# MSM9225B
# User's Manual

CAN (Controller Area Network) Controller

# Preface

This manual describes the hardware and operation of the MSM9225B CAN Controller which conforms to the CAN protocol specification (Bosch, V2.0 part B/Active).

# Notation

| Classification | Notation | Description |
|---|---|---|
| ♦ Numeric value | xxhex, xxh, xxH | Indicates a hexadecimal number.    x: Any value in the range of 0 to F |
| | xxb | Indicates a binary number; "b" may be omitted.    x: A value 0 or 1 |
| ♦ Unit | word, W | 1 word = 16 bits |
| | byte, B | 1 byte = 2 nibbles = 8 bits |
| | nibble, N | 1 nibble = 4 bits |
| | maga-, M | $10^6$ |
| | kilo-, K | $2^{10} = 1024$ |
| | kilo-, k | $10^3 = 1000$ |
| | milli-, m | $10^{-3}$ |
| | micro-, μ | $10^{-6}$ |
| | nano-, n | $10^{-9}$ |
| | second, s (lower case) | second |
| ♦ Symbol | x0hex | x indicates any value in the range of 0 to F of the high-order 4 bits. |
| ♦ Terminology | "H" level, "1" level | Indicates high voltage signal levels $V_{IH}$ and $V_{OH}$ as specified by the electrical characteristics. |
| | "L" level, "0" level | Indicates low voltage signal levels $V_{IL}$ and $V_{OL}$ as specified by the electrical characteristics. |

♦ Register description

Read/write attribute:  R indicates a readable bit and W indicates a writable bit.
MSB/LSB:  Most significant bit of the 8-bit register (memory)/least significant bit of the 8-bit register (memory).
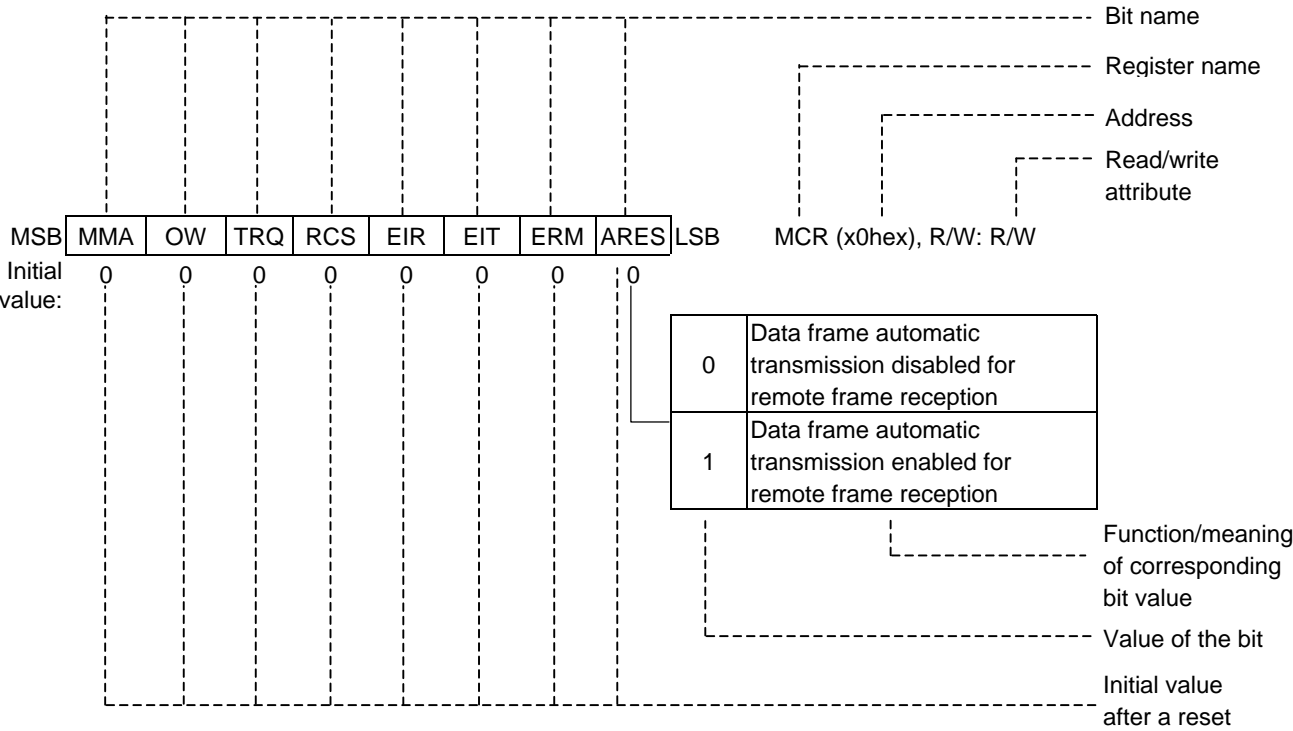
# Table of Contents

## Chapter 4      Microcontroller Interface

## Chapter 5      Electrical Characteristics

## Appendixes

## Revision History

*Chapter 1*

# Overview

# Chapter 1 Overview

## 1.1 Overview

The MSM9225B is a microcontroller peripheral LSI which conforms to the CAN protocol for high-speed LANs in automobiles.

## 1.2 Features

- Conforms to CAN protocol specification (Bosch, V2.0 part B/Active)
- Maximum of 1 Mbps bit rate
- Communication method:
  - Transmission line is bi-directional, two-wire serial communication
  - NRZ (Non-Return to Zero) system using bit stuff function
  - Multi-master system
  - Broadcast system
- Up to 16 message boxes can be used, and messages up to 8 bytes long can be transmitted or received for each message box.
  - Number of received messages can be extended by group message function (up to 2 groups can be set)
  - Overwrite flag is provided
- Priority control by identifier
  - 2032 types in standard format, $2032 \times 2^{18}$ types in extended format
- Microcontroller interface
  - Corresponding to both parallel and serial interface
    - Parallel interface: Separate address/data bus type (with address latch signal/no address latch signal) and multiplexed address/data bus type
    - Serial interface: Synchronous communication type
  - Three interrupt sources: transmission/receive/error
- Error control:
  - Bit error/stuff error/CRC error/form error/acknowledgment error detection functions
  - Retransmission/error status monitoring function when error occurs
  - Bit error flag/stuff error flag/CRC error flag/form error flag/acknowledge error flag are provided
- Communication control by remote data request function
- Sleep/Stop mode function
- Supply voltage: 5 V±10%
- Operating temperature: –40 to +125°C
- Package: 44-pin plastic QFP (QFP44-P-910-0.80-2K) (MSM9225BGA)

## 1.3  Block Diagram
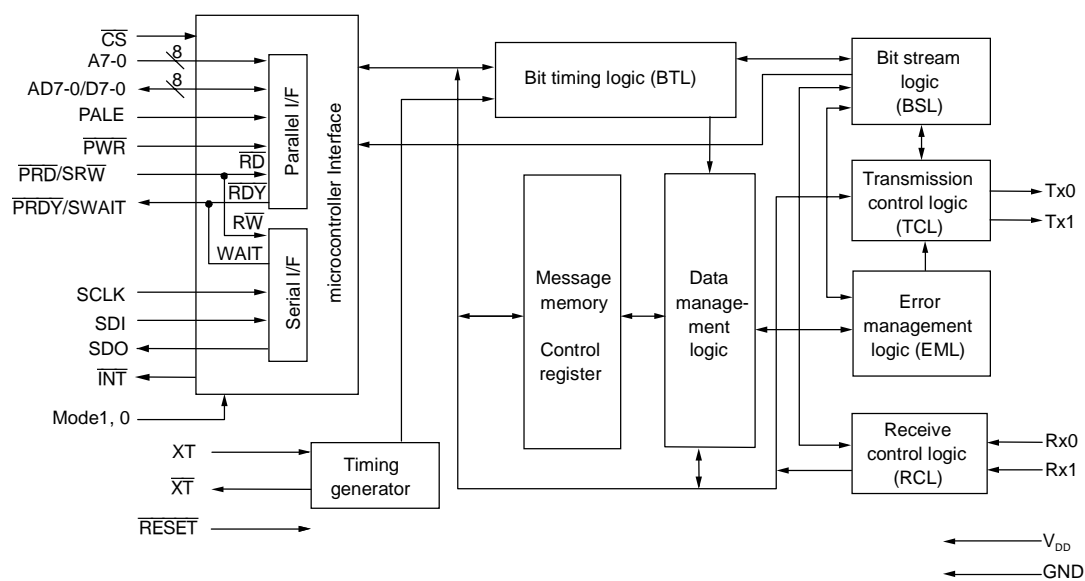


**Figure 1-1  Block Diagram**

## 1.4  Configuration Example
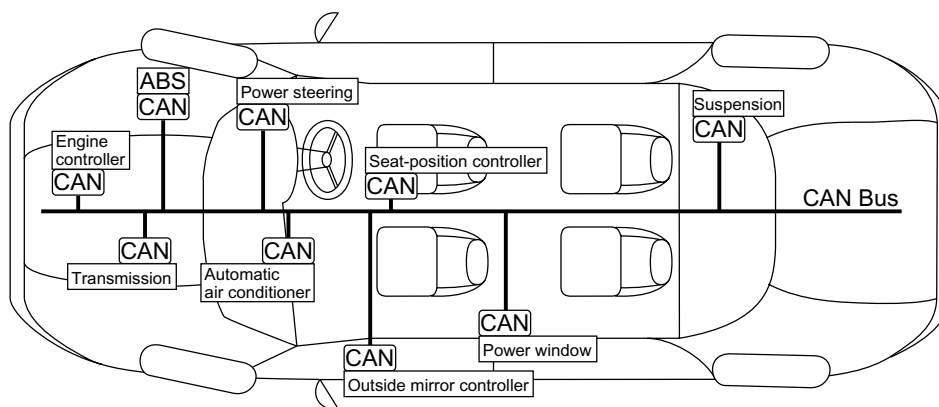


**Figure 1-2  Configuration Example**

## 1.5   Pin Configuration
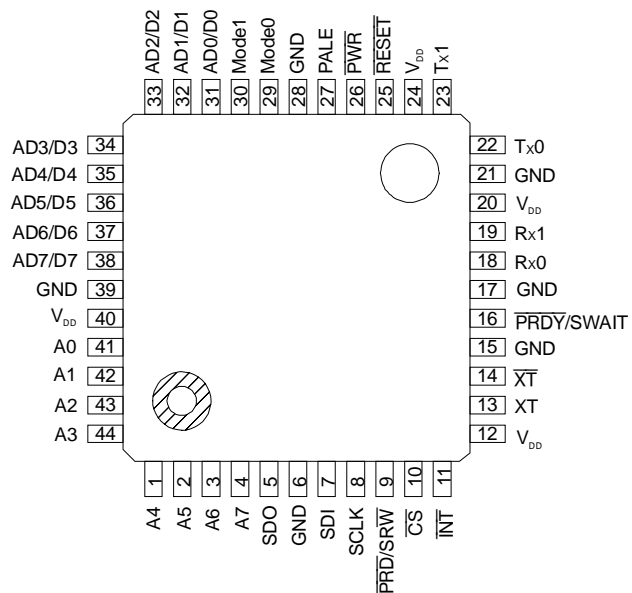


Connect all $V_{DD}$ pins.  Connect all GND pins.

**Figure 1-3  44-Pin Plastic QFP (Top View)**

## 1.6   Pin Descriptions

**Table 1-1  Pin Description**

| Symbol | Pin | Type | Description |
|---|---|---|---|
| $\overline{\text{CS}}$ | 10 | I | Chip select pin.  When "L", PALE, $\overline{\text{PWR}}$, $\overline{\text{PRD}}$/SR$\overline{\text{W}}$, SCLK and SDO pins (microcontroller interface pins) are valid.<br>When "H", these pins are invalid. |
| A7-0 | 41-44, 1-4 | I | Address bus pins (when using separate buses).  If used with a multiplexed bus or if used in the serial mode, fix these pins at "H" or "L" levels. |
| AD7-0/ D7-0 | 31-38 | I/O | Multiplexed bus:  Address/data pins (AD7-0)<br>Separate buses:  Data pins (D7-0)<br>If used in the serial mode, fix these pins at a "L" level. |
| $\overline{\text{PWR}}$ | 26 | I | Write input pin if used in the parallel mode.  Data is captured when this pin is at a "L" level.<br>If used in the serial mode, fix this pin at a "L" level. |
| $\overline{\text{PRD}}$/ SR$\overline{\text{W}}$ | 9 | I | Parallel mode:  Read signal pin ($\overline{\text{PRD}}$)<br>When at a "L" level, data is output from the data pins.<br>Serial mode:  Read/write signal pin (SR$\overline{\text{W}}$)<br>When at a "H" level, data is output from the SDO pin.<br>When at a "L" level, the SDO pin is at high impedance, and data is captured beginning with the second byte of data input from the SDI pin. |
| PALE | 27 | I | Address latch signal pin<br>When at a "H" level, addresses are captured.<br>If used in the parallel mode and the address latch signal is unnecessary or in the serial mode, fix this pin at a "H" or "L" level. |
| SDI | 7 | I | Serial data input pin<br>Addresses (1st byte) and data (beginning from the 2nd byte) are input to this pin, LSB first.  If used in the parallel mode, fix this pin at a "H" or "L" level. |
| SDO | 5 | O | Serial data output pin<br>When the $\overline{\text{CS}}$ pin is at a "H" level, this pin is at high impedance.  When $\overline{\text{CS}}$ is at a "L" level, data is output from this pin, LSB first.<br>If used in the parallel mode, fix this pin at a "H" or "L" level. |
| SCLK | 8 | I | Shift clock input pin for serial data<br>At the rising edge of the shift clock, SDI pin data is captured.  At the falling edge, data is output from the SDO pin. |
| $\overline{\text{PRDY}}$/ SWAIT | 16 | O | Ready output pin<br>When required by the MSM9225B, a signal may be output to extend the bus cycle until the internal access is completed. |

| | Internal access in progress | After completion of access |
|---|---|---|
| Parallel mode ($\overline{\text{PRDY}}$) | "L" level output | High impedance output |
| Serial mode (SWAIT) | "H" level output | "L" level output |

**Table 1-1  Pin Description (continued)**

| Symbol | Pin | Type | Description |
|--------|-----|------|-------------|
| Mode1, 0 | 29, 30 | I | Microcontroller interface select pins<br><br>| Mode1 | Mode0 | Interface | | |<br>|---|---|---|---|---|<br>| 0 | 0 | Parallel mode | Separate buses | No address latch signal |<br>| 0 | 1 | | | With address latch signal |<br>| 1 | 0 | | Multiplexed buses | |<br>| 1 | 1 | Serial mode | | | |
| $\overline{\text{INT}}$ | 11 | O | Interrupt request output pin<br>When an interrupt request occurs, a "L" level is output. This pin automatically outputs a "H" level after 32 Ts (T = 1/fosc).<br>Three types of interrupts share this pin:  transmission complete, reception complete, and error. |
| $\overline{\text{RESET}}$ | 25 | I | Reset pin<br>System is reset when this pin is at a "L" level. |
| XT | 13 | I | Clock pins.  If internal oscillator is used, connect a crystal (ceramic resonator). |
| $\overline{\text{XT}}$ | 14 | O | If external clock is used, input clock via XT pin.  The $\overline{\text{XT}}$ pin should be left open. |
| Rx0, Rx1 | 18, 19 | I | Receive input pin.  Differential amplifier included. |
| Tx0, Tx1 | 22, 23 | O | Transmission output pin |
| $V_{DD}$ | 12, 20, 24, 40 | — | Power supply pin: Connect all $V_{DD}$ pins to the power supply source. |
| GND | 6, 15, 17, 21, 28, 39 | — | GND pin: Connect all GND pins to ground. |

# Register Descriptions

# Chapter 2 Register Descriptions

## 2.1 Memory Space

The MSM9225B has 256 bytes of memory space for the message memory and control registers. Before starting communication, messages for communication and various control registers must be set.
Figure 2-1 shows the configuration of memory space.
The message memory and the control registers are selected by an 8-bit address.
The message memory consists of 16 message boxes (message box 0 to message box F). Each message box is selected by the high-order 4 bits (0hex to Fhex) of the address, and the message control register, the identifier, and the area for storing the message contents are selected by the low-order 4 bits (0hex to Dhex) of the address. It is possible to store a 2-byte (standard format) or a 5-byte (extended format) identifier and a message of a maximum of 8 bytes can be stored in each message box. The control registers are selected by the low-order 4 bits (Ehex, Fhex) of the address. Table 2-1 shows the configuration of the control registers.

| H(hex) / L(hex) | Higher addresses 0 to F |
|---|---|
| Lower addresses 0 to D | Message memory |
| E | Control registers |
| F | |

"H" indicates high-order 4 bits and
"L" indicates low-order 4 bits.

| H(hex) / L(hex) | Message boxes specified by high-order 4 bits of addresses | |
|---|---|---|
| | Standard format | Extended format |
| 0 | Message control register (MCR) | |
| 1 | Identifier 0 (IDR0) | |
| 2 | Identifier 1 (IDR1) | |
| 3 | Message 0 (MSG0) | Identifier 2 (IDR2) |
| 4 | Message 1 (MSG1) | Identifier 3 (IDR3) |
| 5 | Message 2 (MSG2) | Identifier 4 (IDR4) |
| 6 | Message 3 (MSG3) | Message 0 (TMSG0) |
| 7 | Message 4 (MSG4) | Message 1 (TMSG1) |
| 8 | Message 5 (MSG5) | Message 2 (TMSG2) |
| 9 | Message 6 (MSG6) | Message 3 (TMSG3) |
| A | Message 7 (MSG7) | Message 4 (TMSG4) |
| B | Not used | Message 5 (TMSG5) |
| C | Not used | Message 6 (TMSG6) |
| D | Not used | Message 7 (TMSG7) |

**Figure 2-1 Memory Space Configuration**

2 – 2

**Table 2-1  Control Register Configuration**

| Address | Symbol | Name |
|---------|--------|------|
| 0EH | CANC | CAN control register |
| 0FH | CANI | CAN interrupt control register |
| 1EH | NMES | Message box count setting register |
| 1FH | BTR0 | CAN bus timing register 0 |
| 2EH | BTR1 | CAN bus timing register 1 |
| 2FH | TIOC | Communication input/output control register |
| 3EH | GMR0 | Group message register 0 |
| 3FH | GMR1 | Group message register 1 |
| 4EH | GMSK00 | Message mask register 00 |
| 4FH | GMSK01 | Message mask register 01 |
| 5EH | GMSK02 | Message mask register 02 |
| 5FH | GMSK03 | Message mask register 03 |
| 6EH | GMSK10 | Message mask register 10 |
| 6FH | GMSK11 | Message mask register 11 |
| 7EH | GMSK12 | Message mask register 12 |
| 7FH | GMSK13 | Message mask register 13 |
| 8EH | STBY | Standby control register |
| 8FH | CANC2 | CAN control register 2 |
| 9EH | TMN | Communication message box number register |
| 9FH | CANS | CAN status register |
| AEH | TEC | Transmission error counter |
| AFH | REC | Receive error counter |
| BEH | CANS2 | CAN status register 2 |
| BFH | BOCO | Bus off release counter |
| CEH CFH DEH DFH EEH EFH FEH FFH | — | Not used (reserve area) |

## 2.2  Message Memory

The message memory is the memory for setting and storing messages to be transmitted and received.  The message memory consists of 16 message boxes from message box 0 to message box F.

It is possible to transmit only the messages that have been stored in the message boxes, and the transmission is done starting from the message box with the higher priority for which a transmission request is present.

Reception is possible only of messages having identifiers stored in the message boxes.  When a message has been received normally without generating an error, and if the identifier matches with the identifier stored in a message box, the data of the message is stored in the corresponding message box in the message memory.  Set the highest message box number to be used in the register NMES (see Section 2.4.3).

**Note when reading Message Memory Related Register**

When the Message Memory Related Register (MCR, IDR0, IDR1, MSG0-7 in the case the standard format, IDR2-4 and TMSG0-7 in the case of the extended format) is polled, the same data are read out from it as long as the same address is specified consecutively even if the Message Memory Related Register is overwritten by the completion of message transfer between each polling.

However the MMA bit of Message Control Register (MCR, ×0hex) and Control Registers located at ×Ehex and ×Fhex addresses are excluded.

When the Message Memory Related Register is polled, insert the dummy read access to the different address after each reading out.

## 2.3   Message Memory Related Register

### 2.3.1   Message Control Register (MCR: x0hex)

This register performs various controls for a message.
Set this register for each message box.
The bit configuration is as follows:



| MSB | MMA | OW | TRQ | RCS | EIR | EIT | FRM | ARES | LSB | MCR (x0hex), R/W: R/W |
|-----|-----|-----|-----|-----|-----|-----|-----|------|-----|------------------------|

Initial value: 0 0 0 0 0 0 0 0

| 0 | Data frame automatic transmission disabled for remote frame reception. |
|---|---|
| 1 | Data frame automatic transmission enabled for remote frame reception. |

| 0 | Frame type specification |
|---|---|
| 1 | See Table 2-2 for details. |

| 0 | Setting the transmission completion interrupt request flag (ITF) is disabled. |
|---|---|
| 1 | Setting the transmission completion interrupt request flag (ITF) is enabled. |

| 0 | Setting the receive completion interrupt request flag (IRF) is disabled. |
|---|---|
| 1 | Setting the receive completion interrupt request flag (IRF) is enabled. |

| 0 | Cleared to "0" by the microcontroller. |
|---|---|
| 1 | "1" is set at the completion of reception. |

| 0 | Cleared to "0" at the end of transmission. |
|---|---|
| 1 | Write a "1" for transmission (transmission request). |

| 0 | No message overwrite |
|---|---|
| 1 | Message has been overwritten |

| 0 | Writing disabled from the microcontroller to the message box.  Transmission and reception are possible. |
|---|---|
| 1 | Writing enabled from the microcontroller to the message box.  Transmission and reception stopped. |

**Figure 2-2  Message Control Register (MCR)**

(1)  Automatic transmission: ARES

If the automatic transmission of the data frame is used for remote frame reception, set this bit to "1".

At reset, the ARES bit is set to "0".  The ARES bit is invalid if the message is specified as a group message.

**Notes on Automatic Transmission**

Following shows how the transmission is carried out for the messages for which ARES is set to "1" when a remote frame is received.

The MSM9225B detects the transmission priority of all the messages for which the TRQ (transmission request) bit is set to "1", then transmits the messages in sequence from the one with the highest priority.

Note, therefore, that messages for which automatic transmission is set will not always be transmitted immediately after remote frame reception if there are any other messages to be transmitted.

Also in cases where there are some messages for which TRQ is set to "1", whereas the TIRS bit of CANC is not set to "1" because it is not yet desired to transmit them, those messages for which TRQ bit is set to "1" will be transmitted.

(2)  Frame type setting: FRM

This flag sets the frame type of the message to be transmitted/received. A message of a frame type other than the specified frame type cannot be transmitted/received.

Table 2-2 shows the relationship between setting and frame type.

At reset, the FRM bit is set to "0".

**Table 2-2  Frame Types**

| Specified as group message | FRM | Transmission frame | Receive frame |
|---|---|---|---|
| No | 0 | Data frame | Remote frame |
| | 1 | Remote frame | Data frame |
| Yes | 0 | Transmission not activated | Data frame |
| | 1 | | Remote frame |

(3)  Transmission completion interrupt enable: EIT

This is a flag to enable setting ("1") the transmission interrupt request flag (ITF) when transmission completes.

Set this flag from the microcontroller.

The EIT bit is valid when the EINTT bit of the CANI register is "1". (See Section 2.4.2.)

At reset, the EIT bit is set to "0".

(4)  Receive completion interrupt enable: EIR

This is a flag to enable setting ("1") the receive interrupt request flag (IRF) when receiving completes.

Set this flag from the microcontroller.

The EIR bit is valid when the EINTR bit of the CANI register is "1". (See Section 2.4.2.)

At reset, the EIR bit is set to "0".

(5)  Receive status: RCS

When receiving completes, the RCS bit becomes "1". Write "0" to the RCS bit before the micro-controller reads receive data.  When receiving the remote frame, the RCS bit becomes "1" just after the reception.  When receiving the data frame, it becomes "1" after receive data is written to the message box.

At reset, the RCS bit is set to "0".

(6)  Transmission request: TRQ
When a message box is used for transmission, write "1" to this bit from the microcontroller.  When transmission ends normally, "0" is written to this bit. This means that the TRQ bit is "1" during transmission. Therefore, to request transmission, confirm that the TRQ bit is "0" first, and then write "1" to the TRQ bit.  Set the TIRS bit of CANC to start transmission.
When the remote frame is received while the ARES bit is "1", the TRQ bit is set to "1".
At reset, the TRQ bit is set to "0"

(7)  Overwrite flag: OW
When the RCS bit is "1", this bit is set to "1" if data has been received by the same message box again. That is, OW is a flag to indicate that receive data has been overwritten.
At reset, the OW bit is set to "0".

(8)  Message box access request/enable bit: MMA
Be sure to write a "1" to the MMA bit before writing to a message box from the microcontroller.  Then read the MMA bit.  If "1" is read, the message box is accessible.  If "0" is read, write a "1" in a loop until the MMA bit actually becomes "1".
After a "1" has been written to the MMA bit and the message box has been rewritten, be sure to write a "0" to the MMA bit.  Then read the MMA bit.  If "1" is read, write a "0" in a loop until the MMA bit actually becomes "0".
The initialization bit INIT of the CAN control register (CANC: 0Ehex) has priority over the MMA bit. That is, when INIT is "1", the MMA bit is read as "1" irrespective of whether the MMA bit content is "0" or "1", so that the message box becomes accessible.  In addition, after INIT is reset to "0", all the MMA bits will be set to "0".
At reset, the MMA bit is set to "0".  It is possible to rewrite the contents of the other bits in the message control register (MCR) at the same time that the MMA bit is overwritten.
When the MMA bit of a message box is set to "1", do not set the MMA bit of other message box to "1".

2.3.2 Identifier 0 (IDR0: x1hex)

This register sets the frame format, data length code, and a part of the identifier.
The bit configuration is as follows:

MSB | IDFM | DLC3 | DLC2 | DLC1 | DLC0 | IDB28 | IDB27 | IDB26 | LSB   IDR0 (x1hex), R/W: R/W

Initial value: Undefined Undefined Undefined Undefined Undefined Undefined Undefined Undefined

| | |
|---|---|
| 0 | ID26 |
| 1 | |

| | |
|---|---|
| 0 | ID27 |
| 1 | |

| | |
|---|---|
| 0 | ID28 |
| 1 | |

| DLC3 | DLC2 | DLC1 | DLC0 | Number of bytes of a data field |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| • | • | • | • | • |
| • | • | • | • | • |
| 0 | 1 | 1 | 1 | 7 |
| 1 | 0 | 0 | 0 | 8 |

| | |
|---|---|
| 0 | Standard format (ID = 11 bits) |
| 1 | Extended format (ID = 29 bits) |
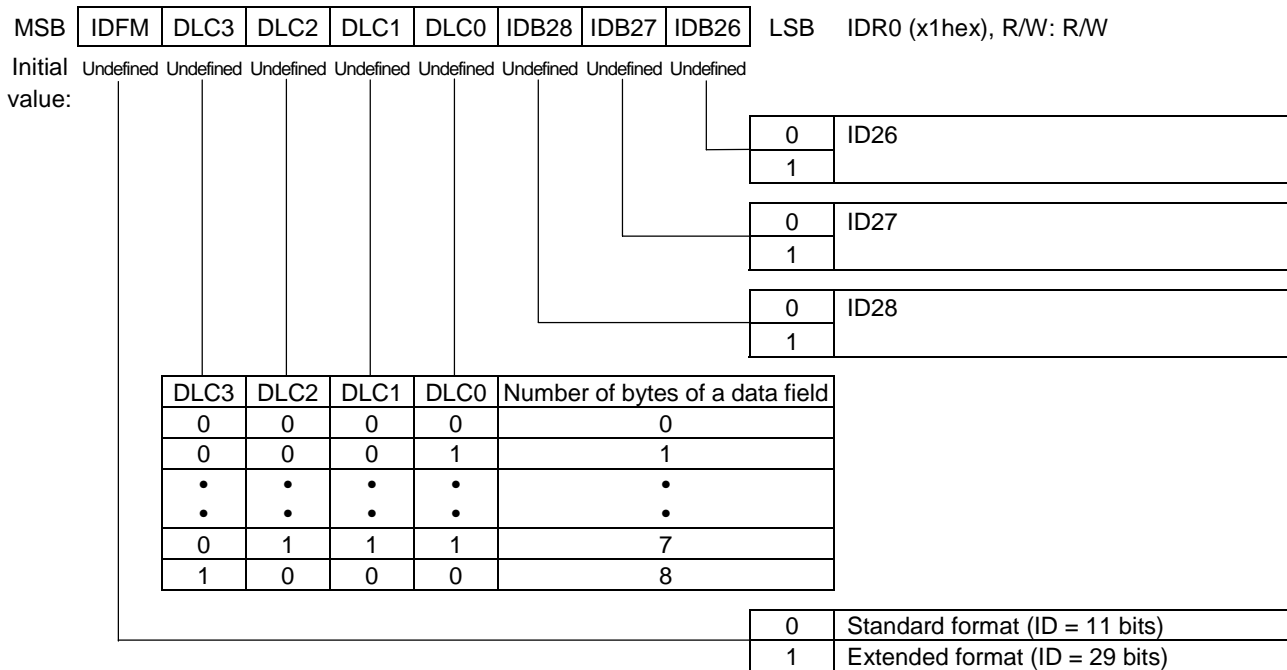
**Figure 2-3  Identifier 0 (IDR0)**

(1) Identifier: IDB28 to IDB26
These bits set the identifier field.
For standard format (IDFM = 0), the higher 3 bits (ID28 to ID26) of the 11 bits (ID28 to ID18) are set.
For extended format (IDFM = 1), the higher 3 bits (ID28 to ID26) of the 29 bits (ID28 to ID0) are set.
At reset, these bits are undefined.

**Note on Identifier**
The identifier field (ID28 to ID18 for standard format, and ID28 to ID0 for extended format) is overwritten with the received message's identifier, when the message box for which the group message function has been specified receives the message.

(2) Data length code: DLC3 to DLC0
These bits set the number of bytes of a data field. 0 to 8 can be set.  Do not set values other than 0 to 8.
At reset, these bits are undefined.

**Notes on Data Length Code**
When the received data length code (hereafter DLC) matches the DLC set in the message box, the number of bytes of data indicated by the received DLC is received and written to the message box.

When the received DLC does not match the DLC set in the message box, the MSM9225B operates as follows:
- The received DLC is written into the DLC field in the message box.
- The number of bytes of data indicated by the received DLC is received and written to the message box.

(3)  Frame format setting: IDFM
This bit sets the frame format.
At reset, the IDFM bit is undefined.

**Table 2-3  Frame Format**

| IDFM | Operation |
| --- | --- |
| 0 | Standard format (ID = 11 bits) |
| 1 | Extended format (ID = 29 bits) |

2.3.3 Identifier 1 (IDR1: x2hex)

This register sets the identifier.
The bit configuration is as follows:
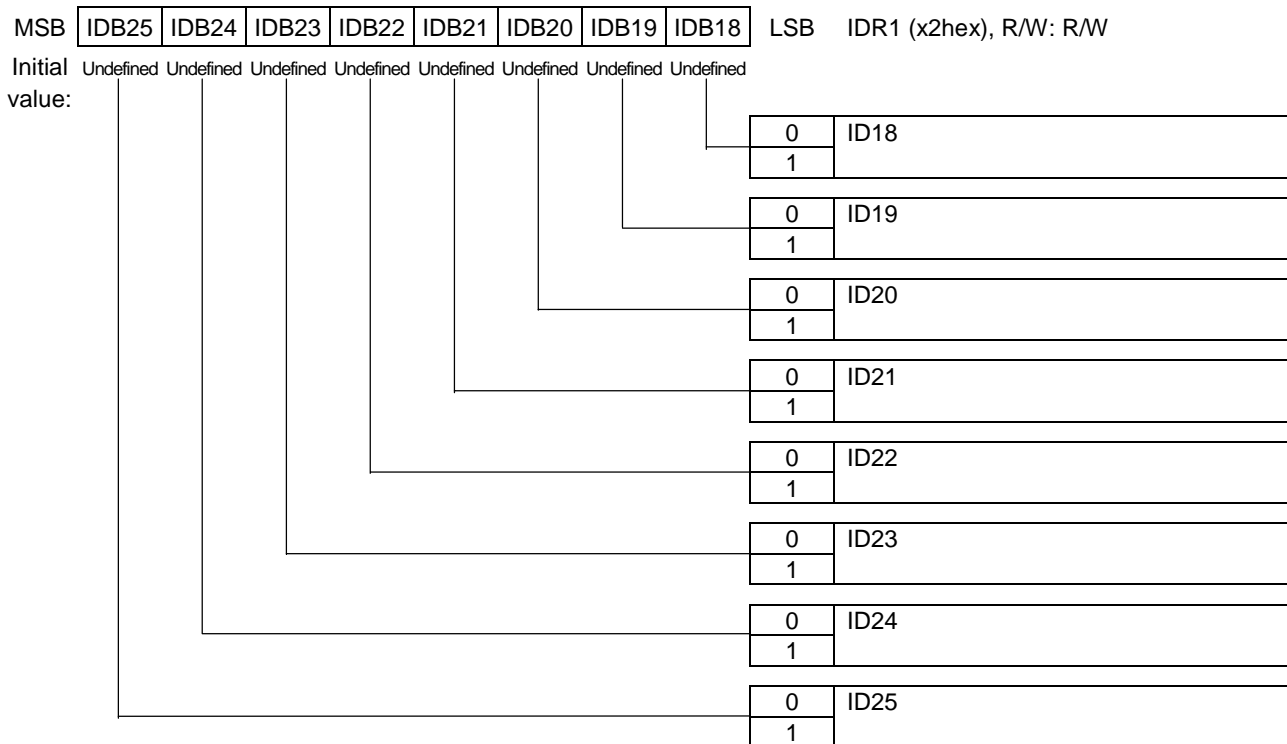


**Figure 2-4  Identifier 1 (IDR1)**

(1) Identifier: IDB25 to IDB18
These bits set the 8 bits of the identifier.
For standard format (IDFM = 0), the lower 8 bits (ID25 to ID18) of the 11 bits (ID28 to ID18) are set.
For extended format (IDFM = 1), ID25 to ID18 of the 29 bits (ID28 to ID0) are set.
At reset, these bits are undefined.

2.3.4 Identifiers 2, 3, 4/Messages 0-7 (MSG0-7 in the case of standard format; IDR2-4, TMSG0-7 in the case of extended format: x3 to xDhex)

In the case of the standard format (IDFM = 0), the addresses x3 to xAhex (MSG0-7) become the registers for storing the transmit/receive data.
In the case of the extended format (IDFM = 1), the addresses x3 to x5hex (IDR2-4) are used for setting the identifier field and the addresses x6 to xDhex (TMSG0-7) are used for the registers for storing the transmit/receive data.
In either case, the transmit/receive data can be stored up to a maximum of 8 bytes, and it is necessary to set beforehand the number of bytes that can be transmitted or received by the data length code. (See the explanation about the data length code (DLC) in Section 2.3.2.)
The contents of these registers after a reset will not be definite.
The bit configurations are shown below.

\* The top rows indicate the ID for the extended format setting and the bottom rows indicate the content of message 0 for the standard format setting.

| MSB | IDB17 | IDB16 | IDB15 | IDB14 | IDB13 | IDB12 | IDB11 | IDB10 | LSB | IDR2 (x3hex), R/W: R/W |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSG07 | MSG06 | MSG05 | MSG04 | MSG03 | MSG02 | MSG01 | MSG00 | | MSG0 (x3hex), R/W: R/W |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | | |

| | |
|---|---|
| 0 | ID10 for extended format, message 0 (bit |
| 1 | 0) for standard format. |

| | |
|---|---|
| 0 | ID11 for extended format, message 0 (bit |
| 1 | 1) for standard format. |

| | |
|---|---|
| 0 | ID12 for extended format, message 0 (bit |
| 1 | 2) for standard format. |

| | |
|---|---|
| 0 | ID13 for extended format, message 0 (bit |
| 1 | 3) for standard format. |

| | |
|---|---|
| 0 | ID14 for extended format, message 0 (bit |
| 1 | 4) for standard format. |

| | |
|---|---|
| 0 | ID15 for extended format, message 0 (bit |
| 1 | 5) for standard format. |

| | |
|---|---|
| 0 | ID16 for extended format, message 0 (bit |
| 1 | 6) for standard format. |

| | |
|---|---|
| 0 | ID17 for extended format, message 0 (bit |
| 1 | 7) for standard format. |

**Figure 2-5  Identifier 2/Message 0 (IDR2/MSG0)**

\* The top rows indicate the ID for the extended format setting and the bottom rows indicate the content of message 1 for the standard format setting.

| MSB | IDB9 | IDB8 | IDB7 | IDB6 | IDB5 | IDB4 | IDB3 | IDB2 | LSB | IDR3 (x4hex), R/W: R/W |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSG17 | MSG16 | MSG15 | MSG14 | MSG13 | MSG12 | MSG11 | MSG10 | | MSG1 (x4hex), R/W: R/W |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | | |

| | |
|---|---|
| 0 | ID2 for extended format, message 1 |
| 1 | (bit 0) for standard format. |

| | |
|---|---|
| 0 | ID3 for extended format, message 1 |
| 1 | (bit 1) for standard format. |

| | |
|---|---|
| 0 | ID4 for extended format, message 1 |
| 1 | (bit 2) for standard format. |

| | |
|---|---|
| 0 | ID5 for extended format, message 1 |
| 1 | (bit 3) for standard format. |

| | |
|---|---|
| 0 | ID6 for extended format, message 1 |
| 1 | (bit 4) for standard format. |

| | |
|---|---|
| 0 | ID7 for extended format, message 1 |
| 1 | (bit 5) for standard format. |

| | |
|---|---|
| 0 | ID8 for extended format, message 1 |
| 1 | (bit 6) for standard format. |

| | |
|---|---|
| 0 | ID9 for extended format, message 1 |
| 1 | (bit 7) for standard format. |

**Figure 2-6  Identifier 3/Message 1 (IDR3/MSG1)**

* The top rows indicate the ID for the extended format setting and the bottom rows indicate the content of message 2 for the standard format setting.

| MSB | IDB1 | IDB0 | Not used | Not used | Not used | Not used | Not used | Not used | LSB | IDR4 (x5hex), R/W: R/W |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSG27 | MSG26 | MSG25 | MSG24 | MSG23 | MSG22 | MSG21 | MSG20 | | MSG2 (x5hex), R/W: R/W |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | | |

| | |
|---|---|
| 0 | Not used for extended format, message 2 |
| 1 | (bit 0) for standard format |

| | |
|---|---|
| 0 | Not used for extended format, message 2 |
| 1 | (bit 1) for standard format |

| | |
|---|---|
| 0 | Not used for extended format, message 2 |
| 1 | (bit 2) for standard format |

| | |
|---|---|
| 0 | Not used for extended format, message 2 |
| 1 | (bit 3) for standard format |

| | |
|---|---|
| 0 | Not used for extended format, message 2 |
| 1 | (bit 4) for standard format |

| | |
|---|---|
| 0 | Not used for extended format, message 2 |
| 1 | (bit 5) for standard format |

| | |
|---|---|
| 0 | ID0 for extended format, message 2 |
| 1 | (bit 6) for standard format. |

| | |
|---|---|
| 0 | ID1 for extended format, message 2 |
| 1 | (bit 7) for standard format. |

Write "0" to the unused bits.

**Figure 2-7  Identifier 4/Message 2 (IDR4/MSG2)**

* The top rows indicate the content of message 0 for the extended format setting and the bottom rows indicate the content of message 3 for the standard format setting.

| MSB | TMSG07 | TMSG06 | TMSG05 | TMSG04 | TMSG03 | TMSG02 | TMSG01 | TMSG00 | LS | TMSG0 (x6hex), R/W: R/W |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSG37 | MSG36 | MSG35 | MSG34 | MSG33 | MSG32 | MSG31 | MSG30 | B | MSG3 (x6hex), R/W: R/W |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | | |

| | |
|---|---|
| 0 | Message 0 (bit 0) for extended format, |
| 1 | message 3 (bit 0) for standard format |

| | |
|---|---|
| 0 | Message 0 (bit 1) for extended format, |
| 1 | message 3 (bit 1) for standard format |

| | |
|---|---|
| 0 | Message 0 (bit 2) for extended format, |
| 1 | message 3 (bit 2) for standard format |

| | |
|---|---|
| 0 | Message 0 (bit 3) for extended format, |
| 1 | message 3 (bit 3) for standard format |

| | |
|---|---|
| 0 | Message 0 (bit 4) for extended format, |
| 1 | message 3 (bit 4) for standard format |

| | |
|---|---|
| 0 | Message 0 (bit 5) for extended format, |
| 1 | message 3 (bit 5) for standard format |

| | |
|---|---|
| 0 | Message 0 (bit 6) for extended format, |
| 1 | message 3 (bit 6) for standard format |

| | |
|---|---|
| 0 | Message 0 (bit 7) for extended format, |
| 1 | message 3 (bit 7) for standard format |

**Figure 2-8  Message 0/Message 3 (TMSG0/MSG3)**

\* The top rows indicate the content of message 1 for the extended format setting and the bottom rows indicate the content of message 4 for the standard format setting.

| MSB | TMSG17 | TMSG16 | TMSG15 | TMSG14 | TMSG13 | TMSG12 | TMSG11 | TMSG10 | LSB | TMSG1 (x7hex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|--------|
| | MSG47 | MSG46 | MSG45 | MSG44 | MSG43 | MSG42 | MSG41 | MSG40 | | MSG4 (x7hex), R/W: R/W |

Initial value:

| | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
|---|---|---|---|---|---|---|---|---|

| 0 | Message 1 (bit 0) for extended format, |
|---|---|
| 1 | message 4 (bit 0) for standard format |

| 0 | Message 1 (bit 1) for extended format, |
|---|---|
| 1 | message 4 (bit 1) for standard format |

| 0 | Message 1 (bit 2) for extended format, |
|---|---|
| 1 | message 4 (bit 2) for standard format |

| 0 | Message 1 (bit 3) for extended format, |
|---|---|
| 1 | message 4 (bit 3) for standard format |

| 0 | Message 1 (bit 4) for extended format, |
|---|---|
| 1 | message 4 (bit 4) for standard format |

| 0 | Message 1 (bit 5) for extended format, |
|---|---|
| 1 | message 4 (bit 5) for standard format |

| 0 | Message 1 (bit 6) for extended format, |
|---|---|
| 1 | message 4 (bit 6) for standard format |

| 0 | Message 1 (bit 7) for extended format, |
|---|---|
| 1 | message 4 (bit 7) for standard format |

**Figure 2-9  Message 1/Message 4 (TMSG1/MSG4)**

\* The top rows indicate the content of message 2 for the extended format setting and the bottom rows indicate the content of message 5 for the standard format setting.

| MSB | TMSG27 | TMSG26 | TMSG25 | TMSG24 | TMSG23 | TMSG22 | TMSG21 | TMSG20 | LSB | TMSG2 (x8hex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|--------|
| | MSG57 | MSG56 | MSG55 | MSG54 | MSG53 | MSG52 | MSG51 | MSG50 | | MSG5 (x8hex), R/W: R/W |

Initial value:

| | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
|---|---|---|---|---|---|---|---|---|

| 0 | Message 2 (bit 0) for extended format, |
|---|---|
| 1 | message 5 (bit 0) for standard format |

| 0 | Message 2 (bit 1) for extended format, |
|---|---|
| 1 | message 5 (bit 1) for standard format |

| 0 | Message 2 (bit 2) for extended format, |
|---|---|
| 1 | message 5 (bit 2) for standard format |

| 0 | Message 2 (bit 3) for extended format, |
|---|---|
| 1 | message 5 (bit 3) for standard format |

| 0 | Message 2 (bit 4) for extended format, |
|---|---|
| 1 | message 5 (bit 4) for standard format |

| 0 | Message 2 (bit 5) for extended format, |
|---|---|
| 1 | message 5 (bit 5) for standard format |

| 0 | Message 2 (bit 6) for extended format, |
|---|---|
| 1 | message 5 (bit 6) for standard format |

| 0 | Message 2 (bit 7) for extended format, |
|---|---|
| 1 | message 5 (bit 7) for standard format |

**Figure 2-10  Message 2/Message 5 (TMSG2/MSG5)**

* The top rows indicate the content of message 3 for the extended format setting and the bottom rows indicate the content of message 6 for the standard format setting.

| MSB | TMSG37 | TMSG36 | TMSG35 | TMSG34 | TMSG33 | TMSG32 | TMSG31 | TMSG30 | LSB | TMSG3 (x9hex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|-------------------------|
|     | MSG67  | MSG66  | MSG65  | MSG64  | MSG63  | MSG62  | MSG61  | MSG60  |     | MSG6 (x9hex), R/W: R/W  |

Initial value: Undefined Undefined Undefined Undefined Undefined Undefined Undefined Undefined

| 0 | Message 3 (bit 0) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 0) for standard format  |

| 0 | Message 3 (bit 1) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 1) for standard format  |

| 0 | Message 3 (bit 2) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 2) for standard format  |

| 0 | Message 3 (bit 3) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 3) for standard format  |

| 0 | Message 3 (bit 4) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 4) for standard format  |

| 0 | Message 3 (bit 5) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 5) for standard format  |

| 0 | Message 3 (bit 6) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 6) for standard format  |

| 0 | Message 3 (bit 7) for extended format, |
|---|----------------------------------------|
| 1 | message 6 (bit 7) for standard format  |

**Figure 2-11  Message 3/Message 6 (TMSG3/MSG6)**

* The top rows indicate the content of message 4 for the extended format setting and the bottom rows indicate the content of message 7 for the standard format setting.

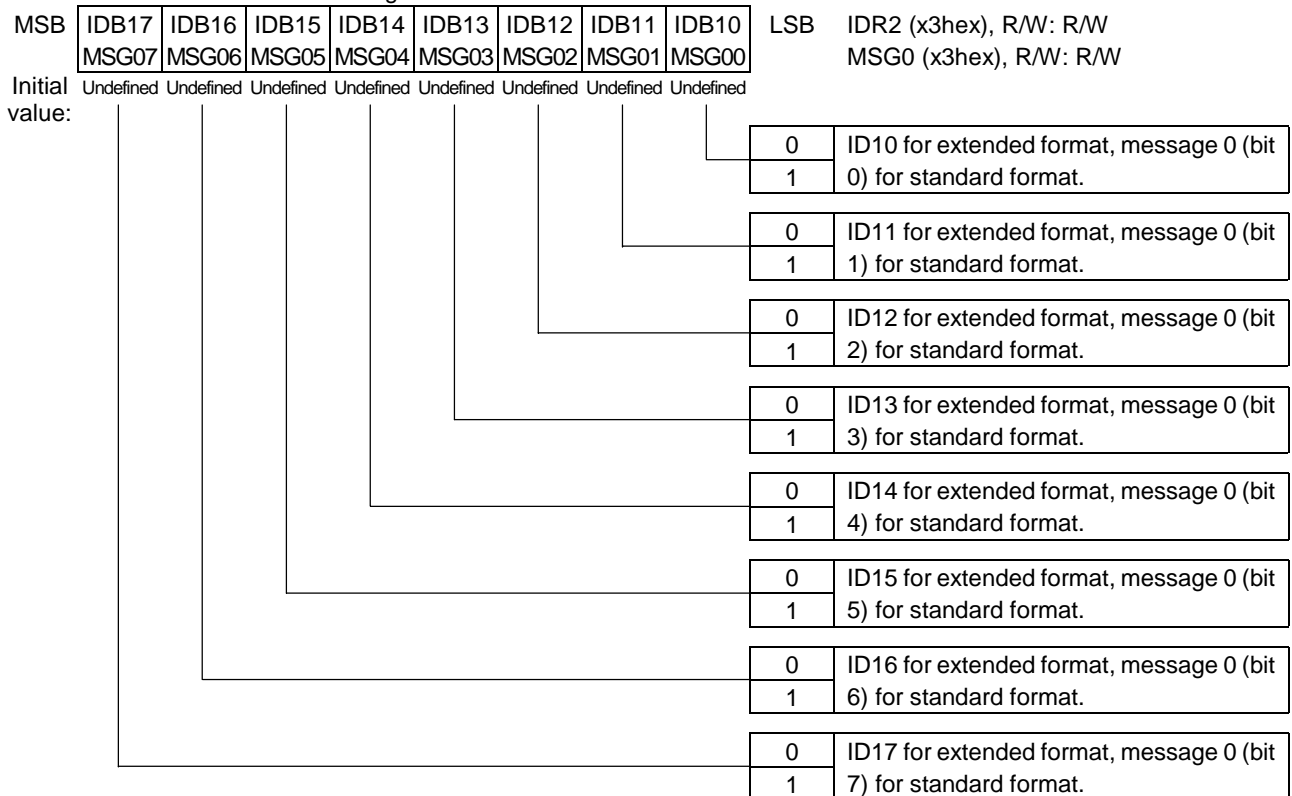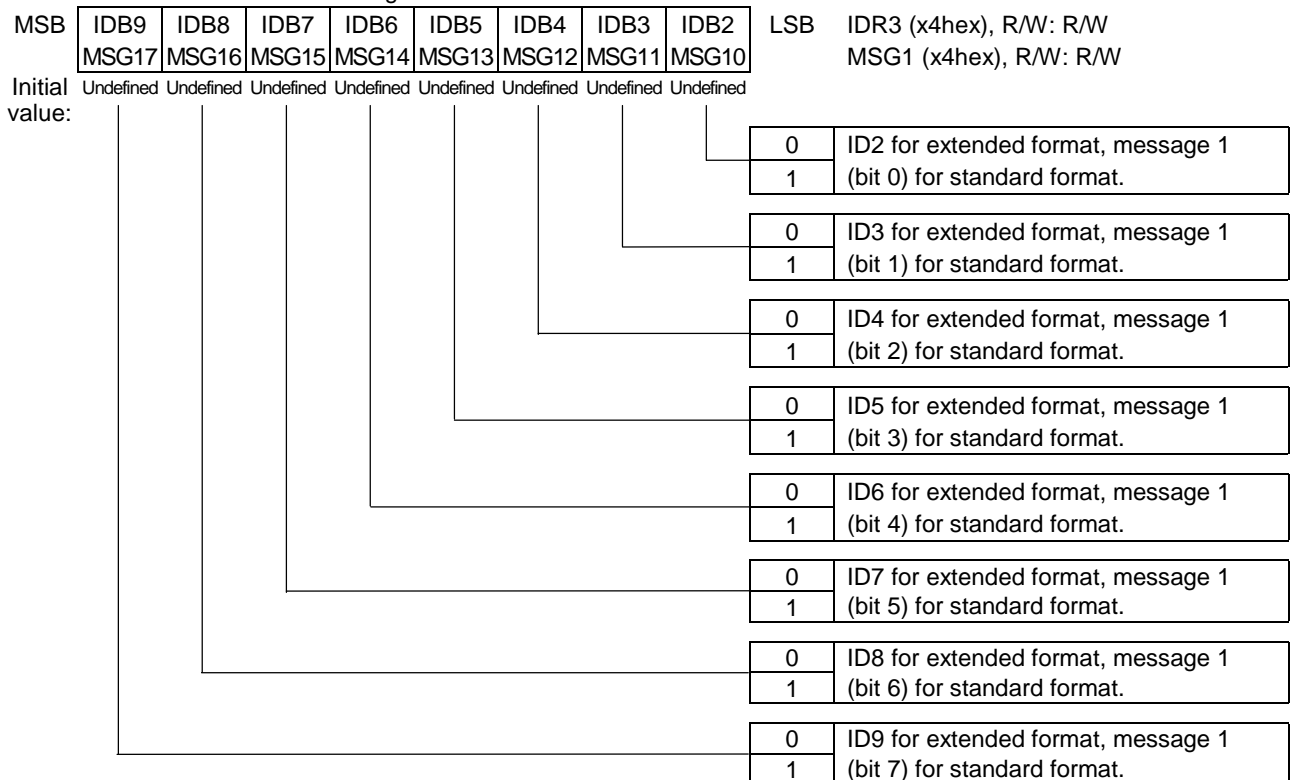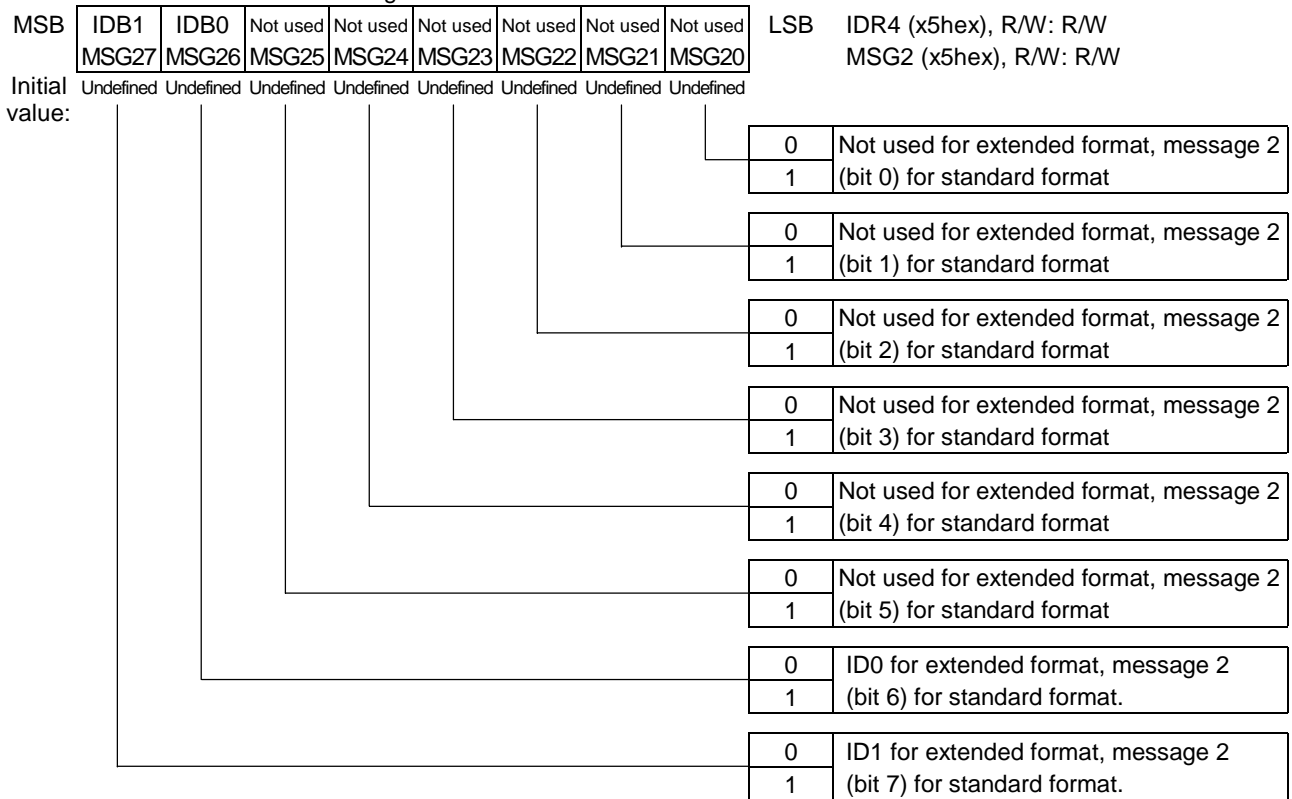| MSB | TMSG47 | TMSG46 | TMSG45 | TMSG44 | TMSG43 | TMSG42 | TMSG41 | TMSG40 | LSB | TMSG4 (xAhex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|-------------------------|
|     | MSG77  | MSG76  | MSG75  | MSG74  | MSG73  | MSG72  | MSG71  | MSG70  |     | MSG7 (xAhex), R/W: R/W  |

Initial value: Undefined Undefined Undefined Undefined Undefined Undefined Undefined Undefined

| 0 | Message 4 (bit 0) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 0) for standard format  |

| 0 | Message 4 (bit 1) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 1) for standard format  |

| 0 | Message 4 (bit 2) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 2) for standard format  |

| 0 | Message 4 (bit 3) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 3) for standard format  |

| 0 | Message 4 (bit 4) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 4) for standard format  |

| 0 | Message 4 (bit 5) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 5) for standard format  |

| 0 | Message 4 (bit 6) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 6) for standard format  |

| 0 | Message 4 (bit 7) for extended format, |
|---|----------------------------------------|
| 1 | message 7 (bit 7) for standard format  |

**Figure 2-12  Message 4/Message 7 (TMSG4/MSG7)**

* The content of message 5 for the extended format setting is shown below.

MSB | TMSG57 | TMSG56 | TMSG55 | TMSG54 | TMSG53 | TMSG52 | TMSG51 | TMSG50 | LSB  TMSG5 (xBhex), R/W: R/W

Initial value: Undefined Undefined Undefined Undefined Undefined Undefined Undefined Undefined

| 0 | Message 5 (bit 0) for extended format |
| 1 | setting |

| 0 | Message 5 (bit 1) for extended format |
| 1 | setting |

| 0 | Message 5 (bit 2) for extended format |
| 1 | setting |

| 0 | Message 5 (bit 3) for extended format |
| 1 | setting |

| 0 | Message 5 (bit 4) for extended format |
| 1 | setting |

| 0 | Message 5 (bit 5) for extended format |
| 1 | setting |

| 0 | Message 5 (bit 6) for extended format |
| 1 | setting |

| 0 | Message 5 (bit 7) for extended format |
| 1 | setting |

**Figure 2-13  Message 5 (TMSG5)**

* The content of message 6 for the extended format setting is shown below.

MSB | TMSG67 | TMSG66 | TMSG65 | TMSG64 | TMSG63 | TMSG62 | TMSG61 | TMSG60 | LSB  TMSG6 (xChex), R/W: R/W

Initial value: Undefined Undefined Undefined Undefined Undefined Undefined Undefined Undefined

| 0 | Message 6 (bit 0) for extended format |
| 1 | setting |

| 0 | Message 6 (bit 1) for extended format |
| 1 | setting |

| 0 | Message 6 (bit 2) for extended format |
| 1 | setting |

| 0 | Message 6 (bit 3) for extended format |
| 1 | setting |

| 0 | Message 6 (bit 4) for extended format |
| 1 | setting |

| 0 | Message 6 (bit 5) for extended format |
| 1 | setting |

| 0 | Message 6 (bit 6) for extended format |
| 1 | setting |

| 0 | Message 6 (bit 7) for extended format |
| 1 | setting |

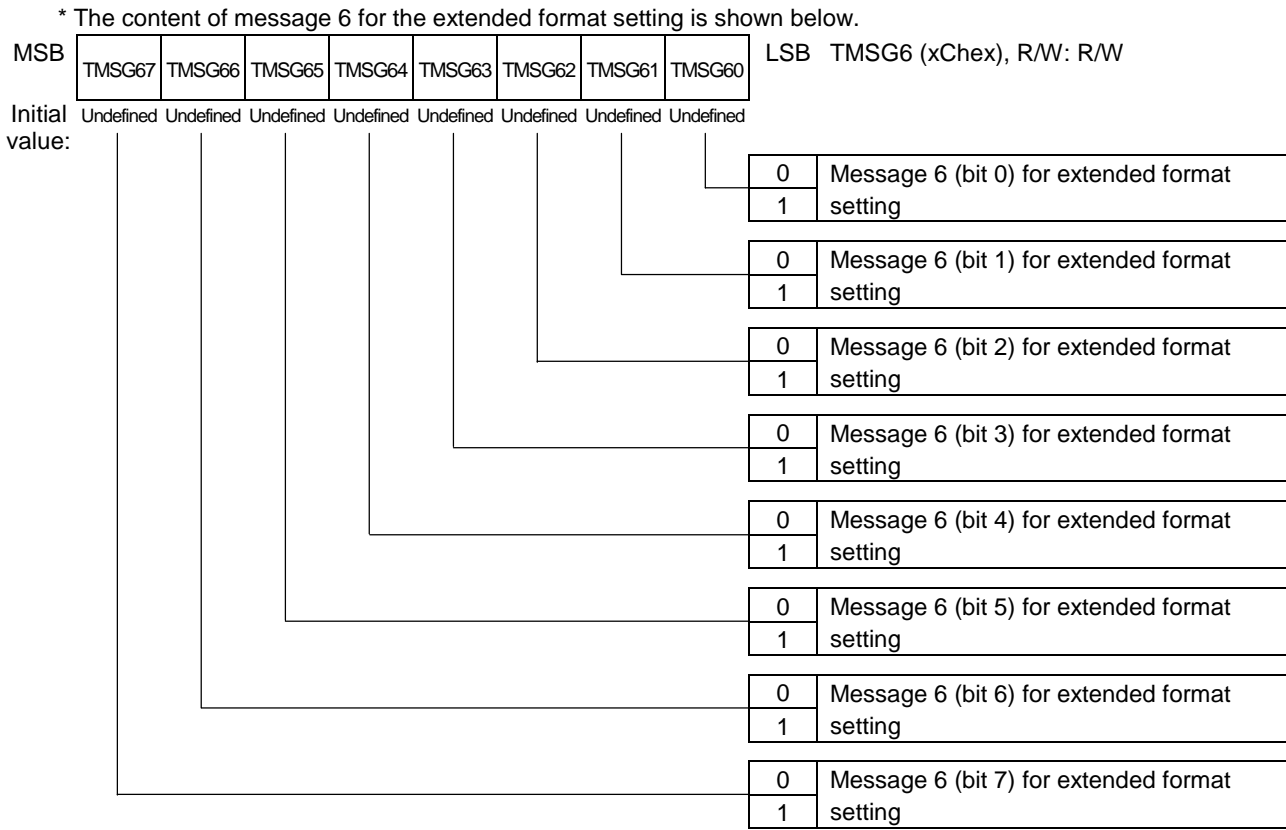**Figure 2-14  Message 6 (TMSG6)**

\* The content of message 7 for the extended format setting is shown below.

MSB

| TMSG77 | TMSG76 | TMSG75 | TMSG74 | TMSG73 | TMSG72 | TMSG71 | TMSG70 |
|--------|--------|--------|--------|--------|--------|--------|--------|

LSB   TMSG7 (xDhex), R/W: R/W

Initial value:

| Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|

| 0 | Message 7 (bit 0) for extended format |
| 1 | setting |

| 0 | Message 7 (bit 1) for extended format |
| 1 | setting |

| 0 | Message 7 (bit 2) for extended format |
| 1 | setting |

| 0 | Message 7 (bit 3) for extended format |
| 1 | setting |

| 0 | Message 7 (bit 4) for extended format |
| 1 | setting |

| 0 | Message 7 (bit 5) for extended format |
| 1 | setting |

| 0 | Message 7 (bit 6) for extended format |
| 1 | setting |

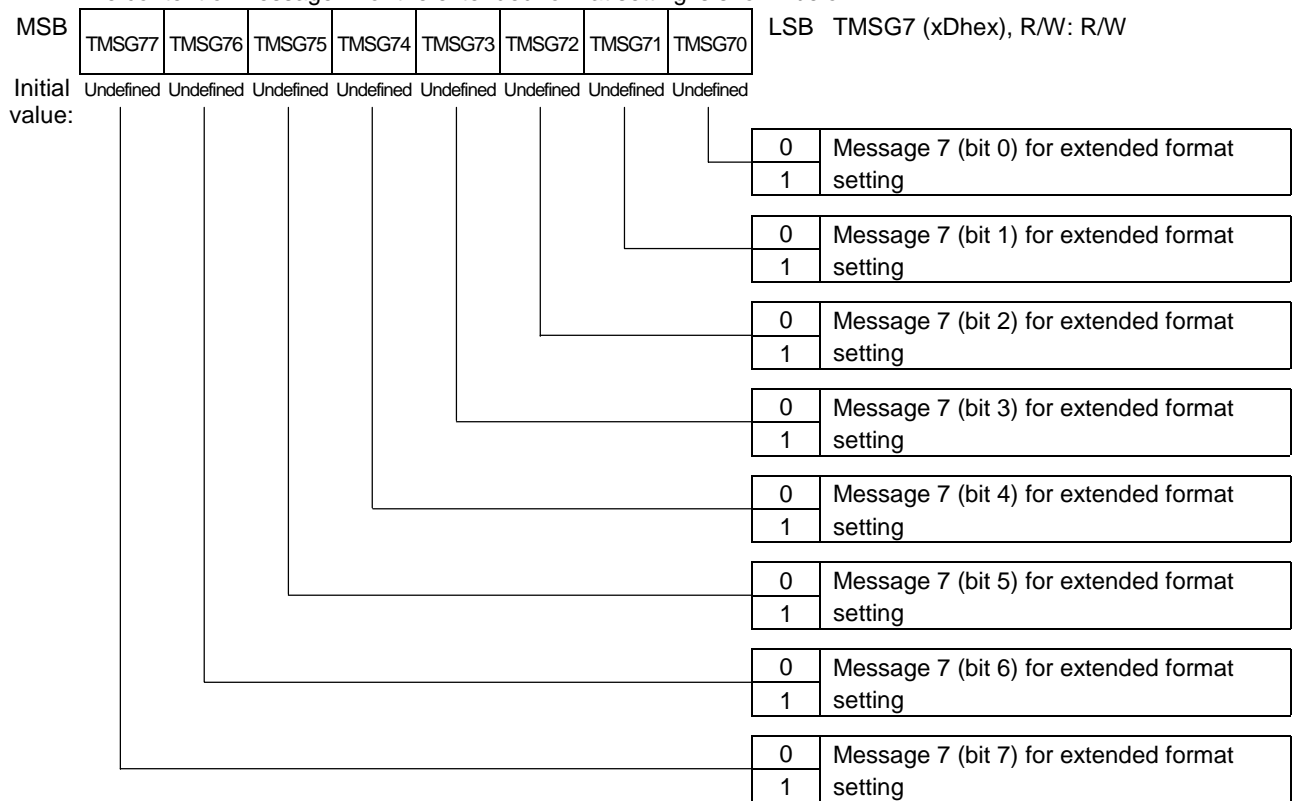| 0 | Message 7 (bit 7) for extended format |
| 1 | setting |

**Figure 2-15  Message 7 (TMSG7)**

**Notes on Identifier (ID) and Message Priority**

- Priority of message
A message has the priority determined by the identifier setting. To determine priority, identifiers of messages are compared from the higher bit, and the identifier (set to "0") detected first has the higher priority (see the example below).



In this example, priority is determined at the shaded bits.

**Figure 2-16  Message Priority**

- When same identifiers are set to multiple messages boxes
When same identifiers are set to multiple message boxes, operations are as follows:

1. Transmit operation
Messages are transmitted sequentially from the smaller message box number.

2. Receive operation
Data is always written to the message box with the smallest message box number, and never written to other message boxes.

For example, if the same identifier is set at message box numbers 1 to 4, as shown in Figure 2-17, operations are as follows:

- Transmit operation
If every message box below is set for transmission, messages are transmitted sequentially in the order of message box number $5 \rightarrow 0 \rightarrow 6 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4$.

- Receive operation
When the identifier "11100111001" is received from the CAN bus, received data is always written to the message box which is indicated by the message box number 1.



**Figure 2-17  Setting of the Same Identifier**

## 2.4 Control Registers

These registers listed below control various operations of CAN.
Table 2-4 lists the control registers.
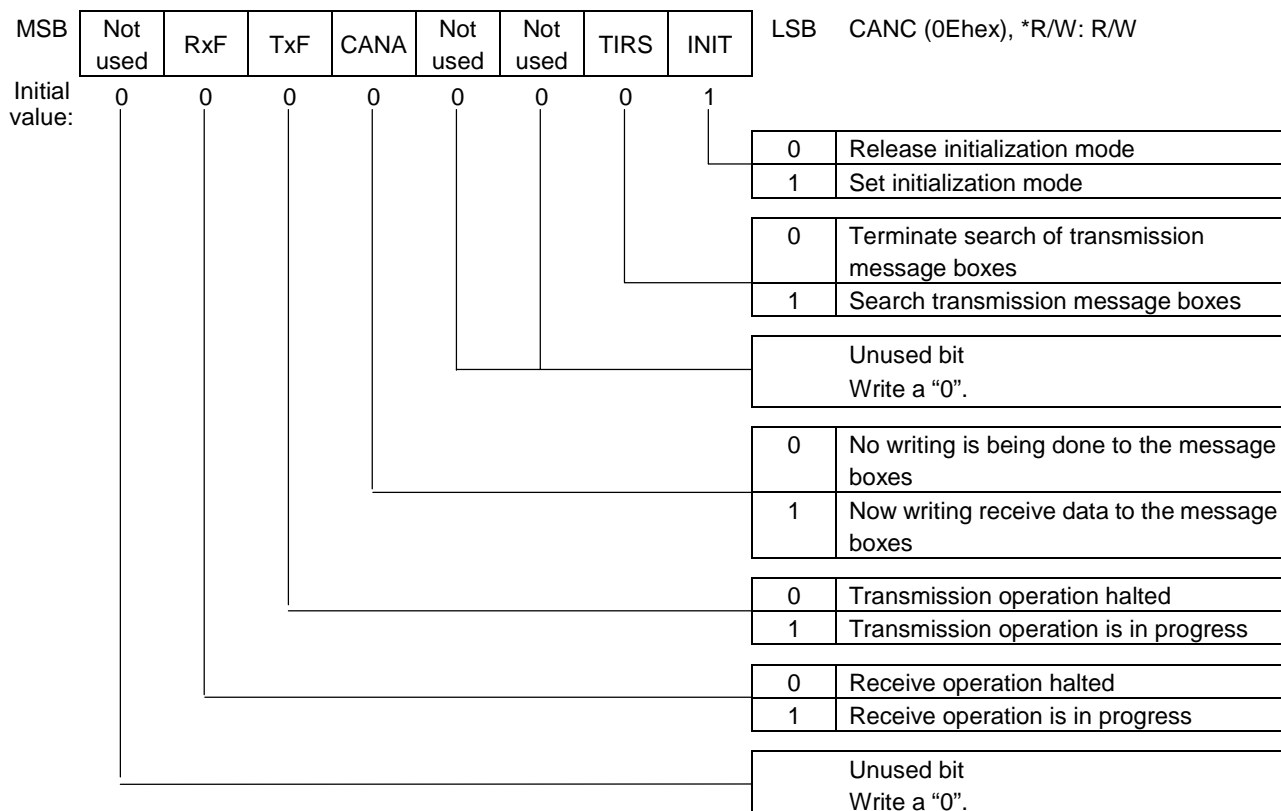
**Table 2-4  Control Registers**

| Address | Symbol | Name | Initial value | R/W |
|---|---|---|---|---|
| 0EH | CANC | CAN control register | 01hex | Bits 4, 5 and 6 are read-only; bits 2 and 7 are not used; the other bits are R/W accessible. |
| 0FH | CANI | CAN interrupt control register | 00hex | Bit 3 is not used. The other bits are R/W accessible. |
| 1EH | NMES | Message box count setting register | 00hex | Bits 4 to 7 are not used. The other bits are R/W accessible. |
| 1FH | BTR0 | CAN bus timing register 0 | 00hex | R/W |
| 2EH | BTR1 | CAN bus timing register 1 | 00hex | Bit 7 is not used. The other bits are R/W accessible. |
| 2FH | TIOC | Communication input/output control register | 00hex | R/W |
| 3EH | GMR0 | Group message register 0 | 00hex | Bits 4 to 6 are not used. The other bits are R/W accessible. |
| 3FH | GMR1 | Group message register 1 | 00hex | Bits 4 to 6 are not used. The other bits are R/W accessible. |
| 4EH | GMSK00 | Message mask register 00 | 00hex | R/W |
| 4FH | GMSK01 | Message mask register 01 | 00hex | R/W |
| 5EH | GMSK02 | Message mask register 02 | 00hex | R/W |
| 5FH | GMSK03 | Message mask register 03 | 00hex | Bits 0 to 2 are not used. The other bits are R/W accessible. |
| 6EH | GMSK10 | Message mask register 10 | 00hex | R/W |
| 6FH | GMSK11 | Message mask register 11 | 00hex | R/W |
| 7EH | GMSK12 | Message mask register 12 | 00hex | R/W |
| 7FH | GMSK13 | Message mask register 13 | 00hex | Bits 0 to 2 are not used. The other bits are R/W accessible. |
| 8EH | STBY | Standby control register | 00hex | Bits 2 to 7 are not used. The other bits are R/W accessible. |
| 8FH | CANC2 | CAN control register 2 | 00hex | R/W |
| 9EH | TMN | Communication message box number register | *–0hex | Bits 4 to 7 are not used. The other bits are read-only. |
| 9FH | CANS | CAN status register | 00hex | Bits 2, 3, 7 are not used. The other bits are read-only. |
| AEH | TEC | Transmission error counter | 00hex | Read-only |
| AFH | REC | Receive error counter | 00hex | Read-only |
| BEH | CANS2 | CAN status register 2 | 00hex | R/W |
| BFH | BOCO | Bus off release counter | 00hex | Read-only |
| CEH | — | Not used (reserve area) Do not access this area. | | |
| CFH | — | | | |
| DEH | — | | | |
| DFH | — | | | |
| EEH | — | | | |
| EFH | — | | | |
| FEH | — | | | |
| FFH | — | | | |

\* "–" in "–0hex" indicates that the value of the nibble is undefined.
Write a "0" to the unused bits.

### 2.4.1 CAN Control Register (CANC: 0Ehex)

This register controls the operation of CAN.
The bit configuration is as follows:

| MSB | Not used | RxF | TxF | CANA | Not used | Not used | TIRS | INIT | LSB | CANC (0Ehex), *R/W: R/W |
|-----|------|-----|-----|------|------|------|------|------|-----|-------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | | |

| 0 | Release initialization mode |
|---|------------------------------|
| 1 | Set initialization mode |

| 0 | Terminate search of transmission message boxes |
|---|-------------------------------------------------|
| 1 | Search transmission message boxes |

| Unused bit Write a "0". |
|--------------------------|

| 0 | No writing is being done to the message boxes |
|---|-----------------------------------------------|
| 1 | Now writing receive data to the message boxes |

| 0 | Transmission operation halted |
|---|-------------------------------|
| 1 | Transmission operation is in progress |

| 0 | Receive operation halted |
|---|--------------------------|
| 1 | Receive operation is in progress |

| Unused bit Write a "0". |
|--------------------------|

* TIRS is in the read-only state when "0", and CANA, TxF, RxF are read-only bits.

**Figure 2-18  CAN Control Register (CANC)**

(1) Initialization bit: INIT
This is the bit for setting the initialization mode of the communication control section.
At the time of initialization, start the initialization after writing a "1" to INIT and reading it to ensure that INIT has been set to "1".  Also, at the end of initialization, write a "0" to INIT, then read this bit to make sure that it has been set to "0".  In either case, make sure to carry out the above operations because neither "1" nor "0" will be set immediately.  Note that data cannot be written to the INIT bit while the CAN bus is at the dominant level.
If INIT is set to "1" during transmission or reception, the initialization is started after completing the communication.  Although the communication operation stops when INIT is set to "1", the contents of the message memory and the control registers will be retained, except the content of the MMA bit of the message control register within the message box.
To initialize the message memory, first write the number of message boxes to be used in the message box count setting register NMES, and then write the message control register, identifier 1, and identifier 2 in sequence from the message box number 0 for all the message boxes to be used.
At reset, INIT is set to "1".

(2) Search for the transmit identifier: TIRS
When this bit is set, the identifiers are scanned starting from the message box 0 up to last message box specified by NMES, the messages with the transmission request bit TRQ set to "1" are detected, and the transmission is started from the one with the highest priority.  TIRS will be set to "0" when all the messages with the transmission request TRQ set to "1" have been transmitted, or when no message box with TRQ being "1" is detected as a result of the search.
Writing a "0" to TIRS when it has already been set to "1" will stop the transmission of the message after the transmission which has already been started is completed.
At reset, TIRS is set to "0".
* See Appendix C "Transmission Failure of MSM9225B" for transmission operation.

(3) CAN write flag: CANA
This bit is used to indicate receive data write status to the message box. CANA is "1" while CAN is writing receive data to the message box.
This is a read-only flag.

(4) Transmission flag: TxF
This bit is used to indicate transmission operation status.
When TxF is "0", CAN is in transmission operation stop status.
When TxF is "1", CAN is in transmission operation status.
This is a read-only flag.

(5) Receive flag: RxF
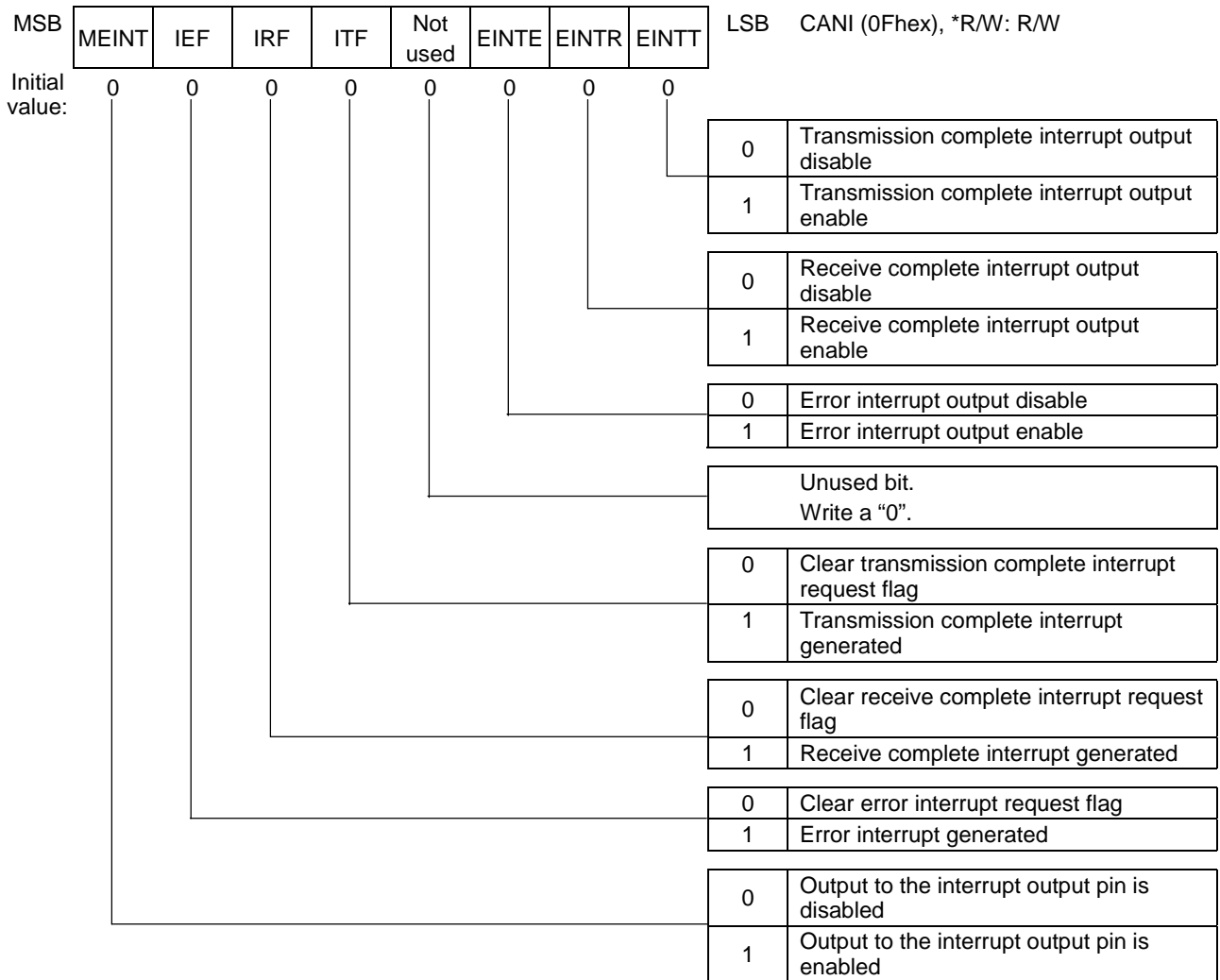This bit is used to indicate receive operation status.
When RxF is "0", CAN is in receive operation stop status.
When RxF is "1", CAN is in receive operation status.
This is a read-only flag.

### 2.4.2 CAN Interrupt Control Register (CANI: 0Fhex)

This register controls CAN interrupts.
The bit configuration is as follows:

| MSB | MEINT | IEF | IRF | ITF | Not used | EINTE | EINTR | EINTT | LSB | CANI (0Fhex), *R/W: R/W |
|-----|-------|-----|-----|-----|----------|-------|-------|-------|-----|-------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| | |
|---|---|
| 0 | Transmission complete interrupt output disable |
| 1 | Transmission complete interrupt output enable |

| | |
|---|---|
| 0 | Receive complete interrupt output disable |
| 1 | Receive complete interrupt output enable |

| | |
|---|---|
| 0 | Error interrupt output disable |
| 1 | Error interrupt output enable |

| | |
|---|---|
| | Unused bit. Write a "0". |

| | |
|---|---|
| 0 | Clear transmission complete interrupt request flag |
| 1 | Transmission complete interrupt generated |

| | |
|---|---|
| 0 | Clear receive complete interrupt request flag |
| 1 | Receive complete interrupt generated |

| | |
|---|---|
| 0 | Clear error interrupt request flag |
| 1 | Error interrupt generated |

| | |
|---|---|
| 0 | Output to the interrupt output pin is disabled |
| 1 | Output to the interrupt output pin is enabled |

\* ITF, IRF, and IEF are in the read-only state when they are "1". "1" cannot be written.

**Figure 2-19  CAN Interrupt Control Register (CANI)**

(1) Transmission complete interrupt output enable: EINTT
This bit is used to output transmission complete interrupt signal INTT from interrupt pin $\overline{\text{INT}}$ when transmission completes.
When EINTT is "0", a transmission complete interrupt signal is not output from the interrupt pin.
When EINTT is "1", a transmission complete interrupt signal is output from the interrupt pin.
At reset, EINTT is set to "0".

(2) Receive complete interrupt output enable: EINTR
This bit is used to output receive complete interrupt signal INTR from interrupt pin $\overline{\text{INT}}$ when reception completes.
When EINTR is "0", a receive complete interrupt signal is not output from the interrupt pin.
When EINTR is "1", a receive complete interrupt signal is output from the interrupt pin.
At reset, EINTR is set to "0".

(3) Error interrupt output enable: EINTE

When an error occurs, this bit is used to output error interrupt signal INTE from interrupt pin $\overline{\text{INT}}$.
When EINTE is "0", an error interrupt signal is not output from the interrupt pin.
When EINTE is "1", an error interrupt signal is output from the interrupt pin.
At reset, EINTE is set to "0".

(4) Transmission complete interrupt request flag: ITF

ITF becomes "1" when a transmission complete interrupt is generated. Only "0" can be written to this bit.
At reset, ITF is set to "0".

(5) Receive complete interrupt request flag: IRF

IRF becomes "1" when a receive complete interrupt is generated. Only "0" can be written to this bit.
At reset, IRF is set to "0".

(6) Error interrupt request flag: IEF

IEF becomes "1" when an error occurs. Only "0" can be written to this bit.
At reset, IEF is set to "0".

(7) Master interrupt control enable: MEINT

This bit is used to set enable/disable of interrupts.
The outline of interrupt control is shown in Figure 2-20.
When MEINT is "0", interrupt request control is disabled.
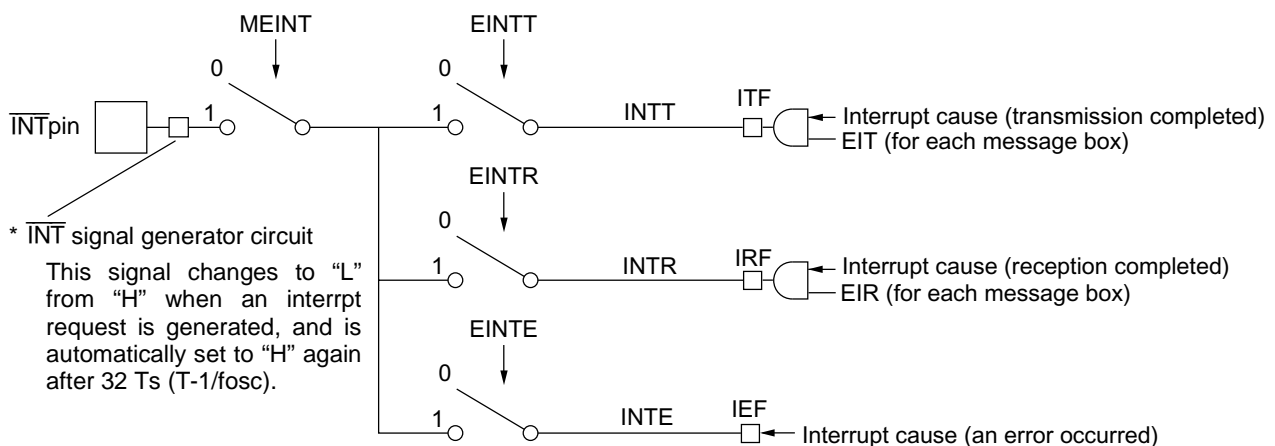When MEINT is "1", interrupt request control is enabled.
At reset, MEINT is set to "0".



**Figure 2-20  Interrupt Control**

2.4.3　Message Box Count Setting Register (NMES: 1Ehex)

This is a register to set the number of message boxes to be used.
A maximum of 16 message boxes can be set, with message box numbers 0 to F.
Writing to NMES is enabled when initialize bit INIT of the CAN control register (CANC: 0Ehex) is "1".
At reset, NMES is set to "00000000"
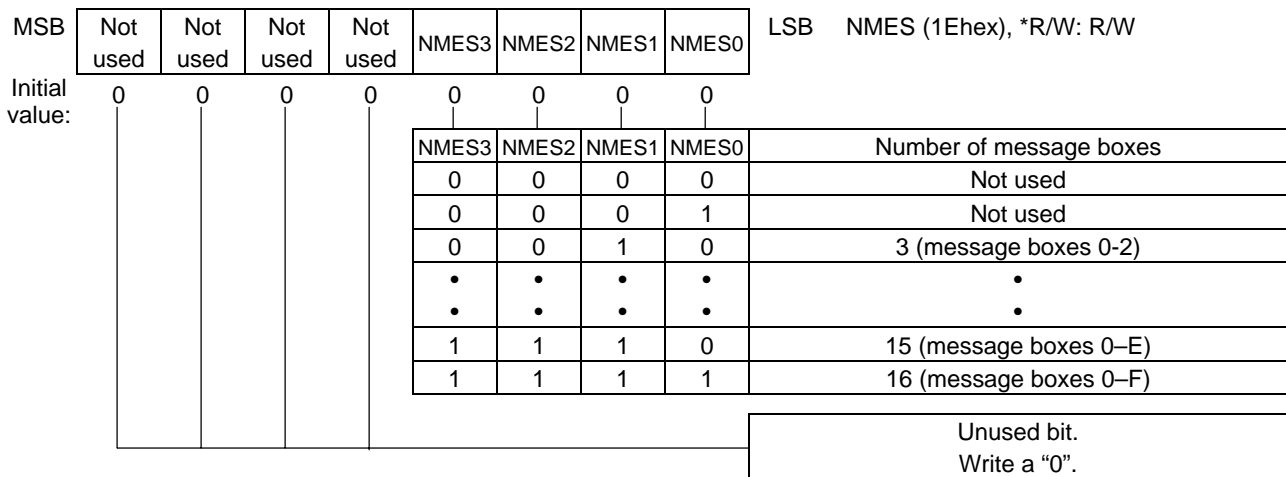The bit configuration and relationship between message box number and number of message boxes are as follows:

| MSB | Not used | Not used | Not used | Not used | NMES3 | NMES2 | NMES1 | NMES0 | LSB | NMES (1Ehex), *R/W: R/W |
|---|---|---|---|---|---|---|---|---|---|---|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| NMES3 | NMES2 | NMES1 | NMES0 | Number of message boxes |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Not used |
| 0 | 0 | 0 | 1 | Not used |
| 0 | 0 | 1 | 0 | 3 (message boxes 0-2) |
| • | • | • | • | • |
| • | • | • | • | • |
| 1 | 1 | 1 | 0 | 15 (message boxes 0–E) |
| 1 | 1 | 1 | 1 | 16 (message boxes 0–F) |

Unused bit.
Write a "0".

**Figure 2-21　Message Box Count Setting Register (NMES)**

**Notes on how to set the NMES register**
Number of message boxes must use three message boxes. This three message boxes not use group message function.

| Message box number | Message box |
|---|---|
| 0 | |
| 1 | |
| 2 | |

*Not use group message function.

**(a)　When 3 message boxes are used.**

| Message box number | Message box |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | GM0 or GM1 |

*Can use group message function for only one message box

**(b)　When 4 message boxes are used.**

| Message box number | Message box |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | GM0 or GM1 |
| 4 | GM1 or GM0 |

**(c)　When 5 message boxes are used.**

**Figure2-22 Message box setting example.**

2.4.4  CAN Bus Timing Register 0 (BTR0: 1Fhex)

The MSM9225B has an internal baud rate prescaler that generates the BTL (Bit Timing Logic) signal by dividing the system clock by a factor of 1 to 64.  BTL is the system clock for the communication function.
The register BTR0 sets the baud rate prescaler and the SJW width.
Writing to BTR0 is enabled only when the initialization bit INIT of the CAN control register (CANC: 0Ehex) has been set to "1".
The bit configuration is shown below.

MSB | SJWB | SJWA | BRP5 | BRP4 | BRP3 | BRP2 | BRP1 | BRP0 | LSB  BTR0 (1Fhex), *R/W: R/W

Initial value:  0  0  0  0  0  0  0  0

| BRP5 | BRP4 | BRP3 | BRP2 | BRP1 | BRP0 | BTL Cycle time |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 × system clock period |
| 0 | 0 | 0 | 0 | 0 | 1 | 2 × system clock period |
| • | • | • | • | • | • | • |
| • | • | • | • | • | • | • |
| 1 | 1 | 1 | 1 | 1 | 0 | 63 × system clock period |
| 1 | 1 | 1 | 1 | 1 | 1 | 64 × system clock period |

System clock: the clock 1/2 the frequency of the oscillator clock.

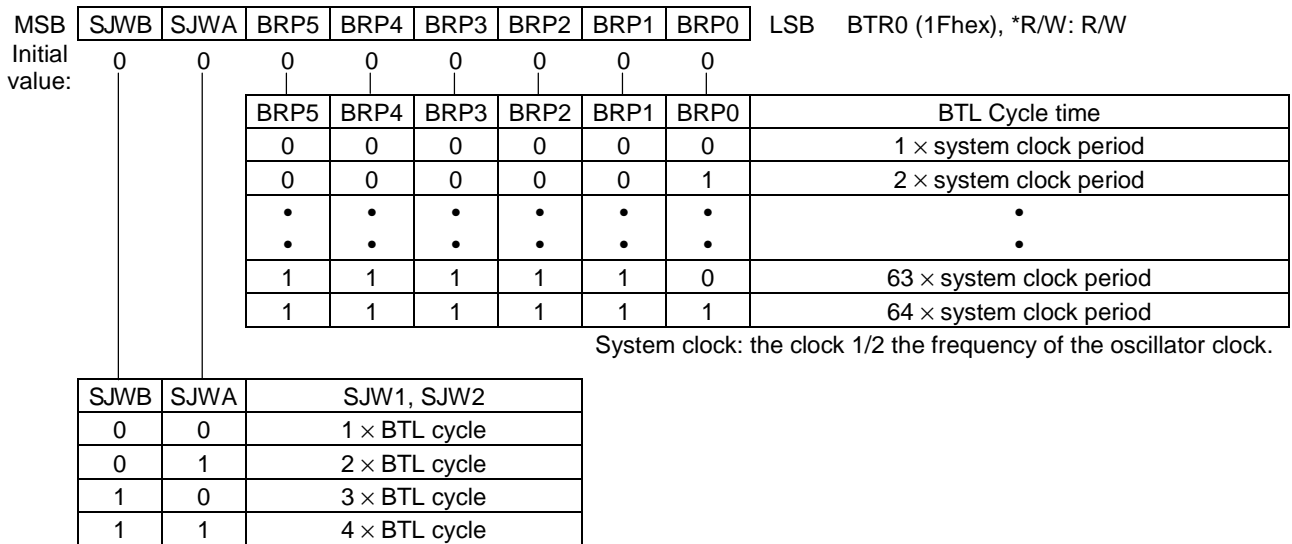| SJWB | SJWA | SJW1, SJW2 |
|---|---|---|
| 0 | 0 | 1 × BTL cycle |
| 0 | 1 | 2 × BTL cycle |
| 1 | 0 | 3 × BTL cycle |
| 1 | 1 | 4 × BTL cycle |

**Figure 2-23  CAN Bus Timing Register 0 (BTR0)**

(1)  Baud rate prescaler: BRP5 to BRP0
This is a 6-bit field to set the BTL cycle time of the basic clock for communication operation.
Table 2-5 shows the relationship between the bit content and BTL.
The BTL cycle time is given by the following equation:

BTL cycle time = $2 \times$ (BRP setting value + 1)/$f_{OSC}$
where $f_{OSC}$ is the oscillation frequency.

At reset, BRP5 to BRP0 are set to "000000".

**Table 2-5  BTL Cycle Time Setting**

| BRP5 | BRP4 | BRP3 | BRP2 | BRP0 | BRP0 | BTL cycle time |
|------|------|------|------|------|------|----------------|
| 0 | 0 | 0 | 0 | 0 | 0 | $1 \times$ System clock period |
| 0 | 0 | 0 | 0 | 0 | 1 | $2 \times$ System clock period |
| • | • | • | • | • | • | • |
| • | • | • | • | • | • | • |
| • | • | • | • | • | • | • |
| 1 | 1 | 1 | 1 | 1 | 0 | $63 \times$ System clock period |
| 1 | 1 | 1 | 1 | 1 | 1 | $64 \times$ System clock period |

System clock: the clock 1/2 the frequency of the oscillation frequency

(2)  SJW: SJWA, SJWB
This is a 2-bit field to set SJW.
Table 2-6 shows the relationship between bit content and SJW.
At reset, SJWA and SJWB are set to "00".

**Table 2-6  Setting of SJW1 and SJW2**

| SJWB | SJWA | SJW1, SJW2 |
|------|------|------------|
| 0 | 0 | $1 \times$ BTL cycle |
| 0 | 1 | $2 \times$ BTL cycle |
| 1 | 0 | $3 \times$ BTL cycle |
| 1 | 1 | $4 \times$ BTL cycle |

### 2.4.5 CAN Bus Timing Register 1 (BTR1: 2Ehex)

This register sets the sampling point used for bus timing.
Writing to the BTR1 bit is enabled, when the INIT bit of the CAN control register (CANC: 0Ehex) is "1".
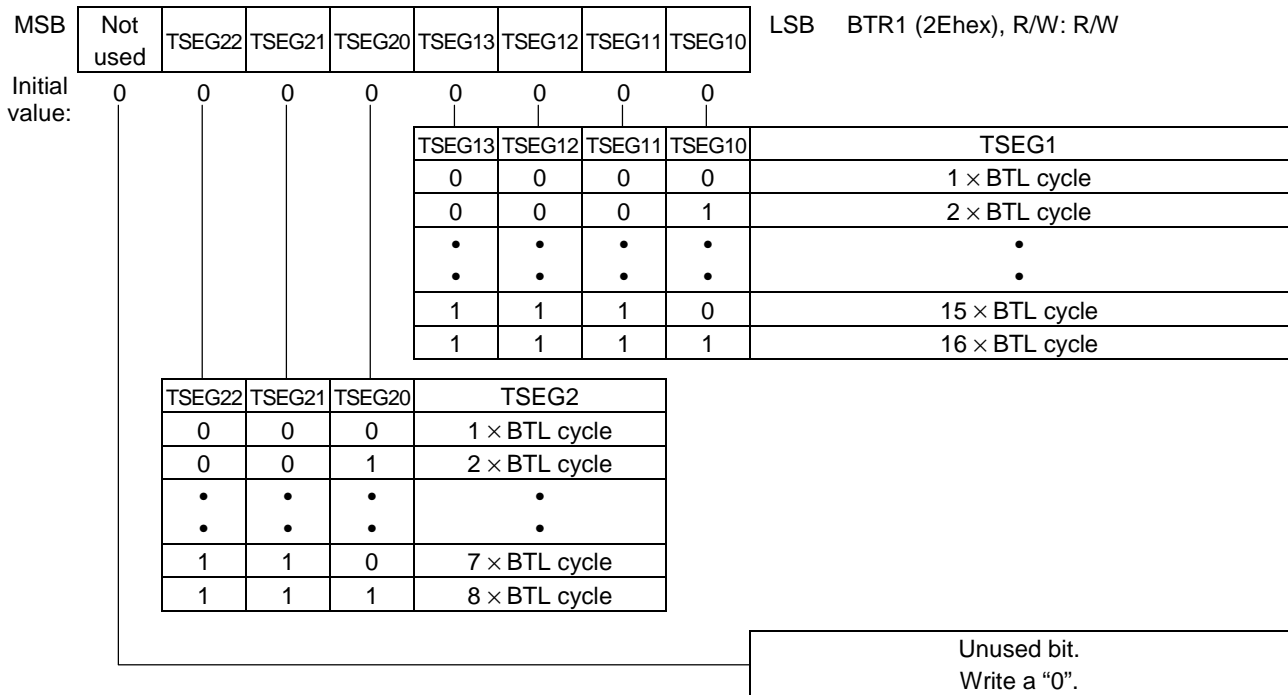The bit configuration is as follows:

| MSB | Not used | TSEG22 | TSEG21 | TSEG20 | TSEG13 | TSEG12 | TSEG11 | TSEG10 | LSB | BTR1 (2Ehex), R/W: R/W |
|---|---|---|---|---|---|---|---|---|---|---|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| TSEG13 | TSEG12 | TSEG11 | TSEG10 | TSEG1 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $1 \times$ BTL cycle |
| 0 | 0 | 0 | 1 | $2 \times$ BTL cycle |
| • | • | • | • | • |
| • | • | • | • | • |
| 1 | 1 | 1 | 0 | $15 \times$ BTL cycle |
| 1 | 1 | 1 | 1 | $16 \times$ BTL cycle |

| TSEG22 | TSEG21 | TSEG20 | TSEG2 |
|---|---|---|---|
| 0 | 0 | 0 | $1 \times$ BTL cycle |
| 0 | 0 | 1 | $2 \times$ BTL cycle |
| • | • | • | • |
| • | • | • | • |
| 1 | 1 | 0 | $7 \times$ BTL cycle |
| 1 | 1 | 1 | $8 \times$ BTL cycle |

Unused bit.
Write a "0".

**Figure 2-24  CAN Bus Timing Register 1 (BTR1)**

(1) Time segment 1: TSEG13 to TSEG10
This is a 4-bit field to set the sampling point.
Table 2-7 shows the relationship between bit content and TSEG1.
At reset, TSEG13 to TSEG10 are set to "0000".

**Table 2-7  TSEG1 Setting**

| TSEG13 | TSEG12 | TSEG11 | TSEG10 | TSEG1 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | $1 \times$ BTL cycle |
| 0 | 0 | 0 | 1 | $2 \times$ BTL cycle |
| • | • | • | • | • |
| • | • | • | • | • |
| • | • | • | • | • |
| 1 | 1 | 1 | 0 | $15 \times$ BTL cycle |
| 1 | 1 | 1 | 1 | $16 \times$ BTL cycle |

(2) Time segment 2: TSEG22 to TSEG20
This is a 3-bit field to set TSEG2.
Table 2-8 shows the relationship between the bit content and TSEG2.
At reset, TSEG22 to TSEG20 are set to "000".

**Table 2-8  TSEG2 Setting**

| TSEG22 | TSEG21 | TSEG20 | TSEG2 |
|:------:|:------:|:------:|:------|
| 0 | 0 | 0 | $1 \times$ BTL cycle |
| 0 | 0 | 1 | $2 \times$ BTL cycle |
| • | • | • | • |
| • | • | • | • |
| • | • | • | • |
| 1 | 1 | 0 | $7 \times$ BTL cycle |
| 1 | 1 | 1 | $8 \times$ BTL cycle |

(3)  Bit timing

Bit timing is set by CAN bus timing registers 0 and 1 (BTR0, 1). Figure 2-25 shows the relationship between 1 bit time of a message and CAN bus timing.
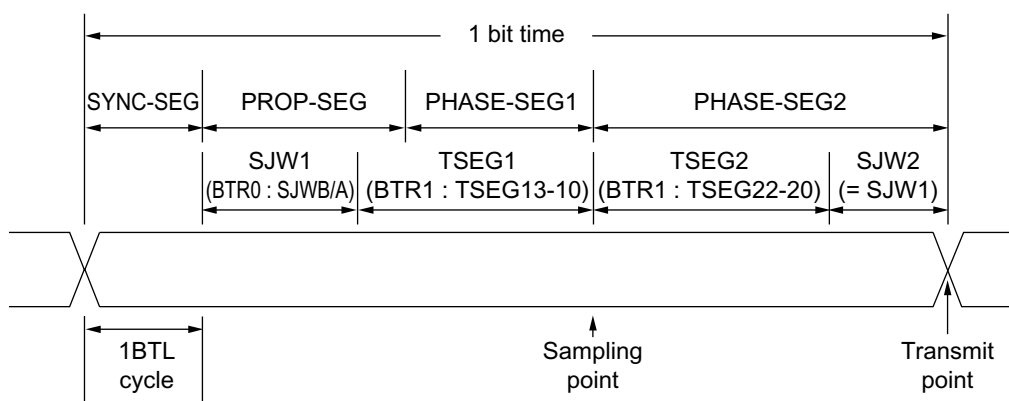


**Figure 2-25  Bit Timing**

Explanation of Terms Used

- Sync segment: The segment in which the falling edge is detected and the bit synchronization is started.
- Prop segment: The segment for compensating for the delay of the output buffer, the CAN bus, and the input buffer.  Set so as to wait until ACK is returned before the start of phase segment 1.

  Period of prop segment time $\geq$ (output buffer delay + CAN bus delay + input buffer delay)
- Phase segment 1, 2: These are the segments for compensating the error in the data bit time.  Although a larger tolerance can be established when these are large, the communication speed becomes slower.
- SJW1, 2: These are the bits for setting the range of bit synchronization.  SJW is the abbreviation for reSynchronization Jump Width.

If setting is :

BTR0 = "01000001"        SJWB = "0", SJWA = "1", BRP5-0 = "000001"
BTR1 = "00000001"        TSEG2 = "000", TSEG1 = "0001"

then the bit timing is as follows:

| | |
|:---|:---|
| Sync segment | 1 BTL cycle (fixed) |
| SJW 1 | 2 BTL cycle |
| TSEG 1 | 2 BTL cycle |
| TSEG 2 | 1 BTL cycle |
| SJW 2 | 2 BTL cycle |
| 1 bit time | 8 BTL cycle |

Sampling point = 5 BTL cycle

If $f_{OSC}$ = 16 MHz, then 1 BTL cycle is :

BTL cycle = $2 \times$ (BRP setting value + 1)/$f_{OSC}$ = $2 \times (1 + 1)$ / 16 MHz = 0.25 µs

Therefore 1 bit time is :

8 BTL cycle = $8 \times 0.25$ µs = 2.0 µs
(= 500 kbps)

(4) Resynchronization
When a negative edge of the CAN bus signal is detected in the period of SJW, the internal bit status is shifted for resynchronization as shown in Figure 2-26.
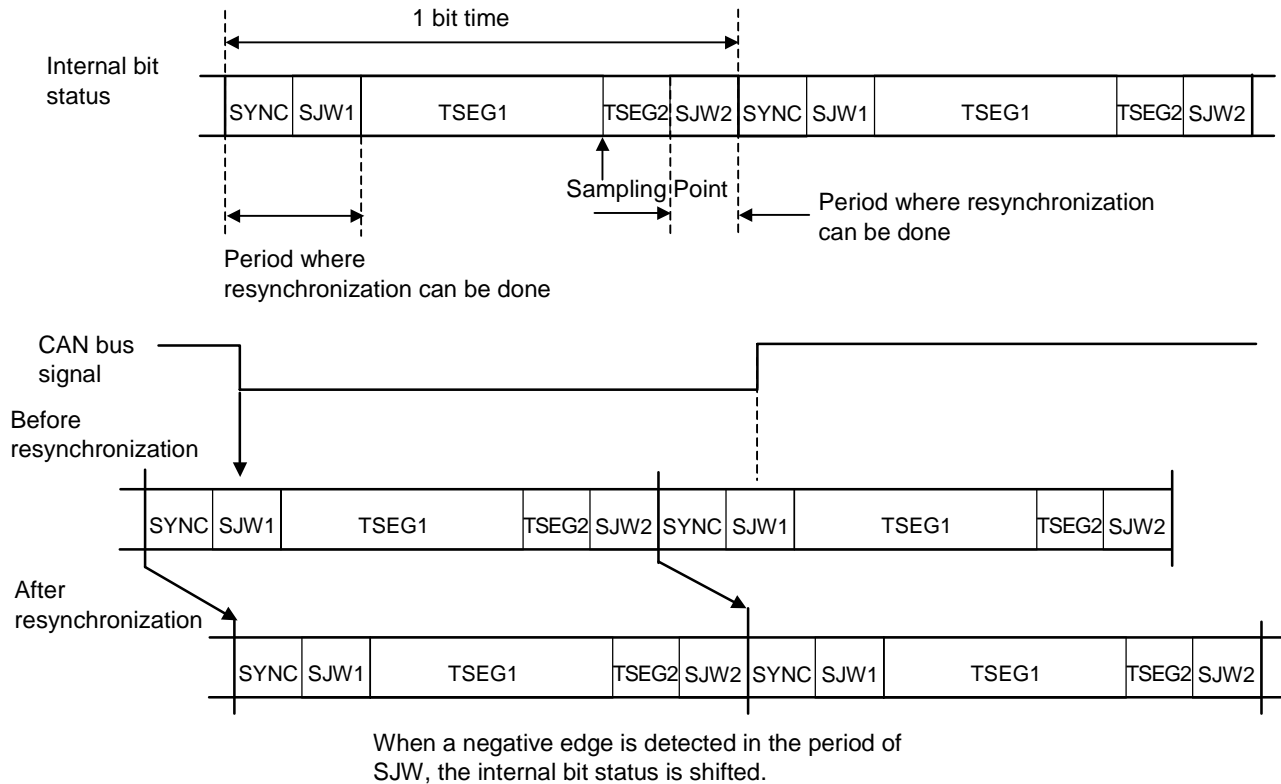


**Figure 2-26  Resynchronization in the period of SJW**

When a negative edge is detected outside the period of SJW, the resynchronization method shown in Figure 2-27 will be done.

(a)    If a negative edge lies before the sampling point
The sampling point is shifted afterward by SJW to resynchronize the internal bit status with the CAN bus signal.
(b)    If a negative edge lies after the sampling point
The period of SJW is deleted to resynchronize the internal bit status with the CAN bus signal.

CAN bus signal

Before resynchronization

SYNC | SJW1 | TSEG1 | TSEG2 | SJW2 | SYNC | SJW1 | TSEG1 | TSEG2 | SJW2

Sampling Point

Sampling Point

After resynchronization

SYNC | SJW1 | TSEG1 | SJW | TSEG2 | SJW2 | SYNC | SJW1 | TSEG1 | TSEG2 | SJW2

Sampling Point

Sampling Point

Sampling Point

Sampling Point

The sampling point is shifted by SJW.

(a) If a negative edge lies before the sampling point

CAN bus signal

Before resynchronization

SYNC | SJW1 | TSEG1 | TSEG2 | SJW2 | SYNC | SJW1 | TSEG1 | TSEG2 | SJW2

Sampling Point

Sampling Point

After resynchronization

SYNC | SJW1 | TSEG1 | TSEG2 | SYNC | SJW1 | TSEG1 | TSEG2 | SJW2

Sampling Point

Sampling Point

Sampling Point

Sampling Point

The period of SJW2 is deleted.

(b) If a negative edge lies after the sampling point

**Figure 2-27  Resynchronization outside the period of SJW**

2.4.6  Communication Input/Output Control Register (TIOC: 2Fhex)

This register sets the input/output mode and output driver format of output pins Tx0 and Tx1.
Writing to the TIOC bit is enabled when the INIT bit of the CAN control register (CANC: 0Ehex) is "1".
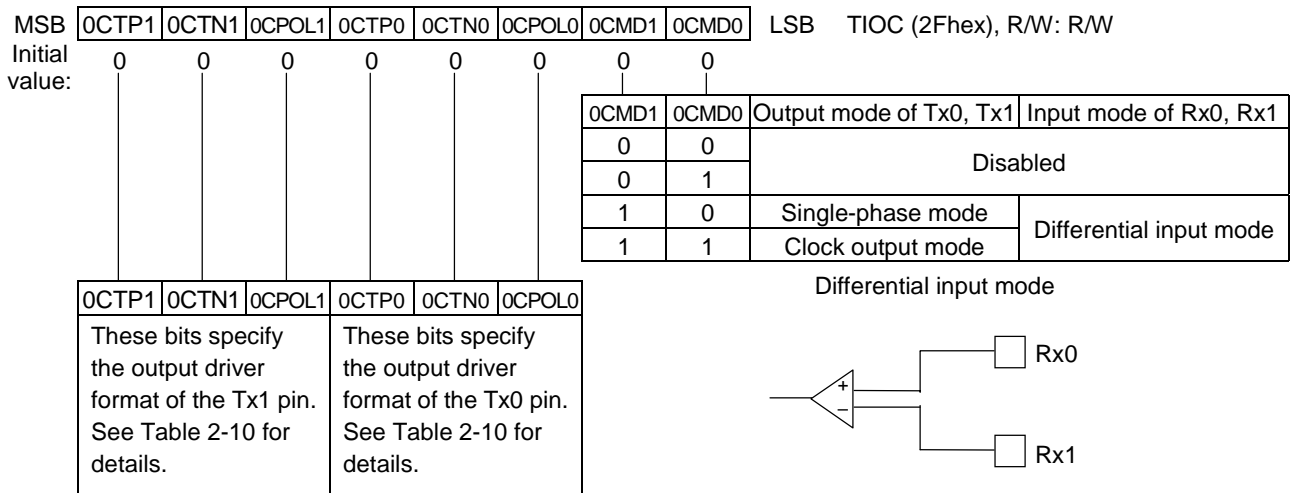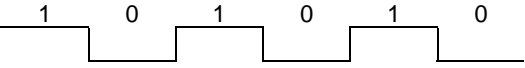The bit configuration is as follows:

MSB | 0CTP1 | 0CTN1 | 0CPOL1 | 0CTP0 | 0CTN0 | 0CPOL0 | 0CMD1 | 0CMD0 | LSB    TIOC (2Fhex), R/W: R/W

Initial value:  0   0   0   0   0   0   0   0

| 0CMD1 | 0CMD0 | Output mode of Tx0, Tx1 | Input mode of Rx0, Rx1 |
|---|---|---|---|
| 0 | 0 | Disabled | |
| 0 | 1 | | |
| 1 | 0 | Single-phase mode | Differential input mode |
| 1 | 1 | Clock output mode | |

| 0CTP1 | 0CTN1 | 0CPOL1 | 0CTP0 | 0CTN0 | 0CPOL0 |
|---|---|---|---|---|---|
| These bits specify the output driver format of the Tx1 pin. See Table 2-10 for details. | | | These bits specify the output driver format of the Tx0 pin. See Table 2-10 for details. | | |

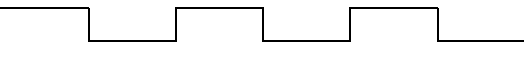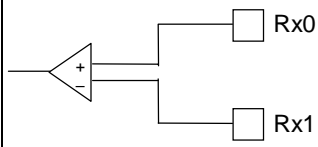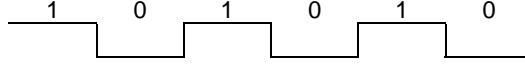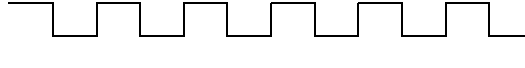Differential input mode

Rx0

Rx1

**Figure 2-28  Communication Input/Output Control Register (TIOC)**

(1)  Input/output mode setting: OCMD1 to OCMD0
These bits are used to set the output mode of output pins Tx0 and Tx1 and the input mode of input pins Rx0 and Rx1.
Table 2-9 shows the relationship between the bit content and input/output mode.
At reset, OCMD1 to OCMD0 are set to "00".

**Table 2-9  Input/output Mode Setting**

| OCMD1 | OCMD0 | Output mode of Tx0 and Tx1 | Input mode of Rx0, Rx1 |
|:-----:|:-----:|---|---|
| 0 | 0 | [Disabled] | |
| 0 | 1 | | |
| 1 | 0 | [Single-phase mode]<br>Same bit string data is output from both Tx0 and Tx1.<br>  Output example<br><br>Data   1   0   1   0   1   0<br>Tx0<br>Tx1 | [differential input mode]<br><br>Rx0<br>Rx1 |
| 1 | 1 | [Clock output mode]<br>Bit string data is output from Tx0.<br>Synchronization clock is output from Tx1.<br>  Output example<br><br>Data   1   0   1   0   1   0<br>Tx0<br>Tx1 | |

(2)  Output driver format setting: OCPOL, OCTN, OCTP
OCPOL is used to set the polarity of output.
OCTN is used to set the open drain mode of the Nch transistor of the output driver.
OCTP is used to set the open drain mode of the Pch transistor of the output driver.
Figure 2-29 shows the circuit configuration of the output driver, and Table 2-10 the relationship between bit content and output driver format.
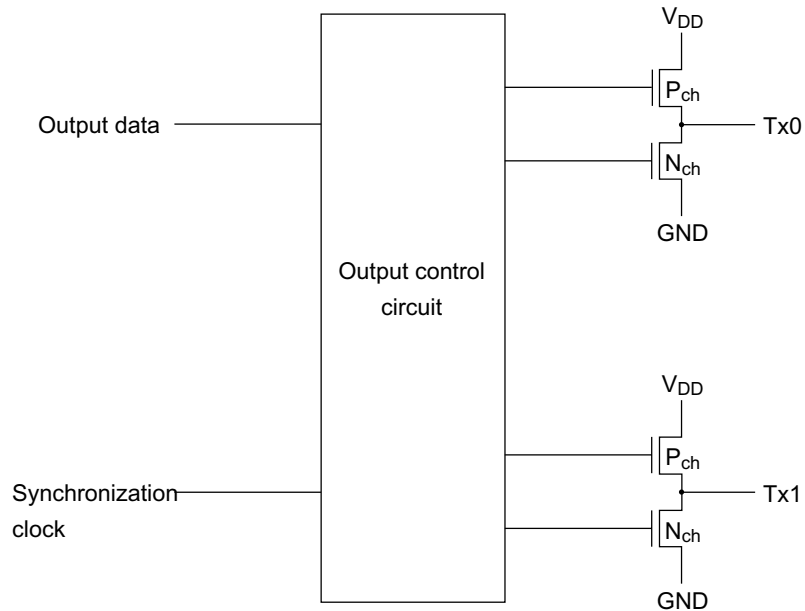


**Figure 2-29  Circuit Configuration of Output Driver**

**Table 2-10  Output Driver Format Setting**

| Mode | OCTPn* | OCTNn* | OCPOLn* | Output data | Pch Tr | Nch Tr | Txn* pin output level |
|---|---|---|---|---|---|---|---|
| Floating | 0 | 0 | 0 | 0 | off | off | Floating |
| | | | | 1 | off | off | Floating |
| | | | 1 | 0 | off | off | Floating |
| | | | | 1 | off | off | Floating |
| Pull-down | | 1 | 0 | 0 | off | on | "0" |
| | | | | 1 | off | off | Floating |
| | | | 1 | 0 | off | off | Floating |
| | | | | 1 | off | on | "0" |
| Pull-up | 1 | 0 | 0 | 0 | off | off | Floating |
| | | | | 1 | on | off | "1" |
| | | | 1 | 0 | on | off | "1" |
| | | | | 1 | off | off | Floating |
| Push-pull | | 1 | 0 | 0 | off | on | "0" |
| | | | | 1 | on | off | "1" |
| | | | 1 | 0 | on | off | "1" |
| | | | | 1 | off | on | "0" |

* n = 0 or 1

### 2.4.7 Group Message Register (GMR0: 3Ehex, GMR1: 3Fhex)

These are registers to set the group message function.

Group Message Function

If the group message function is used, a part of an identifier can be masked. This can increase the number of receivable identifiers.

To use the group message function, set the message box number of the target message box to set the group message function in the GMR register. Then set the bits to be masked in the GMSK register.

If a received message corresponds to both a message box for which the group message function is specified and a message box for which that is not specified, the message will be received in the message box for which the group message function is not specified.

If the identifier of received message fits with the identifier of both group messages, the received message will be written to the message box whose identifier (unmasked identifier set in the identifier field) has the high priority.

The group message function can be set for two message boxes.
The group message function is valid when the EGM0/EGM1 bit is "1".
Using GMR03 to GMR00 and GMR13 to GMR10, set the message box numbers of the message boxes for which the group message function is to be set.
The numbers of the message boxes for which the group message function is to be set must be specified in the descending order of box numbers in the range of message box numbers that correspond to the message box count predetermined by the message box count setting register (NMES: 1Ehex).
If any message box numbers outside that range are specified, all bits in GMR0 and GMR1 are set to "0".
At reset, all bits are set to "0". (See Figure 2-30.)
The bit configuration is as follows:

| MSB | | | | | | | LSB | |
|---|---|---|---|---|---|---|---|---|
| EGM0 | Not used | Not used | Not used | GMR03 | GMR02 | GMR01 | GMR00 | GMR0 (3Ehex), R/W: R/W |
| Initial value: 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| MSB | | | | | | | LSB | |
|---|---|---|---|---|---|---|---|---|
| EGM1 | Not used | Not used | Not used | GMR13 | GMR12 | GMR11 | GMR10 | GMR1 (3Fhex), R/W: R/W |
| Initial value: 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Write a "0" to the unused bits.

**Figure 2-30  Group Message Register (GMR0, GMR1)**

### 2.4.8  Group Message Mask Register (GMSK)

These are registers to mask the identifier of the message box specified by the group message registers GMR0 and GMR1.
Using MnID28 to MnID0 (n = 0, 1), set the bits to mask the identifier.  Setting "1" masks the bit, and setting "0" does not mask the bit.
29 bits M0ID28 to M0ID0 and M1ID28 to M1ID0 are used for the extended format setting.
11 bits M0ID28 to M0ID18 and M1ID28 to M1ID18 are used for the standard format setting.
At reset, all bits are set to "0".
The bit configuration is as follows:

| MSB | M0ID28 | M0ID27 | M0ID26 | M0ID25 | M0ID24 | M0ID23 | M0ID22 | M0ID21 | LSB | GMSK00 (4Ehex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| MSB | M0ID20 | M0ID19 | M0ID18 | M0ID17 | M0ID16 | M0ID15 | M0ID14 | M0ID13 | LSB | GMSK01 (4Fhex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| MSB | M0ID12 | M0ID11 | M0ID10 | M0ID9 | M0ID8 | M0ID7 | M0ID6 | M0ID5 | LSB | GMSK02 (5Ehex), R/W: R/W |
|-----|--------|--------|--------|-------|-------|-------|-------|-------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| MSB | M0ID4 | M0ID3 | M0ID2 | M0ID1 | M0ID0 | Not used | Not used | Not used | LSB | GMSK03 (5Fhex), R/W: R/W |
|-----|-------|-------|-------|-------|-------|----------|----------|----------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| MSB | M1ID28 | M1ID27 | M1ID26 | M1ID25 | M1ID24 | M1ID23 | M1ID22 | M1ID21 | LSB | GMSK10 (6Ehex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| MSB | M1ID20 | M1ID19 | M1ID18 | M1ID17 | M1ID16 | M1ID15 | M1ID14 | M1ID13 | LSB | GMSK11 (6Fhex), R/W: R/W |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| MSB | M1ID12 | M1ID11 | M1ID10 | M1ID9 | M1ID8 | M1ID7 | M1ID6 | M1ID5 | LSB | GMSK12 (7Ehex), R/W: R/W |
|-----|--------|--------|--------|-------|-------|-------|-------|-------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| MSB | M1ID4 | M1ID3 | M1ID2 | M1ID1 | M1ID0 | Not used | Not used | Not used | LSB | GMSK13 (7Fhex), R/W: R/W |
|-----|-------|-------|-------|-------|-------|----------|----------|----------|-----|---------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

Write a "0" to the unused bits.

**Figure 2-31  Group Message Mask Register (GMSK)**

**Notes on how to set the Group Message function**

When setting the Group Message (GM) function in the message box that selects the extended format as a frame format, specify the message box numbers consecutively, beginning with the message box with the largest message box number.

Figure 2-32 (a) shows an example of how to set the GM function in two message boxes when 16 message boxes are used. First, set NMES0 to NMES3 of the message box count setting register (NMES: 1Ehex) to "1111". Next procedure is to set the GM function in the message boxes F and E. To do so, set GMR03 to GMR00 of the GMR0 (group message register) and GMR13 to GMR10 of the GMR1 to "1111" (FH) (message box number) and "1110" (EH) (message box number), respectively. In this case, it is also possible to set GMR03 to GMR00 to "E" (message box number) and GMR13 to GMR10 to "F" (message box number).

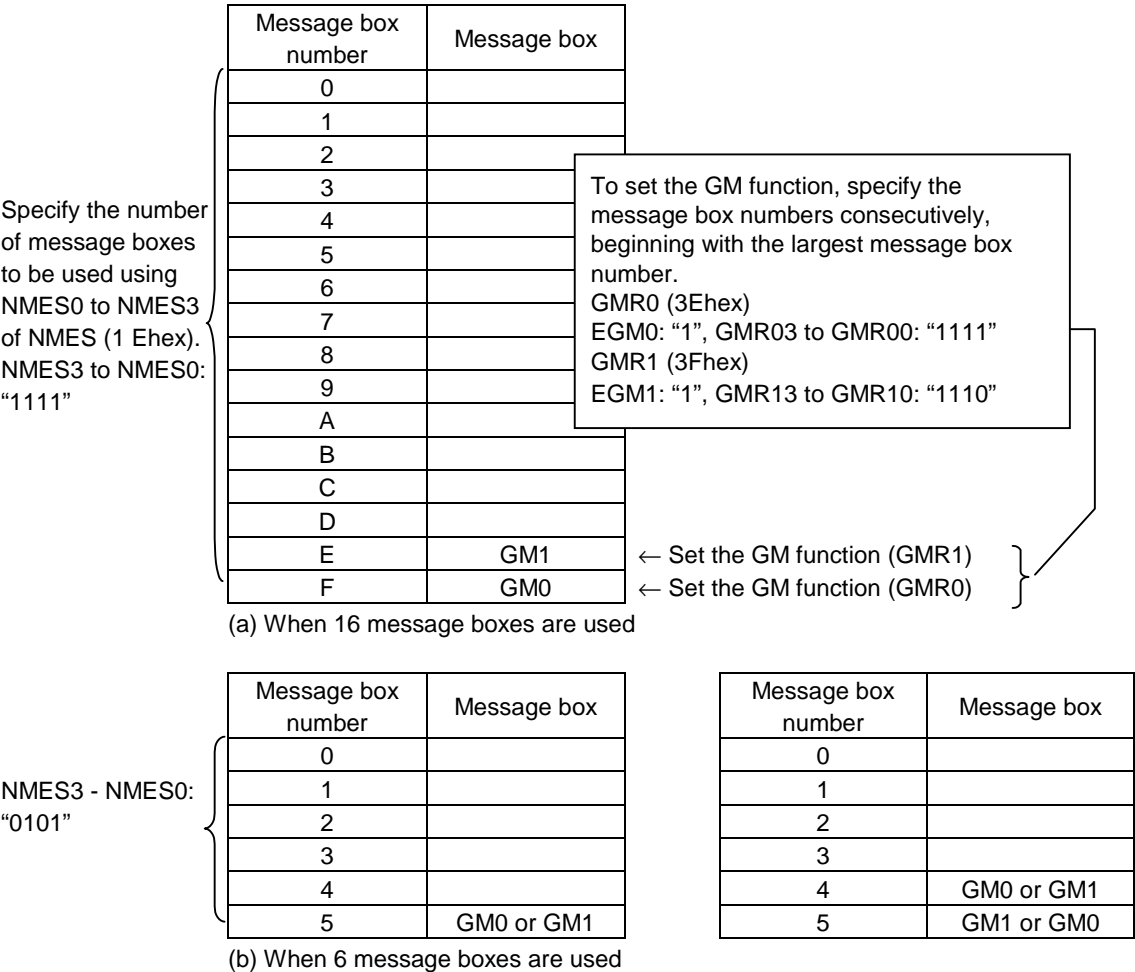Figure 2-32 (b) shows an example of how to set the GM function when 6 message boxes are used.

| Message box number | Message box |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |
| 7 | |
| 8 | |
| 9 | |
| A | |
| B | |
| C | |
| D | |
| E | GM1 |
| F | GM0 |

Specify the number of message boxes to be used using NMES0 to NMES3 of NMES (1 Ehex). NMES3 to NMES0: "1111"

To set the GM function, specify the message box numbers consecutively, beginning with the largest message box number.
GMR0 (3Ehex)
EGM0: "1", GMR03 to GMR00: "1111"
GMR1 (3Fhex)
EGM1: "1", GMR13 to GMR10: "1110"

← Set the GM function (GMR1)
← Set the GM function (GMR0)

(a) When 16 message boxes are used

NMES3 - NMES0: "0101"

| Message box number | Message box |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | GM0 or GM1 |

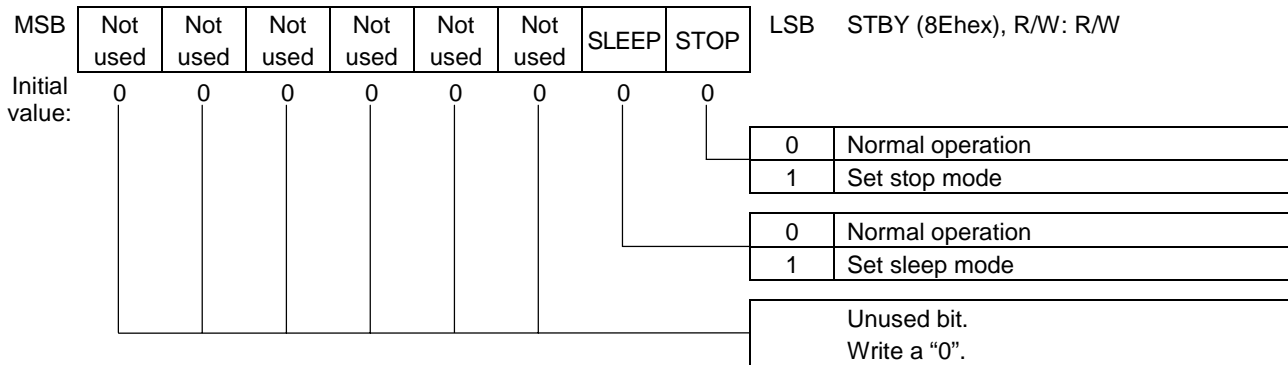| Message box number | Message box |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | GM0 or GM1 |
| 5 | GM1 or GM0 |

(b) When 6 message boxes are used

**Figure 2-32  Example of how to set the GM function**

### 2.4.9  Standby Control Register (STBY: 8Ehex)

This register is used for setting the stop mode and the sleep mode.
A "0" is read after the stop or sleep mode is terminated.
The bit configuration is as follows:

| MSB | Not used | Not used | Not used | Not used | Not used | Not used | SLEEP | STOP | LSB | STBY (8Ehex), R/W: R/W |
|-----|----------|----------|----------|----------|----------|----------|-------|------|-----|------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| | |
|---|---|
| 0 | Normal operation |
| 1 | Set stop mode |

| | |
|---|---|
| 0 | Normal operation |
| 1 | Set sleep mode |

| |
|---|
| Unused bit. Write a "0". |

Note:  A "0" is read out after the sleep mode or the stop mode is terminated.

**Figure 2-33  Standby Control Register (STBY)**

(1)  Stop mode: STOP
   If STOP is set to "1", the MSM9225B will enter the stop mode when the CAN bus is idle.
   In stop mode, the contents of the message memory and control registers are held but the oscillator and all circuits stop to save power consumption. Access to/from external units is therefore disabled.
   Stop mode is terminated by a reset signal input from the $\overline{\text{RESET}}$ pin or a "L" level input to the $\overline{\text{CS}}$ pin.
   At reset, STOP is set to "0".

(2)  Sleep mode: SLEEP
   If SLEEP is set to "1", the MSM9225B will enter the sleep mode when the CAN bus is idle.
   In sleep mode, the contents of the message memory and control registers are held and the differential input of Rx0 and Rx1 operates, but the oscillator and other circuits stop operation. Access to/from external units is therefore disabled.
   Sleep mode is terminated by a reset signal input from the $\overline{\text{RESET}}$ pin or a "L" level input to the $\overline{\text{CS}}$ pin, or by the differential input of Rx0 and Rx1.
   When both stop mode and sleep mode are set at the same time, the MSM9225B enters stop mode.
   At reset, SLEEP is set to "0".

### 2.4.10 CAN Control Register 2 (CANC2: 8Fhex)

This is a register to control bus off release and error counter operation.
The bit configuration is shown below.
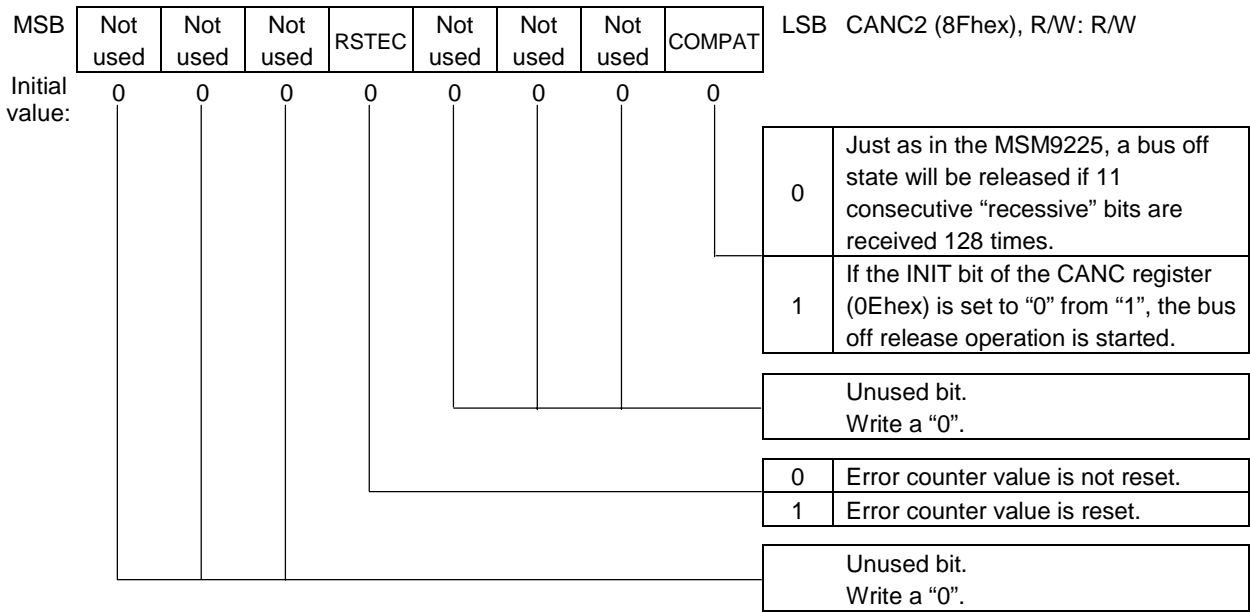At reset, this register is set to "0000 0000".



**Figure 2-34  CAN Control Register 2 (CANC2)**

(1)  Bus off release start timing: COMPAT

This bit specifies bus off release start operation.

When this bit is "0", the bus off state will be released if 11 consecutive "recessive" bits have been received 128 times since the time immediately after the bus off state was entered.  This is the same operation as the MSM9225.

When this bit is "1", the bus off state will be released if 11 consecutive "recessive" bits have been received 128 times since the point of time that the INIT bit of the CAN control register (CANC: 0Ehex) was set to "1", then set to "0".

(2)  Error counter reset: RSTEC

This bit specifies whether to reset the error counters (both transmit error counter and receive error counter).

Setting this bit to "0" does not reset the error counters.

The error counters will be reset if the RSTEC bit is set to "1" when the COMPAT bit is "1" and the INIT bit of CANC is "1".  Set the RSTEC bit to "0" after setting it to "1".

When the MSM9225B is in the bus off state, this operation is invalid.  (Even if the above operation is done, the error counters are not cleared and the bus off state also is not released.)

2.4.11  Communication Message Box Number Register (TMN: 9Ehex)

The message box number when a message is transmitted/received is stored in this register.
The bit configuration is as follows:

| MSB | Not used | Not used | Not used | Not used | TRSN3 | TRSN2 | TRSN1 | TRSN0 | LSB | TMN (9Ehex), R/W: R |
|---|---|---|---|---|---|---|---|---|---|---|
| Initial value: | Undefined | Undefined | Undefined | Undefined | 0 | 0 | 0 | 0 | | |

| TRSN3 | TRSN2 | TRSN1 | TRSN0 | Message box number |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| • | • | • | • | • |
| • | • | • | • | • |
| 1 | 1 | 1 | 0 | E |
| 1 | 1 | 1 | 1 | F |

Unused bit.
When it is read, the value is undefined.

**Figure 2-35  Communication Message Box Number Register (TMN)**

(1)  Communication message box number: TRSN3 to TRSN0
The message box number when a message is transmitted/received is stored.
When transmission completes, the transmitted message box number is stored.  When receiving completes, the received message box number is stored.  And when an error occurs, the message box number of the message box being transmitted/received at that time is stored.
This is a read-only register and is set to "0000" at reset.

2.4.12  CAN Status Register (CANS: 9Fhex)

This is a register to indicate the error status of the MSM9225B.
Bit 6 to bit 4 are flags for the transmitter and bit 1 and bit 0 are for the receiver, and this register is read-only.
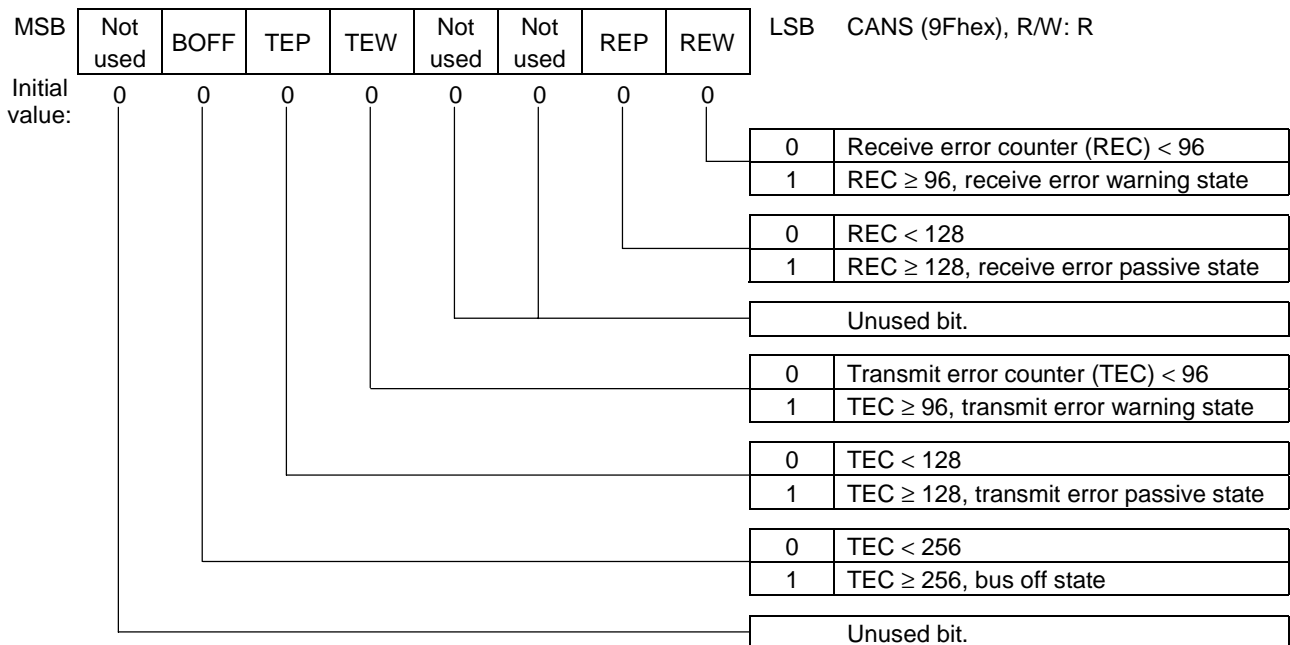The bit configuration is shown below.



| MSB | Not used | BOFF | TEP | TEW | Not used | Not used | REP | REW | LSB | CANS (9Fhex), R/W: R |

| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 0 | Receive error counter (REC) < 96 |
| 1 | REC $\geq$ 96, receive error warning state |

| 0 | REC < 128 |
| 1 | REC $\geq$ 128, receive error passive state |

| Unused bit. |

| 0 | Transmit error counter (TEC) < 96 |
| 1 | TEC $\geq$ 96, transmit error warning state |

| 0 | TEC < 128 |
| 1 | TEC $\geq$ 128, transmit error passive state |

| 0 | TEC < 256 |
| 1 | TEC $\geq$ 256, bus off state |

| Unused bit. |

**Figure 2-36  CAN Status Register (CANS)**

(1)  Receive Error Warning: REW
When the Receive Error Counter (REC) $\geq$ 96, REW becomes "1".  If REW = "1", it is probable that the bus has been damaged.  Testing the bus for this condition is recommended.
At reset or when REC < 96, REW becomes "0".

(2)  Receive Error Passive: REP
When the Receive Error Counter (REC) $\geq$ 128, REP becomes "1" (error passive).
At reset or when REC < 128, REP becomes "0" (error active).

(3)  Transmit Error Warning: TEW
When the Transmit Error Counter (TEC) $\geq$ 96, TEW becomes "1".
If TEW = "1", it is probable that the bus has been damaged.  Testing the bus for this condition is recommended.
At reset or when TEC < 96, TEW becomes "0".

(4)  Transmit Error Passive: TEP
When the Transmit Error Counter (TEC) $\geq$ 128, TEP becomes "1" (error passive).
At reset or when TEC < 128, TEP becomes "0" (error active).

(5)  Bus Off: BOFF
This flag indicates the CAN bus status.
When the Transmit Error Counter (TEC) $\geq$ 256, BOFF becomes "1" and the CAN bus is in the bus off state.
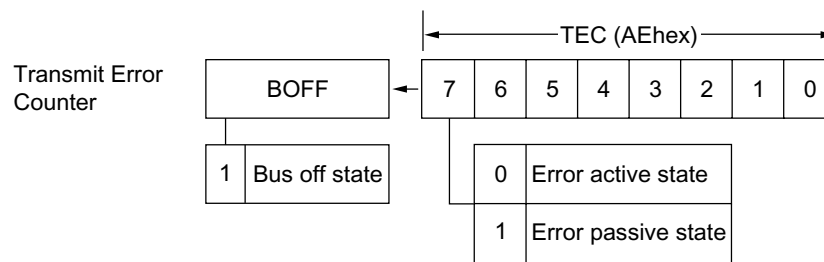At reset or when TEC < 256, BOFF becomes "0".

2.4.13  Transmit Error Counter (TEC: AEhex)

TEC is a register to indicate the Transmit Error Counter value.  This register is read-only.
At reset or when in the bus off state, TEC is set to "0000 0000".
The bit configuration is shown below.

| MSB | TEC7 | TEC6 | TEC5 | TEC4 | TEC3 | TEC2 | TEC1 | TEC0 | LSB | TEC (AEhex), R/W: R |
|---|---|---|---|---|---|---|---|---|---|---|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Figure 2-37  Transmit Error Counter (TEC)**

Figure 2-38 shows the relation between the Transmit Error Counter and the bus off flag (BOFF).
Bit 8 of the transmit error counter, which consists of 9 bits, indicates the BOFF value of the CAN status register (CANS: 9Fhex); bit 7 indicates the TEP value of CANS.



CANS (9Fhex): bit 6 = BOFF, bit 5 = TEP

**Figure 2-38  Relation between the Transmit Error Counter and the Bus OFF flag**

2.4.14  Receive Error Counter (REC: AFhex)

REC is a register to indicate the Receive Error Counter value.  This register is read-only.
At reset or when in the bus off state, REC is set to "0000 0000".

Bit 7 of the Receive Error Counter indicates the value of REP bit of the CAN status register (CANS: 9Fhex).
The bit configuration is shown below.

| MSB | REC7 | REC6 | REC5 | REC4 | REC3 | REC2 | REC1 | REC0 | LSB | REC (AFhex), R/W: R |
|---|---|---|---|---|---|---|---|---|---|---|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Figure 2-39  Receive Error Counter (REC)**

2.4.15  CAN Status Register 2 (CANS2: BEhex)

This is a register to indicate the error contents for when an error occurs.
If an error occurs, the corresponding flag is set to "1".  It is set to "0" when "0" is written to it from the microcontroller.
Once a flag of this register is set to "1", it will not be reset to "0" unless "0" is written to it.  Therefore, when a corresponding error flag is set to "1", and after that if another error occurs when "0" is not written from the microcontroller, it results in two error flags, the previous one and the current one, being set to "1".
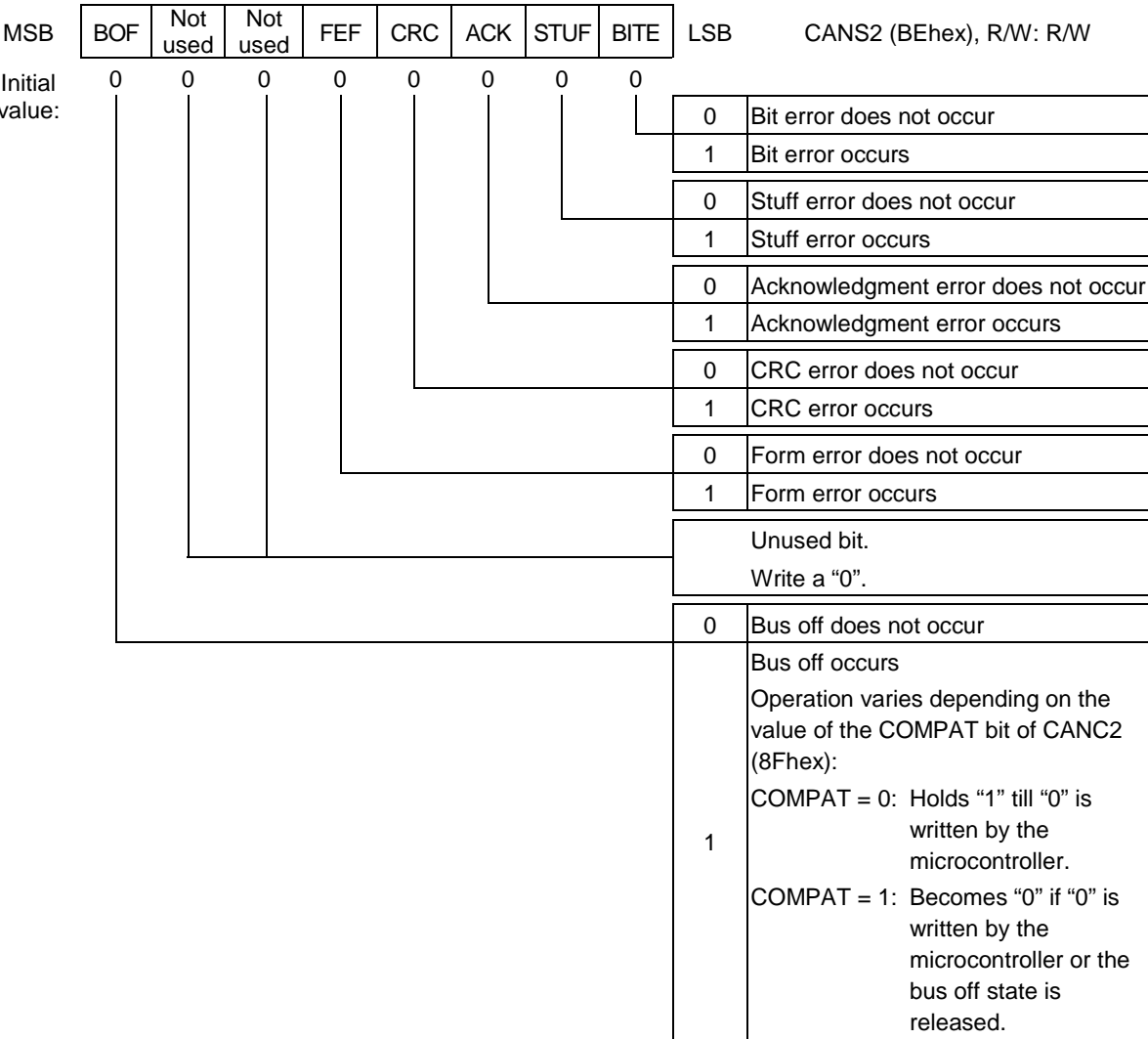The bit configuration is shown below.

| MSB | BOF | Not used | Not used | FEF | CRC | ACK | STUF | BITE | LSB | CANS2 (BEhex), R/W: R/W |
|-----|-----|----------|----------|-----|-----|-----|------|------|-----|-------------------------|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

| | |
|---|---|
| 0 | Bit error does not occur |
| 1 | Bit error occurs |
| 0 | Stuff error does not occur |
| 1 | Stuff error occurs |
| 0 | Acknowledgment error does not occur |
| 1 | Acknowledgment error occurs |
| 0 | CRC error does not occur |
| 1 | CRC error occurs |
| 0 | Form error does not occur |
| 1 | Form error occurs |
| | Unused bit. Write a "0". |
| 0 | Bus off does not occur |
| 1 | Bus off occurs. Operation varies depending on the value of the COMPAT bit of CANC2 (8Fhex): COMPAT = 0: Holds "1" till "0" is written by the microcontroller. COMPAT = 1: Becomes "0" if "0" is written by the microcontroller or the bus off state is released. |

**Figure 2-40  CAN Status Register 2 (CANS2)**

(1)  Bit error flag: BITE
This bit becomes "1" when a bit error occurs.
At reset or after release of the bus off state, this bit becomes "0".

(2)  Stuff error flag: STUF
This bit becomes "1" when a stuff error occurs.
At reset or after release of the bus off state, this bit becomes "0".

(3) Acknowledgment error flag: ACK
This bit becomes "1" when an acknowledgment error occurs.
At reset or after release of the bus off state, this bit becomes "0".

(4) CRC error flag: CRC
This bit becomes "1" when a CRC error occurs.
At reset or after release of the bus off state, this bit becomes "0".

(5) Form error flag: FEF
This bit becomes "1" when a form error occurs.
At reset or after release of the bus off state, this bit becomes "0".

(6) Bus off flag: BOF
This bit becomes "1" when a bus off error occurs.
It is set to "0" when "0" is written from the microcontroller. Also, when the COMPAT bit of the CAN control register 2 (CANC2: 8Fhex) is "1", BOF is automatically set to "0" if the bus off state is released (that is, if 11 consecutive "recessive" bits are received).
A bus off flag (BOFF: bit 6) is provided in the CAN status register (CANS: 9Fhex). This flag is assigned to bit 8 of the transmit error counter. Therefore, when the transmit error counter is set to "0_0000_0000" because of bus off release, BOFF is also set to "0". For this reason, if the microcontroller does not access the CANS register during the period from the time the bus off state is entered to the time it is released, the microcontroller cannot detect whether the CAN controller has entered a bus off state.
In the case of BOF, the CANS2 bus off flag, it holds "1" until "0" is written to it from the microcontroller (when COMPAT = "0") or the bus off release operation does not start before the initialization bit INIT of the CAN control register (CANC: 0Ehex) is manipulated. Therefore, the microcontroller can detect whether the CAN controller is (or was) in the bus off state.
At reset, BOF is set to "0".

## 2.4.16 Bus Off Release Counter (BOCO: BFhex)

BOCO is a read-only register for checking the bus off release.
This counter is incremented by 1 every time 11 consecutive "recessive" bits are detected. If the counter has been incremented up to a maximum of 128, the bus off state will be released.
If any change is found in the value of this register after the bus off state is entered, it can be confirmed that the CAN bus is not fixed as "dominant" due to some effect.
At reset or after the bus off state is released, BOCO is set to "0".

| MSB | BOC7 | BOC6 | BOC5 | BOC4 | BOC3 | BOC2 | BOC1 | BOC0 | LSB | BOCO (BFhex), R/W: R |
|---|---|---|---|---|---|---|---|---|---|---|
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

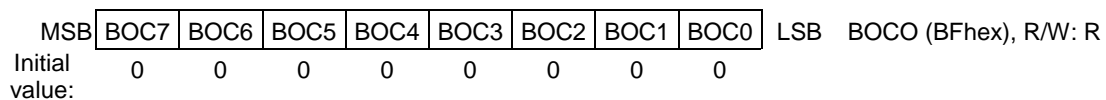**Figure 2-41  Bus Off Release Counter (BOCO)**

*Chapter 3*

# Operational Description

# Chapter 3    Operational Description

MSM9225B operation is described below.

## 3.1    Operational Procedure

Procedures to set and operate various communication protocols are indicated below.

### 3.1.1    Initial Setting
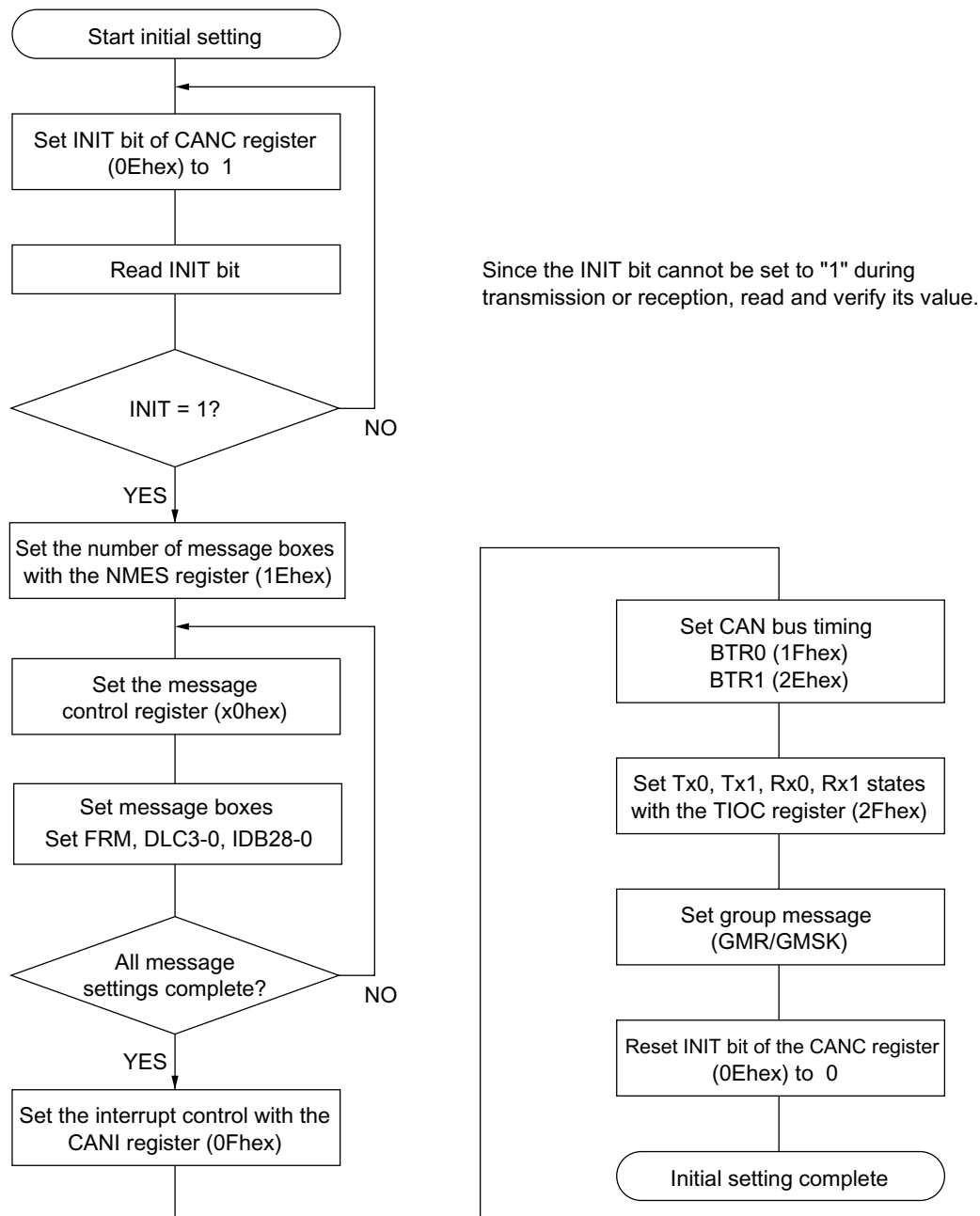
Figure 3-1 shows the initial setting procedure.



**Figure 3-1  Initial Setting Flowchart**

### 3.1.2 Transmit Procedure
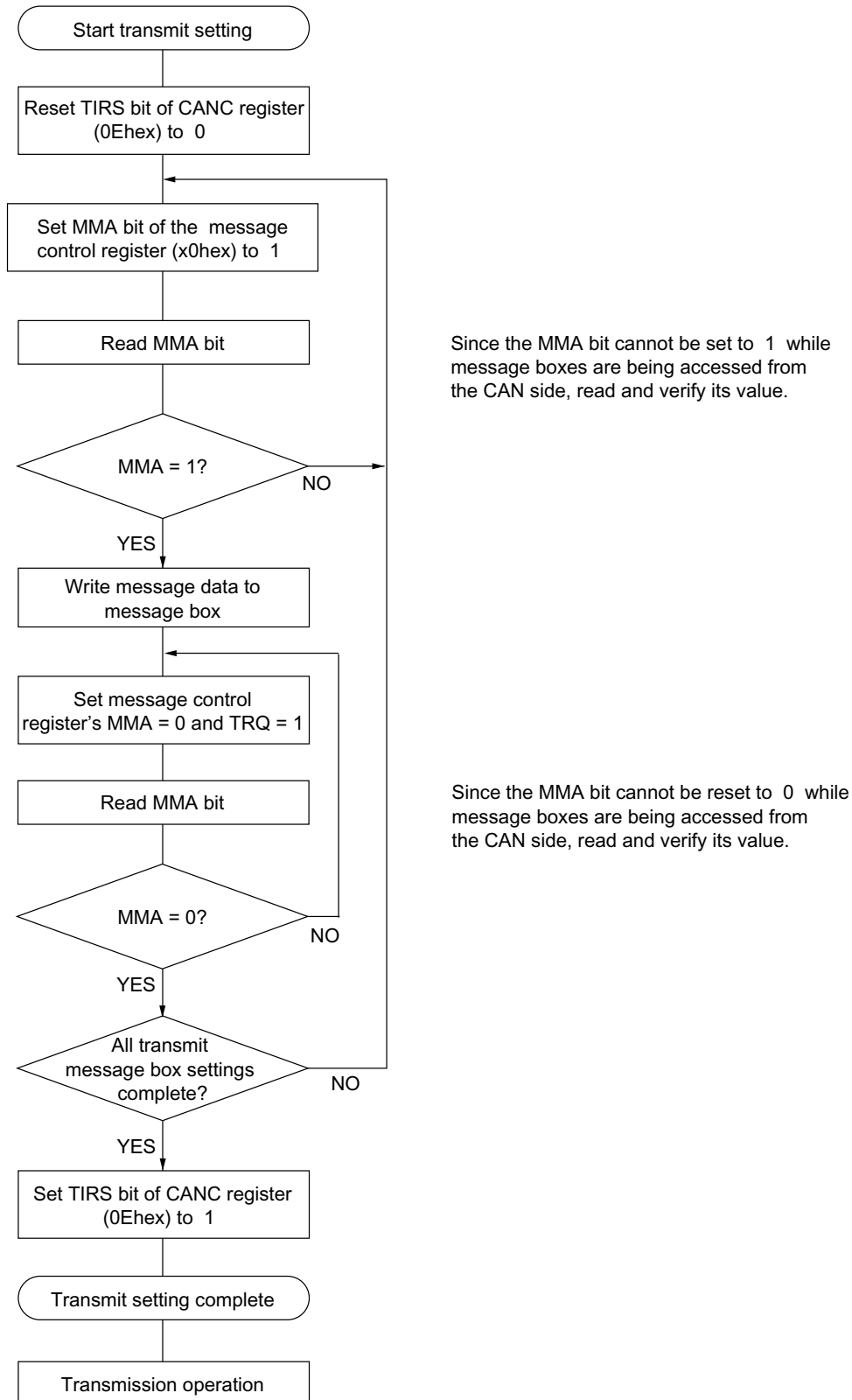
Figure 3-2 shows the transmit procedure.

```
                    ( Start transmit setting )
                              │
          ┌───────────────────────────────────┐
          │ Reset TIRS bit of CANC register   │
          │        (0Ehex) to  0              │
          └───────────────────────────────────┘
                              │
          ┌───────────────────────────────────┐
          │ Set MMA bit of the  message       │
          │ control register (x0hex) to  1    │
          └───────────────────────────────────┘
                              │
          ┌───────────────────────────────────┐            Since the MMA bit cannot be set to  1  while
          │         Read MMA bit              │            message boxes are being accessed from
          └───────────────────────────────────┘            the CAN side, read and verify its value.
                              │
                         < MMA = 1? >──────NO──────┐
                              │ YES                 │
          ┌───────────────────────────────────┐    │
          │     Write message data to         │    │
          │         message box               │    │
          └───────────────────────────────────┘    │
                              │                     │
          ┌───────────────────────────────────┐    │
          │     Set message control           │    │
          │ register's MMA = 0 and TRQ = 1    │    │
          └───────────────────────────────────┘    │
                              │                     │
          ┌───────────────────────────────────┐            Since the MMA bit cannot be reset to  0  while
          │         Read MMA bit              │            message boxes are being accessed from
          └───────────────────────────────────┘            the CAN side, read and verify its value.
                              │                     │
                         < MMA = 0? >──────NO       │
                              │ YES                 │
                         < All transmit             │
                       message box settings >──NO───┘
                           complete?
                              │ YES
          ┌───────────────────────────────────┐
          │ Set TIRS bit of CANC register     │
          │        (0Ehex) to  1              │
          └───────────────────────────────────┘
                              │
                    ( Transmit setting complete )
                              │
          ┌───────────────────────────────────┐
          │     Transmission operation        │
          └───────────────────────────────────┘
```

**Figure 3-2  Transmit Flowchart**

### 3.1.3 Receive Procedure

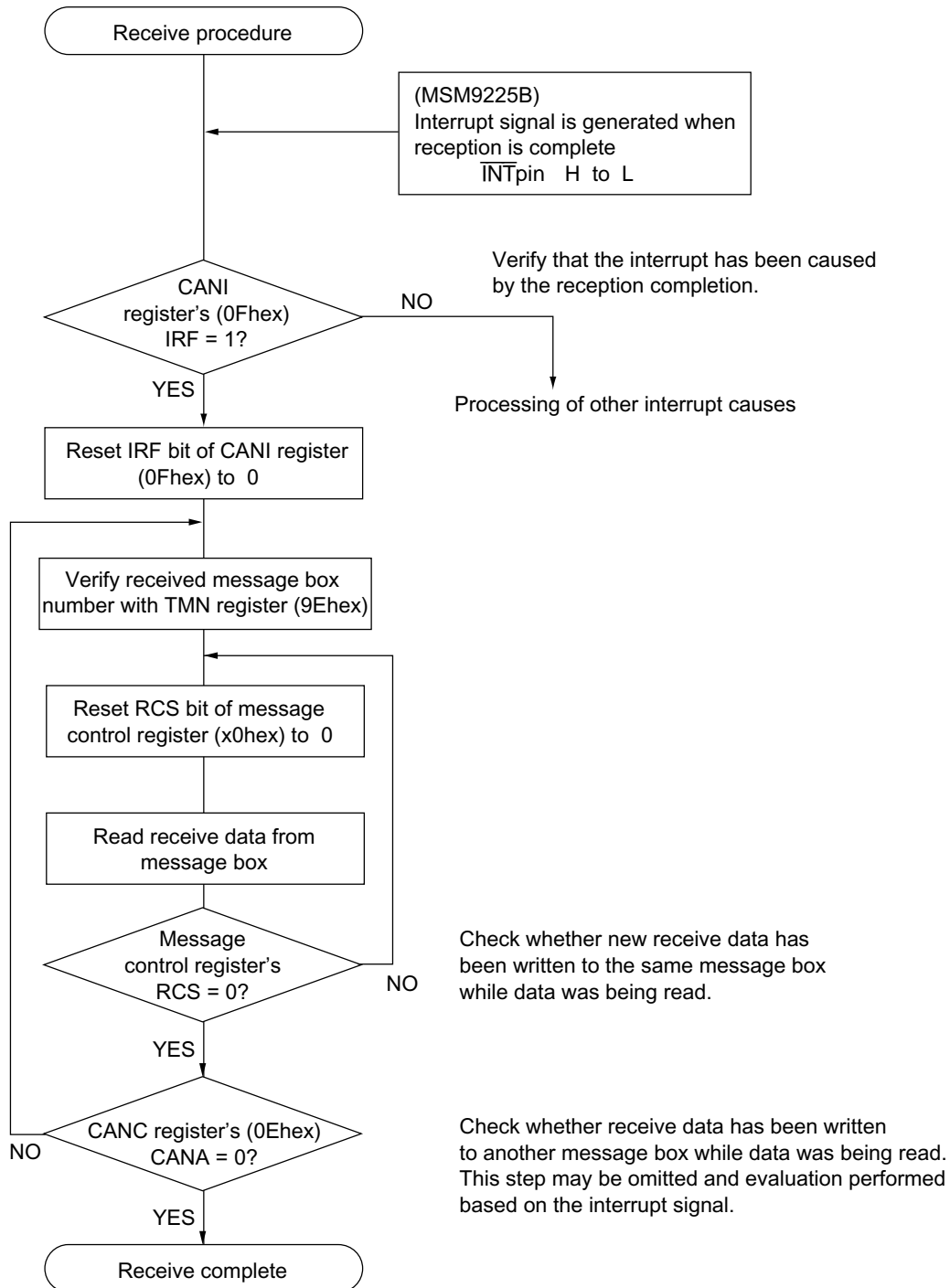Figure 3-3 shows the receive procedure.



**Figure 3-3 Receive Flowchart**

### 3.1.4  Message Box Rewrites during Operation

The procedure to rewrite the Identifier (ID) and Data Length Code (DLC) during operation, excluding the time that initial settings are made for the message boxes, is indicated below.  The number of message boxes set in the NMES register at the initial setting is the number of (valid) message boxes that can be rewritten.
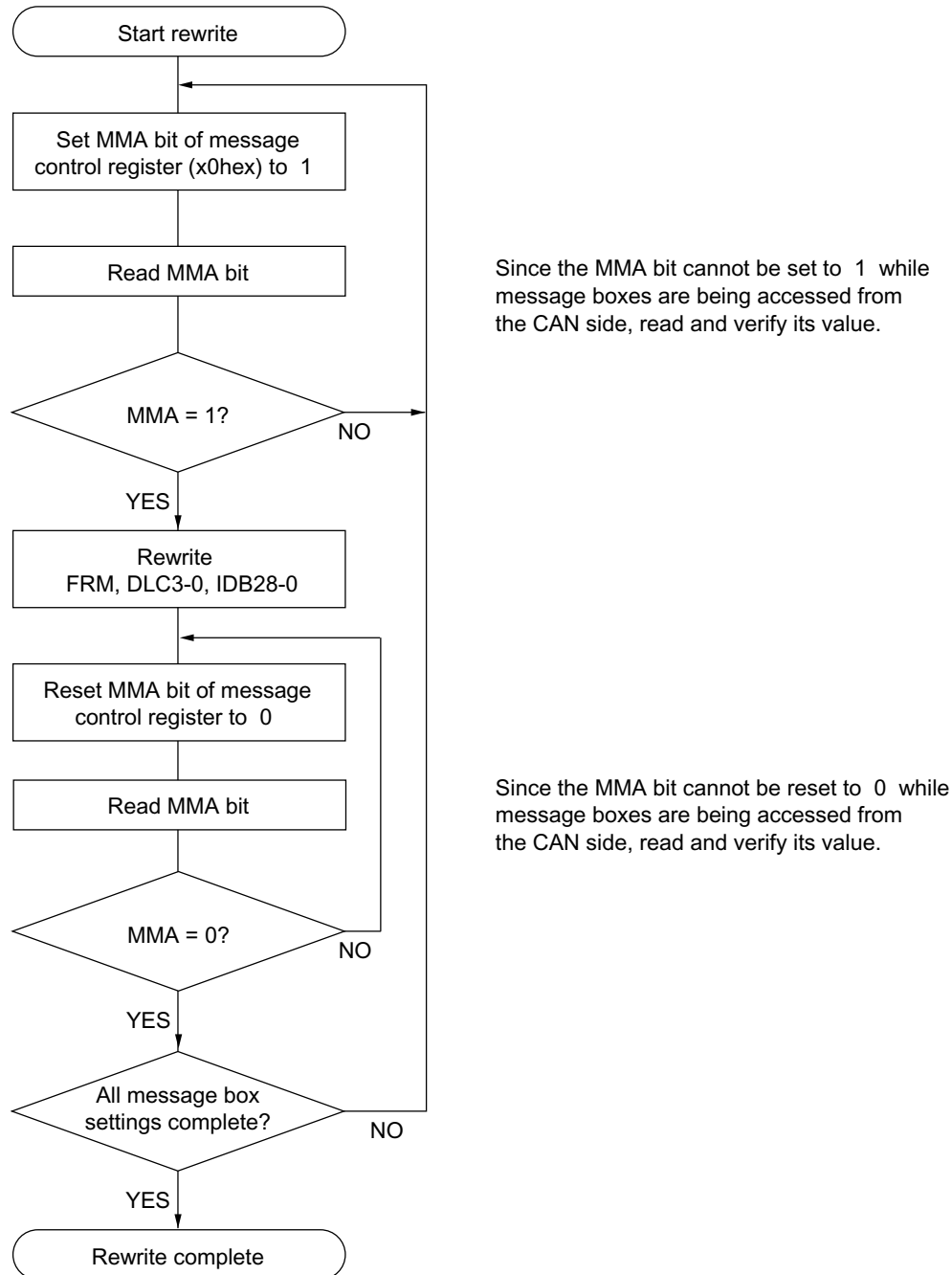
```
                    ( Start rewrite )
                           │
                           ▼◄──────────────────┐
              ┌──────────────────────────┐     │
              │   Set MMA bit of message │     │
              │ control register (x0hex) │     │       Since the MMA bit cannot be set to 1 while
              │          to 1            │     │       message boxes are being accessed from
              └──────────────────────────┘     │       the CAN side, read and verify its value.
                           │                    │
              ┌──────────────────────────┐     │
              │      Read MMA bit        │     │
              └──────────────────────────┘     │
                           │                    │
                       ╱───────╲                │
                      ╱ MMA = 1? ╲──── NO ───────┤
                      ╲          ╱               │
                       ╲───────╱                 │
                           │ YES                  │
              ┌──────────────────────────┐       │
              │        Rewrite           │       │
              │  FRM, DLC3-0, IDB28-0    │       │
              └──────────────────────────┘       │
                           │                      │
                           ▼◄────────────────┐    │
              ┌──────────────────────────┐   │    │
              │  Reset MMA bit of message │   │    │
              │  control register to 0    │   │    │       Since the MMA bit cannot be reset to 0 while
              └──────────────────────────┘   │    │       message boxes are being accessed from
                           │                  │    │       the CAN side, read and verify its value.
              ┌──────────────────────────┐   │    │
              │      Read MMA bit        │   │    │
              └──────────────────────────┘   │    │
                           │                  │    │
                       ╱───────╲              │    │
                      ╱ MMA = 0? ╲──── NO ─────┤    │
                      ╲          ╱             │    │
                       ╲───────╱               │    │
                           │ YES               │    │
                       ╱───────────╲           │    │
                      ╱ All message  ╲         │    │
                      ╲ box settings  ╱── NO ───────┘
                      ╲  complete?   ╱
                       ╲───────────╱
                           │ YES
                    ( Rewrite complete )
```

**Figure 3-4  Message Box Rewrite Flowchart**

3.1.5  Remote Frame Operation

The following two methods are available for transmission after remote frame reception.
(1)  Automatic response: automatically transmit preset message data in message box
(2)  Manual response: set message data and then transmit

3.1.5.1  Automatic Response

After remote frame reception, this method automatically transmits preset message data in message box. Table 3-1 lists the settings of the message control register.

**Table 3-1  Message Control Register Settings for Automatic Response**

|  | Bit | Symbol | Value | Comments |
|---|---|---|---|---|
| Message control register (x0hex) | Bit 5 | TRQ | 0 | When reception is complete, TRQ bit changes from 0 to 1. |
| | Bit 3 | EIR | 0/1 | To enable receive interrupts, set this bit to "1". |
| | Bit 2 | EIT | 1 | Set interrupt to verify the end of transmission. |
| | Bit 1 | FRM | 0 | Specify remote frame as the receive frame type. |
| | Bit 0 | ARES | 1 | Set automatic response. |

Figure 3-5 is the operation flowchart.

Microcontroller (user) operation                    MSM9225B operation

( Start automatic response )

Set MMA bit of message
control register (x0hex) to  1

Read MMA bit

MMA = 1?
NO
YES

**Transmit data setting**

Since the MMA bit cannot be set to  1  while
message boxes are being accessed from
the CAN side, read and verify its value.

Set the message control register
(x0hex) according to Table 3-1

Write transmit data to message box

Reset MMA bit to  0

Read MMA bit

Since the MMA bit cannot be reset to  0  while
message boxes are being accessed from
the CAN side, read and verify its value.

MMA = 0?
NO
YES

**Remote reception and transmission**

Remote frame received?
NO
YES

Data frame transmission

Transmission completion generates
interrupt
INTpin  H  to  L

**Remote transmission verification**

CANI
register's (0Fhex)
ITF is 1?
NO
YES

Processing of other interrupt causes

Reset ITF bit to  0

Reset RCS bit of message
control register to  0

( Remote transmission complete )

**Figure 3-5  Flowchart of Automatic Response after Remote Frame Reception**

3.1.5.2    Manual Response

In this method, after remote frame reception, the transmit data is set and then transmission begins.
Table 3-2 lists the settings of the message control register.

**Table 3-2  Message Control Register Settings for Manual Response**

| | Bit | Symbol | Value | Comments |
|---|---|---|---|---|
| Message control register (x0hex) | Bit 5 | TRQ | 0 | Set to receive message box. |
| | Bit 3 | EIR | 1 | Set interrupt to verify (remote frame) reception. |
| | Bit 2 | EIT | 1 | Set interrupt to verify the end of transmission. |
| | Bit 1 | FRM | 0 | Specify remote frame as the receive frame type. |
| | Bit 0 | ARES | 0 | Specify that automatic response is disabled. |

Figure 3-6 is the operation flow chart.
The basic operation is a combination of receive and transmit procedures.

Microcontroller (user) operation                    MSM9225B operation

```
      ( Start Manual response )
                │
                │                              ┌──────────────────────┐
                │                              ▼                      │
                │                    ◇ Remote frame received? ◇──NO───┘
                │                              │
                │                             YES
                │                              ▼
                │                    ┌──────────────────────┐
                │                    │ Message reception    │
                │                    │ generates interrupt  │
                │                    │ INTpin  H  to  L      │
                │                    └──────────────────────┘
 Remote                              │
 reception         ┌─────────────────┘
                   ▼
            ◇  CANI          ◇──NO──┐
            ◇ register's (0Fhex)◇   │
            ◇   IRF = 1?      ◇     │
                   │                 ▼
                  YES       Processing of other interrupt causes
         ┌──────────────────────────┐
         │ Verify receive message box number │
         │ with TMN retgister (9Ehex), then set │
         │ the message box of that number │
         └──────────────────────────┘
                   │
         ┌──────────────────────────┐
         │ Reset RCS bit of message │
         │ control register (x0hex) to  0 │
         └──────────────────────────┘
                   │
                   ▼  ◄──────────────┐
         ┌──────────────────────┐    │        Since the MMA bit cannot be set to  1  while
         │ Set MMA bit of message │  │        message boxes are being accessed from
         │ control register to  1 │    │        the CAN side, read and verify its value.
         ├──────────────────────┤    │
         │ Read MMA bit         │    │
         └──────────────────────┘    │
                   │                  │
            ◇ MMA = 1? ◇──NO──────────┘
                   │
                  YES
                   ▼
         │ Write transmit data to message box │
                   │
                   ▼  ◄──────────────┐
         ┌──────────────────────┐    │        Since the MMA bit cannot be reset to  0  while
         │ Set  MMA = 0 and TRQ = 1 │ │        message boxes are being accessed from
         │ (message control register bits) │  │  the CAN side, read and verify its value.
         ├──────────────────────┤    │
         │ Read MMA bit         │    │
         └──────────────────────┘    │
                   │                  │
            ◇ MMA = 0? ◇──NO──────────┘
                   │
                  YES
                   ▼
         ┌──────────────────────┐
         │ Set TIRS bit of CANC register │
         │ (0Ehex) to  1 │
         └──────────────────────┘
                                     │
                          ┌──────────────────────┐
                          │ Data frame transmission │
                          ├──────────────────────┤
 Remote                   │ Transmission completion generates │
 transmission             │ interrupt │
                          │ INTpin  H  to  L │
                          └──────────────────────┘
         ┌──────────────────────┐
         │ Verify transmission is complete │
         └──────────────────────┘
                   │
          ( Manual response complete )
```

**Figure 3-6  Manual Response Operation Flowchart**

*Chapter 4*

# Microcontroller Interface

# Chapter 4 Microcontroller Interface

There are two methods of interfacing to the microcontroller.
(1) Synchronous serial interface (serial mode)
(2) Parallel bus interface (parallel mode)
Each interface is selected with the Mode1 and Mode0 pins.
Table 4-1 shows the relation between Mode1 and Mode0 pin values and interface selection.

**Table 4-1  Interface Setting**

| Mode1 | Mode0 | Interface | | |
|-------|-------|-----------|---|---|
| 0 | 0 | Parallel mode | Separate buses | No address latch signal |
| 0 | 1 | | | With address latch signal |
| 1 | 0 | | Multiplexed buses | |
| 1 | 1 | Serial mode | | |

## 4.1  Serial Interface

Figure 4-1 shows the transfer timing.

Address/data transfers begin when the $\overline{CS}$ pin is at a "L" level and end when it changes to a "H" level. Because the MSM9225B has an address increment function, the basic transfer consists of " transfer start address + multiple data." Therefore, to access a nonconsecutive address, the $\overline{CS}$ must be first pulled to a "H" level, and then the address set.

Perform address/data transfers LSB first, in 8-bit units. During a transfer, an interval is necessary between address and data and between consecutive data transfers. (Refer to Chapter 5, "Electrical Characteristics", for interval values.) Note that the SWAIT signal is only generated during the interval between address and data transfers.

(1) Data write
Data write operations are performed with the following procedure.
After setting the $\overline{CS}$ pin and $\overline{PRD}$/SR$\overline{W}$ pin to "L" levels, input an address to the SDI pin. Synchronized to the rising edge of synchronous clock SCLK, the MSM9225B captures the address in an internal register. When 8 SCLK clocks are received, the MSM9225B loads the address into the internal address counter and waits for data reception.
Next, input data to the SDI pin. An internal register captures data in a similar manner to the address capture, at the rising edge of SCLK. When 8 bits of data have been captured, the MSM9225B writes the data to the message memory or control register specified by the address that was received previously, and then increments the address counter by 1. If data is to be written to consecutive addresses, continue the data transfer. After all data has been transferred, set the $\overline{CS}$ pin to a "H" level.

(2) Data read
Data read operations are performed with the following procedure.
After setting the $\overline{CS}$ pin to a "L" level and the $\overline{PRD}$/SR$\overline{W}$ pin to a "H" level, input an address to the SDI pin in the same manner as for the data write operation. When 8 SCLK clocks are received, the MSM9225B loads the address into the internal address counter, reads data from the message memory or control register specified by the address, latches data into a shift register for data output and increments the address counter. Then, when SCLK is input, latched data is output from the SDO pin synchronized to the falling edge of SCLK. At this time, the contents of the data input from the SDI pin does not matter. If there exists remaining data to be read, input another 8 SCLK clocks. After all the data at consecutive addresses has been read, set the $\overline{CS}$ pin to a "H" level.

If the count value of the lower 4 bits of an address overflows (exceeds xFh), the address increment function will reset the count value of the lower 4 bits to 0 without changing the upper 4 bits of the address, and will continue counting.

Figure 4-1  Serial Interface Transfer Timing

## 4.2  Parallel Interface

The following three types of parallel interfaces are available.
(1)  Address/data separate bus type, no address latch signal
(2)  Address/data separate bus type, with address latch signal
(3)  Multiplexed bus type

For transfer timings, refer to Section 5.2, "Timing Diagrams".

## 4.3  MSM9225B Connection Examples

The following examples are for recommendation only. Oki does not guarantee any operation on customer's systems.

### 4.3.1  Microcontroller Interface

#### 4.3.1.1  Address/Data Separate Bus (No Address Latch Signal)
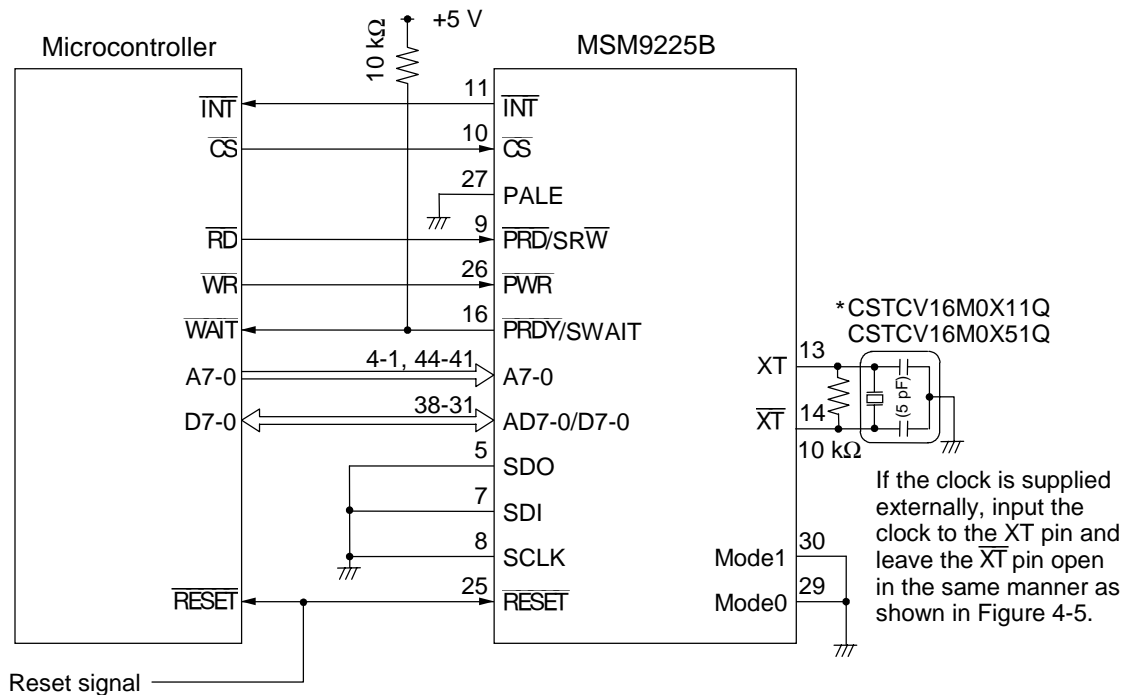


* Ceramic resonator of Murata MFG. (CSTCV16M0X11Q) is recommended. (125 kbps)

**Figure 4-2  Address/Data Separate Bus (No Address Latch Signal)**

4.3.1.2    Address/Data Separate Bus (With Address Latch Signal)
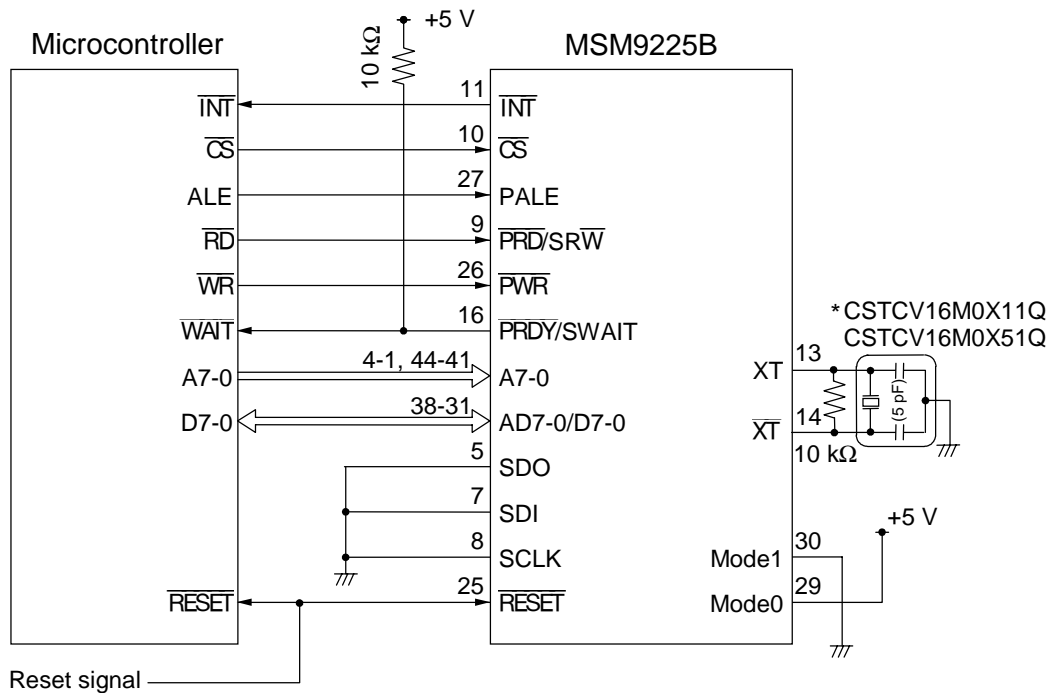


* Ceramic resonator of Murata MFG. (CSTCV16M0X11Q) is recommended. (125 kbps)

**Figure 4-3  Address/Data Separate Bus (With Address Latch Signal)**

4.3.1.3    Address/Data Multiplexed Bus
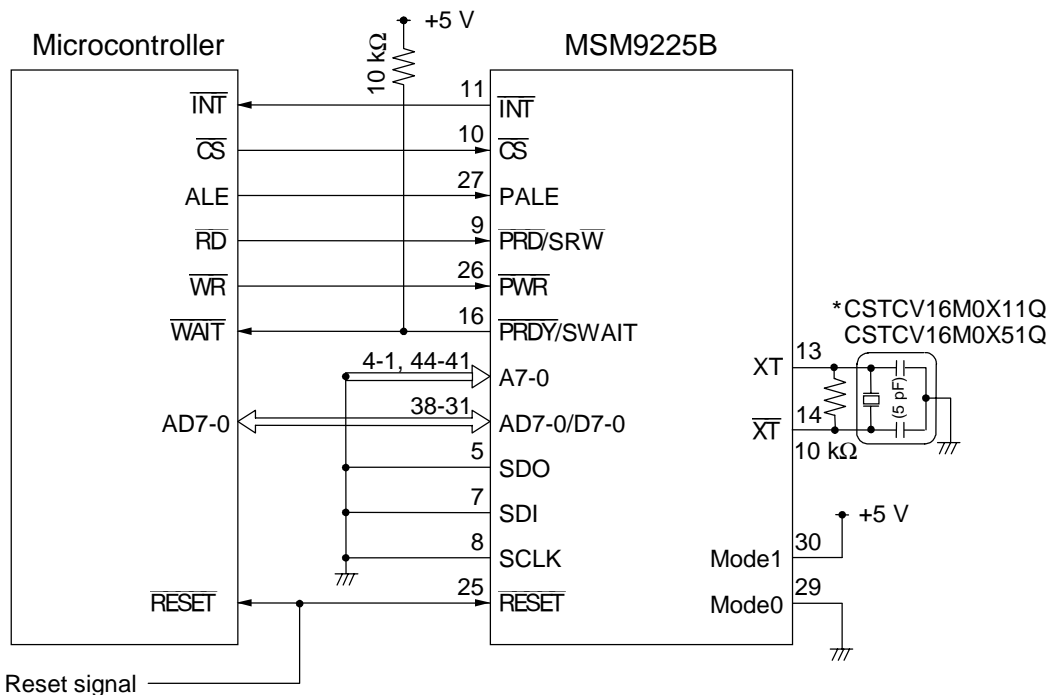


* Ceramic resonator of Murata MFG. (CSTCV16M0X11Q) is recommended. (125 kbps)

**Figure 4-4  Address/Data Multiplexed Bus**

### 4.3.1.4    Serial Interface
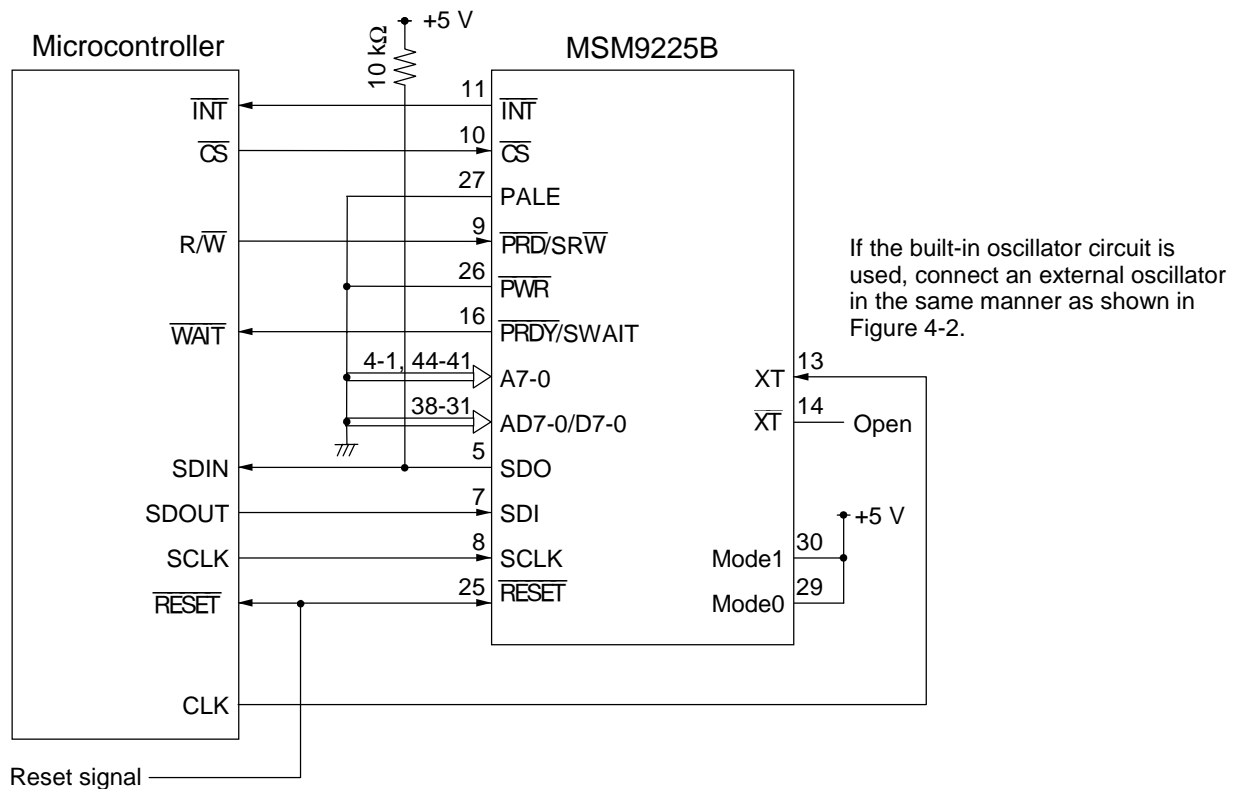


**Figure 4-5  Serial Interface**

4.3.2   CAN Bus Interface

4.3.2.1   Electrically Isolated from Bus Transceiver (PCA82C250)
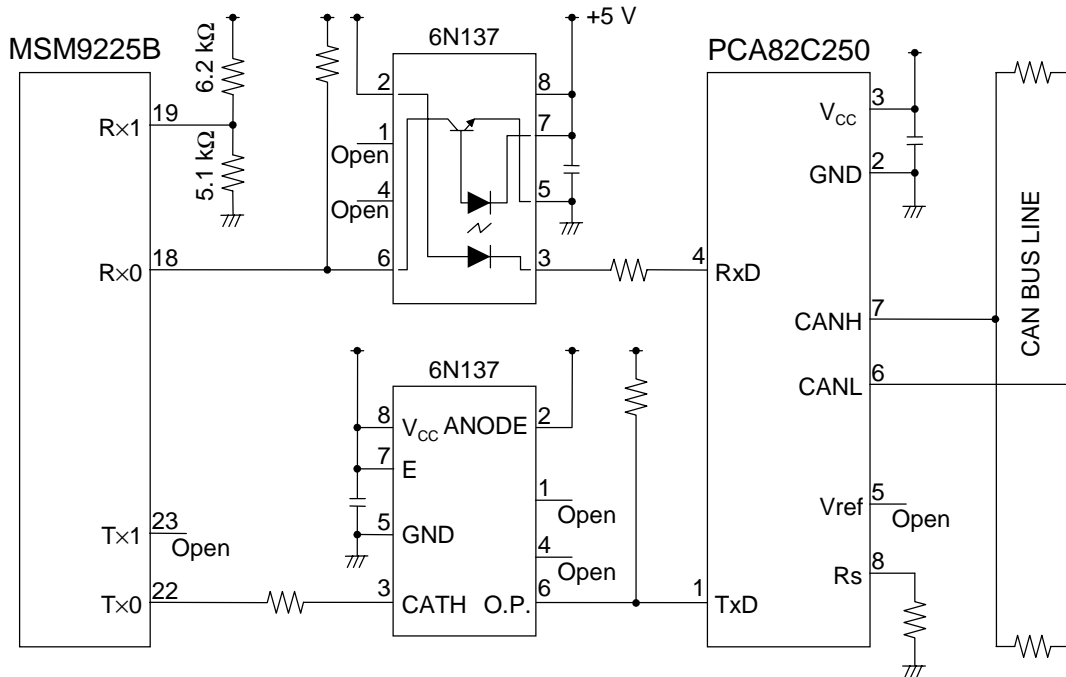


**Figure 4-6  Electrically Isolated from Bus Transceiver (PCA82C250)**

4.3.2.2   Directly Connected to Bus Transceiver (PCA82C250)
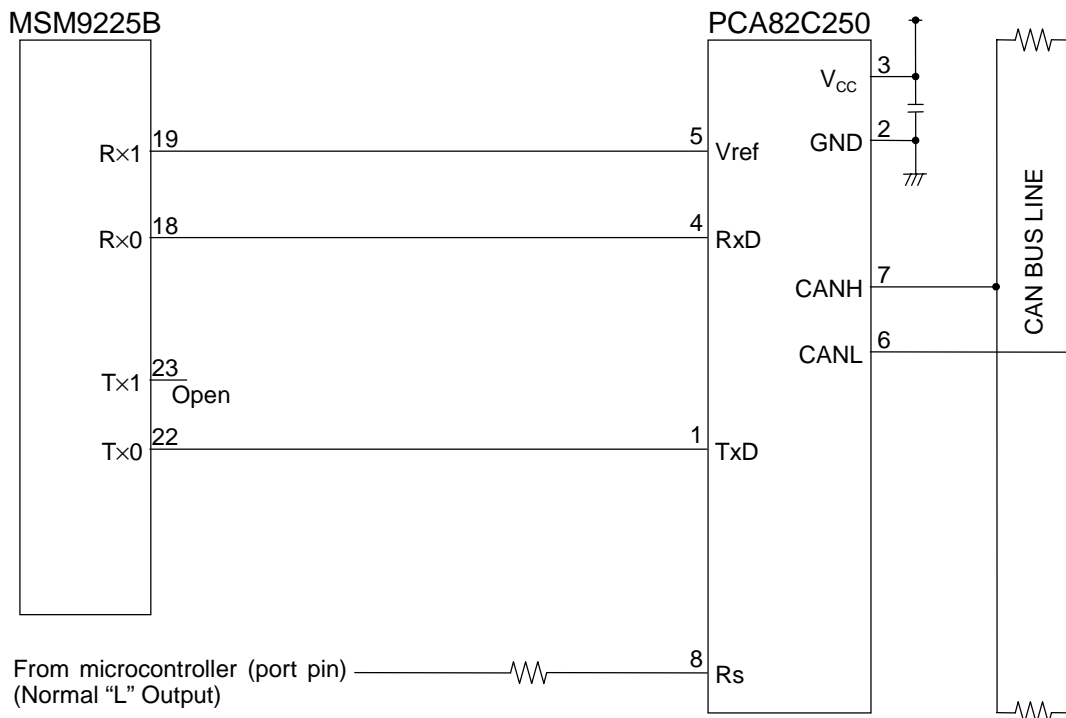


**Figure 4-7  Directly Connected to Bus Transceiver (PCA82C250)**

4.3.2.3    Monitoring the CAN Bus
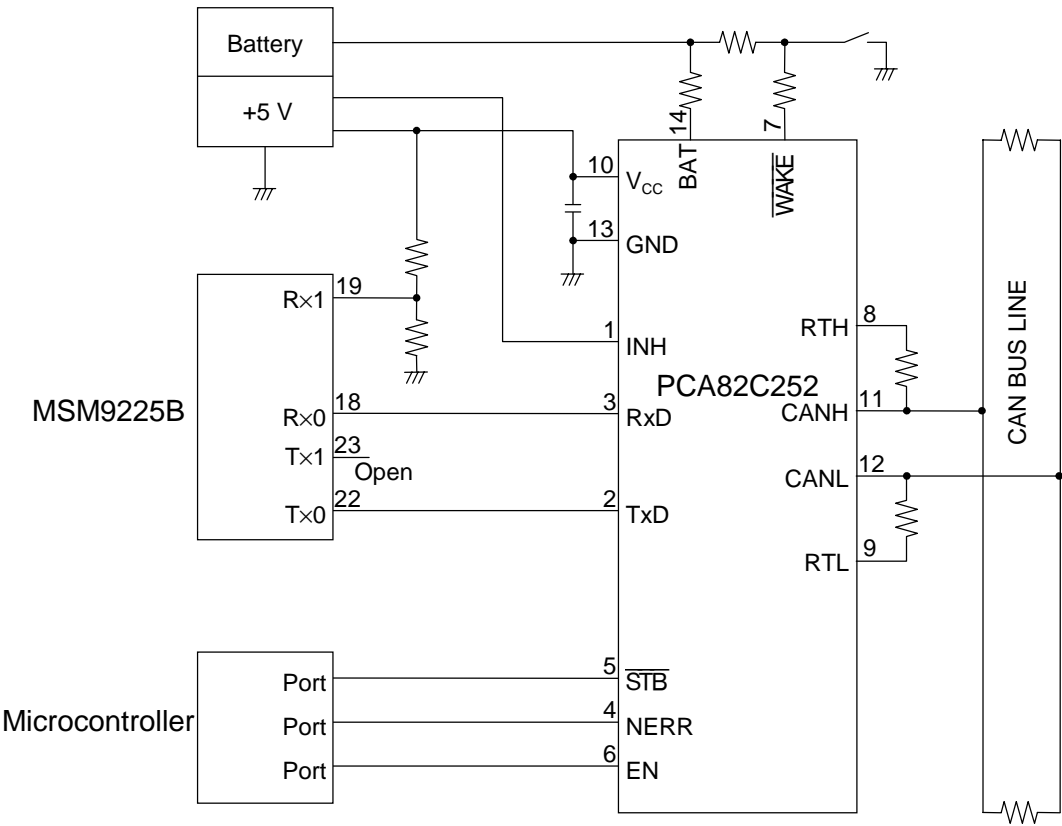


**Figure 4-8  Monitoring the CAN Bus**

# Electrical Characteristics

# Chapter 5    Electrical Characteristics

## 5.1   Electrical Characteristics

### 5.1.1  Absolute Maximum Ratings

| Parameter | Symbol | Condition | Rating | Unit |
|---|---|---|---|---|
| Power Supply Voltage | $V_{DD}$ | Ta = 25°C | −0.3 to +7.0 | V |
| Input Voltage | $V_I$ | — | −0.3 to $V_{DD}$ +3.0 | V |
| Output Voltage | $V_O$ | — | −0.3 to $V_{DD}$ +3.0 | V |
| Power Dissipation | $P_D$ | Ta ≤ 25°C | 615 | mW |
| Operating Temperature | $T_{OP}$ | — | −40 to +125 | °C |
| Storage Temperature | $T_{STG}$ | — | −65 to +150 | °C |

### 5.1.2  Recommended Operating Conditions

| Parameter | Symbol | Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Power Supply Voltage | $V_{DD}$ | $V_{DD} = AV_{DD}$ | 4.5 | 5.0 | 5.5 | V |
| Operating Temperature | $T_{OP}$ | — | −40 | +25 | +125 | °C |

### 5.1.3  DC Characteristics

($V_{DD}$ = 4.5 to 5.5 V, Ta = −40 to +125°C)

| Parameter | Symbol | Applicable pin | Condition | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| "H" Input Voltage | $V_{IH}$ | Applies to all inputs | — | $0.8V_{DD}$ | $V_{DD}$ +0.3 | V |
| "L" Input Voltage | $V_{IL}$ | Applies to all inputs | — | −0.3 | +0.2 $V_{DD}$ | V |
| "H" Input Current | $I_{IH1}$ | XT | $V_I = V_{DD}$ | 3 | 25 | µA |
|  | $I_{IH2}$ | Other inputs |  | −1.0 | +1.0 | µA |
| "L" Input Current | $I_{IL1}$ | XT | $V_I$ = 0V | −25 | −3 | µA |
|  | $I_{IL2}$ | Other input |  | −1.0 | +1.0 | µA |
| "H" Output Voltage | $V_{OH1}$ | $\overline{INT}$, $\overline{PRDY}$/SWAIT | $I_{OH1}$ = −80 µA | $V_{DD}$ −1.0 | — | V |
|  | $V_{OH2}$ | AD7-0/D7-0 | $I_{OH2}$ = −400 µA | $V_{DD}$ −1.0 | — | V |
| "L" Output Voltage | $V_{OL1}$ | $\overline{INT}$, $\overline{PRDY}$/SWAIT | $I_{OL1}$ = 1.6 mA | — | 0.4 | V |
|  | $V_{OL2}$ | AD7-0/D7-0 | $I_{OL2}$ = 3.2 mA | — | 0.4 | V |
| Output Leakage Current | $I_{IH1}$ | $\overline{PRDY}$/SWAIT AD7-0/D7-0 | $V_I = V_{DD}$/0 V | −1.0 | +1.0 | µA |
| Dynamic Supply Current | $I_{DD}$ | — | $f_{OSC}$ = 16 MHz, No Load | — | 9 | mA |
| Static Supply Current | $I_{DDS}$ | — | SLEEP Mode | — | 400 | µA |
|  |  | — | STOP Mode | — | 100 | µA |

### 5.1.4  Rx0, Rx1 Characteristics

Differential input mode

($V_{DD}$ = 4.5 to 5.5 V, Ta = −40 to +125°C)

| Parameter | Symbol | Applicable pin | Condition | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| 'dominant' Input Voltage | VRx0(d) | Rx0 | VRx1 = 0.4 $V_{DD}$ to 0.6 $V_{DD}$ | −0.3 | VRx1 −0.4 | V |
| 'recessive' Input Voltage | VRx0(r) | Rx0 |  | VRx1 +0.4 | $V_{DD}$ +3 | V |
| Input Leakage Current | $I_{LK}$ | Rx0, Rx1 | $VR_{XI} = V_{DD}$/0 V | −1 | +1 | µA |

### 5.1.5  Tx0, Tx1 Characteristics

($V_{DD}$ = 4.5 to 5.5 V, Ta = −40 to +125°C)

| Parameter | Symbol | Condition | Min. | Max. | Unit |
|---|---|---|---|---|---|
| "H" Output Voltage | $V_{OH}$ | $I_{OH}$ = −3.0 mA | $V_{DD}$ −0.4 | — | V |
| "L" Output Voltage | $V_{OL}$ | $I_{OL}$ = 10.0 mA | — | 0.4 | V |

### 5.1.6  AC Characteristics

5 – 3

Parallel mode

$(V_{DD} = 4.5$ to 5.5 V, Ta = −40 to +125°C, $f_{OSC}$ = 16 MHz)

| Parameter | | Symbol | Condition | Min. | Max. | Unit |
|---|---|---|---|---|---|---|
| ALE Address Setup Time | | $t_{AS}$ | — | 10 | — | ns |
| ALE Address Hold Time | | $t_{AH}$ | — | 10 | — | ns |
| $\overline{PRD}$ Output Data Delay Time | | $t_{RDLY}$ | — | — | 60[*1] | ns |
| $\overline{PRD}$ Output Data Hold Time | | $t_{RDH}$ | — | 5 | — | ns |
| ALE "H" Level Width | | $t_{WALEH}$ | — | 16.5 | — | ns |
| Max Access Cycle | When $\overline{PRDY}$ is not generated | $t_{cyc}$ | — | 4T | — | ns |
| | When $\overline{PRDY}$ is generated | | | 7T | — | ns |
| Address Hold Time from $\overline{PRD}$ | | $t_{RAH}$ | — | 0 | — | ns |
| ALE Delay Time from $\overline{PRD}$ | | $t_{HRA}$ | — | 27 | — | ns |
| $\overline{PRD}$ "H" Level Width | | $t_{WRDH}$ | — | 27 | — | ns |
| $\overline{PRDY}$ "L" Delay Time | | $t_{ARLDLY}$ | — | — | 35 | ns |
| $\overline{PRDY}$ "L" Level Width | | $t_{WRDYL}$ | — | 0 | 2.5T | ns |
| Data Output Delay Time from $\overline{PRDY}$ | | $t_{ARDDLY}$ | — | — | 35 | ns |
| $\overline{PWR}$ Hold Time from $\overline{PRDY}$ | | $t_{ARWDLY}$ | — | 10 | — | ns |
| Input Data Setup Time | | $t_{WDS}$ | — | 30 | — | ns |
| Input Data Hold Time | | $t_{WDH}$ | — | 4 | — | ns |
| $\overline{PRD}$ Delay Time | | $t_{RS}$ | — | 10 | — | ns |
| $\overline{PWR}$ Delay Time | | $t_{WS}$ | — | 10 | — | ns |
| Address Hold Time from $\overline{PWR}$ | | $t_{WAH}$ | — | 10 | — | ns |
| ALE Delay Time from $\overline{PWR}$ | | $t_{HWA}$ | — | 27 | — | ns |
| $\overline{PWR}$ "H" Level Width | | $t_{WRH}$ | — | 40 | — | ns |
| $\overline{PWR}$ "L" Level Width | | $t_{WRL}$ | — | 20[*1] | — | ns |
| $\overline{CS}$ Delay Time from $\overline{PRD}$ | | $t_{HRC}$ | — | 0 | — | ns |
| $\overline{CS}$ Delay Time from $\overline{PWR}$ | | $t_{HWC}$ | — | 0 | — | ns |

$T = 1/f_{OSC}$

The values with *1 indicate those when $\overline{PRDY}$ is not generated.

The values with *1 when $\overline{PRDY}$ is generated are defined by "Data Output Delay Time from $\overline{PRDY}$" $t_{ARDDLY}$ and "$\overline{PWR}$ Hold Time from $\overline{PRDY}$" $t_{ARWDLY}$.

Serial mode

$(V_{DD}$ = 4.5 to 5.5 V, Ta = –40 to +125°C, $f_{OSC}$ = 16 MHz)

| Parameter | Symbol | Condition | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $\overline{CS}$ Setup Time | $t_{CS}$ | — | 10 | — | ns |
| $\overline{CS}$ Hold Time | $t_{CH}$ | — | 8T | — | ns |
| SCLK Cycle | $t_{CP}$ | — | 167 | — | ns |
| SCLK Pulse Width | $t_{CW}$ | — | 83 | — | ns |
| SDI Setup Time | $t_{DS}$ | — | 30 | — | ns |
| SDI Hold Time | $t_{DH}$ | — | 5 | — | ns |
| SDO Output Enable Time | $t_{CSODLY}$ | — | — | 30 | ns |
| SDO Output Disable Time | $t_{CSZDLY}$ | — | — | 30 | ns |
| SDO Output Delay Time | $t_{PD}$ | — | — | 30 | ns |
| SR$\overline{W}$ Setup Time | $t_{RS}$ | — | 10 | — | ns |
| SR$\overline{W}$ Hold Time | $t_{RH}$ | | 0 | — | ns |
| SWAIT Output Delay Time | $t_{SRDLY}$ | — | — | 2T | ns |
| SWAIT "H" Level Width | $t_{WRDY}$ | — | — | 6T | ns |
| Byte Delay | $t_{WAIT}$ | — | 8T | — | ns |

T = 1/$f_{OSC}$

Other timing characteristics

$(V_{DD}$ = 4.5 to 5.5 V, Ta = –40 to +125°C)

| Parameter | Symbol | Condition | Min. | Max. | Unit |
|---|---|---|---|---|---|
| System Clock Cycle | $t_{clkcy}$ | — | 62 | — | ns |
| $\overline{RESET}$ "H" Level Input Width | $t_{WRSTH}$ | — | 5 | — | µs |
| $\overline{RESET}$ "L" Level Input Width | $t_{WRSTL}$ | — | 5 | — | µs |
| $\overline{INT}$ "L" Level Output Width | $t_{WINTL}$ | — | 32T | — | ns |

T = 1/$f_{OSC}$

## 5.2   Timing Diagrams

### 5.2.1   Separate Bus Mode

Read access timing
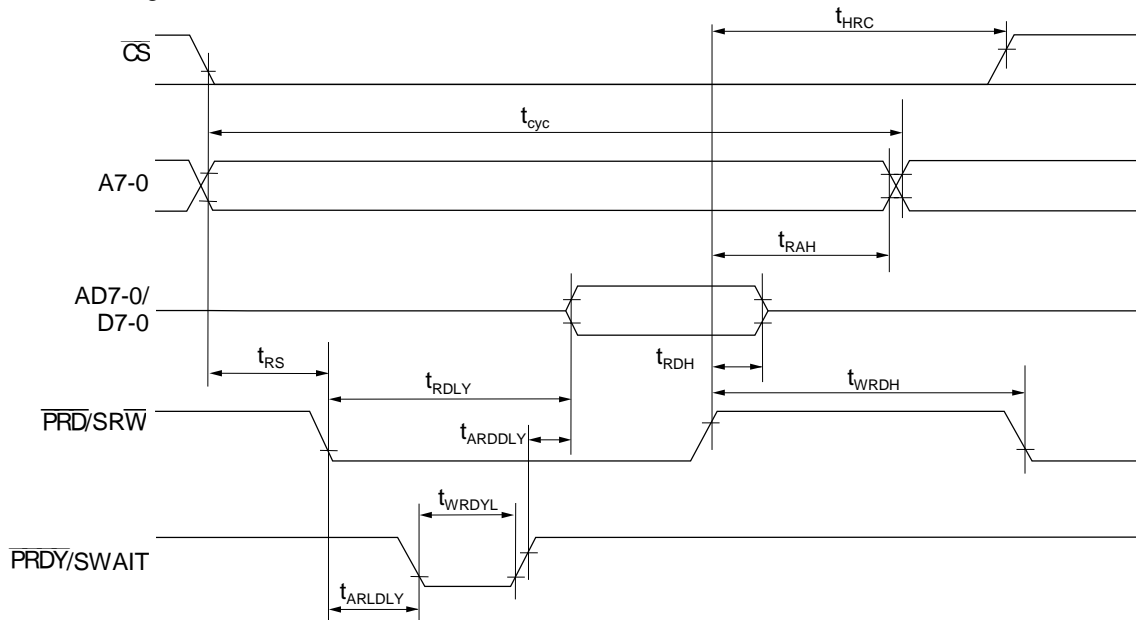


Note:   The $\overline{\text{PRDY}}$ signal may be output depending on the internal state of the MSM9225B.

**Figure 5-1    Read Access Timing**

Write access timing
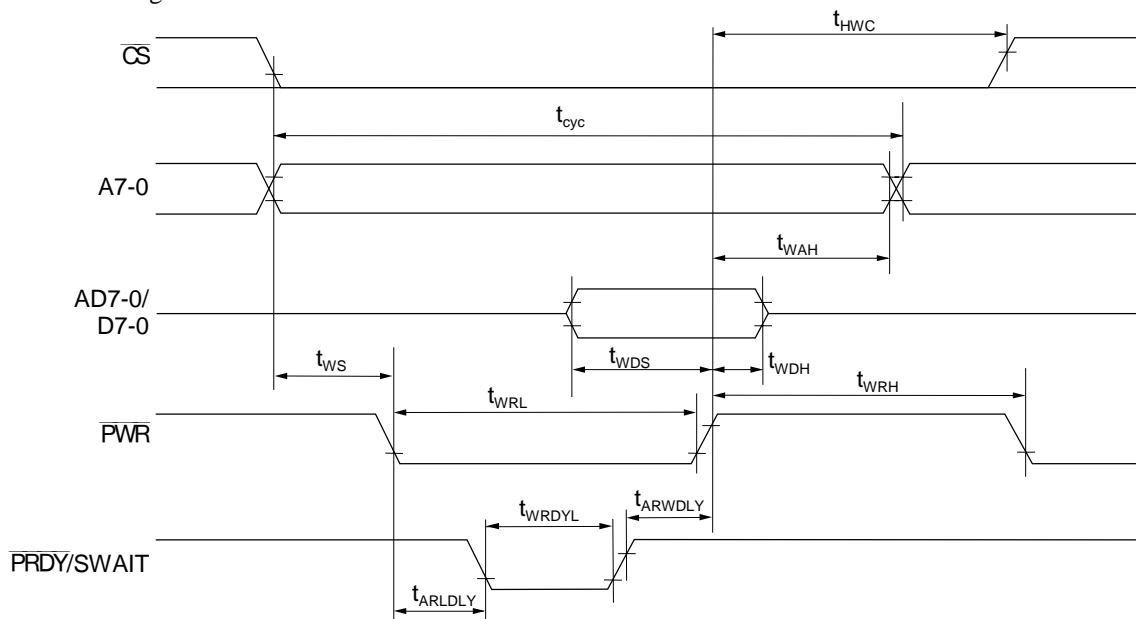


Note:   The $\overline{\text{PRDY}}$ signal may be output depending on the internal state of the MSM9225B.

**Figure 5-2    Write Access Timing**

### 5.2.2   Separate Bus/Address Latch Mode

Read access timing
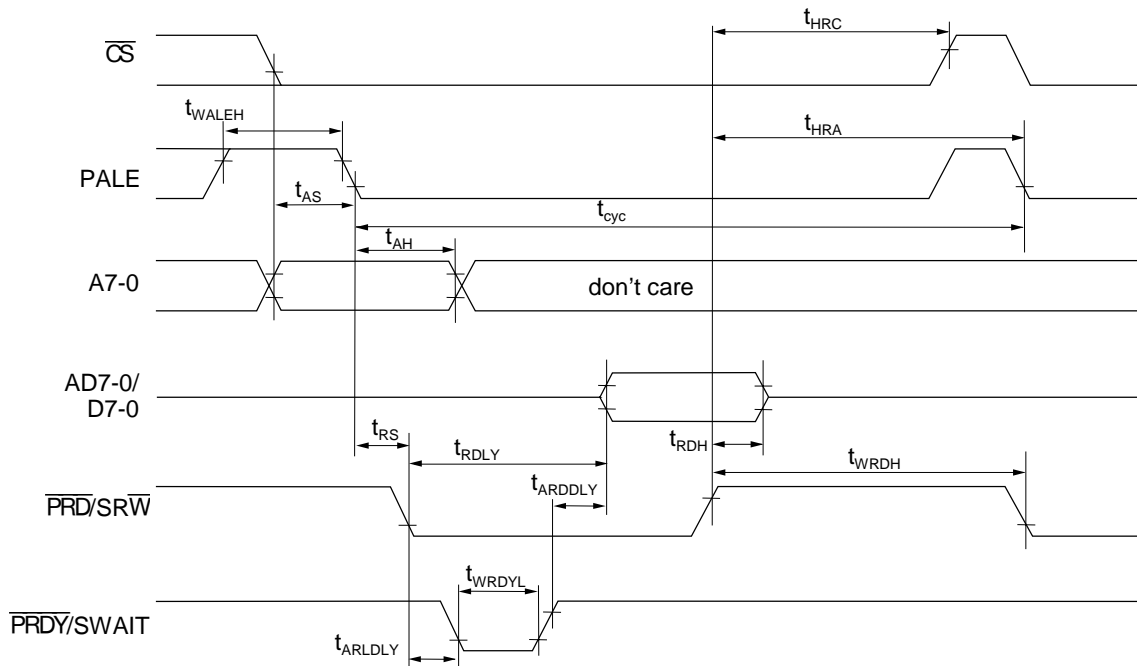


Note:   The $\overline{PRDY}$ signal may be output depending on the internal state of the MSM9225B.

**Figure 5-3    Read Access Timing**

Write access timing
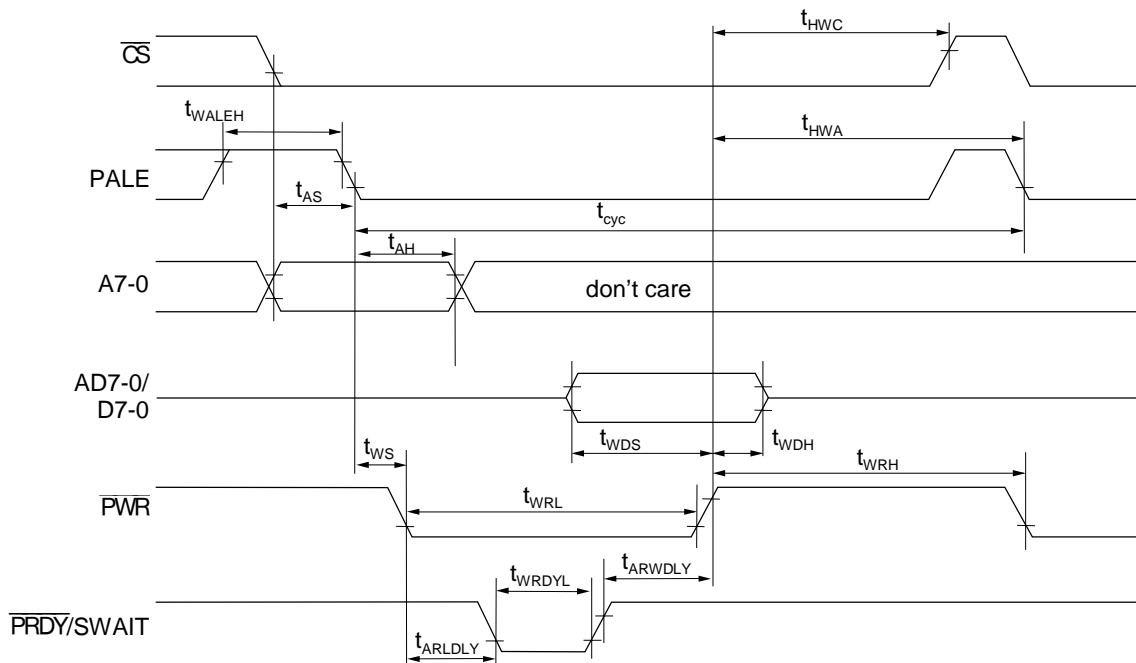


Note:   The $\overline{PRDY}$ signal may be output depending on the internal state of the MSM9225B.

**Figure 5-4    Write Access Timing**

### 5.2.3  Multiplexed Bus Mode

Read access timing
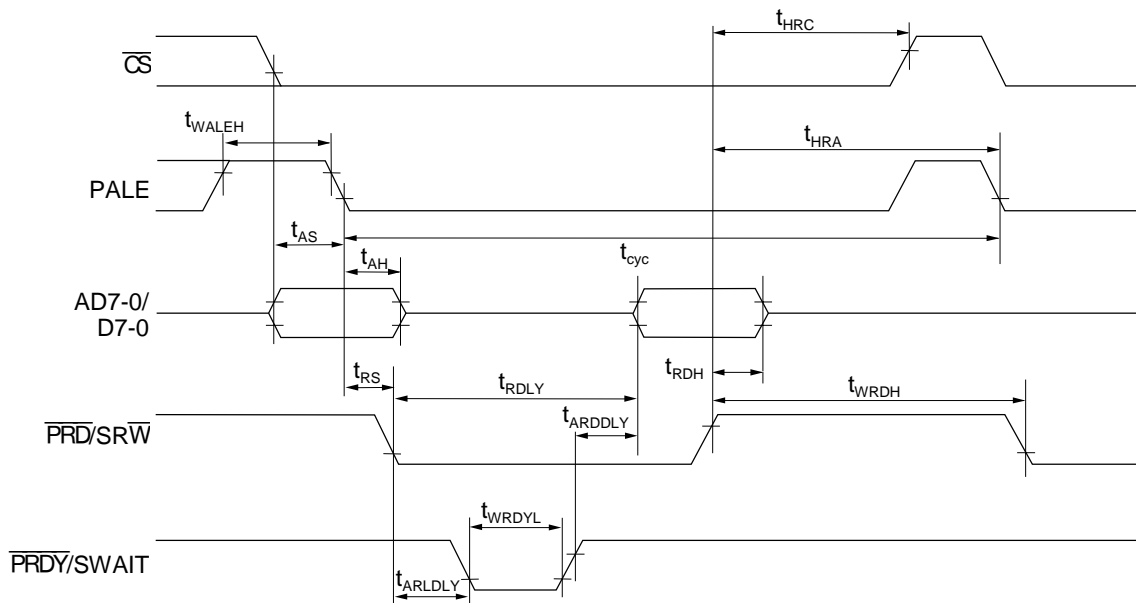


Note:   The $\overline{\text{PRDY}}$ signal may be output depending on the internal state of the MSM9225B.

**Figure 5-5    Read Access Timing**

Write access timing
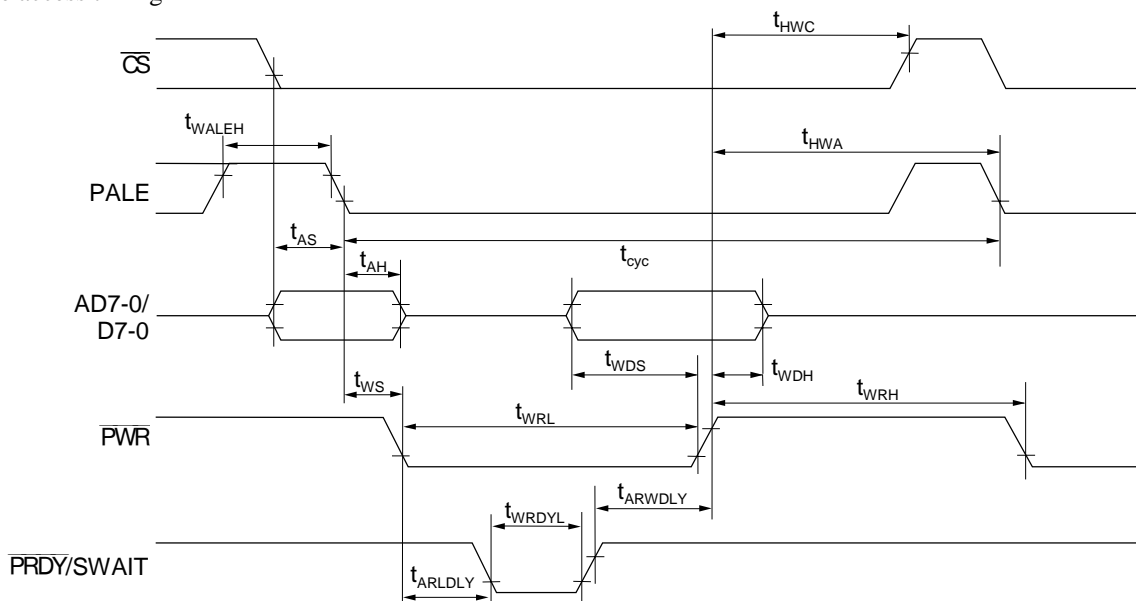


Note:   The $\overline{\text{PRDY}}$ signal may be output depending on the internal state of the MSM9225B.

**Figure 5-6    Write Access Timing**

### 5.2.4  Serial Mode

Read access timing
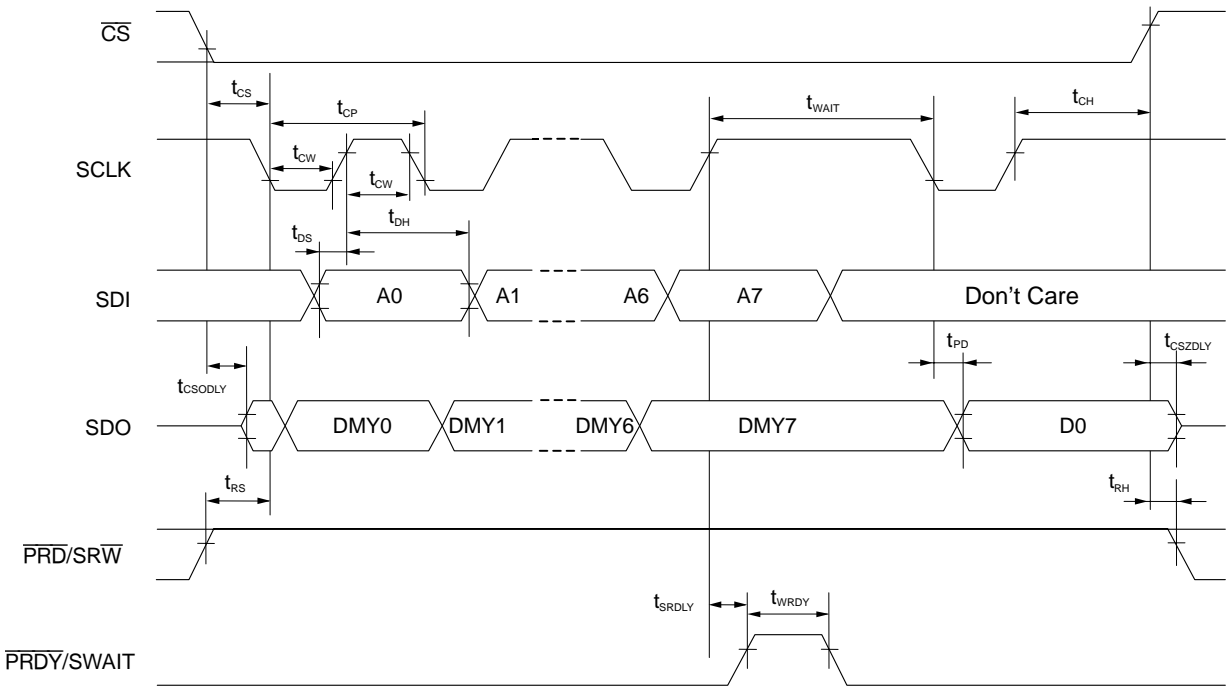


Note:    The SWAIT signal will be output during the interval between address and data transfers.

**Figure 5-7    Read Access Timing**
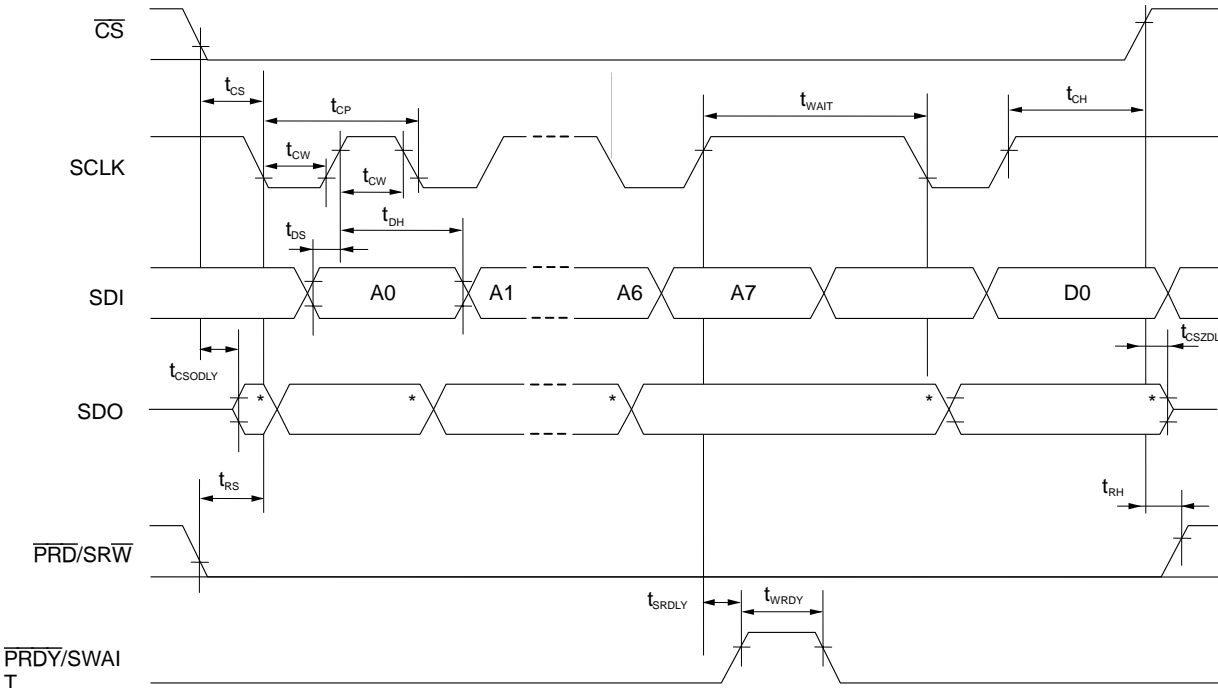
Write access timing



Note:    The SWAIT signal will be output during the interval between address and data transfers.

* : don't care

**Figure 5-8    Write Access Timing**

## 5.2.5  Other Timing
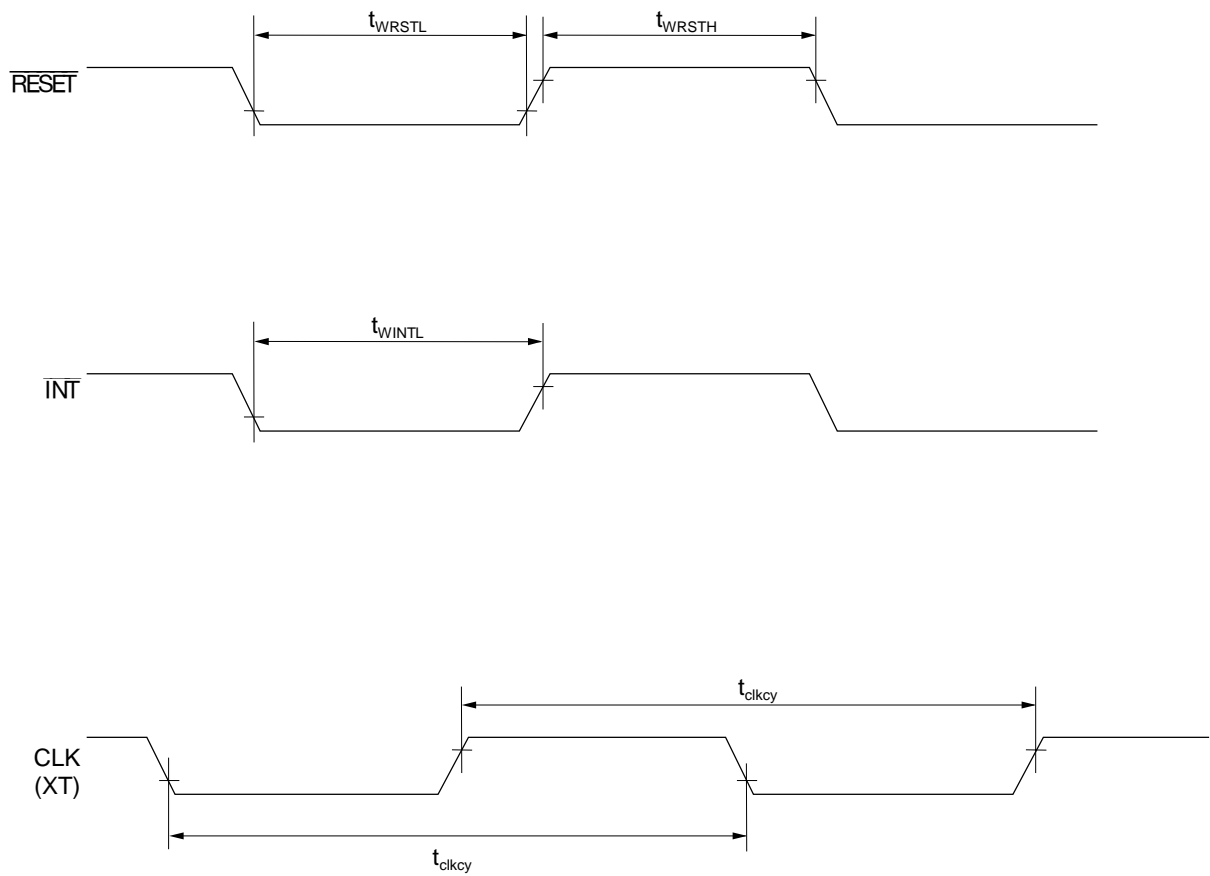


**Figure 5-9    Other Timing**

# Appendixes

# Appendix A   Package Dimensions

(Unit: mm)



QFP44-P-910-0.80-2K

| Package material | Epoxy resin |
|---|---|
| Lead frame material | 42 alloy |
| Pin treatment | Solder plating (≥5μm) |
| Package weight (g) | 0.41 TYP. |
| Rev. No./Last Revised | 4/Nov. 28, 1996 |

**Oki Electric Industry Co., Ltd.**
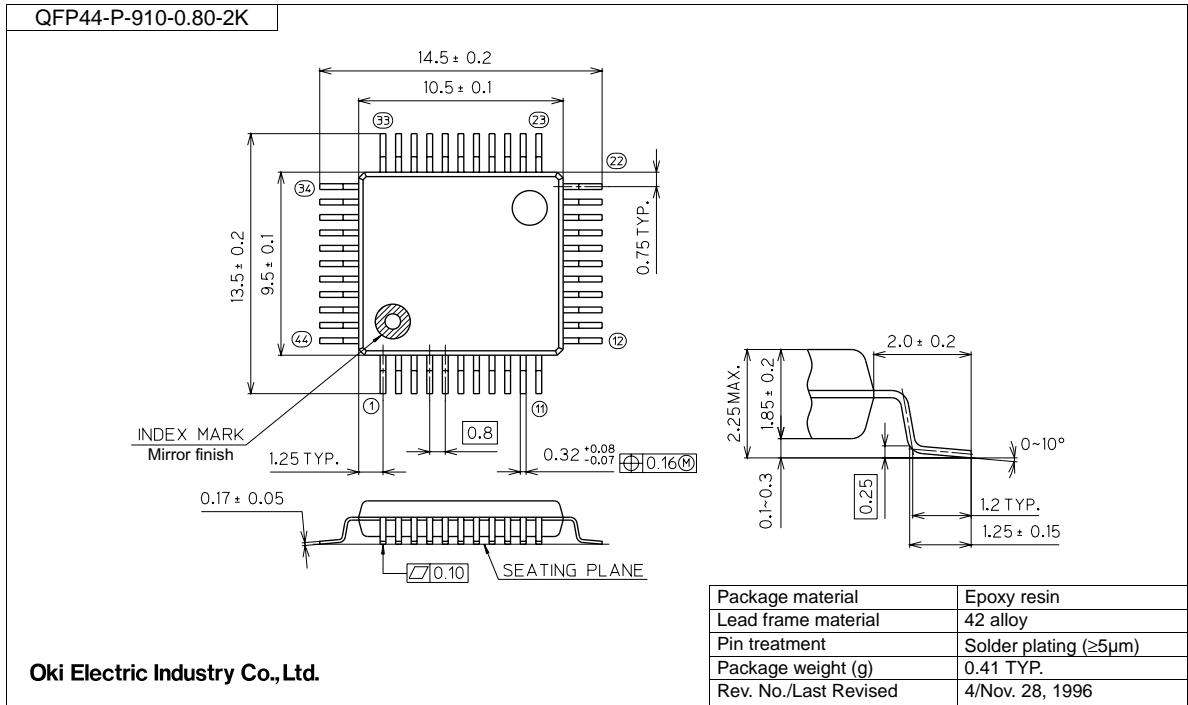
Notes for Mounting the Surface Mount Type Package

The surface mount type packages are very susceptible to heat in reflow mounting and humidity absorbed in storage. Therefore, before you perform reflow mounting, contact Oki's responsible sales person on the product name, package name, pin number, package code and desired mounting conditions (reflow method, temperature and times).

# Appendix B   MSM9225B Memory Map

The following is the memory map of the entire memory space of the MSM9225B.
The symbols 'Mn' (for example, M0MCR, M0MSG0, M0IDR2, etc., n = 0 to 15) correspond to the number of the respective message box among the message boxes 0 to 15.

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 0000 | Message control register | | M0MCR | | R/W | 8 | 00 |
| 0001 | Identifier 0 | | M0IDR0 | | R/W | 8 | Undefined |
| 0002 | Identifier 1 | | M0IDR1 | | R/W | 8 | Undefined |
| 0003 | Message 0 | Identifier 2 | M0MSG0 | M0IDR2 | R/W | 8 | Undefined |
| 0004 | Message 1 | Identifier 3 | M0MSG1 | M0IDR3 | R/W | 8 | Undefined |
| 0005 (∗) | Message 2 | Identifier 4 | M0MSG2 | M0IDR4 | R/W | 8 | Undefined |
| 0006 | Message 3 | Message 0 | M0MSG3 | M0TMSG0 | R/W | 8 | Undefined |
| 0007 | Message 4 | Message 1 | M0MSG4 | M0TMSG1 | R/W | 8 | Undefined |
| 0008 | Message 5 | Message 2 | M0MSG5 | M0TMSG2 | R/W | 8 | Undefined |
| 0009 | Message 6 | Message 3 | M0MSG6 | M0TMSG3 | R/W | 8 | Undefined |
| 000A | Message 7 | Message 4 | M0MSG7 | M0TMSG4 | R/W | 8 | Undefined |
| 000B | — | Message 5 | — | M0TMSG5 | R/W | 8 | Undefined |
| 000C | — | Message 6 | — | M0TMSG6 | R/W | 8 | Undefined |
| 000D | — | Message 7 | — | M0TMSG7 | R/W | 8 | Undefined |
| 000E∗ | CAN control register | | CANC | | R/W | 8 | 01 |
| 000F∗ | CAN interrupt control register | | CANI | | R/W | 8 | 00 |
| 0010 | Message control register | | M1MCR | | R/W | 8 | 00 |
| 0011 | Identifier 0 | | M1IDR0 | | R/W | 8 | Undefined |
| 0012 | Identifier 1 | | M1IDR1 | | R/W | 8 | Undefined |
| 0013 | Message 0 | Identifier 2 | M1MSG0 | M1IDR2 | R/W | 8 | Undefined |
| 0014 | Message 1 | Identifier 3 | M1MSG1 | M1IDR3 | R/W | 8 | Undefined |
| 0015 (∗) | Message 2 | Identifier 4 | M1MSG2 | M1IDR4 | R/W | 8 | Undefined |
| 0016 | Message 3 | Message 0 | M1MSG3 | M1TMSG0 | R/W | 8 | Undefined |
| 0017 | Message 4 | Message 1 | M1MSG4 | M1TMSG1 | R/W | 8 | Undefined |
| 0018 | Message 5 | Message 2 | M1MSG5 | M1TMSG2 | R/W | 8 | Undefined |
| 0019 | Message 6 | Message 3 | M1MSG6 | M1TMSG3 | R/W | 8 | Undefined |
| 001A | Message 7 | Message 4 | M1MSG7 | M1TMSG4 | R/W | 8 | Undefined |
| 001B | — | Message 5 | — | M1TMSG5 | R/W | 8 | Undefined |
| 001C | — | Message 6 | — | M1TMSG6 | R/W | 8 | Undefined |
| 001D | — | Message 7 | — | M1TMSG7 | R/W | 8 | Undefined |
| 001E∗ | Message box count setting register | | NMES | | R/W | 8 | 00 |
| 001F | CAN bus timing register 0 | | BTR0 | | R/W | 8 | 00 |

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.

A bit can be missing at address 00x5h only in the case of the extended format.

A dash "—" indicates that the memory space is not used in the case of the standard format.

A – 3

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 0020 | Message control register | | M2MCR | | R/W | 8 | 00 |
| 0021 | Identifier 0 | | M2IDR0 | | R/W | 8 | Undefined |
| 0022 | Identifier 1 | | M2IDR1 | | R/W | 8 | Undefined |
| 0023 | Message 0 | Identifier 2 | M2MSG0 | M2IDR2 | R/W | 8 | Undefined |
| 0024 | Message 1 | Identifier 3 | M2MSG1 | M2IDR3 | R/W | 8 | Undefined |
| 0025 (∗) | Message 2 | Identifier 4 | M2MSG2 | M2IDR4 | R/W | 8 | Undefined |
| 0026 | Message 3 | Message 0 | M2MSG3 | M2TMSG0 | R/W | 8 | Undefined |
| 0027 | Message 4 | Message 1 | M2MSG4 | M2TMSG1 | R/W | 8 | Undefined |
| 0028 | Message 5 | Message 2 | M2MSG5 | M2TMSG2 | R/W | 8 | Undefined |
| 0029 | Message 6 | Message 3 | M2MSG6 | M2TMSG3 | R/W | 8 | Undefined |
| 002A | Message 7 | Message 4 | M2MSG7 | M2TMSG4 | R/W | 8 | Undefined |
| 002B | — | Message 5 | — | M2TMSG5 | R/W | 8 | Undefined |
| 002C | — | Message 6 | — | M2TMSG6 | R/W | 8 | Undefined |
| 002D | — | Message 7 | — | M2TMSG7 | R/W | 8 | Undefined |
| 002E∗ | CAN bus timing register 1 | | BTR1 | | R/W | 8 | 00 |
| 002F | Communication input/output control register | | TIOC | | R/W | 8 | 00 |
| 0030 | Message control register | | M3MCR | | R/W | 8 | 00 |
| 0031 | Identifier 0 | | M3IDR0 | | R/W | 8 | Undefined |
| 0032 | Identifier 1 | | M3IDR1 | | R/W | 8 | Undefined |
| 0033 | Message 0 | Identifier 2 | M3MSG0 | M3IDR2 | R/W | 8 | Undefined |
| 0034 | Message 1 | Identifier 3 | M3MSG1 | M3IDR3 | R/W | 8 | Undefined |
| 0035 (∗) | Message 2 | Identifier 4 | M3MSG2 | M3IDR4 | R/W | 8 | Undefined |
| 0036 | Message 3 | Message 0 | M3MSG3 | M3TMSG0 | R/W | 8 | Undefined |
| 0037 | Message 4 | Message 1 | M3MSG4 | M3TMSG1 | R/W | 8 | Undefined |
| 0038 | Message 5 | Message 2 | M3MSG5 | M3TMSG2 | R/W | 8 | Undefined |
| 0039 | Message 6 | Message 3 | M3MSG6 | M3TMSG3 | R/W | 8 | Undefined |
| 003A | Message 7 | Message 4 | M3MSG7 | M3TMSG4 | R/W | 8 | Undefined |
| 003B | — | Message 5 | — | M3TMSG5 | R/W | 8 | Undefined |
| 003C | — | Message 6 | — | M3TMSG6 | R/W | 8 | Undefined |
| 003D | — | Message 7 | — | M3TMSG7 | R/W | 8 | Undefined |
| 003E∗ | Group message register 0 | | GMR0 | | R/W | 8 | 00 |
| 003F∗ | Group message register 1 | | GMR1 | | R/W | 8 | 00 |

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.

A bit can be missing at address 00x5h only in the case of the extended format.

A dash "—" indicates that the memory space is not used in the case of the standard format.

A – 4

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 0040 | Message control register | | M4MCR | | R/W | 8 | 00 |
| 0041 | Identifier 0 | | M4IDR0 | | R/W | 8 | Undefined |
| 0042 | Identifier 1 | | M4IDR1 | | R/W | 8 | Undefined |
| 0043 | Message 0 | Identifier 2 | M4MSG0 | M4IDR2 | R/W | 8 | Undefined |
| 0044 | Message 1 | Identifier 3 | M4MSG1 | M4IDR3 | R/W | 8 | Undefined |
| 0045 (∗) | Message 2 | Identifier 4 | M4MSG2 | M4IDR4 | R/W | 8 | Undefined |
| 0046 | Message 3 | Message 0 | M4MSG3 | M4TMSG0 | R/W | 8 | Undefined |
| 0047 | Message 4 | Message 1 | M4MSG4 | M4TMSG1 | R/W | 8 | Undefined |
| 0048 | Message 5 | Message 2 | M4MSG5 | M4TMSG2 | R/W | 8 | Undefined |
| 0049 | Message 6 | Message 3 | M4MSG6 | M4TMSG3 | R/W | 8 | Undefined |
| 004A | Message 7 | Message 4 | M4MSG7 | M4TMSG4 | R/W | 8 | Undefined |
| 004B | — | Message 5 | — | M4TMSG5 | R/W | 8 | Undefined |
| 004C | — | Message 6 | — | M4TMSG6 | R/W | 8 | Undefined |
| 004D | — | Message 7 | — | M4TMSG7 | R/W | 8 | Undefined |
| 004E | Message mask register 00 | | GMSK00 | | R/W | 8 | 00 |
| 004F | Message mask register 01 | | GMSK01 | | R/W | 8 | 00 |
| 0050 | Message control register | | M5MCR | | R/W | 8 | 00 |
| 0051 | Identifier 0 | | M5IDR0 | | R/W | 8 | Undefined |
| 0052 | Identifier 1 | | M5IDR1 | | R/W | 8 | Undefined |
| 0053 | Message 0 | Identifier 2 | M5MSG0 | M5IDR2 | R/W | 8 | Undefined |
| 0054 | Message 1 | Identifier 3 | M5MSG1 | M5IDR3 | R/W | 8 | Undefined |
| 0055 (∗) | Message 2 | Identifier 4 | M5MSG2 | M5IDR4 | R/W | 8 | Undefined |
| 0056 | Message 3 | Message 0 | M5MSG3 | M5TMSG0 | R/W | 8 | Undefined |
| 0057 | Message 4 | Message 1 | M5MSG4 | M5TMSG1 | R/W | 8 | Undefined |
| 0058 | Message 5 | Message 2 | M5MSG5 | M5TMSG2 | R/W | 8 | Undefined |
| 0059 | Message 6 | Message 3 | M5MSG6 | M5TMSG3 | R/W | 8 | Undefined |
| 005A | Message 7 | Message 4 | M5MSG7 | M5TMSG4 | R/W | 8 | Undefined |
| 005B | — | Message 5 | — | M5TMSG5 | R/W | 8 | Undefined |
| 005C | — | Message 6 | — | M5TMSG6 | R/W | 8 | Undefined |
| 005D | — | Message 7 | — | M5TMSG7 | R/W | 8 | Undefined |
| 005E | Message mask register 02 | | GMSK02 | | R/W | 8 | 00 |
| 005F∗ | Message mask register 03 | | GMSK03 | | R/W | 8 | 00 |

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.

A bit can be missing at address 00x5h only in the case of the extended format.

A dash "—" indicates that the memory space is not used in the case of the standard format.

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 0060 | Message control register | | M6MCR | | R/W | 8 | 00 |
| 0061 | Identifier 0 | | M6IDR0 | | R/W | 8 | Undefined |
| 0062 | Identifier 1 | | M6IDR1 | | R/W | 8 | Undefined |
| 0063 | Message 0 | Identifier 2 | M6MSG0 | M6IDR2 | R/W | 8 | Undefined |
| 0064 | Message 1 | Identifier 3 | M6MSG1 | M6IDR3 | R/W | 8 | Undefined |
| 0065 (∗) | Message 2 | Identifier 4 | M6MSG2 | M6IDR4 | R/W | 8 | Undefined |
| 0066 | Message 3 | Message 0 | M6MSG3 | M6TMSG0 | R/W | 8 | Undefined |
| 0067 | Message 4 | Message 1 | M6MSG4 | M6TMSG1 | R/W | 8 | Undefined |
| 0068 | Message 5 | Message 2 | M6MSG5 | M6TMSG2 | R/W | 8 | Undefined |
| 0069 | Message 6 | Message 3 | M6MSG6 | M6TMSG3 | R/W | 8 | Undefined |
| 006A | Message 7 | Message 4 | M6MSG7 | M6TMSG4 | R/W | 8 | Undefined |
| 006B | — | Message 5 | — | M6TMSG5 | R/W | 8 | Undefined |
| 006C | — | Message 6 | — | M6TMSG6 | R/W | 8 | Undefined |
| 006D | — | Message 7 | — | M6TMSG7 | R/W | 8 | Undefined |
| 006E | Message mask register 10 | | GMSK10 | | R/W | 8 | 00 |
| 006F | Message mask register 11 | | GMSK11 | | R/W | 8 | 00 |
| 0070 | Message control register | | M7MCR | | R/W | 8 | 00 |
| 0071 | Identifier 0 | | M7IDR0 | | R/W | 8 | Undefined |
| 0072 | Identifier 1 | | M7IDR1 | | R/W | 8 | Undefined |
| 0073 | Message 0 | Identifier 2 | M7MSG0 | M7IDR2 | R/W | 8 | Undefined |
| 0074 | Message 1 | Identifier 3 | M7MSG1 | M7IDR3 | R/W | 8 | Undefined |
| 0075 (∗) | Message 2 | Identifier 4 | M7MSG2 | M7IDR4 | R/W | 8 | Undefined |
| 0076 | Message 3 | Message 0 | M7MSG3 | M7TMSG0 | R/W | 8 | Undefined |
| 0077 | Message 4 | Message 1 | M7MSG4 | M7TMSG1 | R/W | 8 | Undefined |
| 0078 | Message 5 | Message 2 | M7MSG5 | M7TMSG2 | R/W | 8 | Undefined |
| 0079 | Message 6 | Message 3 | M7MSG6 | M7TMSG3 | R/W | 8 | Undefined |
| 007A | Message 7 | Message 4 | M7MSG7 | M7TMSG4 | R/W | 8 | Undefined |
| 007B | — | Message 5 | — | M7TMSG5 | R/W | 8 | Undefined |
| 007C | — | Message 6 | — | M7TMSG6 | R/W | 8 | Undefined |
| 007D | — | Message 7 | — | M7TMSG7 | R/W | 8 | Undefined |
| 007E | Message mask register 12 | | GMSK12 | | R/W | 8 | 00 |
| 007F∗ | Message mask register 13 | | GMSK13 | | R/W | 8 | 00 |

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.

A bit can be missing at address 00x5h only in the case of the extended format.

A dash "—" indicates that the memory space is not used in the case of the standard format.

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 0080 | Message control register | | M8MCR | | R/W | 8 | 00 |
| 0081 | Identifier 0 | | M8IDR0 | | R/W | 8 | Undefined |
| 0082 | Identifier 1 | | M8IDR1 | | R/W | 8 | Undefined |
| 0083 | Message 0 | Identifier 2 | M8MSG0 | M8IDR2 | R/W | 8 | Undefined |
| 0084 | Message 1 | Identifier 3 | M8MSG1 | M8IDR3 | R/W | 8 | Undefined |
| 0085 (∗) | Message 2 | Identifier 4 | M8MSG2 | M8IDR4 | R/W | 8 | Undefined |
| 0086 | Message 3 | Message 0 | M8MSG3 | M8TMSG0 | R/W | 8 | Undefined |
| 0087 | Message 4 | Message 1 | M8MSG4 | M8TMSG1 | R/W | 8 | Undefined |
| 0088 | Message 5 | Message 2 | M8MSG5 | M8TMSG2 | R/W | 8 | Undefined |
| 0089 | Message 6 | Message 3 | M8MSG6 | M8TMSG3 | R/W | 8 | Undefined |
| 008A | Message 7 | Message 4 | M8MSG7 | M8TMSG4 | R/W | 8 | Undefined |
| 008B | — | Message 5 | — | M8TMSG5 | R/W | 8 | Undefined |
| 008C | — | Message 6 | — | M8TMSG6 | R/W | 8 | Undefined |
| 008D | — | Message 7 | — | M8TMSG7 | R/W | 8 | Undefined |
| 008E∗ | Standby control register | | STBY | | R/W | 8 | 00 |
| 008F∗ | CAN control register 2 | | CANC2 | | R/W | 8 | 00 |
| 0090 | Message control register | | M9MCR | | R/W | 8 | 00 |
| 0091 | Identifier 0 | | M9IDR0 | | R/W | 8 | Undefined |
| 0092 | Identifier 1 | | M9IDR1 | | R/W | 8 | Undefined |
| 0093 | Message 0 | Identifier 2 | M9MSG0 | M9IDR2 | R/W | 8 | Undefined |
| 0094 | Message 1 | Identifier 3 | M9MSG1 | M9IDR3 | R/W | 8 | Undefined |
| 0095 (∗) | Message 2 | Identifier 4 | M9MSG2 | M9IDR4 | R/W | 8 | Undefined |
| 0096 | Message 3 | Message 0 | M9MSG3 | M9TMSG0 | R/W | 8 | Undefined |
| 0097 | Message 4 | Message 1 | M9MSG4 | M9TMSG1 | R/W | 8 | Undefined |
| 0098 | Message 5 | Message 2 | M9MSG5 | M9TMSG2 | R/W | 8 | Undefined |
| 0099 | Message 6 | Message 3 | M9MSG6 | M9TMSG3 | R/W | 8 | Undefined |
| 009A | Message 7 | Message 4 | M9MSG7 | M9TMSG4 | R/W | 8 | Undefined |
| 009B | — | Message 5 | — | M9TMSG5 | R/W | 8 | Undefined |
| 009C | — | Message 6 | — | M9TMSG6 | R/W | 8 | Undefined |
| 009D | — | Message 7 | — | M9TMSG7 | R/W | 8 | Undefined |
| 009E∗ | Communication message box number register | | TMN | | R | 8 | (*1) |
| 009F∗ | CAN status register | | CANS | | R | 8 | 00 |

(*1) Upper 4 bits are undefined and lower 4 bits are set to "0000". Represented as "XXXX0000[b]" in binary notation.

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.
A bit can be missing at address 00x5h only in the case of the extended format.
A dash "—" indicates that the memory space is not used in the case of the standard format.

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 00A0 | Message control register | | M10MCR | | R/W | 8 | 00 |
| 00A1 | Identifier 0 | | M10IDR0 | | R/W | 8 | Undefined |
| 00A2 | Identifier 1 | | M10IDR1 | | R/W | 8 | Undefined |
| 00A3 | Message 0 | Identifier 2 | M10MSG0 | M10IDR2 | R/W | 8 | Undefined |
| 00A4 | Message 1 | Identifier 3 | M10MSG1 | M10IDR3 | R/W | 8 | Undefined |
| 00A5 (∗) | Message 2 | Identifier 4 | M10MSG2 | M10IDR4 | R/W | 8 | Undefined |
| 00A6 | Message 3 | Message 0 | M10MSG3 | M10TMSG0 | R/W | 8 | Undefined |
| 00A7 | Message 4 | Message 1 | M10MSG4 | M10TMSG1 | R/W | 8 | Undefined |
| 00A8 | Message 5 | Message 2 | M10MSG5 | M10TMSG2 | R/W | 8 | Undefined |
| 00A9 | Message 6 | Message 3 | M10MSG6 | M10TMSG3 | R/W | 8 | Undefined |
| 00AA | Message 7 | Message 4 | M10MSG7 | M10TMSG4 | R/W | 8 | Undefined |
| 00AB | — | Message 5 | — | M10TMSG5 | R/W | 8 | Undefined |
| 00AC | — | Message 6 | — | M10TMSG6 | R/W | 8 | Undefined |
| 00AD | — | Message 7 | — | M10TMSG7 | R/W | 8 | Undefined |
| 00AE | Transmission error counter | | TEC | | R | 8 | 00 |
| 00AF | Receive error counter | | REC | | R | 8 | 00 |
| 00B0 | Message control register | | M11MCR | | R/W | 8 | 00 |
| 00B1 | Identifier 0 | | M11IDR0 | | R/W | 8 | Undefined |
| 00B2 | Identifier 1 | | M11IDR1 | | R/W | 8 | Undefined |
| 00B3 | Message 0 | Identifier 2 | M11MSG0 | M11IDR2 | R/W | 8 | Undefined |
| 00B4 | Message 1 | Identifier 3 | M11MSG1 | M11IDR3 | R/W | 8 | Undefined |
| 00B5 (∗) | Message 2 | Identifier 4 | M11MSG2 | M11IDR4 | R/W | 8 | Undefined |
| 00B6 | Message 3 | Message 0 | M11MSG3 | M11TMSG0 | R/W | 8 | Undefined |
| 00B7 | Message 4 | Message 1 | M11MSG4 | M11TMSG1 | R/W | 8 | Undefined |
| 00B8 | Message 5 | Message 2 | M11MSG5 | M11TMSG2 | R/W | 8 | Undefined |
| 00B9 | Message 6 | Message 3 | M11MSG6 | M11TMSG3 | R/W | 8 | Undefined |
| 00BA | Message 7 | Message 4 | M11MSG7 | M11TMSG4 | R/W | 8 | Undefined |
| 00BB | — | Message 5 | — | M11TMSG5 | R/W | 8 | Undefined |
| 00BC | — | Message 6 | — | M11TMSG6 | R/W | 8 | Undefined |
| 00BD | — | Message 7 | — | M11TMSG7 | R/W | 8 | Undefined |
| 00BE | CAN status register 2 | | CANS2 | | R/W | 8 | — |
| 00BF | Bus OFF release counter | | BOCO | | R | 8 | 00 |

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.

A bit can be missing at address 00x5h only in the case of the extended format.

A dash "—" indicates that the memory space is not used in the case of the standard format.

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 00C0 | Message control register | | M12MCR | | R/W | 8 | 00 |
| 00C1 | Identifier 0 | | M12IDR0 | | R/W | 8 | Undefined |
| 00C2 | Identifier 1 | | M12IDR1 | | R/W | 8 | Undefined |
| 00C3 | Message 0 | Identifier 2 | M12MSG0 | M12IDR2 | R/W | 8 | Undefined |
| 00C4 | Message 1 | Identifier 3 | M12MSG1 | M12IDR3 | R/W | 8 | Undefined |
| 00C5 (∗) | Message 2 | Identifier 4 | M12MSG2 | M12IDR4 | R/W | 8 | Undefined |
| 00C6 | Message 3 | Message 0 | M12MSG3 | M12TMSG0 | R/W | 8 | Undefined |
| 00C7 | Message 4 | Message 1 | M12MSG4 | M12TMSG1 | R/W | 8 | Undefined |
| 00C8 | Message 5 | Message 2 | M12MSG5 | M12TMSG2 | R/W | 8 | Undefined |
| 00C9 | Message 6 | Message 3 | M12MSG6 | M12TMSG3 | R/W | 8 | Undefined |
| 00CA | Message 7 | Message 4 | M12MSG7 | M12TMSG4 | R/W | 8 | Undefined |
| 00CB | — | Message 5 | — | M12TMSG5 | R/W | 8 | Undefined |
| 00CC | — | Message 6 | — | M12TMSG6 | R/W | 8 | Undefined |
| 00CD | — | Message 7 | — | M12TMSG7 | R/W | 8 | Undefined |
| 00CE | Not used (reserved) | | — | | — | — | — |
| 00CF | Not used (reserved) | | — | | — | — | — |
| 00D0 | Message   control register | | M13MCR | | R/W | 8 | 00 |
| 00D1 | Identifier 0 | | M13IDR0 | | R/W | 8 | Undefined |
| 00D2 | Identifier 1 | | M13IDR1 | | R/W | 8 | Undefined |
| 00D3 | Message 0 | Identifier 2 | M13MSG0 | M13IDR2 | R/W | 8 | Undefined |
| 00D4 | Message 1 | Identifier 3 | M13MSG1 | M13IDR3 | R/W | 8 | Undefined |
| 00D5 (∗) | Message 2 | Identifier 4 | M13MSG2 | M13IDR4 | R/W | 8 | Undefined |
| 00D6 | Message 3 | Message 0 | M13MSG3 | M13TMSG0 | R/W | 8 | Undefined |
| 00D7 | Message 4 | Message 1 | M13MSG4 | M13TMSG1 | R/W | 8 | Undefined |
| 00D8 | Message 5 | Message 2 | M13MSG5 | M13TMSG2 | R/W | 8 | Undefined |
| 00D9 | Message 6 | Message 3 | M13MSG6 | M13TMSG3 | R/W | 8 | Undefined |
| 00DA | Message 7 | Message 4 | M13MSG7 | M13TMSG4 | R/W | 8 | Undefined |
| 00DB | — | Message 5 | — | M13TMSG5 | R/W | 8 | Undefined |
| 00DC | — | Message 6 | — | M13TMSG6 | R/W | 8 | Undefined |
| 00DD | — | Message 7 | — | M13TMSG7 | R/W | 8 | Undefined |
| 00DE | Not used (reserved) | | — | | — | — | — |
| 00DF | Not used (reserved) | | — | | — | — | — |

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.

A bit can be missing at address 00x5h only in the case of the extended format.

A dash "—" indicates that the memory space is not used in the case of the standard format.

| Address [H] | Name | | Abbreviated name | | R/W | Access | Value at reset [H] |
|---|---|---|---|---|---|---|---|
| | standard | extended | standard | extended | | | |
| 00E0 | Message control register | | M14MCR | | R/W | 8 | 00 |
| 00E1 | Identifier 0 | | M14IDR0 | | R/W | 8 | Undefined |
| 00E2 | Identifier 1 | | M14IDR1 | | R/W | 8 | Undefined |
| 00E3 | Message 0 | Identifier 2 | M14MSG0 | M14IDR2 | R/W | 8 | Undefined |
| 00E4 | Message 1 | Identifier 3 | M14MSG1 | M14IDR3 | R/W | 8 | Undefined |
| 00E5 (∗) | Message 2 | Identifier 4 | M14MSG2 | M14IDR4 | R/W | 8 | Undefined |
| 00E6 | Message 3 | Message 0 | M14MSG3 | M14TMSG0 | R/W | 8 | Undefined |
| 00E7 | Message 4 | Message 1 | M14MSG4 | M14TMSG1 | R/W | 8 | Undefined |
| 00E8 | Message 5 | Message 2 | M14MSG5 | M14TMSG2 | R/W | 8 | Undefined |
| 00E9 | Message 6 | Message 3 | M14MSG6 | M14TMSG3 | R/W | 8 | Undefined |
| 00EA | Message 7 | Message 4 | M14MSG7 | M14TMSG4 | R/W | 8 | Undefined |
| 00EB | — | Message 5 | — | M14TMSG5 | R/W | 8 | Undefined |
| 00EC | — | Message 6 | — | M14TMSG6 | R/W | 8 | Undefined |
| 00ED | — | Message 7 | — | M14TMSG7 | R/W | 8 | Undefined |
| 00EE | Not used (reserved) | | — | | — | — | — |
| 00EF | Not used (reserved) | | — | | — | — | — |
| 00F0 | Message control register | | M15MCR | | R/W | 8 | 00 |
| 00F1 | Identifier 0 | | M15IDR0 | | R/W | 8 | Undefined |
| 00F2 | Identifier 1 | | M15IDR1 | | R/W | 8 | Undefined |
| 00F3 | Message 0 | Identifier 2 | M15MSG0 | M15IDR2 | R/W | 8 | Undefined |
| 00F4 | Message 1 | Identifier 3 | M15MSG1 | M15IDR3 | R/W | 8 | Undefined |
| 00F5 (∗) | Message 2 | Identifier 4 | M15MSG2 | M15IDR4 | R/W | 8 | Undefined |
| 00F6 | Message 3 | Message 0 | M15MSG3 | M15TMSG0 | R/W | 8 | Undefined |
| 00F7 | Message 4 | Message 1 | M15MSG4 | M15TMSG1 | R/W | 8 | Undefined |
| 00F8 | Message 5 | Message 2 | M15MSG5 | M15TMSG2 | R/W | 8 | Undefined |
| 00F9 | Message 6 | Message 3 | M15MSG6 | M15TMSG3 | R/W | 8 | Undefined |
| 00FA | Message 7 | Message 4 | M15MSG7 | M15TMSG4 | R/W | 8 | Undefined |
| 00FB | — | Message 5 | — | M15TMSG5 | R/W | 8 | Undefined |
| 00FC | — | Message 6 | — | M15TMSG6 | R/W | 8 | Undefined |
| 00FD | — | Message 7 | — | M15TMSG7 | R/W | 8 | Undefined |
| 00FE | Not used (reserved) | | — | | — | — | — |
| 00FF | Not used (reserved) | | — | | — | — | — |

An asterisk (∗) in the address column indicates that there is a bit that is not present in that register.

A bit can be missing at address 00x5h only in the case of the extended format.

A dash "—" indicates that the memory space is not used in the case of the standard format.

# Revision History

R – 1

# Revision History

| Document No. | Date | Page | | Description |
|---|---|---|---|---|
| | | Previous Edition | Current Edition | |
| FEUL9225B-01 | May 2000 | – | – | First Edition. |
| FEUL9225B-02 | Sep. 2000 | – | – | Second Edition |
| FEUL9225B-03 | Feb. 2001 | – | – | Third Edition |
| FEUL9225B-04 | Jul. 2001 | – | – | Fourth Edition |
| FEUL9225B-05 | Jul. 1, 2002 | – | – | Fifth Edition |
| | | 2-18,19 | 2-18,19 | Deleted SYNC bit in CANC register. |
| | | 2-22 | 2-22 | Added notes on how to set the NMES register. |
| | | 2-27,28 | 2-27,28 | Changed resynchronization timing from edge to negative edge. |