DevLab: I2C BMI270 6-Axis IMU Sensor

DEVLAB FACTOR

SKU: UE0068

Product Reference Manual (V1.0)

Description

The BMI270 is a compact 6-degree-of-freedom (6DoF) inertial sensor module that integrates a low-power triaxial accelerometer and a high-performance triaxial gyroscope. It provides accurate measurement of linear acceleration and angular rate, making it well suited for motion sensing applications with low power requirements.

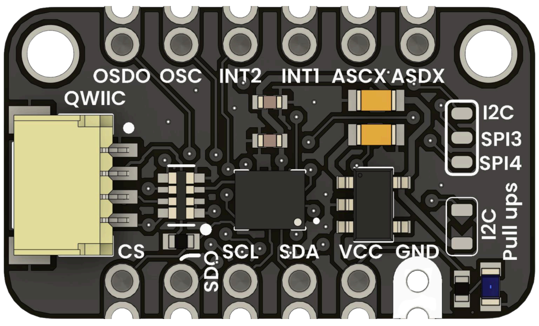
The module is designed for easy integration with microcontrollers and embedded systems, supporting both I²C and SPI interfaces, along with configurable interrupt pins and an auxiliary interface for external sensors. Its small form factor and efficient power management make it ideal for wearables, IoT devices, and motion detection systems.

DevLab format compatibility.

Simplicity and compatibility are the primary objectives of the DevLab form factor. It offers a board layout that is compact and optimized for serial communication, which includes I²C and SPI interfaces.

This format enables the establishment of rapid and dependable connections via a Protocol I²C connector that is entirely compliant with the Qwiic and STEMMA standards.

This design guarantees efficient prototyping, accessibility, and simplicity of integration across a variety of devices and modules.



Key Features

- 6-axis IMU with triaxial accelerometer and triaxial gyroscope
- Accelerometer ranges: ±2 g, ±4 g, ±8 g, ±16 g
- Gyroscope ranges: ±125 dps to ±2000 dps
- Output data rates up to 1.6 kHz (accelerometer) and 6.4 kHz (gyroscope)
- Programmable low-pass filters for accelerometer and gyroscope
- Ultra-low power consumption with integrated power management
- Fast startup time: ~2 ms for accelerometer and gyroscope
- Digital interfaces: I²C (Fast-mode Plus) and SPI (up to 10 MHz)
- Auxiliary I²C interface for external sensors (e.g., magnetometer)
- On-chip FIFO buffer for sensor and timestamp data
- Hardware synchronization and programmable interrupt pins
- Built-in motion detection and activity recognition features
- Small form factor, RoHS compliant

Hardware Features

- Integrated 6-axis IMU (triaxial accelerometer + triaxial gyroscope)
- Selectable acceleration ranges: ±2 g, ±4 g, ±8 g, ±16 g
- Selectable gyroscope ranges: ±125 dps to ±2000 dps
- Digital communication via I²C and SPI
- Programmable interrupt pins for motion and synchronization events
- Auxiliary I²C interface for external sensors
- FIFO buffer for sensor and timestamp data
- Hardware time synchronization between sensor and host
- Integrated power management unit for low-power operation

Software Support

- Official and community-supported drivers available
- Compatible with Arduino, ESP32, RP2040, and embedded Linux platforms
- Register-level configuration for full control of sensor features
- Built-in motion detection and activity recognition functions
- Configurable filters, output data rates, and power modes
- Interrupt-driven operation to reduce host processor load

Applications

- Wearable devices and fitness trackers
- Gesture recognition and motion-based user interfaces
- IoT sensor nodes and smart devices
- Robotics and motion control systems
- Activity monitoring and step counting
- Orientation, tilt, and vibration sensing

CONTENTS

1 The Board

1.1 Accessories

-

2 Ratings

2.1 Recommended Operating Conditions

Symbol	Description	Min	Typ	Max	Unit
VCC	Input supply voltage (external input via VCC pin)	3.6	5.0	6.0	V
V3V3	Regulated 3.3 V output (AP2112K LDO)	3.25	3.3	3.35	V
VIL	Low-level input voltage (I²C/SPI interface)	–0.3	–	0.99	V
VIH	High-level input voltage (I²C/SPI interface)	2.31	–	3.6	V
VOL	Low-level output voltage (at IOL = 3 mA)	–	–	0.4	V
VOH	High-level output voltage (at IOH = –3 mA)	2.4	–	3.3	V
ICC	Typical operating current (BME688 active mode)	–	12	18	mA
ISLEEP	Standby / sleep mode current	–	0.15	0.5	µA
RPU	I²C pull-up resistor to 3.3 V	4.7	–	10	kΩ
TOP	Operating temperature range	–40	–	+85	°C

3 Functional Overview

The BMI270 Sensor Module is a fully integrated inertial sensing solution designed to simplify the use of the Bosch BMI270 IMU in embedded systems. The module combines the inertial sensor, power regulation, interface configuration, and connectivity into a compact, ready-to-use board that can be directly connected to microcontrollers and development platforms without additional external components.

At the core of the module is the BMI270 6-axis inertial measurement unit, which integrates a triaxial accelerometer and a triaxial gyroscope. This sensor provides accurate measurement of linear acceleration and angular velocity, supporting a wide range of motion-sensing use cases such as orientation tracking, activity detection, and gesture recognition. The BMI270 includes internal signal processing, motion detection logic, and FIFO buffering, reducing the computational load on the host processor and enabling efficient, low-latency operation.

Power Architecture and Electrical Operation

The module is designed to operate from a wide external supply voltage applied to the VCC pin, allowing direct connection to common system power rails. An onboard AP2112K low-dropout voltage regulator converts the external supply into a stable 3.3 V internal power domain, which powers the BMI270 sensor and all digital interfaces. This regulated 3.3 V rail ensures consistent logic levels and reliable sensor operation across varying input voltages.

The power architecture supports low-power operation, enabling the module to be used in battery-powered and always-on applications. In sleep and standby modes, the module exhibits extremely low current consumption, while still supporting fast wake-up and rapid sensor startup when motion events occur.

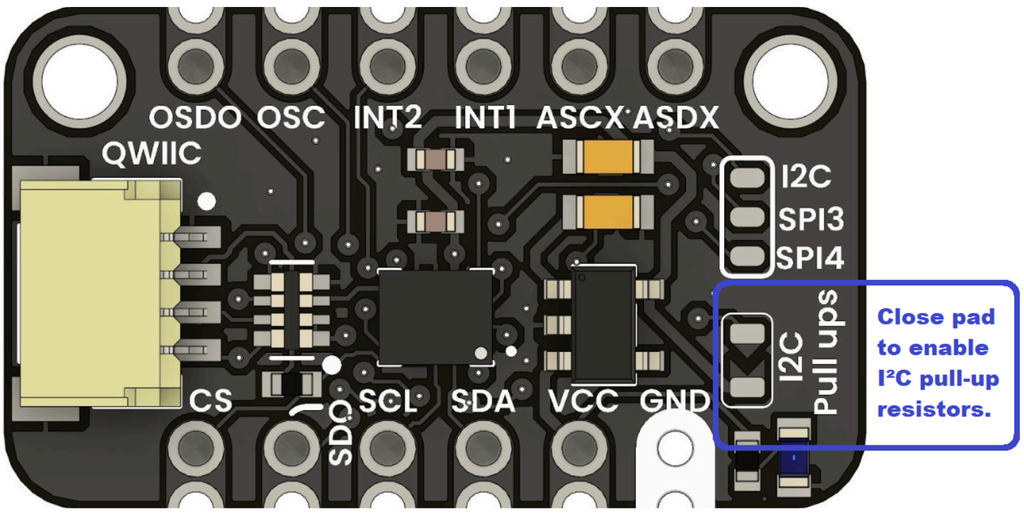
Digital Interfaces and Communication Modes

The BMI270 Sensor Module supports both I²C and SPI communication interfaces, selectable through CS pin header. This flexibility allows the module to be adapted to a wide range of host controllers and system designs.

During power-up or reset, the chip reads the CS level to determine the interface mode:

CS (Chip-Select) start state	Selected mode
CS= HIGH	I²C
CS= LOW	SPI (4 wire)

I²C mode uses the SDA and SCL pins and supports Fast-mode Plus operation. Optional onboard pull-up resistors can be enabled to simplify system integration.



SPI mode can be configured as either 3-wire SPI or 4-wire SPI, allowing compatibility with hosts that require shared or separate data lines.

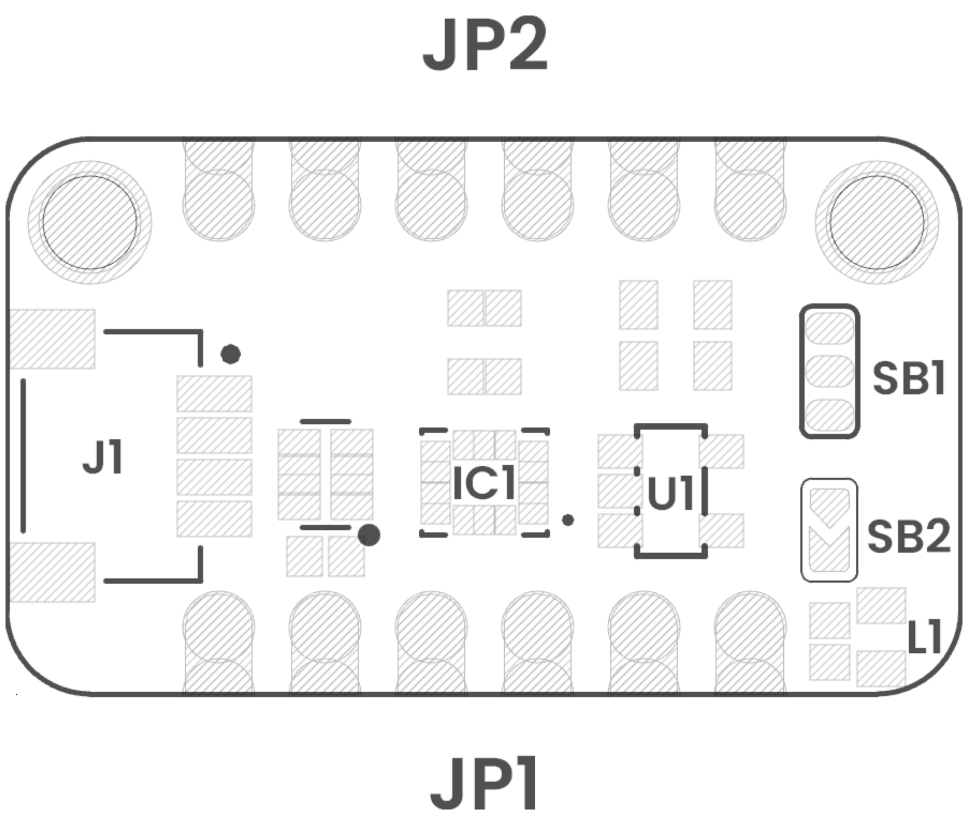
The module also provides a Qwiic-compatible JST connector, enabling fast, solderless I²C connections and full compatibility with Qwiic and STEMMA ecosystems.

Interrupts, Synchronization, and Auxiliary Sensor Support

Two independent interrupt pins (INT1 and INT2) are available and fully configurable. These pins can be used for data-ready signals, FIFO events, motion detection, step counting, and other sensor-generated events. Interrupt-driven operation allows the host processor to remain in low-power states until meaningful motion data is available.

The module exposes the BMI270's auxiliary I²C sensor hub interface, allowing external sensors such as magnetometers or pressure sensors to be connected directly to the BMI270. Data from auxiliary sensors can be synchronized with accelerometer and gyroscope measurements, enabling coherent multi-sensor data acquisition without additional bus management by the host.

3.2 Board Topology



Top View of BMI270 Board

Views of I2C BMI270 Sensor Topology

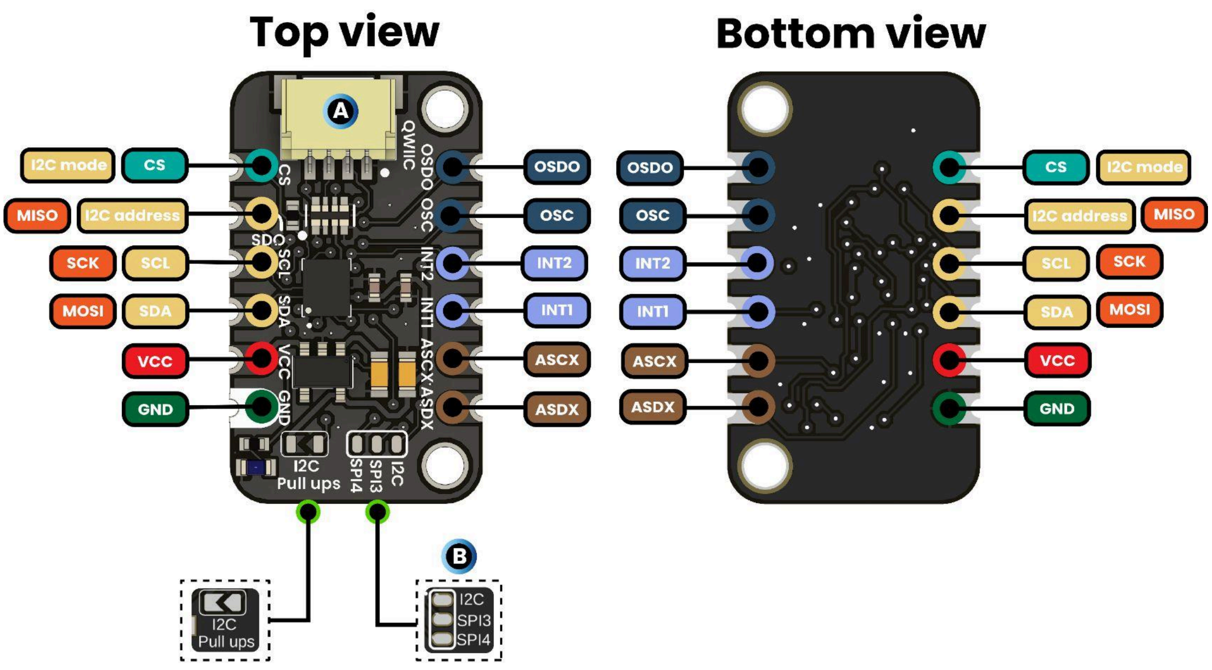
Table 3.2.1 - Components Overview

Ref.	Description
IC1	BMI270 Inertial Sensor
U1	AP2112K 3.3V LDO Voltage Regulator
L1	Power On LED
SB1	Solder bridge to enable mode (I2C, SPI3, SPI4)
SB2	Solder bridge to enable I2C pull-ups
J1	Low-power I2C QWIIC JST Connector
JP1	2.54mm Header
JP2	2.54mm Header

4 Connectors & Pinouts

4.1 General Pinout

BMI270 Sensor Module



Description:

🟥	Supply voltage
🟩	GND
🟨	I2C
🟧	SPI
🟩	Configuration
🟦	Oscillator
🟪	Interruption
🟫	Auxiliary I2C

A qwiic

- GND
- 3.3V
- SDA
- SCL

CS *Chip-Select*

Start state	Selected mode
CS = HIGH	I2C
CS = LOW	SPI (4 wires)

B **Address**
Default = 0x69

I2C SPI3 SPI4	0X68
I2C SPI3 SPI4	0X69

You can change the address by bridging the exposed pads

DevLab BMI270 Sensor Module General Pinout

4.2 Pinout General Description

Pin	Label	Description	Typical Use
OSDO	**OSDO**	Oscillator Output / Crystal Pin (X2)	Optional external clock output or crystal connection.
OSC	**OSC**	Oscillator Input / Crystal Pin (X1)	Optional external clock input or resonator connection.
INT2	**INT2**	Interrupt 2 (polarity configurable)	Generates interrupts for FIFO, step detection, etc.
INT1	**INT1**	Interrupt 1 (polarity configurable)	Data-ready interrupt, motion detection, etc.
AUX_SCL	**ASCX**	Auxiliary I²C clock (sensor-hub)	Clock line for connecting external sensors (e.g. magnetometer).
AUX_SDA	**ASDX**	Auxiliary I²C data (sensor-hub)	Data line for connecting external sensors (e.g. magnetometer).
CS	**CS**	Chip-Select (SPI) – active LOW	Selects the BMI270 when using SPI.
SDO	**SDO**	SPI MISO / I²C-address select	SPI: MISO data output. I²C: sets slave address (0→0x68, 1→0x69).
SCL	**SCL**	I²C clock / SPI clock	Clock line for I²C or SPI bus.
SDA	**SDA**	I²C data / SPI MOSI	Data line for I²C or MOSI in SPI.
VCC	**VCC**	Power supply	Connect to 3.3 V
GND	**GND**	Ground	Connect to system ground.

5 Board Operation

This example demonstrates how to interface the BMI270 Sensor Module with an Arduino-compatible microcontroller using the I²C communication protocol and the SparkFun BMI270 Arduino Library. The sketch initializes the sensor, continuously reads inertial data, and outputs acceleration and angular rate measurements through the serial monitor.

The code is intended as a basic functional reference, showing how to establish communication with the BMI270, verify device presence, and retrieve real-time motion data in a simple and reliable way.

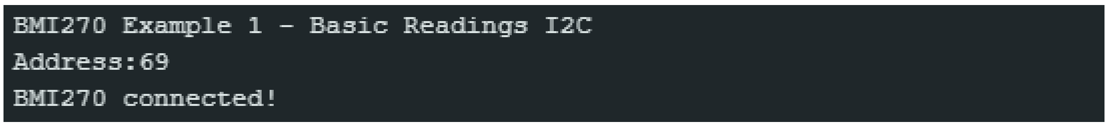
Library Usage and Sensor Initialization

The sketch relies on two main libraries: the Arduino Wire library for I²C communication and the SparkFun BMI270 library, which provides a high-level interface to the sensor's internal registers and data structures. A BMI270 sensor object is created, representing the IMU device and its associated data.

The sensor is configured to use the I²C address (0x69). This address corresponds to the BMI270 standard secondary I²C configuration and matches the module's default hardware setup.

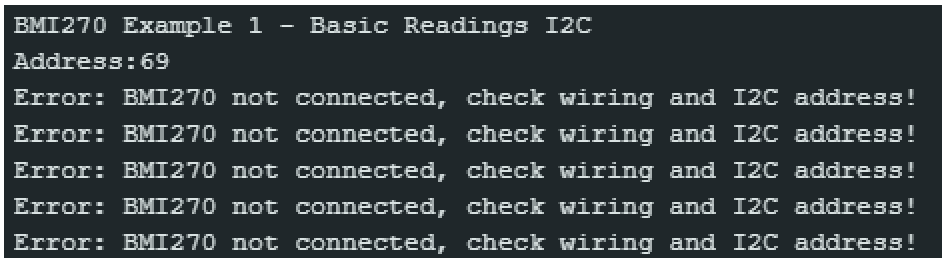
Setup Phase and Communication Check

During the setup phase, serial communication is initialized at 115200 baud to allow data visualization and status messages. The I²C bus is then started, and the sketch attempts to initialize communication with the BMI270 sensor.



```
BMI270 Example 1 - Basic Readings I2C
Address:69
BMI270 connected!
```

If the sensor is not detected, the code repeatedly retries the initialization and reports an error message at one-second intervals. This approach ensures that wiring issues or address mismatches are clearly indicated, and the program only proceeds once the sensor is properly connected and responding.

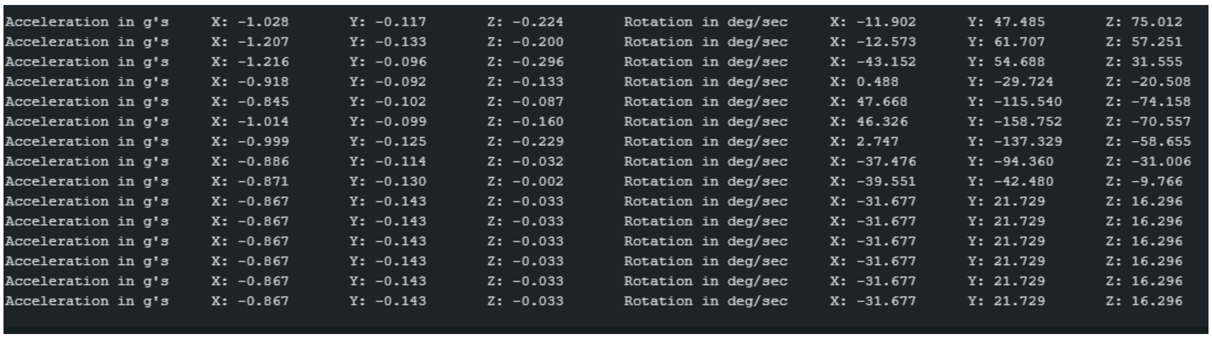


```
BMI270 Example 1 - Basic Readings I2C
Address:69
Error: BMI270 not connected, check wiring and I2C address!
Error: BMI270 not connected, check wiring and I2C address!
Error: BMI270 not connected, check wiring and I2C address!
Error: BMI270 not connected, check wiring and I2C address!
Error: BMI270 not connected, check wiring and I2C address!
```

Sensor Data Acquisition

Inside the main loop, the function responsible for updating sensor measurements is called. This step is essential, as it triggers the BMI270 to refresh its internal data registers. Without this call, the reported sensor values would remain unchanged.

Once updated, the sketch accesses the acceleration and gyroscope data fields provided by the library. Acceleration is reported along the X, Y, and Z axes in g units, while angular velocity is reported in degrees per second (°/s) for each axis. The values are formatted and printed to the serial monitor in a single, readable line.



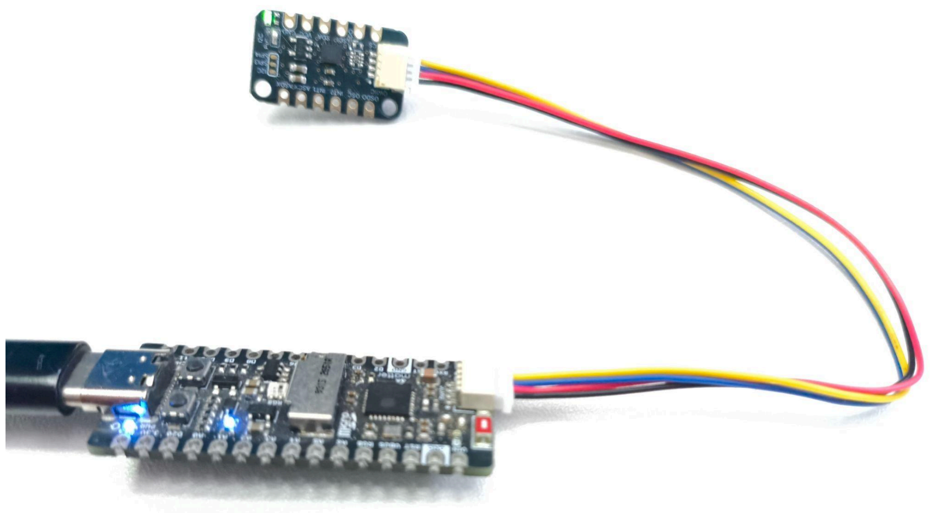
```
Acceleration in g's    X: -1.028    Y: -0.117    Z: -0.224    Rotation in deg/sec    X: -11.902    Y: 47.485      Z: 75.012
Acceleration in g's    X: -1.207    Y: -0.133    Z: -0.200    Rotation in deg/sec    X: -12.573    Y: 61.707      Z: 57.251
Acceleration in g's    X: -1.216    Y: -0.096    Z: -0.296    Rotation in deg/sec    X: -43.152    Y: 54.688      Z: 31.555
Acceleration in g's    X: -0.918    Y: -0.092    Z: -0.133    Rotation in deg/sec    X: 0.488      Y: -29.724     Z: -20.508
Acceleration in g's    X: -0.845    Y: -0.102    Z: -0.087    Rotation in deg/sec    X: 47.668     Y: -115.540    Z: -74.158
Acceleration in g's    X: -1.014    Y: -0.099    Z: -0.160    Rotation in deg/sec    X: 46.326     Y: -158.752    Z: -70.557
Acceleration in g's    X: -0.999    Y: -0.125    Z: -0.229    Rotation in deg/sec    X: 2.747      Y: -137.329    Z: -58.655
Acceleration in g's    X: -0.886    Y: -0.114    Z: -0.032    Rotation in deg/sec    X: -37.476    Y: -94.360     Z: -31.006
Acceleration in g's    X: -0.871    Y: -0.130    Z: -0.002    Rotation in deg/sec    X: -39.551    Y: -42.480     Z: -9.766
Acceleration in g's    X: -0.867    Y: -0.143    Z: -0.033    Rotation in deg/sec    X: -31.677    Y: 21.729      Z: 16.296
Acceleration in g's    X: -0.867    Y: -0.143    Z: -0.033    Rotation in deg/sec    X: -31.677    Y: 21.729      Z: 16.296
Acceleration in g's    X: -0.867    Y: -0.143    Z: -0.033    Rotation in deg/sec    X: -31.677    Y: 21.729      Z: 16.296
Acceleration in g's    X: -0.867    Y: -0.143    Z: -0.033    Rotation in deg/sec    X: -31.677    Y: 21.729      Z: 16.296
Acceleration in g's    X: -0.867    Y: -0.143    Z: -0.033    Rotation in deg/sec    X: -31.677    Y: 21.729      Z: 16.296
Acceleration in g's    X: -0.867    Y: -0.143    Z: -0.033    Rotation in deg/sec    X: -31.677    Y: 21.729      Z: 16.296
```

Sampling Rate and Output Behavior

A fixed delay is introduced at the end of each loop iteration to control the sampling rate. With a delay of 20 milliseconds, the sensor data is read and displayed at approximately 50 Hz, which is suitable for basic motion monitoring, testing, and visualization.

This structure provides a clear example of continuous IMU data acquisition while maintaining predictable timing and low processing overhead.

```
#include <Wire.h>
#include "SparkFun_BMI270_Arduino_Library.h"

// Create a new sensor object
BMI270 imu;

// I2C address selection
uint8_t i2cAddress = BMI2_I2C_PRIM_ADDR+1; // 0x68
//uint8_t i2cAddress = BMI2_I2C_SEC_ADDR; // 0x69

void setup()
{
    // Start serial
    Serial.begin(115200);
    Serial.println("BMI270 Example 1 - Basic Readings I2C");

    // Initialize the I2C library
    Wire.begin();
    // Check if sensor is connected and initialize
    // Address is optional (defaults to 0x68)
    while(imu.beginI2C(i2cAddress) != BMI2_OK)
    {
        // Not connected, inform user
        Serial.println("Error: BMI270 not connected, check wiring and I2C address!");

        // Wait a bit to see if connection is established
        delay(1000);
    }

    Serial.println("BMI270 connected!");
}

void loop()
{
    // Get measurements from the sensor. This must be called before accessing
    // the sensor data, otherwise it will never update
    imu.getSensorData();

    // Print acceleration data
    Serial.print("Acceleration in g's");
    Serial.print("\t");
    Serial.print("X: ");
    Serial.print(imu.data.accelX, 3);
    Serial.print("\t");
    Serial.print("Y: ");
    Serial.print(imu.data.accelY, 3);
    Serial.print("\t");
    Serial.print("Z: ");
    Serial.print(imu.data.accelZ, 3);

    Serial.print("\t");

    // Print rotation data
    Serial.print("Rotation in deg/sec");
    Serial.print("\t");
    Serial.print("X: ");
    Serial.print(imu.data.gyroX, 3);
    Serial.print("\t");
    Serial.print("Y: ");
    Serial.print(imu.data.gyroY, 3);
    Serial.print("\t");
    Serial.print("Z: ");
    Serial.println(imu.data.gyroZ, 3);

    // Print 50x per second
    delay(20);
}
```



Connection of the BMI270 sensor to the UNIT PULSAR C6 development board via the QWIIC I²C interface for demonstrative testing, using the code shown above.

SPI Interface Test and Operation

In addition to the I²C-based example, the BMI270 sensor module was also evaluated using the **SPI communication interface** to demonstrate compatibility with alternative digital interfaces and to validate sensor operation under SPI mode.

This example shows how to interface the BMI270 with an Arduino-compatible microcontroller using the **SPI protocol** and the SparkFun BMI270 Arduino Library. The sketch initializes the sensor over SPI, continuously reads inertial data, and outputs acceleration and angular rate measurements through the serial monitor, in a manner equivalent to the I²C test.

Library Usage and SPI Configuration

The sketch relies on the Arduino **SPI library** for SPI communication and the **SparkFun BMI270 library**, which abstracts the low-level register access and provides a high-level interface to the sensor's data structures.

A BMI270 sensor object is created to represent the IMU device. For SPI operation, two parameters are defined:

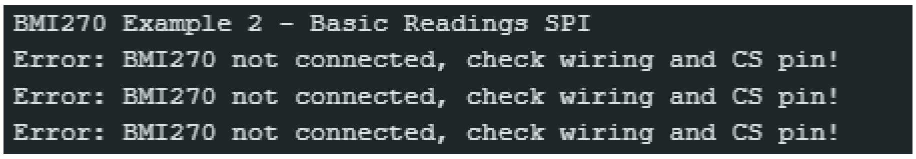
- A **chip select (CS) pin**, used to enable communication with the sensor
- An optional **SPI clock frequency**, set to 100 kHz in this example to ensure reliable communication during testing

These parameters are passed to the library during the sensor initialization process.

Setup Phase and SPI Communication Check

During the setup phase, serial communication is initialized at 115200 baud to display status messages and sensor data. The SPI bus is then initialized, and the sketch attempts to establish communication with the BMI270 using the specified chip select pin.

If the sensor is not detected, the code continuously retries the initialization process and reports an error message every second. This behavior helps identify common issues such as incorrect wiring, an invalid chip select pin, or an incorrect interface mode configuration.



```
BMI270 Example 2 - Basic Readings SPI
Error: BMI270 not connected, check wiring and CS pin!
Error: BMI270 not connected, check wiring and CS pin!
Error: BMI270 not connected, check wiring and CS pin!
```

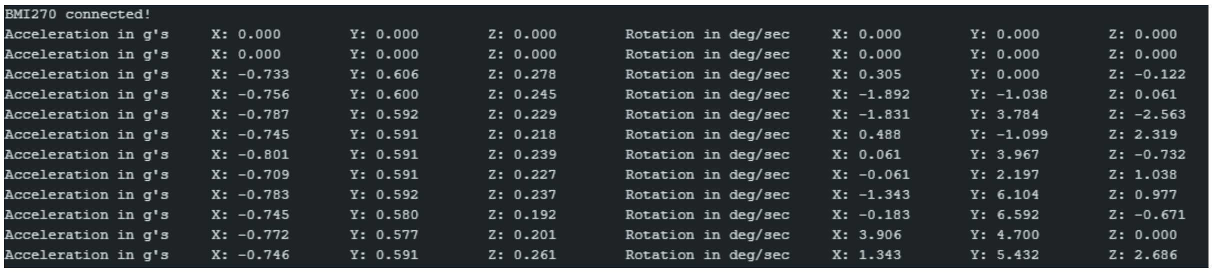
The program proceeds only after successful communication with the sensor has been confirmed.

```
BMI270 connected!
```

Sensor Data Acquisition

Once initialization is complete, the main loop continuously retrieves sensor measurements by calling the data update function provided by the library. This step is required to refresh the BMI270 internal registers and ensure that new measurement data is available.

After updating the sensor data, the sketch accesses the acceleration and gyroscope values for the X, Y, and Z axes. Acceleration data is reported in **g**, while angular velocity is reported in **degrees per second (°/s)**. The values are formatted and printed to the serial monitor in a single, readable line, matching the output format used in the I²C example.



```
BMI270 connected!
Acceleration in g's     X: 0.000       Y: 0.000       Z: 0.000       Rotation in deg/sec     X: 0.000       Y: 0.000       Z: 0.000
Acceleration in g's     X: 0.000       Y: 0.000       Z: 0.000       Rotation in deg/sec     X: 0.000       Y: 0.000       Z: 0.000
Acceleration in g's     X: -0.733      Y: 0.606       Z: 0.278       Rotation in deg/sec     X: 0.305       Y: 0.000       Z: -0.122
Acceleration in g's     X: -0.756      Y: 0.600       Z: 0.245       Rotation in deg/sec     X: -1.892      Y: -1.038      Z: 0.061
Acceleration in g's     X: -0.787      Y: 0.592       Z: 0.229       Rotation in deg/sec     X: -1.831      Y: 3.784       Z: -2.563
Acceleration in g's     X: -0.745      Y: 0.591       Z: 0.218       Rotation in deg/sec     X: 0.488       Y: -1.099      Z: 2.319
Acceleration in g's     X: -0.801      Y: 0.591       Z: 0.239       Rotation in deg/sec     X: 0.061       Y: 3.967       Z: -0.732
Acceleration in g's     X: -0.709      Y: 0.591       Z: 0.227       Rotation in deg/sec     X: -0.061      Y: 2.197       Z: 1.038
Acceleration in g's     X: -0.783      Y: 0.592       Z: 0.237       Rotation in deg/sec     X: -1.343      Y: 6.104       Z: 0.977
Acceleration in g's     X: -0.745      Y: 0.580       Z: 0.192       Rotation in deg/sec     X: -0.183      Y: 6.592       Z: -0.671
Acceleration in g's     X: -0.772      Y: 0.577       Z: 0.201       Rotation in deg/sec     X: 3.906       Y: 4.700       Z: 0.000
Acceleration in g's     X: -0.746      Y: 0.591       Z: 0.261       Rotation in deg/sec     X: 1.343       Y: 5.432       Z: 2.686
```

Sampling Rate and Output Behavior

A fixed delay of 20 milliseconds is introduced at the end of each loop iteration. This results in an effective sampling and output rate of approximately **50 Hz**, which is suitable for functional testing, motion visualization, and interface validation.

The SPI-based example demonstrates that the BMI270 can provide stable and continuous inertial measurements over SPI, offering an alternative communication option for applications requiring higher robustness, flexible pin assignment, or coexistence with multiple I²C devices.

```
#include <SPI.h>
#include "SparkFun_BMI270_Arduino_Library.h"

// Create a new sensor object
BMI270 imu;

// SPI parameters
uint8_t chipSelectPin = D10;
uint32_t clockFrequency = 100000;

void setup()
{
  // Start serial
  Serial.begin(115200);
  Serial.println("BMI270 Example 2 - Basic Readings SPI");

  // Initialize the SPI library
  SPI.begin();

  // Check if sensor is connected and initialize
  // Clock frequency is optional (defaults to 100kHz)
  while(imu.beginSPI(chipSelectPin, clockFrequency) != BMI2_OK)
  {
    // Not connected, inform user
    Serial.println("Error: BMI270 not connected, check wiring and CS pin!");

    // Wait a bit to see if connection is established
    delay(1000);
  }

  Serial.println("BMI270 connected!");
}

void loop()
{
  // Get measurements from the sensor. This must be called before accessing
  // the sensor data, otherwise it will never update
  imu.getSensorData();

  // Print acceleration data
  Serial.print("Acceleration in g's");
  Serial.print("\t");
  Serial.print("X: ");
  Serial.print(imu.data.accelX, 3);
  Serial.print("\t");
  Serial.print("Y: ");
  Serial.print(imu.data.accelY, 3);
  Serial.print("\t");
```

```
    Serial.print("Z: ");
    Serial.print(imu.data.accelZ, 3);

    Serial.print("\t");

    // Print rotation data
    Serial.print("Rotation in deg/sec");
    Serial.print("\t");
    Serial.print("X: ");
    Serial.print(imu.data.gyroX, 3);
    Serial.print("\t");
    Serial.print("Y: ");
    Serial.print(imu.data.gyroY, 3);
    Serial.print("\t");
    Serial.print("Z: ");
    Serial.println(imu.data.gyroZ, 3);

    // Print 50x per second
    delay(20);
}
```

SPI Connection Table

The following table summarizes the required connections between the **BMI270 sensor module** and the microcontroller when operating in **SPI mode**. The module pin names are mapped to their corresponding SPI signals on the microcontroller.

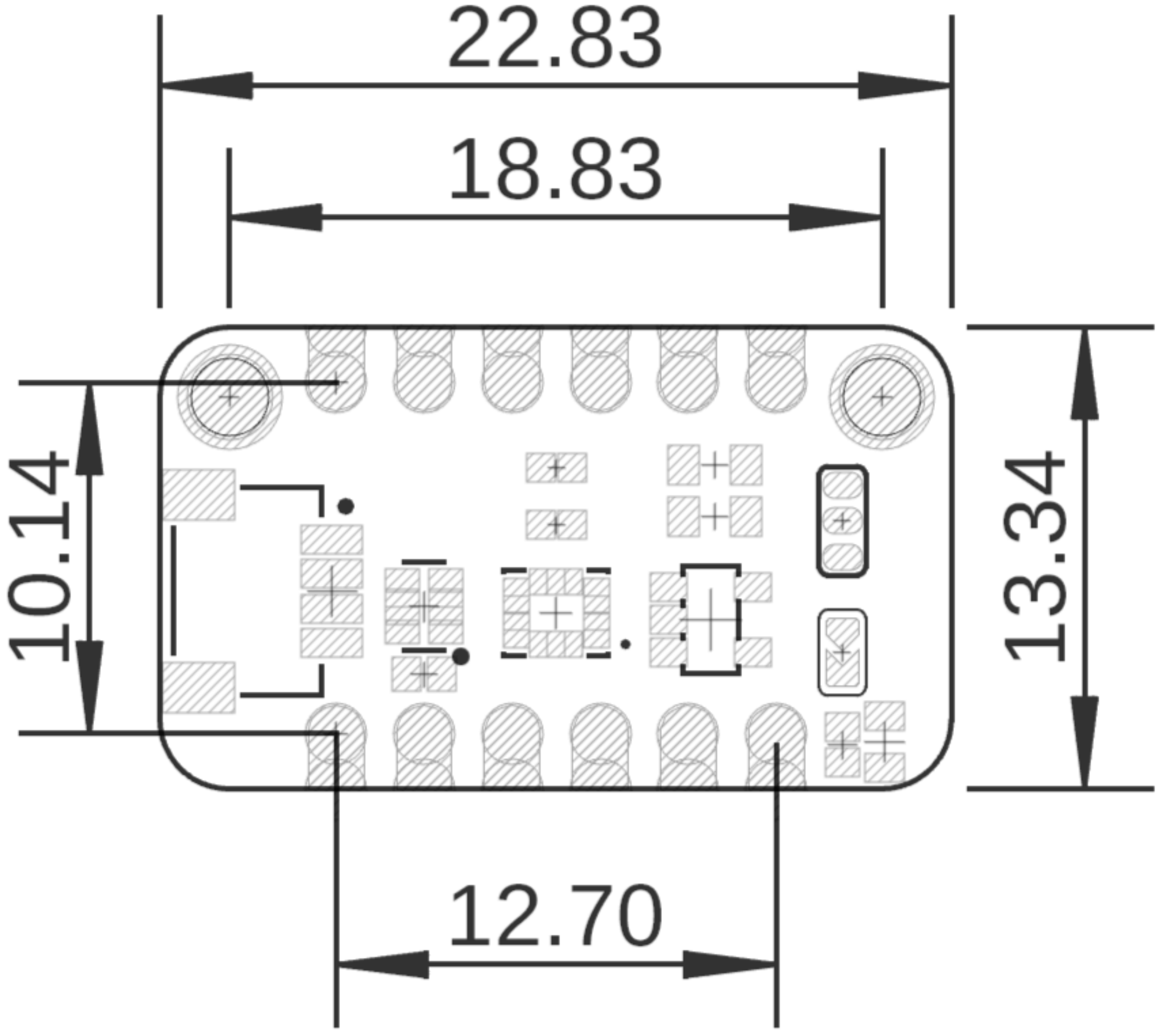
BMI270 Module Pin	Microcontroller Connection
CS	CS (Chip Select)
SDO	MISO
SDA	MOSI
SCL	SCK

Description

The table above shows the SPI signal mapping between the BMI270 module and the microcontroller. When using SPI communication, the CS (Chip Select) pin must be held LOW during power-up or reset to enable SPI mode. Unlike I²C operation, where CS is pulled HIGH, SPI mode requires explicit control of the CS line to select the sensor and allow data exchange.

This configuration ensures proper initialization of the BMI270 in 4-wire SPI mode and reliable communication during sensor operation.

6 Mechanical Information



Mechanical dimensions in millimeters

Mechanical dimensions in millimeters

7 Company Information

Company name	UNIT Electronics
Company website	https://uelectronics.com/
Company Address	Salvador 19, Cuauhtémoc, 06000 Mexico City, CDMX

8 Reference Documentation

Ref	**Link**
Documentation	https://wiki.uelectronics.com/wiki/devlab_bmi270_inertial_sensor_module
Github	https://github.com/UNIT-Electronics-MX/unit_devlab_i2c_bmi270_6_axis_imu_sensor
Thonny IDE	https://thonny.org/
Arduino IDE	https://www.arduino.cc/en/software
Visual Studio Code	https://code.visualstudio.com/download
SparkFun BMI270 Arduino Library	https://github.com/sparkfun/SparkFun_BMI270_Arduino_Library

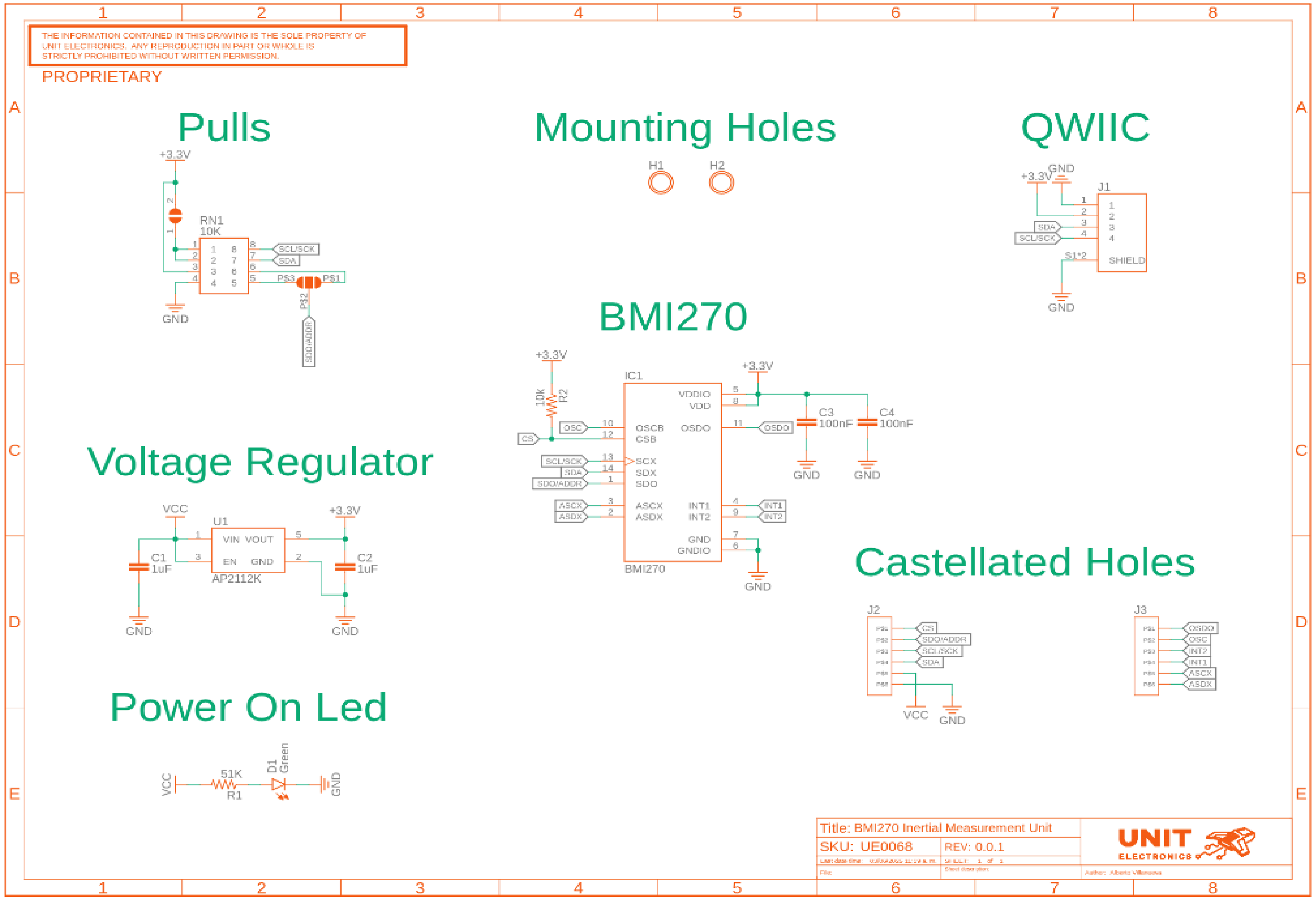
9 Appendix

9.1 Schematic
(https://github.com/UNIT-Electronics-MX/unit_devlab_i2c_bmi270_6_axis_imu_sensor/blob/main/hardware/unit_sch_V_0_0_1_ue0068_bmi270.pdf)

DevLab: I2C BMI270 6-Axis IMU Sensor

DEVLAB FACTOR

SKU: UE0068



THE INFORMATION CONTAINED IN THIS DRAWING IS THE SOLE PROPERTY OF UNIT ELECTRONICS. ANY REPRODUCTION IN PART OR WHOLE IS STRICTLY PROHIBITED WITHOUT WRITTEN PERMISSION.
PROPRIETARY
Pulls
+3.3V
RN1
10K
SCL/SCK
SDA
PS3
PS1
PS2
SDO/ADDR
GND
Mounting Holes
H1
H2
QWIIC
+3.3V
GND
J1
SDA
SCL/SCK
S1*2
SHIELD
GND
BMI270
+3.3V
10K
R2
IC1
VDDIO
VDD
OSC
CS
OSCB
CSB
OSDO
SCL/SCK
SDA
SDO/ADDR
SCX
SDX
SDO
ASCX
ASDX
INT1
INT2
GND
GNDIO
BMI270
C3
100nF
C4
100nF
GND
GND
GND
Voltage Regulator
VCC
U1
VIN
VOUT
EN
GND
AP2112K
+3.3V
C1
1uF
C2
1uF
GND
GND
Castellated Holes
J2
CS
SDO/ADDR
SCL/SCK
SDA
VCC
GND
J3
OSDO
OSC
INT2
INT1
ASCX
ASDX
Power On Led
VCC
51K
R1
D1
Green
GND
Title: BMI270 Inertial Measurement Unit
SKU: UE0068
REV: 0.0.1
UNIT ELECTRONICS
Author: Alberto Villanueva