GigaDevice Semiconductor Inc.

Solution of I2C bus lock based on EEPROM communication

Application Note
AN036

Revision 1.1

(Nov. 2024)

Table of Contents

List of Figures

List of Table

1. Introduction

MCU often uses I2C as the master to communicate with the EEPROM. When the I2C master resets during the communication process, there will be a probability that it will no longer be able to communicate with the EEPROM. This is called a bus lock. To solve this problem, this article provides a method to release the I2C bus using software configuration.

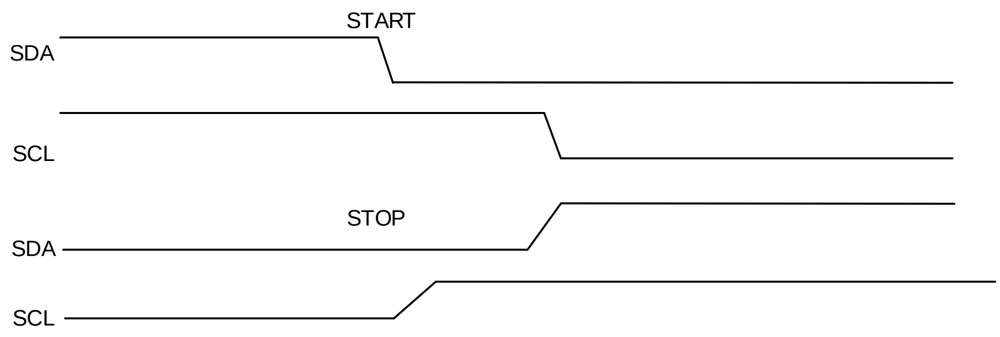
2. I2C bus lockup

2.1. I2C bus lock phenomenon

To complete a normal communication task between the master and slave of the I2C, before establishing the communication, the master must first detect the status of the I2C bus. When the SCL and SDA lines of the I2C bus are both high, the I2C bus is in an idle state. When the SCLA is high, the master pulls down the SDA signal to generate a START start signal.

When the I2C master-slave machine ends a communication task, it needs the master to generate a stop signal, that is, when the SCL is high, the SDA signal is pulled up.

Figure 2-1. I2C bus start and stop signals



Under normal circumstances, the I2C bus protocol can ensure normal read and write operations on the bus. However, when the I2C master device is reset (watchdog action, abnormal on-board power supply causes reset chip action, manual button reset, etc.), and the slave device is not reset, it may cause I2C bus deadlock. In the state of the bus lock, SCL maintains a high state, and SDA maintains a low state.

2.2. Causes of I2C bus lock

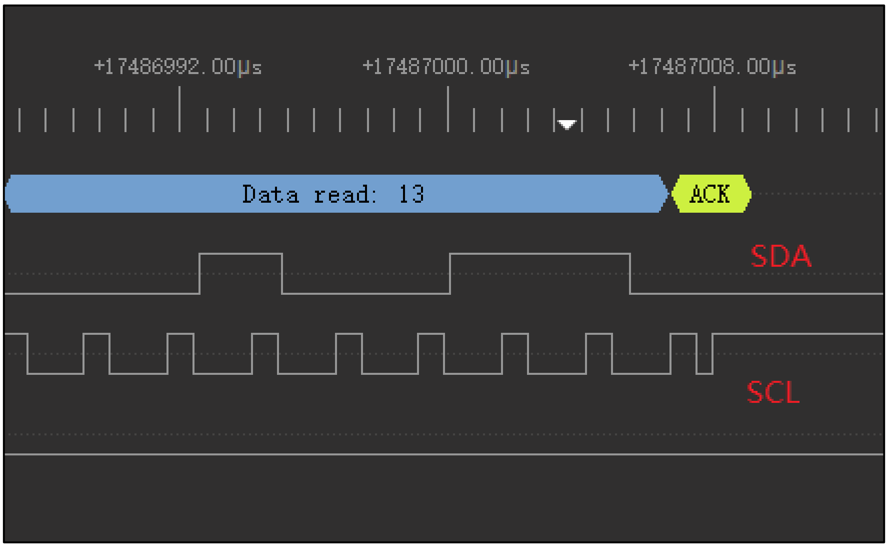
In the process of I2C master reading and writing, there are two situations that will cause the bus to lock up.

1. After the master sends the START signal, it controls the SCL to generate 8 clock pulses, and then pulls the SCL signal low. At this time, the slave outputs the response signal and pulls the SDA signal to the low level. If the host resets abnormally at this time, SCL will be released to high level. At this time, if the slave is not reset, it will continue the I2C response, pulling SDA to a low level all the time, and the response signal will not end until SCL becomes a low level. However, since the I2C host detects the state of the bus after resetting, if the SDA signal is low, the I2C bus is occupied, and it will wait for the SCL and SDA signals to become high. Therefore, when the I2C master is waiting for the slave to

release the SDA signal, the I2C slave is waiting for the master to pull the SCL signal low to release the response signal. The two wait for each other, and the I2C bus enters a deadlock state.

2. When the I2C master is reading data, the I2C slave responds and outputs data. If the I2C master resets abnormally at this moment and the data bit output by the I2C slave is exactly 0, it will also cause the I2C bus to enter a deadlock state.

Figure 2-2. I2C bus lock timing



3. Solution to I2C bus lock

The I2C bus is locked. The bus can also be released by resetting the slave. But when the EEPROM is used as a slave, the software cannot be used to reset the slave, and in some cases, the hardware cannot be reset. Therefore, it is necessary to add the bus release function when the I2C master establishes a new communication. Since the bus lock is probabilistic, the bus BUSY state timeout function can be added. The combination of the two can improve the robustness of the system.

3.1. SCL clock signal release bus

Add the I2C bus recovery program in the I2C master. Every time the I2C master device is reset, if the SDA data line is detected to be pulled low, the SCL clock line in the I2C is controlled to generate 9 clock pulses (for 8-bit data), so that the I2C slave device can be suspended The operation is recovered from the deadlock state.

The I2C master initializes the SCL pin as a normal GPIO function and configures it as a push-pull output. Ensure that 9 clock pulses are sent continuously. In order to ensure normal I2C communication, first reset the I2C module, then set it, and finally configure it as the I2C pin multiplexing function. The software configuration under GD32 project is shown in the following table.

Table 3-1. Configuration of SCL clock signal release bus in GD project

```
/*!
    \brief      reset i2c bus
    \param[in]  none
    \param[out] none
    \retval     none
*/
void i2c_bus_reset()
{
    uint8_t I = 0;
    gpio_init(GPIOB, GPIO_MODE_OUT_PP, GPIO_OSPEED_50MHZ, GPIO_PIN_6);
    /* SCL output clock signal */
    for(I = 0; I < 10; i++){
    gpio_bit_reset(GPIOB, GPIO_PIN_6);
    delay_1us(2);
    gpio_bit_set(GPIOB, GPIO_PIN_6);
    delay_1us(2);
    }
    /* reset I2C */
    i2c_software_reset_config(I2C0, I2C_SRESET_RESET);
    i2c_software_reset_config(I2C0, I2C_SRESET_SET);
```

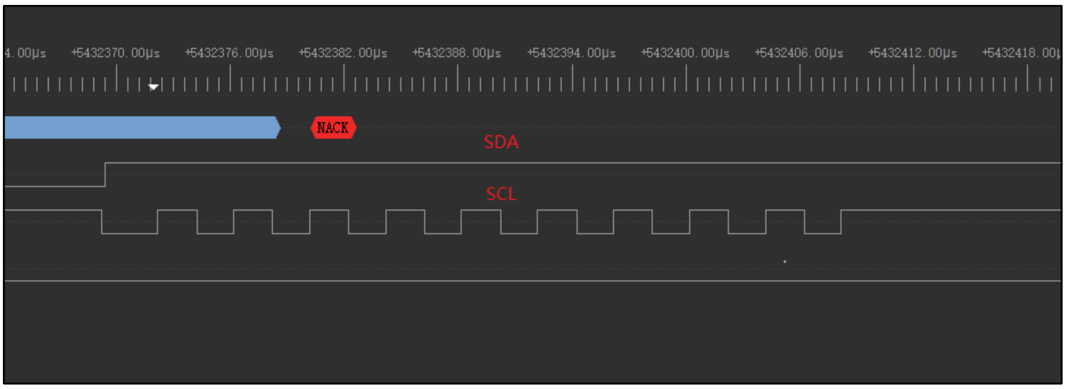
```
    gpio_init(GPIOB,    GPIO_MODE_AF_OD,    GPIO_OSPEED_50MHZ,    GPIO_PIN_6    |
    GPIO_PIN_7);
}

/*!
    \brief      check the I2C is or not busy
    \param[in]  none
    \param[out] none
    \retval     none
*/
void check_bus_status(void)
{
    while(i2c_flag_get(I2C0,I2C_FLAG_I2CBSY))
    {
        if(--time_out == 0){
            i2c_bus_reset();
        }
    }
}
```

3.2. Test results

On the GD32F303 platform, the bus release method was tested, and the test results are shown in ***Figure 3-1. SCL clock signal release bus test***.

The SCL clock signal release bus test is shown in the following figure. After the I2C bus is locked, 9 clock signals are sent continuously. Finally, both SDA and SCL are pulled high. The master detects that the bus is in an idle state and can start to establish a new communication.

Figure 3-1. SCL clock signal release bus test



4. Version history

Table 4-1. Version history

Revision No.	Description	Date
1.0	Initial Release	Nov.30 2021
1.1	1. Remove the method of solving lock-ups by forcibly pulling up the SDA and SCL lines. This method cannot meet all usage scenarios.	Nov.22 2024

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the “Product”), is owned by the Company under the intellectual property laws and treaties of the People’s Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it’s suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it’s suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.

© 2024 GigaDevice – All rights reserved