# Overview

## Overview

This manual explains the I2C device driver in R-Car H3/M3/M3N/E3/D3/V3U/V3H Linux.

## Function

This module transmits/receives data to/from a device connected to the I2C interface on R-Car H3/M3/M3N/E3/D3/V3U/V3H.

### Driver Function

The following table lists the function of this module.

Table 1.2‑1 Driver Function

| **Function** | **Support status**  **(R-Car H3/M3/M3N/V3U)** | **Support status**  **(R-Car E3) \*1** | **Support status**  **(R-Car D3)** | **Support status**  **(R-Car V3H)** |
| --- | --- | --- | --- | --- |
| Number of channels | 7 | 8 | 4 | 6 |
| Channel | Ch0 ~ Ch6 | Ch0 ~ Ch6, Ch8 | Ch0 ~ Ch3 | Ch0 ~ Ch5 |
| Master Mode | Supported | | | |
| Slave Mode | Unsupported | | | |
| DMA function | Supported (Less than 8 bytes will operate as PIO) | | | |

(\*1) Ch7 for R-Car E3 is used in IIC-DVFS. Please refer to "Power Management user’s manual" for detail.

### Transfer Speed

The following table shows the transfer speed that this module supports.

Table 1.2‑2 Transfer speed (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| Interface mode | Real transfer speed | Support |
| --- | --- | --- |
| Standard mode (100KHz) | 100Kbit/s | yes |
| Fast mode (400KHz) | 400Kbit/s | yes |
| Fast mode plus (1MHz) | 1Mbit/s | yes \*1 |

(\*1) Only V3U is supported.

### Connected Device

This module connects the following device on R-Car H3-SiP/M3-SiP/M3N-SiP/E3/D3/V3U/V3H System Evaluation Board.

Table 1.2‑3 Connected device (R-Car H3/M3/M3N)

| Channel | Device | Category | Address | Remark |
| --- | --- | --- | --- | --- |
| I2C0 | EXIO CN B | Connector | - | - |
| I2C1 | EXIO CN A | Connector | - | - |
| I2C2 | AK4613 | SSI CODEC | 0b0010000x | - |
| CS2000 | CLK Synthesizer | 0b1001111x | Up to 100KHz |
| EtherAVB CN | Connector | - | - |
| EXIO CN B | Connector | - | - |
| I2C3 | EXIO CN D | Connector | - |  |
| I2C4 | PCA9654 | I/O Expander | 0b0100000x | - |
| 9FGV0841 | CLK Generator | 0b1101000x | - |
| ADV7482WBBCZ | VIDEO Decoder | 0b1110000x | - |
| MAX9611 | Current-Sense | 0b1111111x | - |
| MAX9611 | Current-Sense | 0b1111100x |  |
| 5P49V5923 | CLK Generator | 0b1101010x |  |
| EXIO CN C | Connector | - | - |
| I2C5 | Pad | Test point | - | - |
| I2C6 | EXIO CN D | Connector | - | - |

Table 1.2‑4 Connected device (R-Car E3)

| Channel | Device | Category | Address | Remark |
| --- | --- | --- | --- | --- |
| I2C0 | PCA9654 | I/O Expander | 0b0100000x | - |
| 9FGV0841 | CLK Generator | 0b1101000x | - |
| ADV7482WBBCZ | VIDEO Decoder | 0b1110000x | - |
| ADV7511WBSWZ | HDMI Transmitter | 0b0111001x | - |
| I2C1 | - | - | - | - |
| I2C2 | - | - | - | - |
| I2C3 | AK4613 | SSI CODEC | 0b0010000x | - |
| CS2000 | CLK Synthesizer | 0b1001111x | Up to 100KHz |
| EtherAVB CN | Connector | - | - |
| BtoB CN | Connector | - | - |
| TestIC | Test point | - | - |
| I2C4 | - | - | - | - |
| I2C5 | - | - | - | - |
| I2C6 | - | - | - | - |
| I2C8 | - | - | - | - |

Note: Ch7 is used in IIC-DVFS. Please refer to "Power Management user’s manual" for detail.

Table 1.2‑5 Connected device (R-Car D3)

| Channel | Device | Category | Address | Remark |
| --- | --- | --- | --- | --- |
| I2C0 | BR24T01FVM-W | EEPROM | 0b1010000x | - |
| ADV7612WBSWZ | HDMI receiver | 0b1001100x | - |
| ADV7180WBCP32Z | Video processor | 0b0100000x | - |
| CS2000 | Clock synthesizer | 0b1001111x | - |
| AK4613 | SSI Codec | 0b0010000x | - |
| ADV7511WBSWZ | HDMI transmitter | 0b0111001x | - |
| MLB CN | Connector | - | - |
| EXIO CN A | Connector | - | - |
| I2C1 | EtherAVB CN | Connector | - | - |

Table 1.2‑6 Connected device (R-Car V3U)

| Channel | Device | Category | Address | Remark |
| --- | --- | --- | --- | --- |
| I2C0 | PCA9654EDTR2G | I/O expander | 0b0100000x |  |
| 9FGV0841AKILF | CLK generator | 0b1101010x |  |
| 10M04SCM153I7G | MAX 10 FPGA | - |  |
| BR24G01FVM-3GTTR | EEPROM | 0b1010000x |  |
| 5P35023-618NLG2 | CLK generator | 0b1101001x |  |
| EXIO CN B | Connector | - |  |
| 5P49V60A554NLG2 | CLK generator | 0b1101000x |  |
| GPIO CN | Connector | - |  |
| BR24G01FVM-3GTTR | EEPROM | 0b1010001x |  |
| Ether | Connector | - |  |
| BR24G01FVM-3GTTR | EEPROM | 0b1010011x |  |
| CSI\_DSI | Connector | - |  |
| PCA9654EDTR2G | I/O expander | 0b0100111x |  |
| PCA9654EDTR2G | I/O expander | 0b0100011x |  |
| PCA9654EDTR2G | I/O expander | 0b0100010x |  |
| PCA9654EDTR2G | I/O expander | 0b0100011x |  |
| I2C1 | SN65DSI86ZQER | DSI to eDP | 0b0101100x |  |
| EXIO CN B | Connector | - |  |
| GPIO CN | Connector | - |  |
| CSI\_DSI | Connector | - |  |
| MAX20087ATPA/VY+ | Power | 0b0101000x |  |
| MAX20087ATPA/VY+ | Power | 0b0101010x |  |
| MAX20087ATPA/VY+ | Power | 0b0101101x |  |
| MAX96789GTN/V+ | Serializer | 0b1000000x |  |
| MAX96712GTB/VY+ | DeSerializer | 0b1001001x |  |
| MAX96712GTB/VY+ | DeSerializer | 0b1001011x |  |
| MAX96712GTB/VY+ | DeSerializer | 0b1101011x |  |

Table 1.2‑6 Connected device (R-Car V3U) (Cont.)

| I2C2 | OcuLink connector | Connector | - |  |
| --- | --- | --- | --- | --- |
| EXIO CN B | Connector | - |  |
| GPIO CN | Connector | - |  |
| I2C3 | OcuLink connector | Connector | - |  |
| EXIO CN B | Connector | - |  |
| GPIO CN | Connector | - |  |
| I2C4 | EXIO CN B | Connector | - |  |
| GPIO CN | Connector | - |  |
| I2C5 | EXIO CN B | Connector | - |  |
| GPIO CN | Connector |  |  |
| I2C6 | RAA271010 | PWM controller | 0b1100100x | - |
| RAA271001 | PMIC | 0b1010100x |  |
|  |  | 0b1010101x |  |
| RAA271001 | PMIC | 0b1011100x |  |
|  |  | 0b1011101x |  |
| R5F10BGGLFB | Micro controller |  |  |

Table 1.2‑7 Connected device (R-Car V3H)

| Channel | Device | Category | Address | Remark |
| --- | --- | --- | --- | --- |
| I2C0 | PCA9654E | I/O expander | 0b0100000x | - |
| PCA9654E | I/O expander | 0b0100001x | - |
| ADV7511W | HDMI Transmitter | 0b0111001x | - |
| 5P35023 | Clock generator | 0b1101000x | - |
| RAA271000 | PMIC | 0b0011101x | - |
| 0b0011110x | - |
| EthernetAVB PHY connector | Connector | - | - |
| EXIO CN A | Connector- CN15 | - | - |
| EXIO CN C | Connector- CN17 | - | - |
| I2C1 | MAX9286 | CSI-2 decoder | 0b1001000x | - |
| MAX9286 | CSI-2 decoder | 0b1001010x | - |
| EXIO CN A | Connector- CN15 | - | - |
| EXIO CN B | Connector- CN16 | - | - |
| I2C2 | EXIO CN A | Connector- CN15 | - | - |
| I2C3 | EXIO CN A | Connector- CN15 | - | - |
| I2C4 | EXIO CN C | Connector- CN17 | - | - |
| I2C5 | EXIO CN A | Connector- CN15 | - | - |

## Reference

### Standards

The following table shows the standard that this module corresponds.

Table 1.3‑1 Standard (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Reference Number** | **Issue** | **Title** | **Edition** | **Date** |
| --- | --- | --- | --- | --- |
| - | Philips Semiconductors | THE I2C-BUS SPECIFICATION | 2.1 | Jan. 2000 |
| - | SBS Implementers Forum | System Management Bus (SMBus) Specification | Version 2.0 | Aug. 03, 2000 |

### Related Documents

The following table shows the document related to this module.

Table 1.3‑2 Related documents (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Number** | **Issue** | **Title** | **Edition** | **Date** |
| --- | --- | --- | --- | --- |
| - | Renesas Electronics | R-Car Series, 3rd Generation User’s Manual: Hardware | Rev.2.20 | Jun. 30, 2020 |
| - | Renesas Electronics | R-CarH3-SiP System Evaluation Board  Salvator-X Hardware Manual  RTP0RC7795SIPB0011S | Rev.1.09 | May. 11, 2017 |
| - | Renesas  Electronics | R-CarM3-SiP System Evaluation Board  Salvator-X Hardware Manual  RTP0RC7796SIPB0011S | Rev.0.04 | Oct. 3.2016 |
| - | Renesas  Electronics | R-CarH3-SiP/M3-SiP/M3N-SiP System Evaluation Board  Salvator-XS Hardware Manual | Rev.2.04 | Jul. 17, 2018 |
| - | Renesas  Electronics | R-CarE3 System Evaluation Board Ebisu Hardware Manual RTP0RC77990SEB0010S | Rev.0.03 | Apr. 11, 2018 |
| - | Renesas  Electronics | R-CarE3 System Evaluation Board Ebisu-4D (E3 board 4xDRAM) Hardware Manual | Rev.1.01 | Jul. 19, 2018 |
| - | Renesas  Electronics | R-CarD3 System Evaluation Board Hardware Manual RTP0RC77995SEB0010S | Rev.1.20 | Jul. 25, 2017 |
| - | Renesas  Electronics | R-Car V3U Series User's Manual | Rev.0.5 | Jul. 31, 2020 |
| - | Renesas  Electronics | R-CarV3U System Evaluation Falcon Hardware Manual | Rev.0.01 | Sep. 11, 2020 |
| - | Renesas Electronics | R-Car V3H\_2, Additional Document for User’s Manual: Hardware | Rev.0.50 | Jul. 31, 2020 |
| - | Renesas  Electronics | R-CarV3H System Evaluation Board Condor-I Hardware Manual | Rev.0.02 | Nov. 11,2019 |

## Restrictions

There is no restriction in this module.

## Notice

* This module supports the ch2 and ch4 of R-Car H3/M3/M3N on Evaluation board.
* This module supports the ch0 and ch3 of R-Car E3 on Evaluation board.
* This module supports the ch0 and ch1 of R-Car D3 on Evaluation board.
* This module supports the ch0 and ch1 of R-Car V3H on Evaluation board.
* Master transfer support. Slave transfer unsupported.
* Usage note for DMA mode of Receive Operation.

If use DMA mode of Receive Operation more than once with repeated START, issue "STOP condition" and start from beginning of transmission and reception procedure instead of repeated START.

# Terminology

The following table shows the terminology related to this module.

Table 1.5‑1 Terminology

| **Terms** | **Explanation** |
| --- | --- |
| I2C | Inter-Integrated Circuit |

# Operating Environment

## Hardware Environment

The following table lists the hardware needed to use this module.

Table 3.1‑1 Hardware specification (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Name** | **Version** | **Manufacture** |
| --- | --- | --- |
| R-CarH3-SiP System Evaluation Board Salvator-X | - | Renesas Electronics |
| R-CarM3-SiP System Evaluation Board Salvator-X | - | Renesas Electronics |
| R-CarH3-SiP/M3-SiP/M3N-SiP System Evaluation Board Salvator-XS | - | Renesas Electronics |
| R-CarE3 System Evaluation Board Ebisu | - | Renesas Electronics |
| R-CarE3 System Evaluation Board Ebisu-4D | - | Renesas Electronics |
| R-CarD3 System Evaluation Board Draak | - | Renesas Electronics |
| R-CarV3U System Evaluation Board Falcon | - | Renesas Electronics |
| R-CarV3H System Evaluation Board Condor-I | - | Renesas Electronics |

## Module Configuration

The following figure shows the configuration of this module.

In the case of R-Car H3-Sip/M3-Sip/M3N System Evaluation Board, you can control the connected device using the /dev/i2c-2 or /dev/i2c-4.

Application

/dev/i2c-0

Kernel Module

**user mode**

MAX9611 x2

9FGV0841

PCA9654

Standard I2C Device Interface

**This module**

I2C Interface

I2C2

I2C4

AK4613

CS2000

**kernel mode**

**hardware**

/dev/i2c-1

/dev/i2c-2

/dev/i2c-3

Standard I2C Core Interface

ADV7482WBBCZ

I2C5 ~ I2C6

5P49V5923

/dev/i2c-5

/dev/i2c-6

/dev/i2c-4

I2C0 ~ I2C1

I2C3

Figure 3‑1 I2C Driver Module configuration (R-Car H3/M3/M3N)

In the case of R-Car E3 System Evaluation Board, you can control the connected device using the /dev/i2c-0 or /dev/i2c-3.

Application

/dev/i2c-0

Kernel Module

**user mode**

9FGV0841

PCA9654

Standard I2C Device Interface

**This module**

I2C Interface

I2C1 ~ I2C2

I2C3

AK4613

CS2000

**kernel mode**

**hardware**

Standard I2C Core Interface

ADV7482WBBCZ

I2C4 ~ I2C6, I2C7

I2C0

/dev/i2c-1

/dev/i2c-2

/dev/i2c-3

/dev/i2c-4

/dev/i2c-5

/dev/i2c-6

/dev/i2c-7

ADV7511WBSWZ

Figure 3‑2 I2C Driver Module configuration (R-Car E3)

In the case of R-Car D3 System Evaluation Board, you can control the connected device using the /dev/i2c-0.

Application

Kernel Module

**user mode**

ADV7180WBCP32Z

CS2000

Standard I2C Device Interface

**This module**

I2C Interface

I2C0

BW24T01FWM-W

ADV7612WBSWZ

**kernel mode**

**hardware**

/dev/i2c-0

/dev/i2c-1

Standard I2C Core Interface

ADV7511WBSWZ

I2C1 ~ I2C3

AK4613

/dev/i2c-3

/dev/i2c-2

Figure 3‑3 I2C Driver Module configuration (R-Car D3)

In the case of R-Car V3U System Evaluation Board, you can control the connected device using the /dev/i2c-0 or /dev/i2c-1

Application

/dev/i2c-0

Kernel Module

**user mode**

PCA9654

Standard I2C Device Interface

**This module**

I2C Interface

I2C1

**kernel mode**

**hardware**

Standard I2C Core Interface

MAX96712

I2C2 ~ I2C6

I2C0

/dev/i2c-1

/dev/i2c-2

/dev/i2c-3

/dev/i2c-4

/dev/i2c-5

/dev/i2c-6

SN65DSI86

Figure 3‑4 I2C Driver Module configuration (R-Car V3U)

In the case of R-Car V3H System Evaluation Board, you can control the connected device using the /dev/i2c-0 or /dev/i2c-1

Application

/dev/i2c-0

Kernel Module

**user mode**

PCA9654E

Standard I2C Device Interface

**This module**

I2C Interface

I2C1

**kernel mode**

**hardware**

Standard I2C Core Interface

I2C2 ~ I2C5

I2C0

/dev/i2c-1

/dev/i2c-2

/dev/i2c-3

/dev/i2c-4

/dev/i2c-5

PCA9654E

ADV7511W

5P35023

RAA271000

MAX9286

MAX9286

Figure 3.5 I2C Driver Module configuration (R-Car V3H)

## State Transition Diagram

There is no state transition diagram for this module.

# External Interface

## Device Node

The following table shows the device node of this module.

Table 4.1‑1 I2C device node (R-Car H3/M3/M3N)

| **Channel** | **Device node** | **Major number** | **Minor number** |
| --- | --- | --- | --- |
| I2C2 | /dev/i2c-2 | 89 | 2 |
| I2C4 | /dev/i2c-4 | 89 | 4 |

Table 4.1‑2 I2C device node (R-Car E3)

| **Channel** | **Device node** | **Major number** | **Minor number** |
| --- | --- | --- | --- |
| I2C0 | /dev/i2c-0 | 89 | 0 |
| I2C3 | /dev/i2c-3 | 89 | 3 |

Table 4.1‑3 I2C device node (R-Car D3)

| **Channel** | **Device node** | **Major number** | **Minor number** |
| --- | --- | --- | --- |
| I2C0 | /dev/i2c-0 | 89 | 0 |
| I2C1 | /dev/i2c-1 | 89 | 1 |

Table 4.1‑4 I2C device node (R-Car V3U)

| **Channel** | **Device node** | **Major number** | **Minor number** |
| --- | --- | --- | --- |
| I2C0 | /dev/i2c-0 | 89 | 0 |
| I2C1 | /dev/i2c-1 | 89 | 1 |

Table 4.1‑5 I2C device node (R-Car V3H)

| **Channel** | **Device node** | **Major number** | **Minor number** |
| --- | --- | --- | --- |
| I2C0 | /dev/i2c-0 | 89 | 0 |
| I2C1 | /dev/i2c-1 | 89 | 1 |

## External Function

This section explains in the following format about the functions this module supplies.

|  |  |
| --- | --- |
| [Overview] | Presents an overview of a function. |
|  |  |
| [Function Name] | Explains the name of the function. |
|  |  |
| [Calling format] | Explains the format for calling the function. |
|  |  |
| [Argument] | Explains the argument(s) of the function. |
|  |  |
| [Return value] | Explains the return value(s) of the function. |
|  |  |
| [Error number] | Explains the error number(s) of the function. |
|  |  |
| [Feature] | Explains the features of the function. |
|  |  |
| [Remark] | Explains points to be noted when using the function. |
|  |  |

The following table lists the interface functions in this module, and Standard I2C core Interface.

Table 4.2‑1 System calls (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Chapter** | **Function name** | **Description** |
| --- | --- | --- |
| 4.2.1 | open | Open I2C. |
| 4.2.2 | close | Close I2C. |
| 4.2.3 | read | Read I2C (8bit data is received). |
| 4.2.4 | write | Write I2C (8bit data is sent). |
| 4.2.5 | ioctl(I2C\_RDWR) | Read/Write processing is performed. |
| 4.2.6 | ioctl(I2C\_FUNCS) | Return the functionality. |
| 4.2.7 | ioctl(I2C\_SLAVE) | The slave address is changed. |
| 4.2.8 | ioctl(I2C\_SMBUS) | It receives and transmits data by SMBus system. |

Table 4.2‑2 Standard I2C device interface (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Chapter** | **Function name** | **Description** |
| --- | --- | --- |
| 4.2.9 | i2c\_new\_probed\_device | Probe function. |
| 4.2.10 | i2c\_register\_board\_info | Register Board Information. |
| 4.2.11 | i2c\_add\_driver | Addition of I2C client driver. |
| 4.2.12 | i2c\_del\_driver | Deletion of I2C client driver. |
| 4.2.13 | i2c\_master\_send | Single(8bit) data transfer (send). |
| 4.2.14 | i2c\_master\_recv | Single(8bit) data transfer (receive). |
| 4.2.15 | i2c\_transfer | Execute a single or combined I2C message. |
| 4.2.16 | i2c\_get\_functionality | Return the functionality. |
| 4.2.17 | i2c\_smbus\_read\_byte | SMBus "receive byte" protocol. |
| 4.2.18 | i2c\_smbus\_write\_byte | SMBus "send byte" protocol. |
| 4.2.19 | i2c\_smbus\_read\_byte\_data | SMBus "read byte" protocol. |
| 4.2.20 | i2c\_smbus\_write\_byte\_data | SMBus "write byte" protocol. |
| 4.2.21 | i2c\_smbus\_read\_word\_data | SMBus "read word(16bit)" protocol. |
| 4.2.22 | i2c\_smbus\_write\_word\_data | SMBus "write word(16bit)" protocol. |
| 4.2.23 | i2c\_smbus\_read\_block\_data | SMBus "block read" protocol. |
| 4.2.24 | i2c\_smbus\_write\_block\_data | SMBus "block write" protocol. |
| 4.2.25 | i2c\_smbus\_read\_i2c\_block\_data | SMBus "read block byte" protocol. |
| 4.2.26 | i2c\_smbus\_write\_i2c\_block\_data | SMBus "write block byte" protocol. |

### open

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Open I2C | | |
|  |  | | |
| [Function Name] | open | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | #include <linux/i2c-dev.h> | | |
|  | int open( const char \*device\_name, int flags); | | |
|  |  | | |
| [Arguments] | device\_name | | Device name ( /dev/i2c-X) to open. |
|  | flags | | Open mode. |
|  |  | | |
| [Returns] | 0 | Success | |
|  | -1 | Error | |
|  |  | | |
| [Feature] | Open I2C device. | | |
|  |  | | |
| [Remark] | - | | |
|  |  | | |

### close

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Close I2C | | |
|  |  | | |
| [Function Name] | close | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | #include <linux/i2c-dev.h> | | |
|  | int close(int fd); | | |
|  |  | | |
| [Arguments] | fd | | File descriptor |
|  |  | | |
| [Returns] | 0 | Success | |
|  | -1 | Error | |
|  |  | | |
| [Feature] | Close I2C device. | | |
|  |  | | |
| [Remark] | - | | |
|  |  | | |

### read

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Read I2C(8bit data) | | |
|  |  | | |
| [Function Name] | read | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | #include <linux/i2c-dev.h> | | |
|  | ssize\_t read(int fd, void \*buf, size\_t count); | | |
|  |  | | |
| [Arguments] | fd | | File descriptor |
|  | buf | | Read data stock area |
|  | count | | Read size |
|  |  | | |
| [Returns] | Positive value | Success (Read size) | |
|  | -1 | Error | |
|  |  | | |
| [Feature] | Read I2C(8bit data). | | |
|  |  | | |
| [Remark] | - | | |
|  |  | | |

### write

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Write I2C(8bit data) | | |
|  |  | | |
| [Function Name] | write | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | #include <linux/i2c-dev.h> | | |
|  | ssize\_t write(int fd, const void \*buf, size\_t count); | | |
|  |  | | |
| [Arguments] | fd | | File descriptor |
|  | buf | | Write data stock area |
|  | count | | Write size |
|  |  | | |
| [Returns] | Positive value | Success (Write size) | |
|  | -1 | Error | |
|  |  | | |
| [Feature] | Write I2C(8bit data). | | |
|  |  | | |
| [Remark] | - | | |
|  |  | | |

### ioctl(I2C\_RDWR)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [Overview] | Read/Write transaction | | | |
|  |  | | | |
| [Function Name] | ioctl(I2C\_RDWR) | | | |
|  |  | | | |
| [Calling format] | #include <linux/i2c.h> | | | |
|  | #include <linux/i2c-dev.h> | | | |
|  | int ioctl(int fd, I2C\_RDWR, struct i2c\_rdwr\_ioctl\_data \*msgset); | | | |
|  |  | | | |
| [Arguments] | fd | | File descriptor | |
|  | I2C\_RDWR | | Fixed control | |
|  | msgset | | Send/Receive data | |
|  |  | | | |
| [Returns] | 0 | Success | | |
|  | -1 | Error | | |
|  |  | | | |
| [Error number] | -EFAULT | | | Bad address |
|  | -EINVAL | | | Invalid argument |
|  | -ENOMEM | | | There is not enough memory area |
|  | -EAGAIN | | | Arbitration lost occurs on I2C bus |
|  | -EOPNOTSUPP | | | Operation not supported |
|  | -ETIMEDOUT | | | Timeout occurs during I2C device access |
|  | -ENXIO | | | NACK occurs on I2C bus |
|  | -EBUSY | | | Busy on I2C bus |
|  |  | | | |
| [Feature] | Do read/write transaction without stop. | | | |
|  | Only valid if the adapter has I2C\_FUNC\_I2C. | | | |
|  |  | | | |
| [Remark] | Refer 4.3.6 about i2c\_rdwr\_ioctl\_data structure. | | | |
|  |  | | | |

### ioctl(I2C\_FUNCS)

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Return the functionality | | |
|  |  | | |
| [Function Name] | ioctl(I2C\_FUNCS) | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | #include <linux/i2c-dev.h> | | |
|  | int ioctl(int fd, I2C\_FUNCS, unsigned long \*funcs); | | |
|  |  | | |
| [Arguments] | fd | | File descriptor |
|  | I2C\_FUNCS | | Fixed control |
|  | funcs | | The index of function support |
|  |  | | |
| [Returns] | 0 | Success | |
|  | -1 | Error | |
|  |  | | |
| [Error number] | -EFAULT | Bad address | |
|  |  | | |
| [Feature] | This function returns whether the function is supported (the command). | | |
|  | Refer to 4.2.16 in detail. | | |
|  |  | | |
| [Remark] | - | | |
|  |  | | |

### ioctl(I2C\_SLAVE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [Overview] | Change the slave address | | | |
|  |  | | | |
| [Function Name] | ioctl(I2C\_SLAVE) | | | |
|  |  | | | |
| [Calling format] | #include <linux/i2c.h> | | | |
|  | #include <linux/i2c-dev.h> | | | |
|  | int ioctl(int fd, I2C\_SLAVE, long slave); | | | |
|  |  | | | |
| [Arguments] | fd | | File descriptor | |
|  | I2C\_SLAVE | | Fixed control | |
|  | slave | | Slave address | |
|  |  | | | |
| [Returns] | 0 | Success | | |
|  | -1 | Error | | |
|  |  | | | |
| [Error number] | -EINVAL | | | Invalid argument |
|  | -EBUSY | | | Busy on I2C bus |
|  |  | | | |
| [Feature] | Change the slave address. | | | |
|  |  | | | |
| [Remark] | - | | | |
|  |  | | | |

### ioctl(I2C\_SMBUS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [Overview] | Send/Receive data by SMBus system | | | |
|  |  | | | |
| [Function Name] | ioctl(I2C\_SMBUS) | | | |
|  |  | | | |
| [Calling format] | #include <linux/i2c.h> | | | |
|  | #include <linux/i2c-dev.h> | | | |
|  | int ioctl(int fd, I2C\_SMBUS, struct i2c\_smbus\_ioctl\_data \*args); | | | |
|  |  | | | |
| [Arguments] | fd | | File descriptor | |
|  | I2C\_SMBUS | | Fixed control | |
|  | args | | Send/Receive data. | |
|  |  | | | |
| [Returns] | 0 | Success | | |
|  | -1 | Error | | |
|  |  | | | |
| [Error number] | -EFAULT | | | Bad address |
|  | -EINVAL | | | Invalid argument |
|  | -EAGAIN | | | Arbitration lost occurs on I2C bus |
|  | -ETIMEDOUT | | | Timeout occurs during I2C device access |
|  | -ENXIO | | | NACK occurs on I2C bus |
|  | -EBUSY | | | Busy on I2C bus |
|  |  | | | |
| [Feature] | SMBus transfer | | | |
|  |  | | | |
| [Remark] | Refer to 4.3.7 about i2c\_smbus\_ioctl\_data structure. | | | |
|  |  | | | |

### i2c\_new\_probed\_device

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Probe function | | |
|  |  | | |
| [Function Name] | i2c\_new\_probed\_device | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | struct i2c\_client \* i2c\_new\_probed\_device(struct i2c\_adapter \*adap, | | |
|  | struct i2c\_board\_info \*info, unsigned short const \*addr\_list) | | |
|  |  | | |
| [Arguments] | adap | Adaptor information. | |
|  | info | Board information. | |
|  | addr\_list | Address list to probe. | |
|  |  | | |
| [Returns] | Positive Number | | Success (The pointer to client information) |
|  | NULL | | Error |
|  |  | | |
| [Feature] | The probe process for the connected device. | | |
|  |  | | |
| [Remark] | Refer to the following chapter about i2c\_client､i2c\_adapter､i2c\_board\_info structure. | | |
|  | 4.3.1 struct i2c\_adapter | | |
|  | 4.3.2 struct i2c\_client | | |
|  | 4.3.3 struct i2c\_board\_info | | |
|  |  | | |

### i2c\_register\_board\_info

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Register board information | | |
|  |  | | |
| [Function Name] | i2c\_register\_board\_info | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | int \_\_init i2c\_register\_board\_info(int busnum, | | |
|  | struct i2c\_board\_info const \*info, unsigned len); | | |
|  |  | | |
| [Arguments] | busnum | A number of the bus connected device. | |
|  | info | The vector of I2C device descriptor. | |
|  | len | The number of descriptor included vector. | |
|  |  | | |
| [Returns] | Positive Number | | Success (A number of the registered board information) |
|  | Negative number | | Error |
|  |  | | |
| [Feature] | Register board information. | | |
|  |  | | |
| [Remark] | This function supports the board dependent part call only. | | |
|  | Refer to 4.3.3 about i2c\_board\_info structure. | | |
|  |  | | |

### i2c\_add\_driver

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Addition of I2C client driver | | |
|  |  | | |
| [Function Name] | i2c\_add\_driver | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | int i2c\_add\_driver(struct i2c\_driver \*driver); | | |
|  |  | | |
| [Arguments] | driver | The information of client driver. | |
|  |  | | |
| [Returns] | 0 | | Success |
|  | Other value | | Error |
|  |  | | |
| [Feature] | Addition of I2C client driver. | | |
|  |  | | |
| [Remark] | Refer to 4.3.4 about i2c\_driver structure. | | |
|  |  | | |

### i2c\_del\_driver

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Deletion of I2C client driver | | |
|  |  | | |
| [Function Name] | i2c\_del\_driver | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | void i2c\_del\_driver(struct i2c\_driver \*driver); | | |
|  |  | | |
| [Arguments] | driver | | The information of client driver. |
|  |  | | |
| [Returns] | - |  | |
|  |  | | |
| [Feature] | Deletion of I2C client driver. | | |
|  |  | | |
| [Remark] | Refer to 4.3.4 about i2c\_driver structure. | | |
|  |  | | |

### i2c\_master\_send

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Single(8bit) data transfer in master transmit mode | | |
|  |  | | |
| [Function Name] | i2c\_master\_send | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | int i2c\_master\_send(struct i2c\_client \*client, const char \*buf, int count); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | buf | The data area sent to slave device. | |
|  | count | Sent data size. | |
|  |  | | |
| [Returns] | Positive Number | | Success (Sent data size) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | Single(8bit) data transfer in master transmit mode. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | 4.3.2 struct i2c\_client | | |
|  |  | | |

### i2c\_master\_recv

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Single(8bit) data transfer in master receive mode | | |
|  |  | | |
| [Function Name] | i2c\_master\_recv | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | int i2c\_master\_recv(struct i2c\_client \*client, char \*buf, int count); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | buf | The data area received from slave device. | |
|  | count | Received data size. | |
|  |  | | |
| [Returns] | Positive Number | | Success (Received data size) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | Single(8bit) data transfer in master receive mode | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_transfer

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Execute a single or combined I2C message | | |
|  |  | | |
| [Function Name] | i2c\_transfer | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | int i2c\_transfer(struct i2c\_adapter \* adap, struct i2c\_msg \*msgs, int num); | | |
|  |  | | |
| [Arguments] | adap | The information of I2C bus | |
|  | msgs | One or more messages which will be performed before operation stops | |
|  |  | by STOP. | |
|  |  | Each message begins with a START. | |
|  | num | A number of messages to be executed. | |
|  |  | | |
| [Returns] | Positive Number | | Success (A number of transmitted messages) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | Execute a single or combined I2C message. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to the following chapter about i2c\_adapter and i2c\_msg structure. | | |
|  | 4.3.1 struct i2c\_adapter | | |
|  | 4.3.5 struct i2c\_msg | | |
|  |  | | |

### i2c\_get\_functionality

|  |  |  |
| --- | --- | --- |
| [Overview] | Return the functionality | |
|  |  | |
| [Function Name] | i2c\_get\_functionality | |
|  |  | |
| [Calling format] | #include <linux/i2c.h> | |
|  | u32 i2c\_get\_functionality(struct i2c\_adapter \*adap); | |
|  |  | |
| [Arguments] | adap | The information of I2C bus |
|  |  | |
| [Returns] | The support information of function (Refer to Table 4.4 in detail) | |
|  | In the "AND" result of the following bit mask to the return value, "0" is unsupported | |
|  | function, "1" is support function. | |
|  |  | |
| [Feature] | Return the functionality. | |
|  |  | |
| [Remark] | This function cannot be called from the interrupt handler. | |
|  | Refer to 4.3.1 about i2c\_adapter structure. | |
|  |  | |

Table 4.2‑3 The support information of function (R-Car H3/M3/M3N/E3/D3/V3U/V3H) (1)

| **Bit mask** | **Function name** |
| --- | --- |
| **Function** |
| 0x00000001 | I2C\_FUNC\_I2C |
| Plane I2C command (Not execute in SMBus adapter) |
| 0x00000002 | I2C\_FUNC\_10BIT\_ADDR |
| Extend 10 bits address |
| 0x00000004 | I2C\_FUNC\_PROTOCOL\_MANGLING |
| Identify the flag of I2C\_M\_REV\_DIR\_ADDR, I2C\_M\_REV\_DIR\_ADDR andI2C\_M\_REV\_DIR\_NOSTART. |
| 0x00000008 | I2C\_FUNC\_SMBUS\_PEC |
| SMBus pec command |
| 0x00008000 | I2C\_FUNC\_SMBUS\_BLOCK\_PROC\_CALL |
| SMBus block\_process\_call command |
| 0x00010000 | I2C\_FUNC\_SMBUS\_QUICK |
| SMBus write\_quick command |
| 0x00020000 | I2C\_FUNC\_SMBUS\_READ\_BYTE |
| SMBus read\_byte command |
| 0x00040000 | I2C\_FUNC\_SMBUS\_WRITE\_BYTE |
| SMBus write\_byte command |
| 0x00080000 | I2C\_FUNC\_SMBUS\_READ\_BYTE\_DATA |
| SMBus read\_byte\_data command |
| 0x00100000 | I2C\_FUNC\_SMBUS\_WRITE\_BYTE\_DATA |
| SMBus write\_byte\_data command |
| 0x00200000 | I2C\_FUNC\_SMBUS\_READ\_WORD\_DATA |
| SMBus read\_word\_data command |
| 0x00400000 | I2C\_FUNC\_SMBUS\_WRITE\_WORD\_DATA |
| SMBus write\_byte\_data command |
| 0x00800000 | I2C\_FUNC\_SMBUS\_PROC\_CALL |
| SMBus process\_call command |
| 0x01000000 | I2C\_FUNC\_SMBUS\_READ\_BLOCK\_DATA |
| SMBus read\_block\_data command |
| 0x02000000 | I2C\_FUNC\_SMBUS\_WRITE\_BLOCK\_DATA |
| SMBus write\_block\_data command |
| 0x04000000 | I2C\_FUNC\_SMBUS\_READ\_I2C\_BLOCK |
| SMBus read\_i2c\_block\_data command |
| 0x08000000 | I2C\_FUNC\_SMBUS\_WRITE\_I2C\_BLOCK |
| SMBus write\_i2c\_block\_data command |
| 0x00060000 | I2C\_FUNC\_SMBUS\_BYTE |
| SMBus read\_byte and write\_byte command |
| 0x00180000 | I2C\_FUNC\_SMBUS\_BYTE\_DATA |
| SMBus read\_byte\_data and write\_byte\_data command |

Table 4.2‑4 The support information of function (R-Car H3/M3/M3N/E3/D3/V3U/V3H) (2)

| **Bit mask** | **Function name** |
| --- | --- |
| **Function** |
| 0x00600000 | I2C\_FUNC\_SMBUS\_WORD\_DATA |
| SMBus read\_word\_data and write\_word\_data command |
| 0x03000000 | I2C\_FUNC\_SMBUS\_BLOCK\_DATA |
| SMBus read\_block\_data and write\_block\_data command |
| 0x0C000000 | I2C\_FUNC\_SMBUS\_I2C\_BLOCK |
| SMBus read\_i2c\_block\_data and write\_i2c\_block\_data command |
| 0x02FF0008 | I2C\_FUNC\_SMBUS\_EMUL |
| All SMBus commands that can be emulated by a real I2C adapter (using the transparent emulation layer) |

### i2c\_smbus\_read\_byte

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "receive byte" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_read\_byte | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_read\_byte(struct i2c\_client \*client); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  |  | | |
| [Returns] | Positive Number | | Success (Received data size) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | SMBus "receive byte" protocol. | | |
|  | Without specifying a register of a device, this function is received 8 bits. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_write\_byte

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | Single(8bit) "send byte" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_write\_byte | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_write\_byte(struct i2c\_client \*client, u8 value); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | value | Sent data size | |
|  |  | | |
| [Returns] | 0 | | Success |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | SMBus "send byte" protocol. | | |
|  | Without specifying a register of a device, this function is sent 8 bits. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_read\_byte\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "read byte" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_read\_byte\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_read\_byte\_data(struct i2c\_client \*client, u8 command); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  |  | | |
| [Returns] | Positive Number | | Success (Read data size) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is received 8 bits data from the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_write\_byte\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "write byte" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_write\_byte\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_write\_byte\_data(struct i2c\_client \*client, u8 command, u8 value); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  | value | Write data size | |
|  |  | | |
| [Returns] | 0 | | Success |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is sent 8 bits data to the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_read\_word\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "read word(16bit)" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_read\_word\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_read\_word\_data(struct i2c\_client \*client, u8 command); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  |  | | |
| [Returns] | Positive Number | | Success (Read data size(16bit unsigned word)) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is received 16 bits data from the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_write\_word\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "write word(16bit)" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_write\_word\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_write\_word\_data(struct i2c\_client \*client, u8 command, u16 value); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  | value | Write data size (16bit word) | |
|  |  | | |
| [Returns] | 0 | | Success |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is sent 16 bits data to the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_read\_block\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "block read" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_read\_block\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_read\_block\_data(struct i2c\_client \*client, u8 command, u8 \*values); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  | values | The pointer to store read data. | |
|  |  | | |
| [Returns] | Positive Number | | Success (Read data block size) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is received data block from the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_write\_block\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "block write" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_write\_block\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_write\_block\_data(struct i2c\_client \*client, u8 command, | | |
|  | u8 length, const u8 \*values); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  | length | Data block size. | |
|  | value | The pointer to store write data. | |
|  |  | | |
| [Returns] | 0 | | Success |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is sent data block to the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_read\_i2c\_block\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "read block byte" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_read\_i2c\_block\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_read\_i2c\_block\_data(struct i2c\_client \*client, u8 command, | | |
|  | u8 length, u8 \*values); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  | length | Data block size. | |
|  | values | The pointer to store read data. | |
|  |  | | |
| [Returns] | Positive Number | | Success (Read data size) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is received data block from the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

### i2c\_smbus\_write\_i2c\_block\_data

|  |  |  |  |
| --- | --- | --- | --- |
| [Overview] | SMBus "write block byte" protocol | | |
|  |  | | |
| [Function Name] | i2c\_smbus\_write\_i2c\_block\_data | | |
|  |  | | |
| [Calling format] | #include <linux/i2c.h> | | |
|  | s32 i2c\_smbus\_write\_i2c\_block\_data(struct i2c\_client \*client, u8 command, | | |
|  | u8 length, const u8 \*values); | | |
|  |  | | |
| [Arguments] | client | The information of slave device. | |
|  | command | The command interpreted by slave. | |
|  | length | Data block size. | |
|  | value | The pointer to store write data. | |
|  |  | | |
| [Returns] | 0 | | Success (Write data size) |
|  | Negative Number | | Error |
|  |  | | |
| [Feature] | This function is sent data block to the specified register of a device. | | |
|  |  | | |
| [Remark] | This function cannot be called from the interrupt handler. | | |
|  | Refer to 4.3.2 about i2c\_client. | | |
|  |  | | |

## Structure

Structure of this module is based on Linux v5.10.41

### struct i2c\_adapter

Table 4.3‑1 struct i2c\_adapter (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Structure name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_adapter | struct module \* | owner | The owner of this module |
| unsigned int | class | The type of I2C device supported by this driver |
| const struct i2c\_algorithm \* | algo | The pointer of the algorithm to access the bus |
| void \* | algo\_data | The algorithm data |
| const struct i2c\_lock\_operations\* | lock\_ops | Lock operations |
| struct rt\_mutex | bus\_lock | The structure specified rt\_mutex |
| struct rt\_mutex | mux\_lock | The structure specified rt\_mutex |
| int | timeout | Timeout value |
| int | retries | The retry number |
| struct device | dev | The adapter device |
| unsigned long | locked\_flags | owned by the I2C core |
| int | nr | The adapter ID |
| char | name[48] | The name of I2C device driver |
| struct completion | dev\_released | The structure used to maintain the state of "completion" |
| struct mutex | userspace\_clients\_lock | The mutex of client |
| struct list\_head | userspace\_clients | The list of client |
| struct i2c\_bus\_recovery\_info \* | bus\_recovery\_info | The pointer of the information for bus recovery |
| const struct i2c\_adapter\_quirks \* | quirks | describe flaws of the i2c adapter |
| struct irq\_domain \* | host\_notify\_domain | Recovery |

### struct i2c\_client

Table 4.3‑2 struct i2c\_client (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Structure name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_client | unsigned short | flags | The support function flag of client |
| unsigned short | addr | The slave address |
| char | name[I2C\_NAME\_SIZE] | The client name |
| struct i2c\_adapter \* | adapter | The adapter information |
| struct device | dev | The driver model device node for the slave |
| int | init\_irq | irq set at initialization |
| int | irq | The interrupt number used by the device |
| struct list\_head | detected | The member of i2c\_driver.clients list |

### struct i2c\_board\_info

Table 4.3‑3 struct i2c\_board\_info (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Structure name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_board\_info | char | type[I2C\_NAME\_SIZE] | The chip type to initialize i2c\_client.name |
| unsigned short | flags | The flag to initialize i2c\_client.flags |
| unsigned short | addr | The device address |
| const char \* | dev\_name | The device name |
| void \* | platform\_data | The Platform Data of the device |
| struct device\_node \* | of\_node | The information of the device node |
| struct fwnode\_handle \* | fwnode | Device node supplied by the platform firmware |
| const struct property\_entry \* | properties | Additional device properties for the device |
| const struct resource \* | resources | Resources associated with the device |
| unsigned int | num\_resources | Number of resources in the @resources array |
| int | irq | The interrupt number used by the device |

### struct i2c\_driver

Table 4.3‑4 struct i2c\_driver (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Structure name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_driver | unsigned int | class | The kind of the I2C device created the instance |
| int (\*probe)(struct i2c\_client \*,  const struct i2c\_device\_id \*) | probe | Callback for device binding |
| int (\*remove)(struct i2c\_client \*) | remove | Callback for device unbinding |
| int (\*probe\_new)(struct i2c\_client \*) | probe\_new | Callback for device probing |
| void (\*shutdown)(struct i2c\_client \*) | shutdown | Callback for device shutdown |
| void (\*alert)(struct i2c\_client \*,  enum i2c\_alert\_protocol protocol,  unsigned int data) | alert | Alert callback |
| int (\*command)(struct i2c\_client \*client,  unsigned int cmd, void \*arg) | command | Callback for bus-wide signaling (optional) |
| struct device\_driver | driver | The device driver structure |
| const struct i2c\_device\_id \* | id\_table | List of I2C devices supported by this driver |
| int (\*detect)(struct i2c\_client \*,  struct i2c\_board\_info \*) | detect | Callback for device detection |
| const unsigned short \* | address\_list | The I2C addresses to probe (for detect) |
| struct list\_head | clients | List of detected clients we created (for i2c-core use only) |

### struct i2c\_msg

Table 4.3‑5 struct i2c\_msg (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Structure name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_msg | \_\_u16 | addr | The slave address |
| \_\_u16 | flags | Specify R/W flag |
| \_\_u16 | len | The message length |
| \_\_u8\* | buf | The pointer to the message data |

### struct i2c\_rdwr\_ioctl\_data

Table 4.3‑6 struct i2c\_rdwr\_ioctl\_data (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Structure name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_rdwr\_ioctl\_data | struct i2c\_msg \* | msgs | The pointer to i2c\_msg structure |
| \_\_u32 | nmsgs | A number of i2c\_msg |

### struct i2c\_smbus\_ioctl\_data

Table 4.3‑7 struct i2c\_smbus\_ioctl\_data (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Structure name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_smbus\_ioctl\_data | \_\_u8 | read\_write | Specify R/W flag |
| \_\_u8 | command | The slave address |
| \_\_u32 | size | Data type (size) |
| union i2c\_smbus\_data \* | data | The pointer to the data |

### union i2c\_smbus\_data

Table 4.3‑8 union i2c\_smbus\_data (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Union name** | **Member** | | |
| --- | --- | --- | --- |
| **Type** | **Member name** | **Overview** |
| i2c\_smbus\_data | \_\_u8 | byte | 8bit data buffer |
| \_\_u16 | word | 16bit data buffer |
| \_\_u8 | block[I2C\_SMBUS\_BLOCK\_MAX + 2] | Block data buffer |

## Global Variables and Constants

### Global Variables

There are no global variables for this module.

### Global Constants

The following table shows the global constants used by standard I2C core.

Table 4.4‑1 List of Global constants (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Global Constant Name** | Value | Remark |
| --- | --- | --- |
| I2C\_CLIENT\_END | 0xfffeU | - |
| I2C\_CLIENT\_PEC | 0x04 | - |
| I2C\_CLIENT\_TEN | 0x10 | - |
| I2C\_CLIENT\_WAKE | 0x80 | - |
| I2C\_FUNC\_SMBUS\_QUICK | 0x00010000 | - |
| I2C\_FUNC\_SMBUS\_READ\_BYTE | 0x00020000 | - |
| I2C\_MODULE\_PREFIX | "i2c:" | - |
| I2C\_M\_RD | 0x0001 | - |
| I2C\_M\_RECV\_LEN | 0x0400 | - |
| I2C\_M\_TEN | 0x0010 | - |
| I2C\_NAME\_SIZE | 20 | - |
| I2C\_SMBUS\_BYTE | 1 | - |
| I2C\_SMBUS\_BYTE\_DATA | 2 | - |
| I2C\_SMBUS\_BLOCK\_DATA | 5 | - |
| I2C\_SMBUS\_BLOCK\_MAX | 32 | - |
| I2C\_SMBUS\_BLOCK\_PROC\_CALL | 7 | - |
| I2C\_SMBUS\_I2C\_BLOCK\_DATA | 8 | - |
| I2C\_SMBUS\_PROC\_CALL | 4 | - |
| I2C\_SMBUS\_QUICK | 0 | - |
| I2C\_SMBUS\_READ | 1 | - |
| I2C\_SMBUS\_WRITE | 0 | - |
| I2C\_SMBUS\_WORD\_DATA | 3 | - |
| I2C\_RDWR | 0x0707 | - |
| I2C\_FUNCS | 0x0705 | - |
| I2C\_SLAVE | 0x0703 | - |
| I2C\_SMBUS | 0x0720 | - |

# Integration

## Directory Configuration

The directory configuration is shown below.

drivers/i2c/busses/

i2c-rcar.c

：I2C Driver source file

Figure 5‑1 Directory configuration (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

## Integration Procedure

To enable the function of this module, make the following setting with Kernel Configuration.

Device Drivers --->

-\*- I2C support --->

<\*> I2C device interface

I2C Hardware Bus support --->

<\*> Renesas R-Car I2C Controller

Figure 5‑2 Kernel configuration (R-Car H3/M3/M3N/E3/D3/V3U/V3H)

## Option Setting

### Module Parameters

#### DT

When operation is in master mode, the SCL clock ratio is generated from an internal clock.

The following equation is used.

SCLfreq = I2Cck/(20 + SCGD × 8 + F[(tICF + tr + IntDelay) × I2Cck])

I2Cck: I2C internal clock frequency (133.33MHz[R-Car H3/M3/M3N/E3/D3/V3U/V3H])

SCGD: SCL Clock Generation Divider (Calculated by the driver)

F[n]: n rounded down to an integer

Table 5.3‑1 Device Tree and Formula Parameter(R-Car H3/M3/M3N/E3/D3/V3U/V3H)

| **Device Tree property** | Formula Parameter | Description | Default value |
| --- | --- | --- | --- |
| clock-frequency | SCLfreq | Frequency of bus clock in Hz | 100000 |
| i2c-scl-rising-time-ns | tr | I2C SCL rising time (depending on external load) | 200 |
| i2c-scl-falling-time-ns | tICF | I2C SCL falling time (depending on external load) | 35 |
| i2c-scl-internal-delay-ns | IntDelay | LSI internal delay corresponds to output buffer type. | 50 |

Please set I2C transfer speed 400000 or 100000 to "clock-frequency" of device tree file (salvator-common.dtsi) in arch/arm64/boot/dts/renesas directory. No setting means 100000 Hz.

&i2c2 {

status = "okay";

clock-frequency = <100000>;

...

}

&i2c4 {

status = "okay";

clock-frequency = <400000>;

...

}

Figure 5‑3 Example of setting I2C transfer speed (R-Car H3/M3/M3N)

I2C0, I2C3, I2C4 and I2C5 have Open drain buffer.

Their IntDelay are set 110ns in arch/arm64/boot/dts/renesas/r8a7795.dtsi, r8a7796.dtsi, r8a77965.dtsi.

I2C1, I2C2 and I2C6 have LVTTL (low drive only) buffer.

Their IntDelay are set 6 ns in arch/arm64/boot/dts/renesas/r8a7795.dtsi, r8a7796.dtsi, r8a77965.dtsi.

Please set I2C transfer speed 400000 or 100000 to "clock-frequency" of device tree file (r8a77990-ebisu.dts, r8a77990-es10-ebisu.dts) in arch/arm64/boot/dts/renesas directory. No setting means 100000 Hz.

&i2c0 {

status = "okay";

clock-frequency = <400000>;

...

}

&i2c3 {

status = "okay";

clock-frequency = <100000>;

...

}

Figure 5‑4 Example of setting I2C transfer speed (R-Car E3)

I2C0, I2C3 have Open drain buffer.

Their IntDelay are set 110ns in arch/arm64/boot/dts/renesas/r8a77990.dtsi.

I2C1, I2C2, I2C4, I2C5, I2C6 and I2C8 have LVTTL (low drive only) buffer.

Their IntDelay are set 6ns in arch/arm64/boot/dts/renesas/r8a77990.dtsi.

Please set I2C transfer speed 400000 or 100000 to "clock-frequency" of device tree file (r8a77995-draak.dts) in arch/arm64/boot/dts/renesas directory. No setting means 100000 Hz.

&i2c0 {

status = "okay";

clock-frequency = <400000>;

...

}

&i2c1 {

status = "okay";

clock-frequency = <100000>;

...

}

Figure 5‑5 Example of setting I2C transfer speed (R-Car D3)

I2C0, I2C1, I2C2 and I2C3 have LVTTL (low drive only) buffer.

Their IntDelay are set 6ns in arch/arm64/boot/dts/renesas/r8a77995.dtsi.

Please set I2C transfer speed 400000 or 100000 to "clock-frequency" of device tree file (r8a779a0-falcon.dts) in arch/arm64/boot/dts/renesas directory. No setting means 100000 Hz.

&i2c0 {

status = "okay";

clock-frequency = <400000>;

...

}

&i2c1 {

status = "okay";

clock-frequency = <100000>;

...

}

Figure 5‑6 Example of setting I2C transfer speed (R-Car V3U)

Please set I2C transfer speed 400000 or 100000 to "clock-frequency" of device tree file (r8a77980-condor.dts) in arch/arm64/boot/dts/renesas directory. No setting means 100000 Hz.

&i2c0 {

status = "okay";

clock-frequency = <400000>;

...

}

&i2c1 {

status = "okay";

clock-frequency = <100000>;

...

}

Figure 5.7 Example of setting I2C transfer speed (R-Car V3H)

I2C0, I2C1, I2C2, I2C3, I2C4 and I2C5 have LVTTL (low drive only) buffer.

Their IntDelay are set 6ns in arch/arm64/boot/dts/renesas/r8a77980.dtsi

#### Multi-master

If there is another master on I2C bus, a clock should always be supplied.

Please set “multi-master” of device tree file (salvator-common.dtsi, r8a77990-ebisu.dts, r8a77990-es10-ebisu.dts, r8a77995-draak.dts, r8a779a0-falcon.dts, r8a77980-condor.dts, …) in arch/arm64/boot/dts/renesas directory.

| **Device Tree property** | Description | Remark |
| --- | --- | --- |
| multi-master | Support the multi-master | A clock should always be supplied when multi-master to keep arbitration working. |

### Kernel Parameters

There are no module parameters.