HLAB09
Interfacing an IMU using I2C (Inter-Integrated Circuit) Serial Communication

This lab will introduce the Atmega328 serial communication using an I2C (Inter-Integrated Circuit), also called a TWI (Two-Wire Interface), which is widely used to connect extra devices to a microprocessor system. The acronym of TWI and I2C will be used interchangeably. TWI is slower than SPI (serial peripheral interface) but has only two wires. TWI is also faster than a UART. Also, a network of maximum 128 devices can be connected to a TWI bus using only two wires. In this lab, we will connect a PNAV sensor, an Inertial Measurement Unit, reading acceleration, angular velocity and magnetic fields.

1. AIMS

- To connect an IMU sensor (operating at 3.3V) to the Atmega328 board which requires soldering a 3.3V power regulator and an optional level-shifter (5V to 3.3V).
- To compile and test the polling-based TWI read/write on the Atmega328 board.
- To stream out the IMU data to a serial Putty terminal in the desktop PC.

2. Connecting a PNAV Sensor

The PNAV sensor is an IMU (inertial measurement unit) which can measure the 3-axis acceleration, angular velocity, magnetic fields and barometric pressure. You can refer to the IMU article from Sparkfun for a more detailed explanation of the sensor.

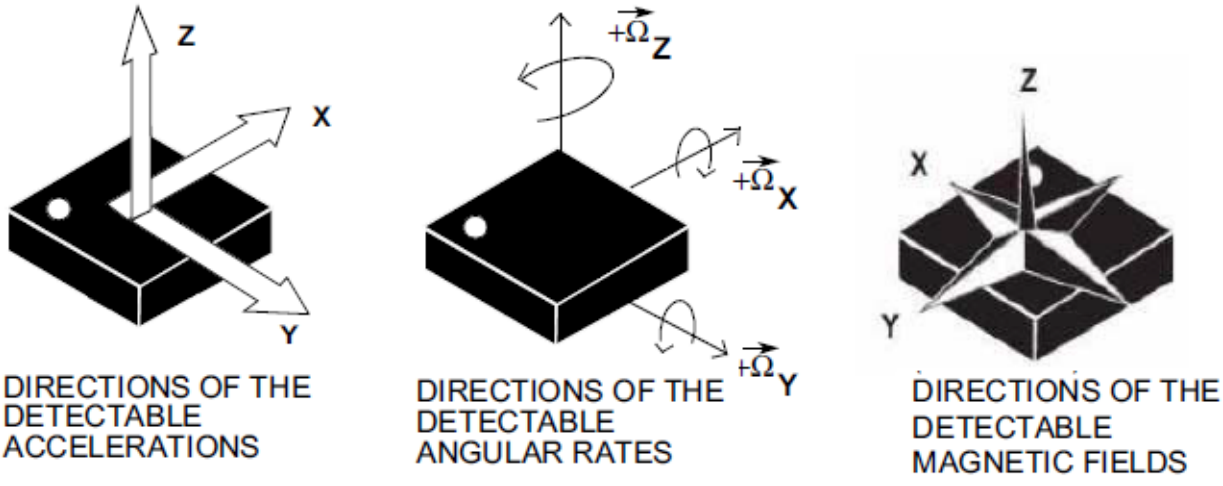


Figure 1 An IMU sensor with 3-axes of accelerometer, gyroscopes and magnetometers

It is operating with 3.3V and can be connected through the TWI or SPI (serial peripheral interface). TWI utilises the open-collector output (for the high impedance) to share the bus line with other devices. The PNAV sensor and TWI unit include internal pull-up resistors, and thus we will not attach extra pull-up resistors to the TWI lines.

The Atmega328P pins related to this lab are shown below.

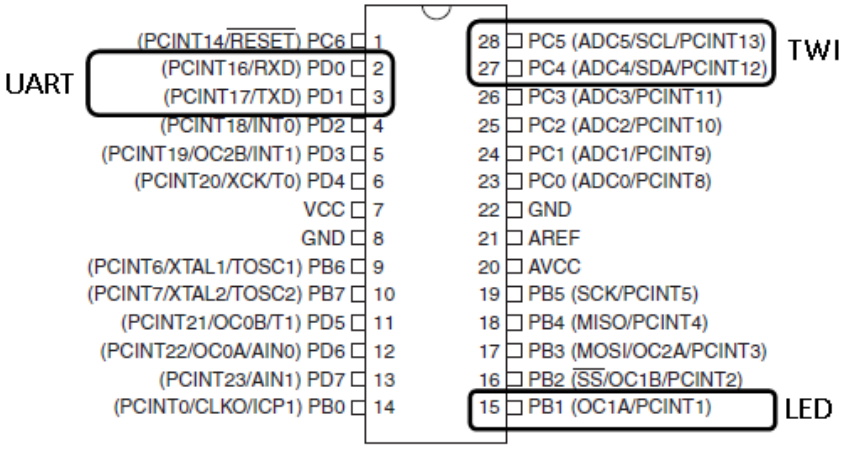


Figure 2. Atmega328 pins used in the work

To connect the IMU sensor,

- Solder a 3.3V regulator (LM2936-3.3) with two capacitors in a spare area of the Atmega board. Figure 3 below shows an example of soldering next to the power area. The regulator has three pins as shown (bottom-view) and for more information, please refer to the LM2936 datasheet. After connection, measure and confirm the output 3.3V voltage using a multi-meter.

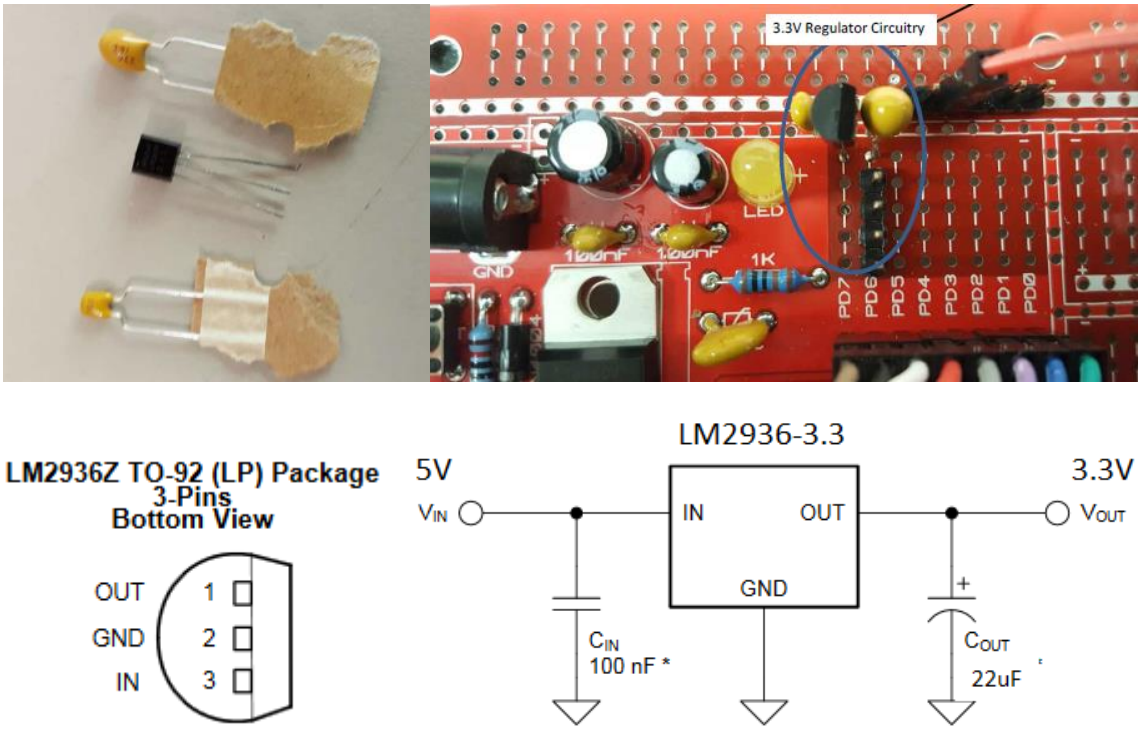


Figure 3. Soldering a 3.3V-Regulator.

- As the Atmega's I/O pins are 5V, their direct connection to a 3.3V IMU chip can damage the IMU I/O line or the 3.3V logic HIGH cannot be high enough. To prevent this problem, a level shifter shown below can be used (A 4-channels bi-directional level converter from Sparkfun, in which "HV' stands for High Voltage and "LV" for Low Voltage.

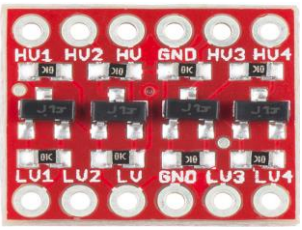


Figure 4. A level-converter between 5V and 3.3V I/O signals

- The level shifter is, however, optional in this lab to simplify the connections. This is possible as the TWI interface relies on pull-up resistors to create a logic-high and the PNAV has internal pull-up resistors connected to 3.3V power. A potential problem will be that the 3.3V logic HIGH is weaker than the 5V logic HIGH, although greater than 2.5V (CMOS logic threshold). If you see a problem, for example, due to noises, you need to use a level shifter provided in the kit.
- Using jumper wires, connect the SDA (Serial Data) and SCL (Serial Clock) lines between the IMU and Atmega328 board. The connection diagram of the system will look like below. We will use a serial terminal in Windows (Putty or Hyper-terminal), with the 115200-baud rate.

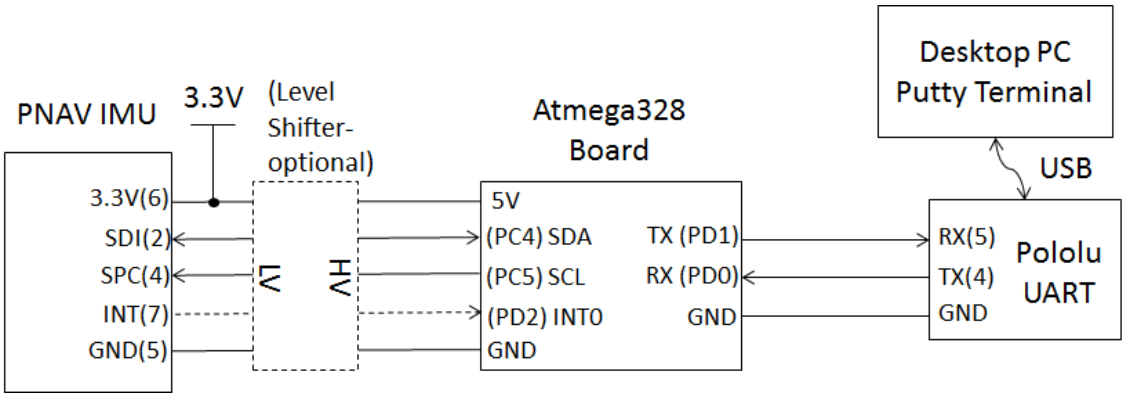


Figure 5. The full system connection diagram.

- It is recommended the IMU be attached to the spare region of the Atemga board using Velcro. We will be using a polling-based data reading, so the interrupt line is not necessary. If you wish to use the interrupt, the PNAV interrupt out pin (INT) can be connected to the Atmega328's external interrupt such as INT0 (PD2).
- An example of the final Atmega board is shown in Figure 6.

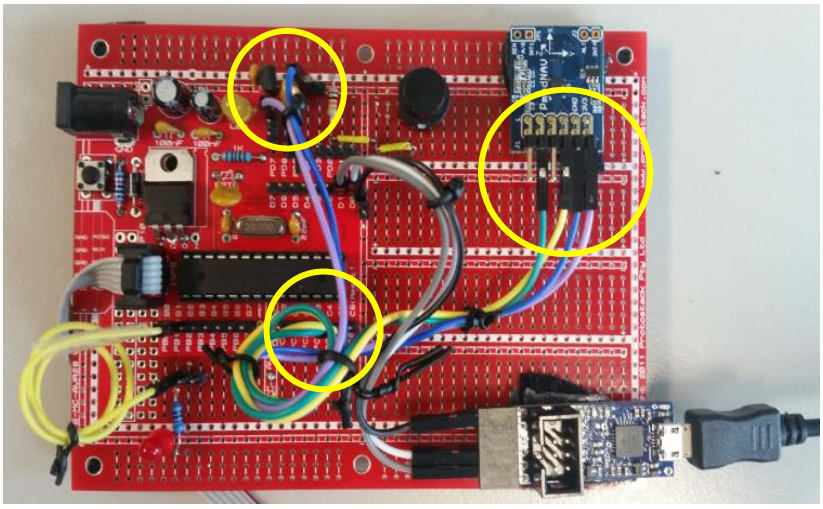


Figure 6. An example of IMU TWI connection to Atmega328 Board.

3. Programming TWI and IMU

Referring to the data flow of TWI communication in Appendix A,

- Download "*10_twi_interface_imu.c"* from Wattle and add them to a project.
- Build and download the executable on the Atmega328 board.
- The program provides three TWI functions to initialise, transmit and read data. The functions are used to read data from the magnetometer, sending the data to the UART serial. By referring to the IMU datasheet (LSM9DS1), complete the program to read Accelerometers and Gyroscopes, and display the data on the Putty terminal in proper units as
    - Acceleration (g)
    - Angular rate (deg/s)
    - Magnetic field (Gauss)
- By referring to the lecture note, compute the Euler angles (roll, pitch and heading in degrees) using the accelerometer and magnetometer.
- Try rotating the sensor/board (without pulling out any wires!) to see whether the values make sense.

- Congratulation! You have connected the IMU sensor successfully. Show the working system to the tutors to get marked.
- Please note that you need to keep working on the IMU data processing to improve the performance towards the final demonstration next week which has 10% marks of the course total. Please refer to the HLAB10 sheet for the suggestion on the further processing.

Appendix A – Two-Wire Serial Interface

A.1 Introduction

The TWI is ideally suited for typical microcontroller applications. The TWI protocol allows the systems designer to interconnect up to 128 different devices using only two bidirectional bus lines: one for clock (SCL) and one for data (SDA). The only external hardware needed to implement the bus is a single pull-up resistor for each of the TWI bus lines as shown in Figure A-1. All devices connected to the bus have individual addresses, and mechanisms for resolving bus contention are inherent in the TWI protocol.

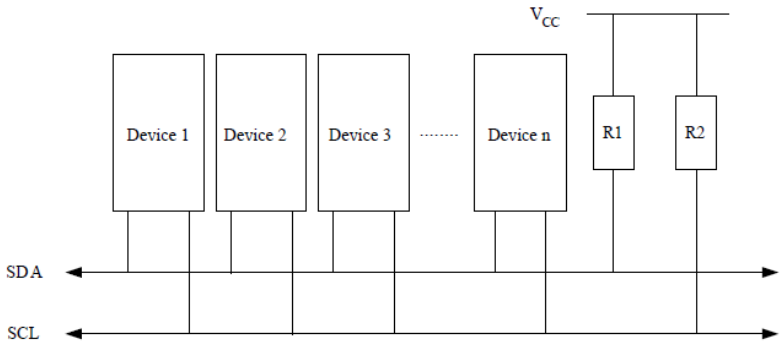


Figure A-1. TWI Bus Interconnection (from Datasheet)

TWI is a half-duplex serial communication unlike UART or SPI, which means the TX and RX lines are shared between all connected devices. In addition, there are no separate address lines to select a certain device, unlike SPI. This means the address should be sent to the device together with the data, as well as the data flow direction (read or write). The result is much-complicated software processing but the simplest hardware (only two wires!). The Digilent Pmod NAV sensor consists of the LSM9DS1 3-axis accelerometer, 3-axis gyroscope, 3-axis magnetometer, plus the LPS25HB digital barometer to provide users with 10-DOF (degrees-of-freedom) functionality. Figure A-2 illustrates the 3 internal sensors connected to a TWI bus system with each factory-set address (from datasheet). Note that the board contains internal pull-up resistors.

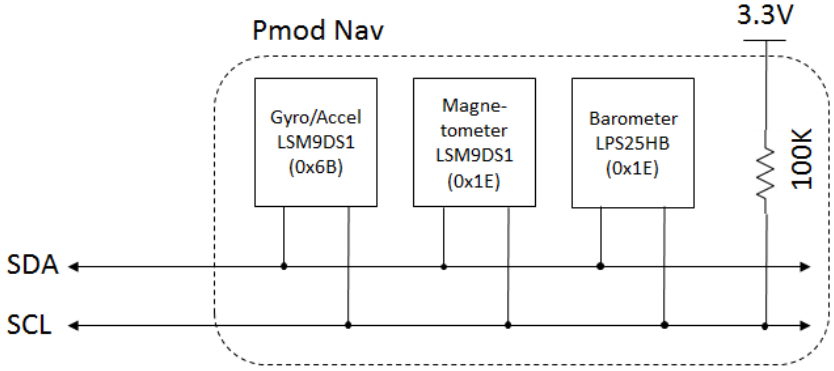
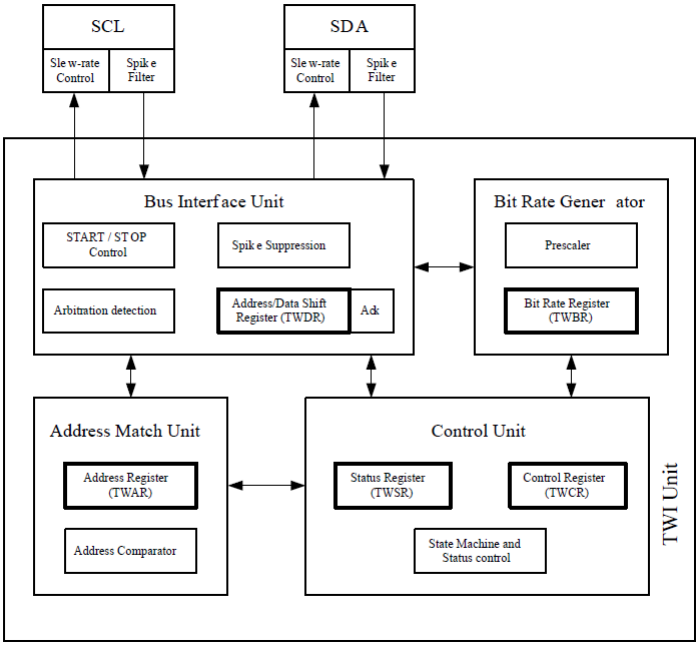


Figure A-2 Pmod Nav sensor which includes 3x TWI sensors with addresses and pull-up resistors connected to SDA and SCL lines.

The figure below illustrates the block diagram of the TWI unit in the Atmega328P controller, showing four relevant TWI registers,

- TWBR (TWI Bit-rate Register) – Set the Bit Rate i.e. 100K bps
- TWSR (TWI Status Register) – Status + Bit-rate Pre-scaler
- TWCR (TWI Control Register) – Interrupt Flag, Ack, Start/Stop, Enable TWI.
- TWDR (TWI Data Register) – TX and RX data

Figure 26-9. Overview of the TWI Module

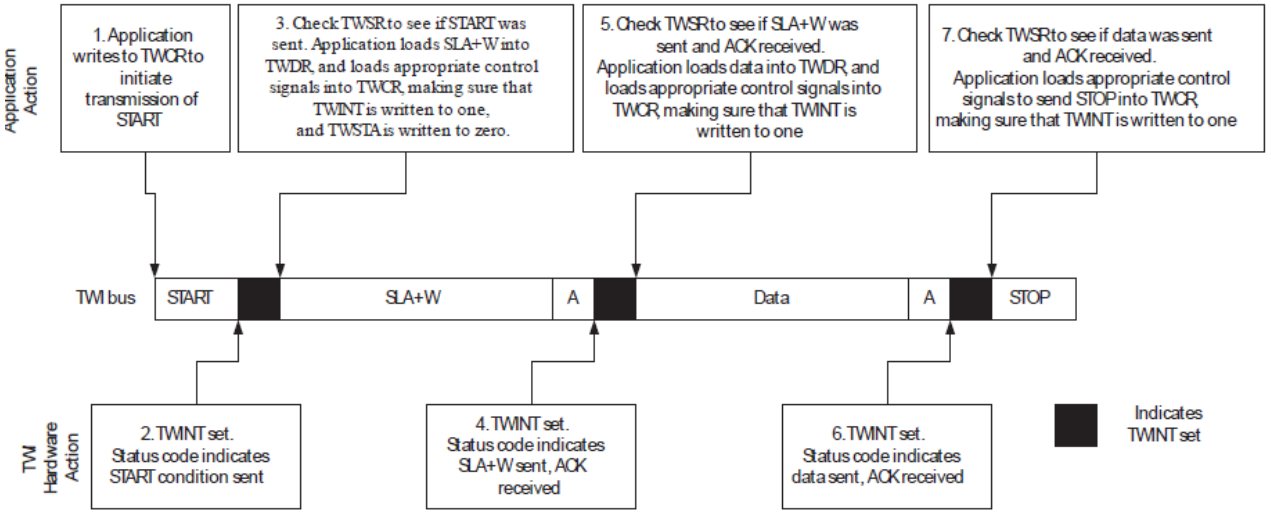


A.2 TWI Transmit and Receive

Transmitting and receiving data using the TWI unit require extensive handshaking (using acknowledge) between the master (which provides the clock and flow control) and slave devices. In addition, due to the single data line in the system, the device address should be sent before sending/receiving any data. The figure below illustrates a typical data packet for the master to transmit a byte date. Please refer to the datasheet for a detailed explanation of this figure. In brief, the TWI unit

- Sends a start signal (ST) to use the bus
- Sends a slave address (SLA) with reading or writing bit
- Upon an acknowledge (A) from the slave, reads or writes a byte
- Once reads the byte, the master sends an Ack. If master writes a byte, the slave should send an Ack
- Sends a stop signal (SP).

Figure 26-10. Interfacing the Application to the TWI in a Typical Transmission



A.3 Accessing IMU Registers using TWI

Reading and writing data from/to the IMU sensor require one more data to access the internal registers in the IMU sensor. The registers are to control the sensors and provide data with status. You need to refer to the datasheets

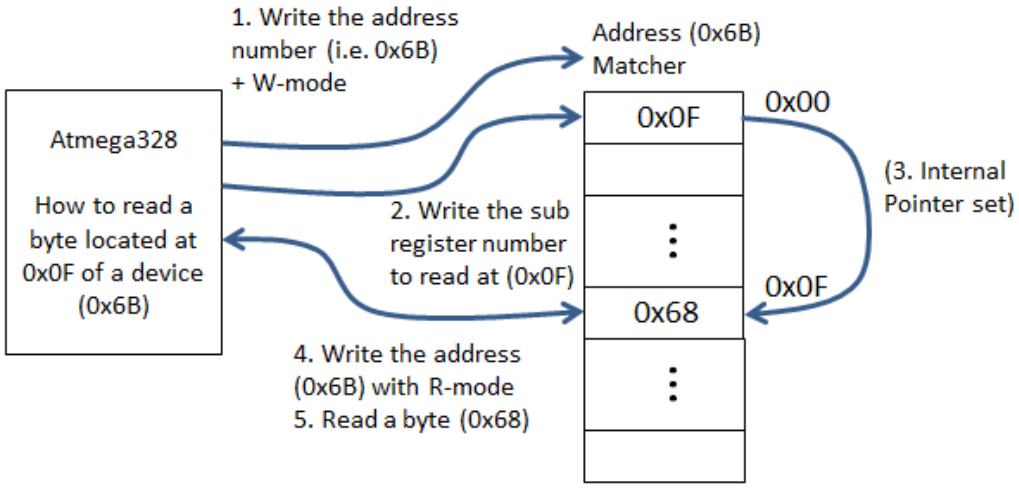
- LSM9DS1 – Accelerometers, Gyroscopes, Magnetometers and temperature (Registers Page 40)
- LPS25HB – Barometer (A static pressure sensor) (Registers on Page 31)

To access the internal registers, we need to write one more data of sub-address of the register during the reading and writing operations. For example, suppose we want to read the “Who_Am_I” register in the IMU (Accel+Gyro) unit which is located at 0x0F with a content value of 0x68. Table 17 and the figure below illustrate the data flow (some different acronyms are used in the IMU datasheet).

- Send a start signal (ST) to use the bus
- Send a slave address (SLA) (0x6B) with a Write-bit
- *Send a byte of sub-address (0x0F) of the register*
- *Send a slave address (0x6B) again with a Read-bit*
- Read a byte (0x68) from the slave
- *Master sends a NACK (Not-Ack) to mark the end of data*
- Send a stop signal (SP)

Table 17. Transfer when master is receiving (reading) one byte of data from slave

Master	ST	SAD + W		SUB		SR	SAD + R			NMAK	SP
Slave			SAK		SAK			SAK	DATA		



To read multiple registers, we can use the auto-increment option of the magnetometer, and thus the master can continuously read data as shown in Table 18.

Table 18. Transfer when master is receiving (reading) multiple bytes of data from slave

Master	ST	SAD+W		SUB		SR	SAD+R			MAK		MAK		NMAK	SP
Slave			SAK		SAK			SAK	DATA		DATA		DATA		

We also need to write data to the sensor to configure the sensor, such as output rate and sensing range. Suppose we wish to write 0x00 to one of the control registers located at 0x10. The master sends the SLA+W first, and then write two bytes of data (subaddress followed by the actual data to write) as illustrated in Table 15 and the figure below.

Table 15. Transfer when master is writing one byte to slave

Master	ST	SAD + W		SUB		DATA		SP
Slave			SAK		SAK		SAK