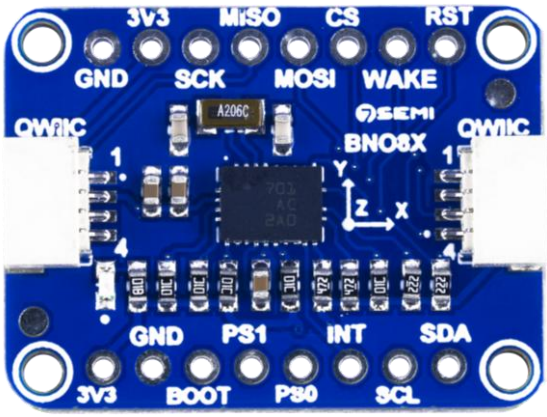
7SEMI

BNO085 9-DOF Absolute Orientation Sensor Breakout I2C Qwiic

Version 1.0

Table of Contents

1.Introduction



The **7Semi BNO085 9-DOF Orientation IMU Fusion Breakout Qwiic** is an advanced sensor module integrating a **triaxial accelerometer, gyroscope, and magnetometer**, making it ideal for **motion tracking, orientation sensing, and VR/AR applications**. The sensor runs **CEVA's SH-2 firmware**, which provides **real-time sensor fusion**, ensuring accurate orientation and motion analysis.

1.1 Features

- 9 Degrees of Freedom (9-DOF): Includes a 3-axis accelerometer, 3-axis gyroscope, and 3-axis magnetometer.
- Sensor Fusion Algorithm: Provides accurate 3D orientation, heading, and angular velocity calculations.
- Multiple Output Formats:
  - Quaternion-based rotation vectors
  - Euler angles (roll, pitch, yaw)
  - Gravity vector and linear acceleration
  - Angular velocity
- Qwiic-Compatible: Seamless integration with Qwiic ecosystem via I2C interface.
- Low Power Consumption: Optimized for battery-powered applications.
- Firmware Upgrade Support: Allows DFU (Device Firmware Update) over I2C, SPI, or UART.

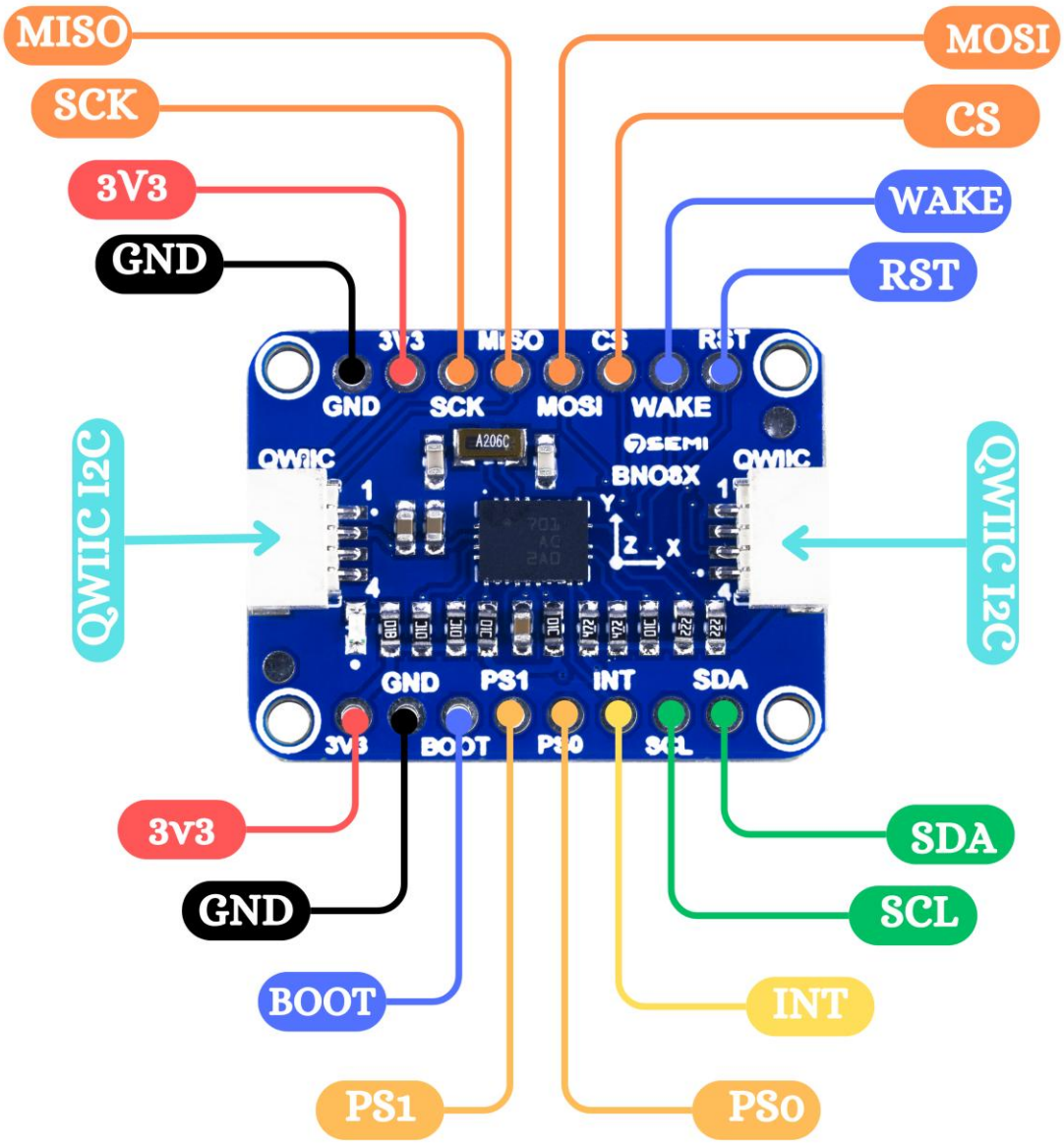
2.Technical Specification

The **Technical Specification** table provides detailed information about **7Semi BNO085 9-DOF Orientation IMU Fusion Breakout Qwiic**, including its operating voltage, current consumption, and electrical characteristics. This data helps users understand the power requirements, communication parameters, and performance capabilities of the sensor. It ensures compatibility with different microcontrollers and embedded systems while providing guidelines for efficient integration into various applications.

BNO085 Specifications

- Accelerometer Range: ±8g
- Gyroscope Range: ±2000°/s
- Magnetometer Range: ±8 Gauss
- Communication Interfaces:
  - I²C (Qwiic)
  - SPI
  - UART
- Operating Voltage: 1.8V - 3.6V
- Power Consumption: Low-power mode available for energy-efficient operation
- Maximum Sensor Data Rates:
  - Gyro Rotation Vector: Up to 1000Hz
  - Rotation Vector & Game Rotation Vector: Up to 400Hz
  - Geomagnetic Rotation Vector: Up to 90Hz
  - Accelerometer: Up to 500Hz
  - Gyroscope: Up to 400Hz
  - Magnetometer: Up to 100Hz
- Firmware Upgrade: Supports Device Firmware Update (DFU) over I²C, SPI, or UART
- Package Dimensions: 3.8mm x 5.2mm x 1.1mm

3.Pinouts

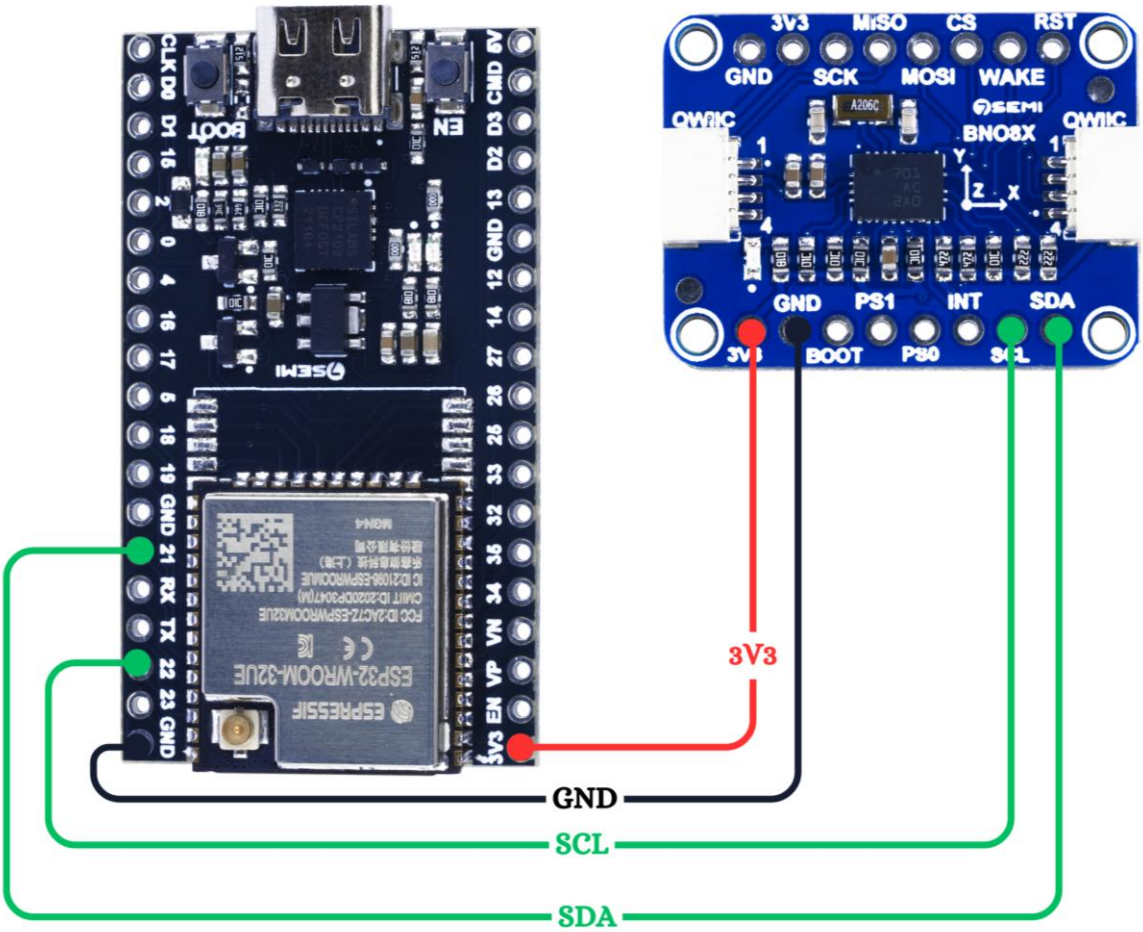

MISO
SCK
3V3
GND
MOSI
CS
WAKE
RST
QWIIC I2C
QWIIC I2C
3V3
GND
BOOT
PS1
INT
PS0
SDA
SCL
GND
SCK
MOSI
WAKE
A206C
BNO8X
QWIIC
QWIIC
GND
PS1
INT
SDA

Pin Number	Pin Name	Description
1	3V3	Power Input (3.3V)
2	GND	Ground Connection
3	MISO	SPI Data Output (Master In Slave Out)
4	SCK	SPI Clock
5	MOSI	SPI Data Input (Master Out Slave In)
6	CS	SPI Chip Select
7	WAKE	Wake-up pin for the IMU
8	RST	Reset pin
9	BOOT	Bootloader mode activation
10	PS1	Mode selection pin
11	PS0	Mode selection pin
12	INT	Interrupt pin
13	SCL	I²C Clock Line
14	SDA	I²C Data Line
15	Qwiic	Easy plug-and-play I²C communication

Connection Guidelines

- For I²C communication, connect SCL to the microcontroller's SCL pin and SDA to the SDA pin.
- For SPI communication, use MOSI, MISO, SCK, and CS accordingly.
- The BOOT pin is used for firmware updates or entering bootloader mode.
- The WAKE pin can be used to wake up the sensor from low-power mode.
- The INT pin serves as an interrupt output for motion detection events.

4.Hardware Interface



Connection Explanation :

- Connect ESP32 3.3V to BNO085 3V3 for power.
- Connect ESP32 GND to BNO085 GND for a common ground.
- Connect ESP32 GPIO 21 (SDA) to BNO085 SDA for data communication.
- Connect ESP32 GPIO 22 (SCL) to BNO085 SCL for the clock signal.
- This configuration enables communication between the ESP32 and BNO085 using the I²C protocol.

5.Example code link

We provide example codes to help you get started with the **7Semi BNO085 9-DOF Absolute Orientation Sensor Breakout I2C Qwiic**. These examples demonstrate how to communicate with the sensor and retrieve **absolute orientation, acceleration, gyroscope, and magnetometer data** using the **I²C protocol**. The code is available for two popular platforms: **Arduino and ESP32**.

Code Explanation

```
#include "Adafruit_BNO08x_RVC.h"

Adafruit_BNO08x_RVC rvc = Adafruit_BNO08x_RVC();

void setup() {

  // Wait for serial monitor to open

  Serial.begin(115200);

  while (!Serial)

    delay(10);

  Serial.println("Adafruit BNO08x IMU - UART-RVC mode");

  Serial1.begin(115200); // This is the baud rate specified by the
datasheet

  while (!Serial1)

    delay(10);

  if (!rvc.begin(&Serial1)) { // connect to the sensor over hardware
serial

    Serial.println("Could not find BNO08x!");

    while (1)

      delay(10);

  }
```

```
  Serial.println("BNO08x found!");
}
void loop() {
  BNO08x_RVC_Data heading;

  if (!rvc.read(&heading)) {
    return;
  }
  Serial.println();
  Serial.println(F("---------------------------------------"));
  Serial.println(F("Principal Axes:"));
  Serial.println(F("---------------------------------------"));
  Serial.print(F("Yaw: "));
  Serial.print(heading.yaw);
  Serial.print(F("\tPitch: "));
  Serial.print(heading.pitch);
  Serial.print(F("\tRoll: "));
  Serial.println(heading.roll);
  Serial.println(F("---------------------------------------"));
  Serial.println(F("Acceleration"));
  Serial.println(F("---------------------------------------"));
  Serial.print(F("X: "));
  Serial.print(heading.x_accel);
  Serial.print(F("\tY: "));
  Serial.print(heading.y_accel);
```

```
    Serial.print(F("\tZ: "));
    Serial.println(heading.z_accel);
    Serial.println(F("---------------------------------------"));
    //  delay(200);
  }
```

1. Library Inclusions

- #include <Arduino.h> → Standard Arduino functions.
- #include <Adafruit_BNO08x.h> → Library to interface with the BNO085 IMU sensor.

2. Pin Definitions

- #define BNO08X_CS 10 → Chip Select (CS) for SPI communication.
- #define BNO08X_INT 9 → Interrupt pin to detect new sensor data.
- #define BNO08X_RESET -1 → Reset pin, not required for I²C/UART.

3. Fast Mode Toggle

- FAST_MODE enabled → Uses SH2_GYRO_INTEGRATED_RV for higher refresh rates (~1000Hz) but potentially more noise.
- FAST_MODE disabled → Uses SH2_ARVR_STABILIZED_RV for more accurate orientation tracking (~250Hz).
- The report frequency is set using reportIntervalUs.

4. Sensor Initialization (setup())

- Starts Serial Communication at 115200 baud rate.
- Waits for Serial Monitor to open (for boards like Leonardo, which require a delay).
- Initializes BNO085 sensor using I²C communication.
- If initialization fails, the program enters an infinite loop.

- Enables quaternion sensor reports based on the selected mode.
- Displays a success message if the sensor is detected.

5. Setting Sensor Reports (setReports())

- Configures the BNO085 sensor to send quaternion data at a specified interval.
- If enabling the report fails, it prints an error message.

6. Converting Quaternion to Euler Angles (quaternionToEuler())

- Processes quaternion data (qr, qi, qj, qk) from the sensor.
- Uses trigonometric functions (atan2, asin) to compute:
  - Yaw (Rotation around Z-axis).
  - Pitch (Rotation around X-axis).
  - Roll (Rotation around Y-axis).
- Converts the values to degrees if required.

7. Handling Different Quaternion Reports

- quaternionToEulerRV() → Converts ARVR Stabilized Rotation Vector to Yaw, Pitch, Roll.
- quaternionToEulerGI() → Converts Gyro Integrated Rotation Vector to Yaw, Pitch, Roll.

8. Main Loop (loop())

- Checks if the sensor has reset and re-enables reports if needed.
- Fetches new sensor data from BNO085.
- Depending on the report type:
  - ARVR Stabilized Report (SH2_ARVR_STABILIZED_RV) → Used for more accurate orientation tracking.

  - Gyro Integrated Report (SH2_GYRO_INTEGRATED_RV) → Used for faster, high-frequency tracking.
- Processes quaternion data into Euler angles.
- Outputs the following data to the Serial Monitor:
  - Time elapsed since the last measurement.
  - Sensor accuracy level (0-3, where 3 is the best accuracy).
  - Yaw, Pitch, and Roll values.

9. Arduino Example Code

This example is designed for Arduino-compatible boards and demonstrates:

- Initializing the I²C communication with the **BNO085** sensor Board.
- Reading absolute orientation, acceleration, gyroscope, and magnetometer data.
- Printing the sensor data to the Serial Monitor.

10.ESP32 Example Code

This example targets ESP32 boards and showcases:

- Configuring the ESP32 I²C interface to communicate with the **BNO085** sensor Board.
- Reading absolute orientation, acceleration, gyroscope, and magnetometer data.
- Printing the sensor data to the Serial Monitor.

How to Access the Code

- **Download Link for Arduino and ESP32 Example**: Click Here

5.1 Sample Serial Output Arduino

Sample output Image of Arduino:

```
---------------------------------------
Principal Axes:
---------------------------------------
Yaw: 23.70      Pitch: -3.76    Roll: 6.17
---------------------------------------
Acceleration
---------------------------------------
X: -0.82        Y: -0.51        Z: 7.50
---------------------------------------

---------------------------------------
Principal Axes:
---------------------------------------
Yaw: 23.70      Pitch: -3.76    Roll: 6.17
---------------------------------------
Acceleration
---------------------------------------
X: -0.82        Y: -0.47        Z: 7.53
---------------------------------------
```

6.Mechanical Specification