4.6 I2C Protocol

The I²C bus uses SCL (= SCx pin, serial clock) and SDA (= SDx pin, serial data input and output) signal lines. Both lines are connected to V_{DDIO} externally via pull-up resistors so that they are pulled high when the bus is free.

The I²C interface of the BNO055 is compatible with the I²C Specification UM10204 Rev. 03 (19 June 2007), available at http://www.nxp.com. The BNO055 supports I²C standard mode and fast mode, only 7-bit address mode is supported. The BNO055 I²C interface uses clock stretching.

The default I²C address of the BNO055 device is 0101001b (0x29). The alternative address 0101000b (0x28), in I2C mode the input pin COM3 can be used to select between the primary and alternative I2C address as shown in Table 4-7.

Table 4-7: I2C address selection

I2C configuration	COM3_state	I2C address
Slave	HIGH	0x29
Slave	LOW	0x28
HID-I2C	X	0x40

The timing specification for I²C of the BNO055 is given in Table 4-8: I²C timings:

Table 4-8: I²C timings

Parameter	Symbol	Condition	Min	Max	Units
Clock Frequency	f_{SCL}			400	kHz
SCL Low Period	t_{LOW}		1.3		µs
SCL High Period	t_{HIGH}		0.6		
SDA Setup Time	t_{SUDAT}		0.1		
SDA Hold Time	t_{HDDAT}		0.0		
Setup Time for a repeated Start Condition	t_{SUSTA}		0.6		
Hold Time for a Start Condition	t_{HDSTA}		0.6		
Setup Time for a Stop Condition	t_{SUSTO}		0.6		
Time before a new Transmission can start	t_{BUF}		1.3		
Idle time between write accesses, normal mode, standby mode, low-power mode 2	$t_{IDLE\_wacc\_nm}$		2		µs
Idle time between write accesses, suspend mode, low-power mode 1	$t_{IDLE\_wacc\_sum}$		450		µs

Figure 5: I²C timing diagram shows the definition of the I²C timings given in Table 4-8:

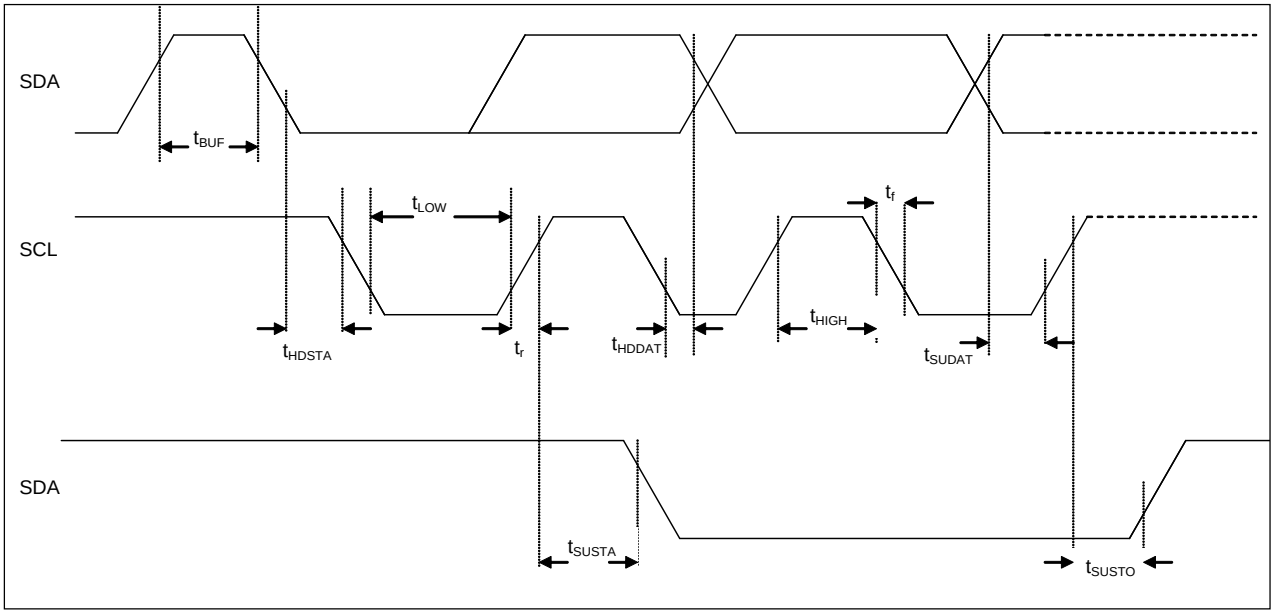

Figure 5: I²C timing diagram

The I²C protocol works as follows:

START: Data transmission on the bus begins with a high to low transition on the SDA line while SCL is held high (start condition (S) indicated by I²C bus master). Once the START signal is transferred by the master, the bus is considered busy.

STOP: Each data transfer should be terminated by a Stop signal (P) generated by master. The STOP condition is a low to HIGH transition on SDA line while SCL is held high.

ACK: Each byte of data transferred must be acknowledged. It is indicated by an acknowledge bit sent by the receiver. The transmitter must release the SDA line (no pull down) during the acknowledge pulse while the receiver must then pull the SDA line low so that it remains stable low during the high period of the acknowledge clock cycle.

In the following diagrams these abbreviations are used:

S	Start
P	Stop
ACKS	Acknowledge by slave
ACKM	Acknowledge by master
NACKM	Not acknowledge by master
RW	Read / Write

A START immediately followed by a STOP (without SCL toggling from ´VDDIO´ to ´GND´) is not supported. If such a combination occurs, the STOP is not recognized by the device.

I²C write access:
I²C write access can be used to write a data byte in one sequence. The sequence begins with start condition generated by the master, followed by 7 bits slave address and a write bit (RW = 0). The slave sends an acknowledge bit (ACK = 0) and releases the bus. Then the master sends the one byte register address. The slave again acknowledges the transmission

and waits for the 8 bits of data which shall be written to the specified register address. After the slave acknowledges the data byte, the master generates a stop signal and terminates the writing protocol.

Example of an I²C write access to the BNO055 (i2c address in this case: 0101000b = 0x28):

Start	Slave address							RW	ACKS	dummy	Register address (0x00 .. 0x7F)								ACKS	Data								ACKS	Stop
S	0	1	0	1	0	0	0	0	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	A	P

Figure 6: I²C write

I²C read access:

I²C read access also can be used to read one or multiple data bytes in one sequence. A read sequence consists of a one-byte I²C write phase followed by the I²C read phase. The two parts of the transmission must be separated by a repeated start condition (Sr). The I²C write phase addresses the slave and sends the register address to be read. After slave acknowledges the transmission, the master generates again a start condition and sends the slave address together with a read bit (RW = 1). Then the master releases the bus and waits for the data bytes to be read out from slave. After each data byte the master has to generate an acknowledge bit (ACK = 0) to enable further data transfer. A NACKM (ACK = 1) from the master stops the data being transferred from the slave. The slave releases the bus so that the master can generate a STOP condition and terminate the transmission.
The register address is automatically incremented and, therefore, more than one byte can be sequentially read out. Once a new data read transmission starts, the start address will be set to the register address specified in the latest I²C write command. By default the start address is set at 0x00. In this way repetitive multi-bytes reads from the same starting address are possible.

Example of an I²C read access to the BNO055:

Start	Slave address							RW	ACKS	dummy	Register address (0x08)								ACKS
S	0	1	0	1	0	0	0	0	A	x	0	0	0	1	0	0	0	A

Start	Slave address							RW	ACKS	Read data (0x08)								ACKM	Read data (0x09)								ACKM	
Sr	0	1	0	1	0	0	0	1	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	A	...

	ACKS	Read data (0x0A)								ACKM	Read data (0x0B)								ACKM	
...	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	A	...

	ACKS	Read data (0x0C)								ACKM	Read data (0x0D)								NACKM	Stop
...	A	x	x	x	x	x	x	x	x	A	x	x	x	x	x	x	x	x	NA	P

Figure 7: I²C multiple read