Freescale Semiconductor
Application Note

Document Number: AN5264
Rev. 0, 05/2016

Emulating I2C Master Mode
In the S12/S12Z Family

by: Agustin Diaz

Contents

1 Introduction

The I^2C protocol is a 2-wire serial communication interface implemented by hardware in numerous devices. However, in certain situations it is necessary to emulate the protocol using the General Purpose Input Outputs (GPIOs) of the microcontroller, usually to act as the “master”.

This application note describes how to communicate via I^2C using input/output pins. This method can be executed on any MCU. However, the scope of this application note are the MCUs of the S12/S12Z family. To develop the example provided in this application note S12ZVL128 was selected. The software example is provided along this document. The application note does not focus on providing a deep explanation on how the I^2C protocol works.

2 Overview

I^2C is a 2-wire communication protocol that requires two lines: the data line SDA and the clock line SCL. The frequency of the SCL line goes from 100 kHz to 400 kHz in fast mode. There are two types of devices: master and slave. The master is the one that initiates the transfer of data, generates the clock signal, addresses a slave device and terminates the transfer of data. The I^2C provides a way of validating the veracity of the data transferred. The

acknowledgment bits, which are either generated by the master or the slave after the received data has been validated. The protocol also provides a solution for handling multiple masters on the same bus.

3 I²C Basic states

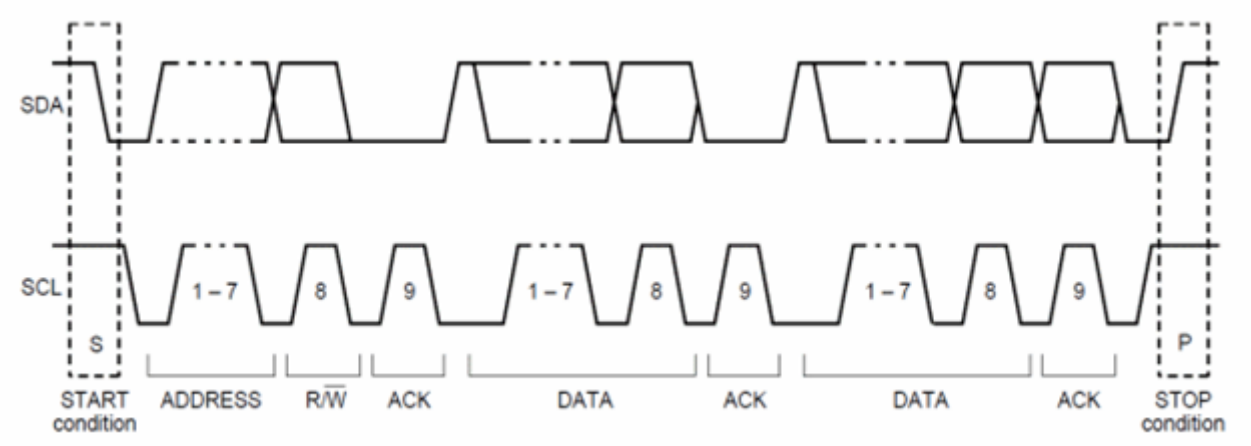

Figure 1. The I²C protocol

- START condition: Falling edge on SDA while SCL is held high.
- STOP condition: Rising edge on SDA while SCL is held high.
- ADRESS: Indicates the address of the slave to whom the master wants to communicate.
- R/W: Indicates if the master wants to read or write from a specific slave.
- ACK (Acknowledge bit): Held low at every ninth SCL pulse if the slaves validates whatever was transferred. Held high otherwise.

4 I²C master application

The application presented in this document illustrates a basic implementation of the I²C protocol using 2 digital I/O and one timer for the time specifications of the protocol. It provides the basic functionality of a device configured as master. The application allows the MCU to send data through the SDA line and controls de SCL signal to communicate with I²C slave devices. This application uses the GPIO of the S12 family in order to simulate the 2-wire line of the I²C protocol. Therefore, it can be used by any MCU of the S12 family.

The software attached with the application note provides the following functionality:

- 7-bit addressing mode of the I2C slave device.
- Single master transmitter.
- SCL frequency of approximately 108 kHz.
- Write and read operations.
- Single data byte within a serial transfer.
- Checking of acknowledge signal.
- Easy routing of SDA and SCL lines.
- The time delays are generated by polling.

In order for this application to function properly SDA and SCL lines must be in the same port. And both should be configured as open drain otherwise the application may not work correctly and the slave devices could be damaged.

On the software provided along this application note, in the *vIIC.h* file there are some configurations that the user may change in order to route SDA and SCL to a different port or different pins. The following three defines must be changed in order to achieve that:

```
#define vIIC_PORT S
#define SDA_PIN 1
#define SCL_PIN 2
```

vIIC indicates at which port the two wires will be routed. SDA_PIN indicates at which pin of that port SDA will be routed. SCL_PIN indicates at which pin of that port SCL will be routed.

There are three main functions in this application:

```
void vIIC_init(void);
void vIIC_write_byte(unsigned char device_address, unsigned char device_register, unsigned
char data);
unsigned char vIIC_read_byte(unsigned char device_address, unsigned char device_register);
```

vIIC_init() set the pins as outputs. It should be called at the beginning of the program.

vIIC_write_byte() receives the address of the I2C device, the register at which the master is trying to write and the data that wants to be written.

vIIC_read_byte() receives the address of the I2C device and the register that wants to be read.

5 Example overview

This application was tested on a S12ZVL128 MCU communicating with a GY-521 slave device. The GY-521 is an I^2C gyroscope and accelerometer. The lines are routed to Port S SDA to bit 1 and SCL to bit 2. The I^2C frame is shown in the following figure.

Figure 2. I^2C example frame

SDA is the yellow signal and SCL the green one.

The Figure 3 shows the read sequence initiated by the master. To test if the protocol functions properly the master initiates a read sequence of the register address 0x75 of the slave, this register contains the address of the slave. If the communication works correctly a 0x68 must be received by the master. A START bit is required to initiate communication, followed by the 7-bit address of the slave (in this case 0x68) and a 0 indicating that the master wants to write in to the slave. Then at the ninth pulse the slave holds the line low indicating acknowledge. Then the master writes 0x75 indicating that the slave must position in that address. Then the master sends a read command and the slave responses with 0x68.

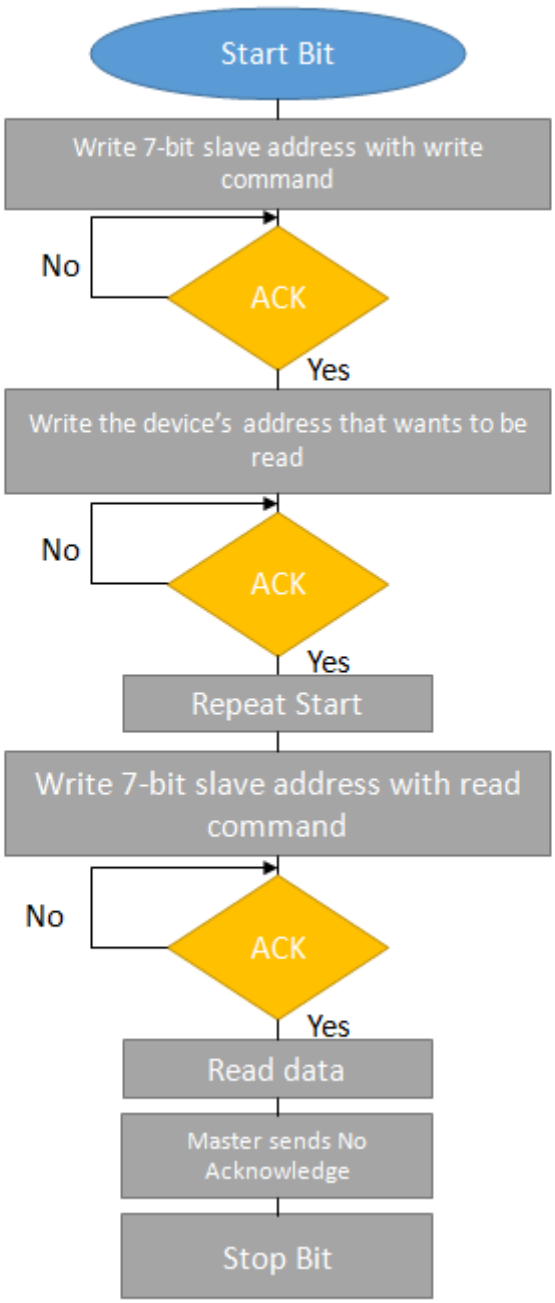

Figure 3. Flow diagram

6 Attachments

This application note includes the software that carries out the application described in the previous section. The *vIIC.h* and *vIIC.c*. as well as the example code that reads the address of the accelerometer, configures the timer and the bus clock of the MCU.

7 References

- AN3317

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number AN5264
Revision 0, 05/2016