMR2				AM0	AM7			CANIDMR3				AM0
	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
CANIDAR (0-3)	AC7			CANIDAR0				AC0	AC7			CANIDAR1				AC0
	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
	AC7			CANIDAR2				AC0	AC7			CANIDAR3				AC0
	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0

Figure 9. MSCAN identifier filter registers configuration for extended data frame

As shown in Figure 9. on page 11, to obtain the expect extend data identifier (0x801, 0x802), the particular bit in CANIDMR0 – CANIDMR3 registers is set to 0, it indicates that the corresponding bit in CANIDAR0 – CANIDAR3 registers can be the same as its identifier bit. In this section, the CANIDAR0 – CANIDAR3 registers can be set as 0x801 or 0x802. The particular bit in CANIDMR0 – CANIDMR3 registers is set to 1, which defines which bit is not used for comparing.

When the configuration of CANIDMR0 – CANIDMR3 registers and CANIDAR0 – CANIDAR3 registers is completed, the ID that can be received is shown as **Filter Value**. In **Filter Value**, **X** means the bit can be 0 or 1. Then the bit 1 (ID0) and bit 2 (ID1) which are **Don't care** can be 1 or 0 in the CANIDAR3 register ase they are meaningless. When a message from MSCAN data frame with identifier 0x801 or 0x802 produces filter 0 hit, the message with the identifier 0x801 or 0x802 will be accepted.

3 Implementation details

There is a use case of the communication between two MSCAN nodes. In this use case, MSCAN node1 transmits a message to MSCAN node2 and receives a message from MSCAN node2. At the same time, MSCAN node2 transmits a message to MSCAN node1 and receives a message from MSCAN node1.

3.1 Example operation

Be sure to confirm the hardware connection before the software implementation. Prepare two KE16Z boards and connect the pins, as shown in Table 2. Connect MSCAN pins between two boards on page 11.

Table 2. Connect MSCAN pins between two boards

BOARD_1	Connects to		BOARD_2
Pin Name	Board location	Board location	Pin name
CAN_L	J12-1	J12-1	CAN_L
CAN_H	J12-3	J12-3	CAN_H
GND	J12-2	J12-2	GND

The following steps can realize the communication with two boards.

1. Connect a USB cable between the host PC and the OpenSDA USB port on the two target boards (node1, node2), as shown in Figure 10. on page 12.

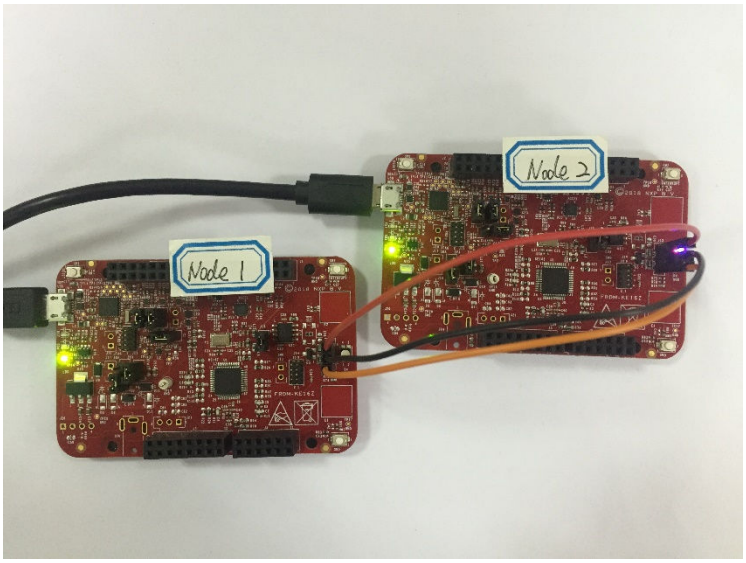

Figure 10. Connection of two FRDM-KE16Z boards

2. Open two serial terminals by setting the baud rate to 115200 to show the output of the MSCAN node demos in the terminal window.
3. Download the two programs to the two target boards.
4. Either press the reset button on your boards or launch the debugger in your IDE to run the demo.

When preparing two programs for two MSCAN nodes, refer to MSCAN node configuration on page 12 for the details.

3.2 MSCAN node configuration

MSCAN node1 is set up to receive data from MSCAN node2 and then transmit data to MSCAN node2. Similarly, MSCAN node2 is set up to receive data from MSCAN node1 and then transmit data to MSCAN node1. This section describes the detailed process for MSCAN node1 to transmit and receive messages.

3.2.1 ID acceptance and mask filter definition

In this use case, node1 is configured as an extended frame with ID 0x801. It receives message from node2 with ID 0x802. In the mode of two 32-bit identifier filters, use macro definitions to fill the values of the mask and acceptance registers (MSCAN_CANIDARn, MSCAN_CANIDMRn).

Similarly, node2 is configured as an extended frame with ID 0x802. It receives message from node1 with ID 0x801.

Take node1 as an example and the following shows the macro definitions acceptance and mask filter.

```
/* Definitions for node1 ID Acceptance and Mask filter */
#define NODE_ID1    0x801                   /* Transmit Extend ID */
#define NODE_ID2    0x802                   /* Receive Extend ID */
#define MSCAN_IDAR0    ((NODE_ID2>>21)<<24)|(((NODE_ID2>>18)&0x7)<<21)\
                       |(((NODE_ID2>>15)&0x7)<<16)|(((NODE_ID2>>7)&0xFF)<<8)\
                       |(((NODE_ID2>>0)&0x7F)<<1)|(1<<19)    /* ID Acceptance Registers of First Bank
*/
```

```
#define MSCAN_IDAR1     ((NODE_ID2>>21)<<24)|(((NODE_ID2>>18)&0x7)<<21)\
                        |(((NODE_ID2>>15)&0x7)<<16)|(((NODE_ID2>>7)&0xFF)<<8)\
                        |(((NODE_ID2>>0)&0x7F)<<1) |(1<<19)  /* ID Acceptance Registers of second Bank
*/
#define MSCAN_IDMR0     0x1|(uint32_t)0x18<<16              /* ID Mask Registers of First Bank */
#define MSCAN_IDMR1     0x1|(uint32_t)0x18<<16              /* ID Mask Registers of Second Bank */
```

3.2.2 MSCAN initial configuration

Select the clock source and configure the bit rate to guarantee the speed of communication. Select the identifier filter mode and configure the acceptance filter and mask filter to determine the acceptance identifier.

Take node1 as an example and the following shows the method of initial configuration.

```
/* Initialize MSCAN Module config struct with default value. */
  MSCAN_GetDefaultConfig(&mscanConfig); //bit rate:1M; clock source:Oscillator clock; loopback
mode:disabled
                                       //Identifier acceptance mode: two 32-bit filters

  /* Acceptance filter and Mask filter configuration. */
  mscanConfig.filterConfig.u32IDAR0 = MSCAN_IDAR0;  // Configure the first bank of Acceptance filter
  mscanConfig.filterConfig.u32IDAR1 = MSCAN_IDAR1;  // Configure the second bank of Acceptance filter
  mscanConfig.filterConfig.u32IDMR0 = MSCAN_IDMR0;  // Configure the first bank of Mask filter
  mscanConfig.filterConfig.u32IDMR1 = MSCAN_IDMR1;  // Configure the second bank of Mask filter
  /* Initialize MSCAN module.*/
  MSCAN_Init(EXAMPLE_MSCAN, &mscanConfig, EXAMPLE_MSCAN_CLK_FREQ);

  /* Enable Rx Buffer interrupt.*/
  MSCAN_EnableRxInterrupts(EXAMPLE_MSCAN, kMSCAN_RxFullInterruptEnable);
  MSCAN_EnableRxInterrupts(EXAMPLE_MSCAN, kMSCAN_OverrunInterruptEnable);
  MSCAN_EnableRxInterrupts(EXAMPLE_MSCAN, kMSCAN_StatusChangeInterruptEnable);
  EnableIRQ(EXAMPLE_MSCAN_IRQn);
```

3.2.3 Transmit data

When transmitting a message to other nodes, configure the MSCAN frame to data frame and select the standard format or extended format to the data frame. The message can be sent out after the completion of data field, arbitration field and others configuration.

Take node1 as an example and the following shows the preparation for transmitting data.

```
    /*Prepare Tx Frame for sending. */
  txFrame.ID_Type.ID = NODE_ID1;              // Configure the transfer message identifier to NODE_ID1
  txFrame.format = kMSCAN_FrameFormatExtend;  // Configure the transfer message format to extended ID
  txFrame.type = kMSCAN_FrameTypeData;        // Configure the MSCAN frame to data frame
  txFrame.DLR = 8;                            // Configure the data length to 8
  txFrame.dataWord0 = 0x44332211;             // Configure the transfer data
  txFrame.dataWord1 = 0x88776655;

  /* Send data through Tx Buffer using polling function. */
  MSCAN_TransferSendBlocking(EXAMPLE_MSCAN,&txFrame);
```

3.2.4 Receive data

When receiving a new message, read the data buffer and clear the reception flag with the receiver interrupt function to keep the module available to receive more messages.

Take node1 as an example and the following shows the process of receive data interrupt handler.

```
  /* Receive Data interrupt handler */
void EXAMPLE_MSCAN_IRQHandler(void)
{
  /* If new data arrived.*/
  if (MSCAN_GetRxBufferFullFlag(EXAMPLE_MSCAN))
  {
      MSCAN_ReadRxMb(EXAMPLE_MSCAN, &rxFrame);      // Read message received from acceptance nodes
      MSCAN_ClearRxBufferFullFlag(EXAMPLE_MSCAN);   // Clear Rx flag
   }
   /* Add for ARM errata 838869, affects Cortex-M4, Cortex-M4F Store immediate overlapping
     exception return operation might vector to incorrect interrupt */
#if defined __CORTEX_M && (__CORTEX_M == 4U)
  __DSB();
#endif
}
```

4 Conclusion

This application note explains the important features, bit rate calculation and the identifier arbitration about MSCAN module. It provides a use case about two node communication to help user understand how to use MSCAN module. Users can easily implement the function by referring to the configuration and modify the configurations on their needs.

5 Reference

You can refer to the following documents on the NXP website for further information.

- *KE16Z Reference Manual* (KE1xZP48M48SF0RM)
- Using MSCAN on the HCS12Family (AN3034)
- *XGATE Library: CAN Driver Providing a Full CAN Mailbox System* (AN2726)
- *Bosch Controller Area Network (CAN) Version 2.0 Protocol Standard* (BCANPSV2)
- *S32K1xx Series Cookbook* (AN5413)

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 25 Febuary 2019
Document identifier: AN12355

arm