# BOSCH SENSORTEC:

## CONSUMER INERTIAL MEMS – HIGH TECH IN YOUR HANDS

PoliMi, 01.12.2023

***Francesco Sechi**, Inertial Sensor Expert*

***Leonardo Gaffuri Pagani**, Inertial Sensor Expert*

01

**Introduction**

Bosch and Bosch Sensortec.

02

**Technology**

Our technical solutions.

03

**Hands-on**

Now it’s your turn.

# University program

01

**Introduction**

Bosch and Bosch Sensortec.

02

**Technology**

Our technical solutions.

03

**Hands-on**

Now it’s your turn.

01

## Introduction

Bosch and Bosch Sensortec.

02

## Technology

Our technical solutions.

03

## Hands-on

Now it’s your turn.

# Hands-on Session
## Low entry barrier

How to easily start working with sensors?

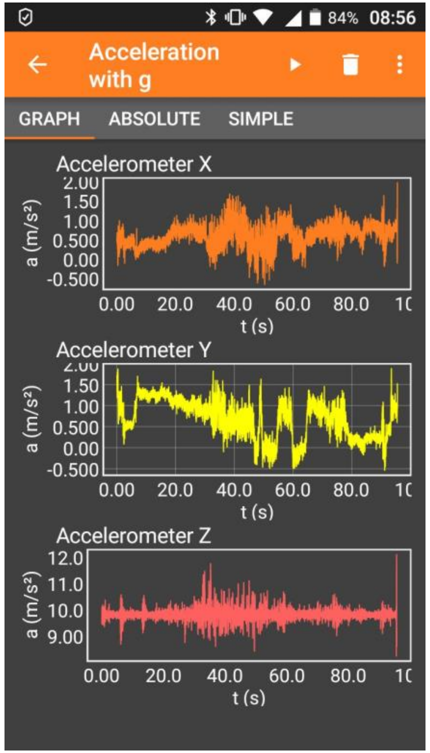
How to evaluate sensor data?

phyphox®
physical phone experiments

# Hands-on Session

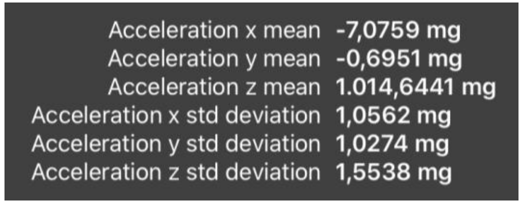
## Use Smart Device and App

1. Get *phyphox* mobile app at https://phyphox.org/
2. Read sensors and log data
3. Download and process data, compute offset, noise, sensitivity, etc
4. Further step…
   create your own experiment at https://phyphox.org/editor/



# Hands-on Session
## Use Smart Device and App

1. **Statistics Accelerometer**
2. Statistics Accelerometer/Gyroscope
3. Sensitivity Gyroscope



```
       Acceleration x mean  -7,0759 mg
       Acceleration y mean  -0,6951 mg
       Acceleration z mean  1.014,6441 mg
Acceleration x std deviation  1,0562 mg
Acceleration y std deviation  1,0274 mg
Acceleration z std deviation  1,5538 mg
```

# Hands-on Session

## Use Smart Device and App

1. Statistics Accelerometer
2. **Statistics Accelerometer/Gyroscope**
3. Sensitivity Gyroscope

# Hands-on Session

## Use Smart Device and App

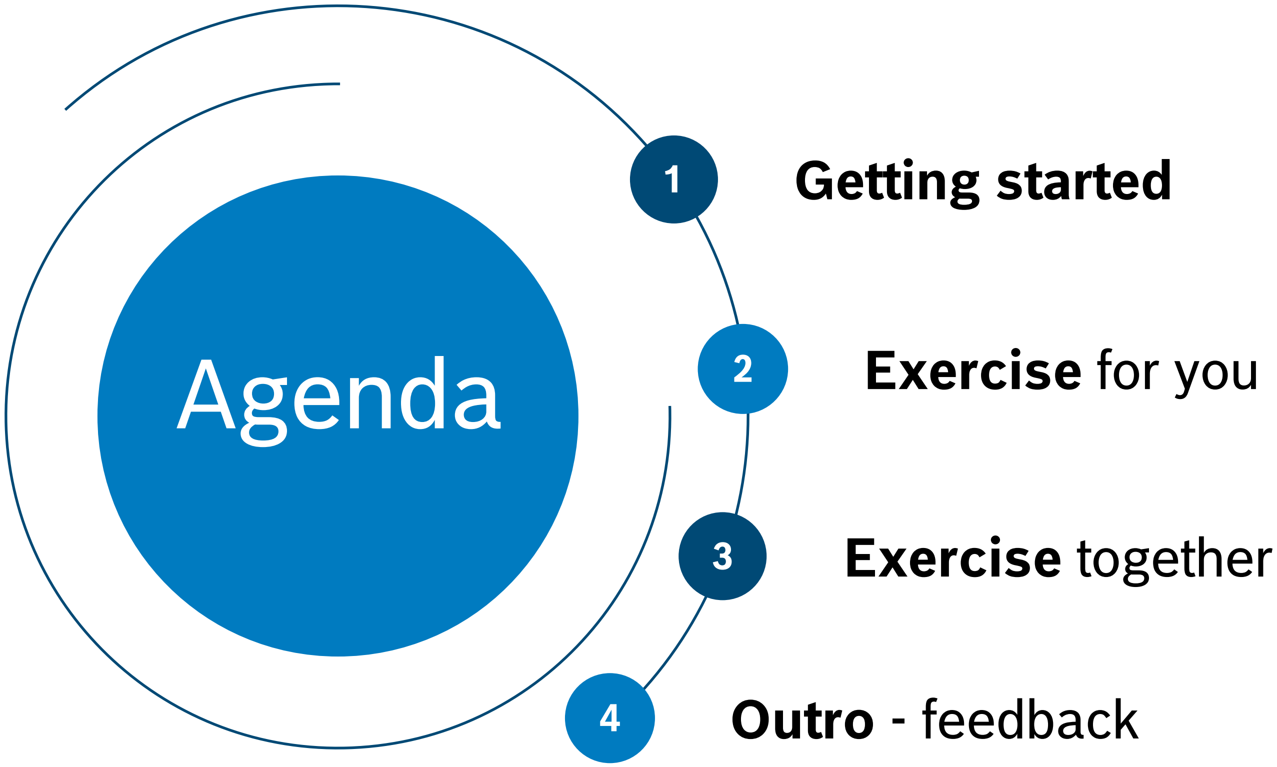
1. Statistics Accelerometer
2. Statistics Accelerometer Gyroscope
3. **Sensitivity Gyroscope**

# Nicla Sense ME
# Hands-on Session

BOSCH

# Nicla Sense ME Hands-on Session

## Agenda



Agenda

1 **Getting started**

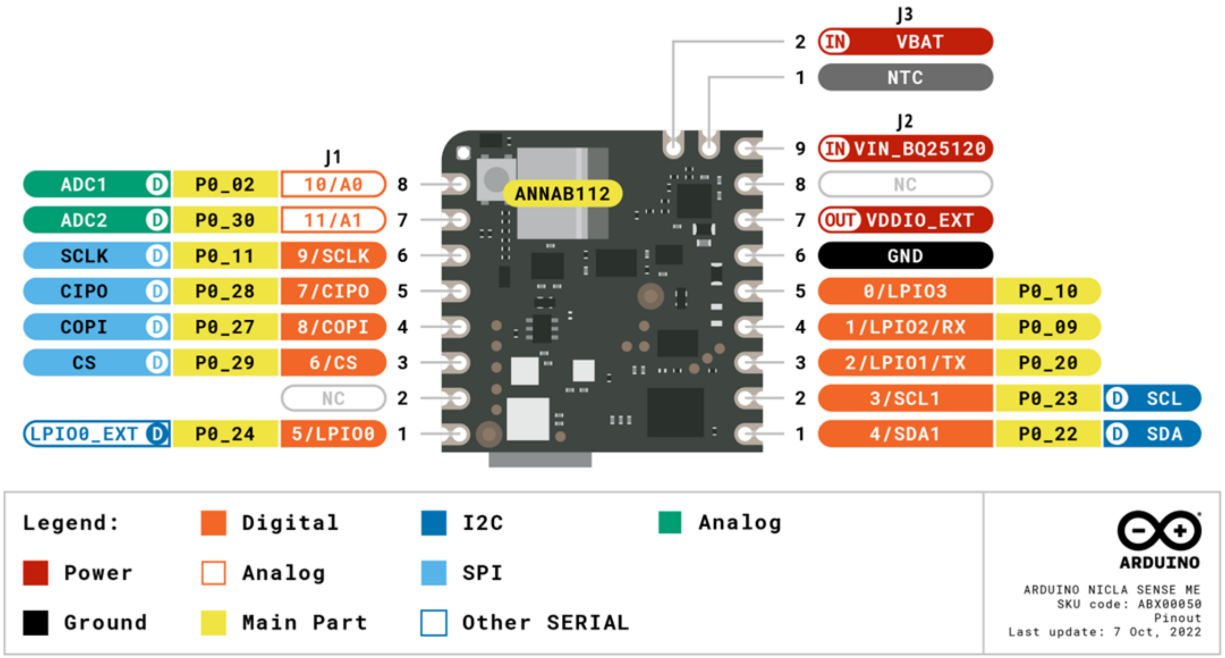
2 **Exercise** for you

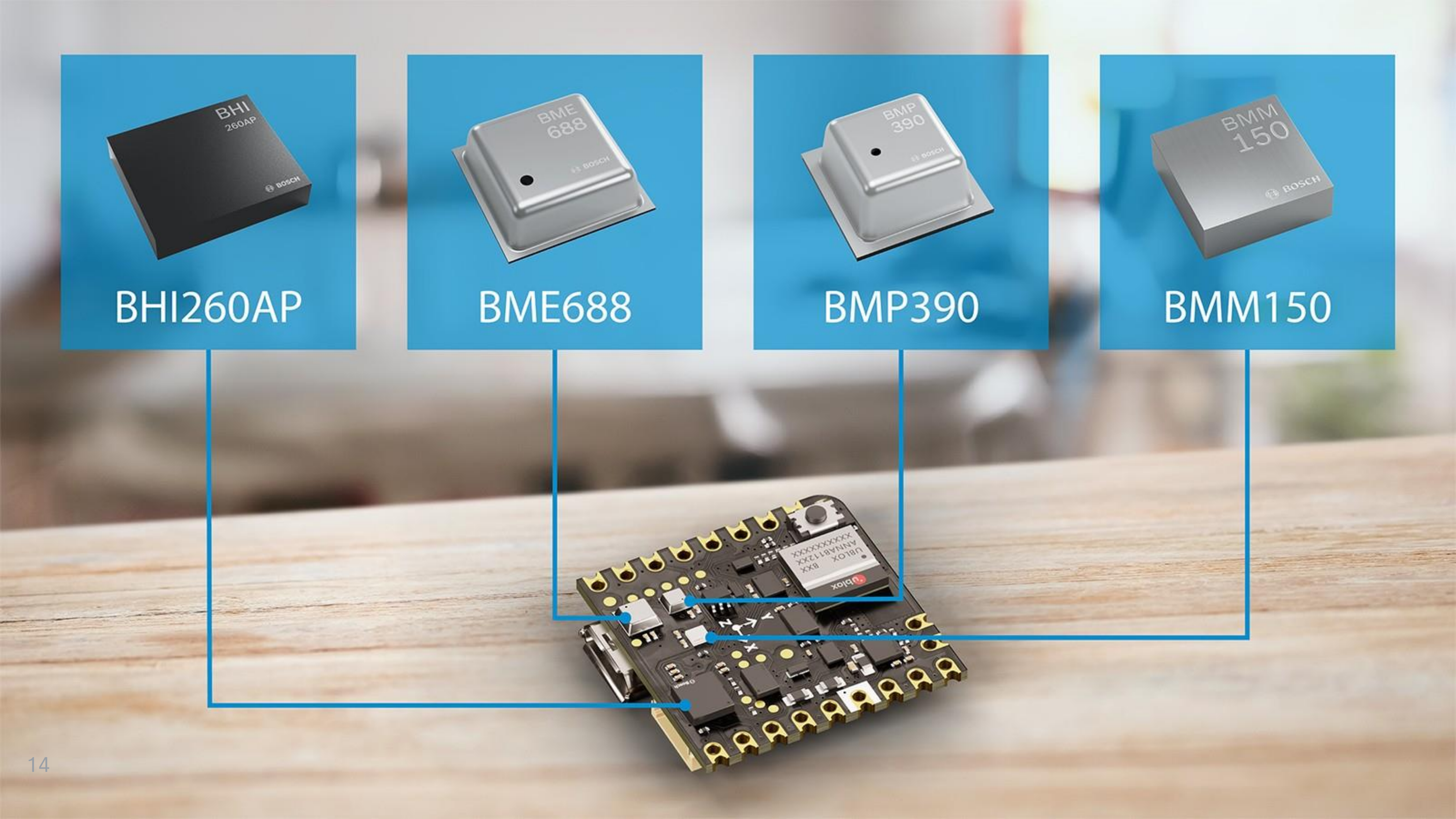
3 **Exercise** together

4 **Outro** - feedback

# Nicla Sense ME

| Parameter | Technical data |
|---|---|
| Processor | 64 MHz Arm® Cortex M4 (nRF52832) |
| I/O | Castellated pins with the following features:<br>• 1x I2C bus (with ext. ESLOV)<br>• 1x serial port<br>• 1x SPI<br>• 2x ADC<br>• Programmable I/O voltage from 1.8-3.3V |
| Dimensions | 22,86 mm x 22,86 mm |
| Power | • USB<br>• Pin Header<br>• 3.7V Li-po battery, Integrated charger |
| Connectivity | Bluetooth 5, BLE |
| Memory | • 512KB Flash / 64KB RAM<br>• 2MB SPI Flash for storage<br>• 2MB QSPI dedicated for BHI260AP |
| Interface | USB interface with debug functionality |





BHI260AP

BME688

BMP390

BMM150

# Introduction
## Bosch Sensors

### BHI260AP

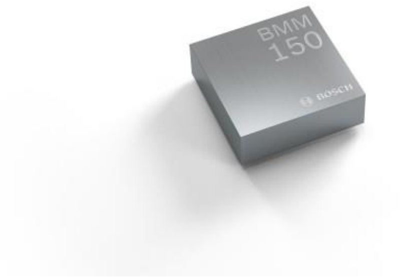
Smart sensor that includes in one package many software functionalities, 32-bit customer programmable microcontroller, 6-axis IMU.

- Self-learning AI software
- Low power pedestrian position tracking
- Personalized fitness tracking & swim analytics

### BMM150

Low-power, low-noise 3-axis digital geomagnetic sensor, provides absolute spatial orientation and motion vectors with high accuracy and dynamics via dedicated data fusion software.

- Outdoor/indoor navigation
- Head movement tracking



# Introduction
## Bosch Sensors

### BME688

The first gas sensor with Artificial Intelligence (AI) and integrated high-linearity and high-accuracy pressure, humidity and temperature sensors.
VOCs and other gases (e.g., CO and H) detection in the ppb range.

- Specific detection of Volatile Sulfur Compounds (VSCs)
- Application-specific gas scanner
- BME AI-Studio software

### BMP390

Small, low-power and low-noise 24-bit absolute barometric pressure sensor.
Digital, high-performance sensor for a wide range of altitude tracking applications (smartphones, GPS modules, wearables, hearables, drones, etc.)

- Unique accuracy and stability
- Lowest noise

# Getting Started
## Environment Setup

Tool Chain Setup:

- Arduino IDE 1.8.19
- Open from Tools -> Manage Libraries: "Arduino_BHY2", "Arduino_BHY2Host", "ArduinoBLE“
- Open from Tools -> Boards Manager:  "Arduino Mbed OS Nicla Boards“
- File -> Examples -> Arduino_BHY2 -> Standalone
  press on *Verify* to check that it compiles

# Getting Started
## Web Resources

**Nicla Sense ME - User Guide, Technical Specifications & Product Documentation**
**Bosch Sensortec Community**

**Arduino IDE Download Page**

Learn more about:

- BHI260AP
- BMM150
- BME688
- BMP390

Sensor Classes
Sensor IDs

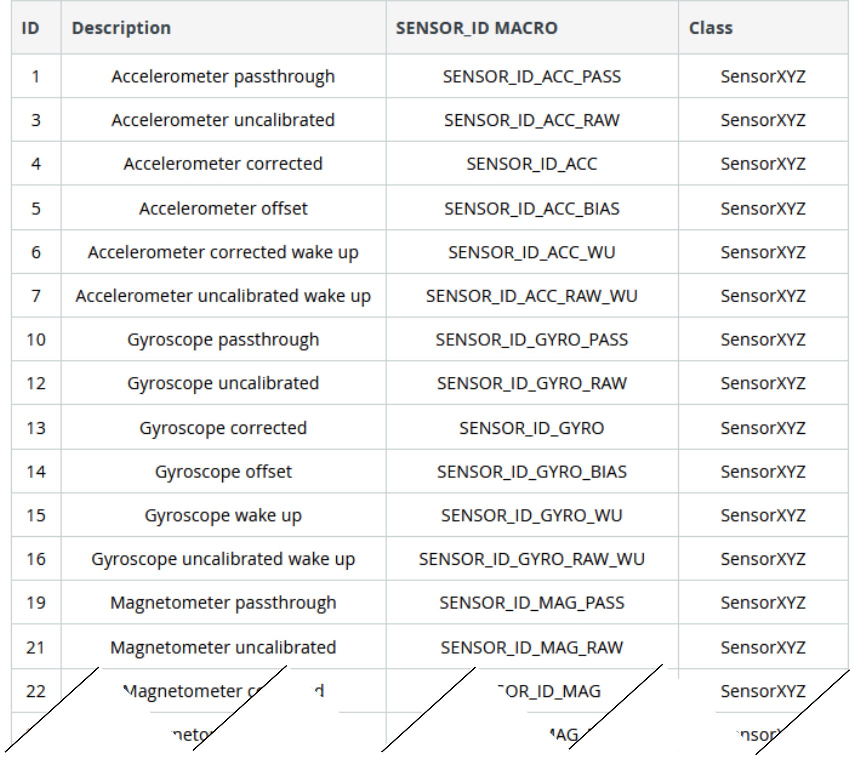
# Getting Started
## Read Sensors Data

```
#include "Arduino_BHY2.h"

// Create a reference to the accel sensor
SensorXYZ accel(SENSOR_ID_ACC);

void setup() {

}

void loop() {

}
```

## Sensor Classes

- Sensor: single values (e.g., temperature, pressure, …)
- SensorXYZ: XYZ values (e.g., accel, gyro, …)
- …

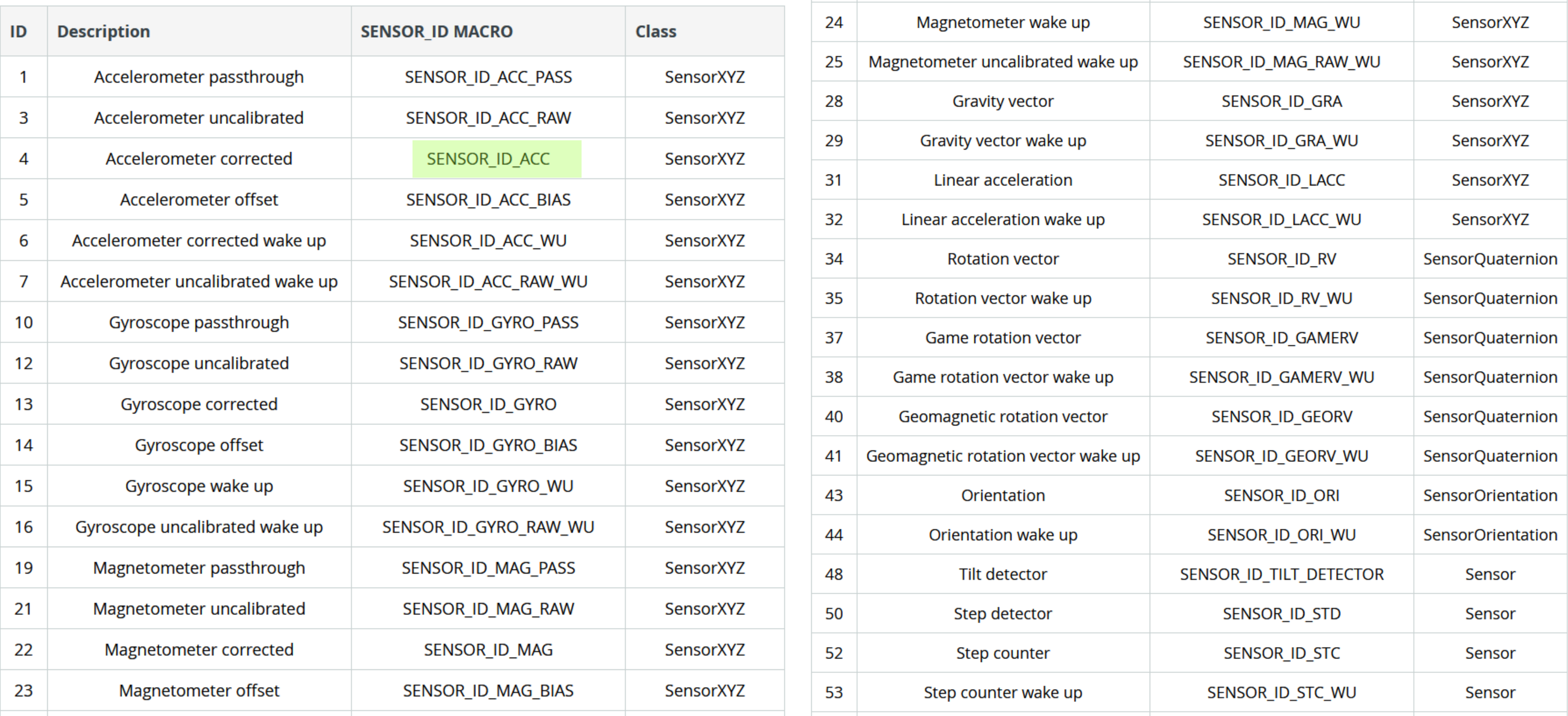
## Sensor IDs



| ID | Description | SENSOR_ID MACRO | Class |
|---|---|---|---|
| 1 | Accelerometer passthrough | SENSOR_ID_ACC_PASS | SensorXYZ |
| 3 | Accelerometer uncalibrated | SENSOR_ID_ACC_RAW | SensorXYZ |
| 4 | Accelerometer corrected | SENSOR_ID_ACC | SensorXYZ |
| 5 | Accelerometer offset | SENSOR_ID_ACC_BIAS | SensorXYZ |
| 6 | Accelerometer corrected wake up | SENSOR_ID_ACC_WU | SensorXYZ |
| 7 | Accelerometer uncalibrated wake up | SENSOR_ID_ACC_RAW_WU | SensorXYZ |
| 10 | Gyroscope passthrough | SENSOR_ID_GYRO_PASS | SensorXYZ |
| 12 | Gyroscope uncalibrated | SENSOR_ID_GYRO_RAW | SensorXYZ |
| 13 | Gyroscope corrected | SENSOR_ID_GYRO | SensorXYZ |
| 14 | Gyroscope offset | SENSOR_ID_GYRO_BIAS | SensorXYZ |
| 15 | Gyroscope wake up | SENSOR_ID_GYRO_WU | SensorXYZ |
| 16 | Gyroscope uncalibrated wake up | SENSOR_ID_GYRO_RAW_WU | SensorXYZ |
| 19 | Magnetometer passthrough | SENSOR_ID_MAG_PASS | SensorXYZ |
| 21 | Magnetometer uncalibrated | SENSOR_ID_MAG_RAW | SensorXYZ |
| 22 | Magnetometer c d | OR_ID_MAG | SensorXYZ |
|  | neto | AG | nsor |

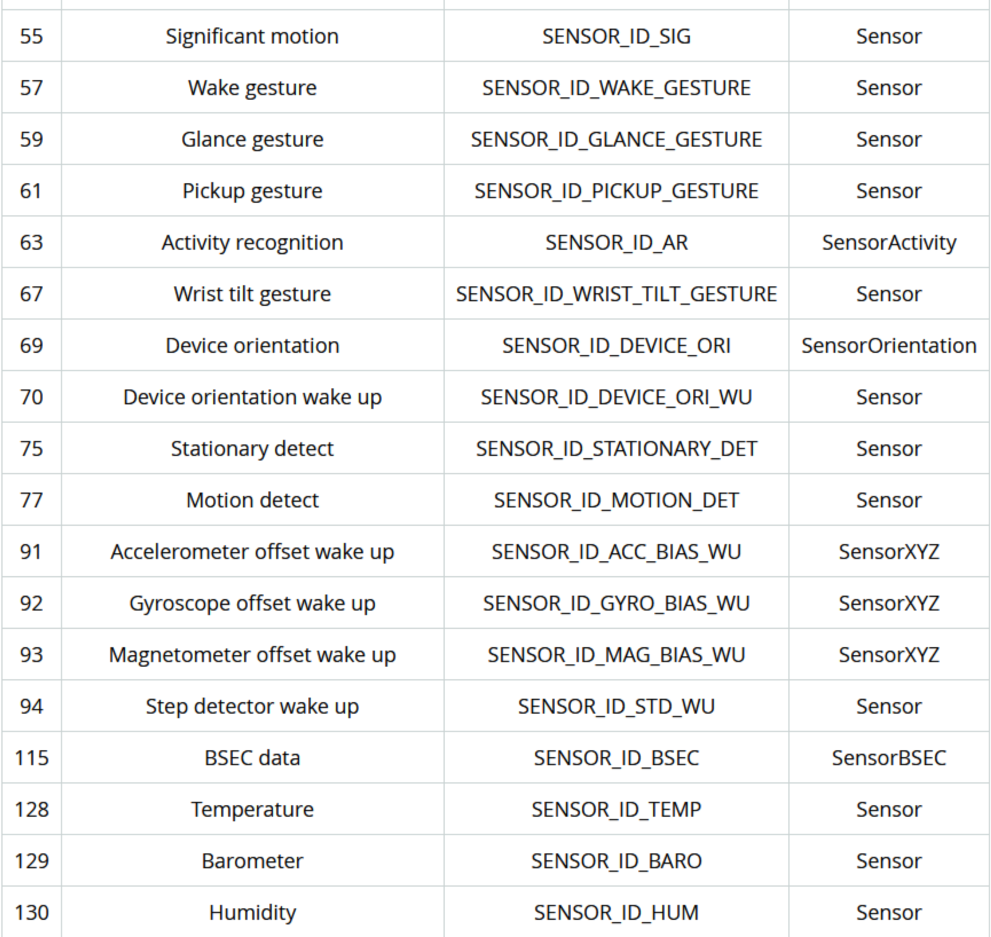
# Getting Started

## Read Sensors Data

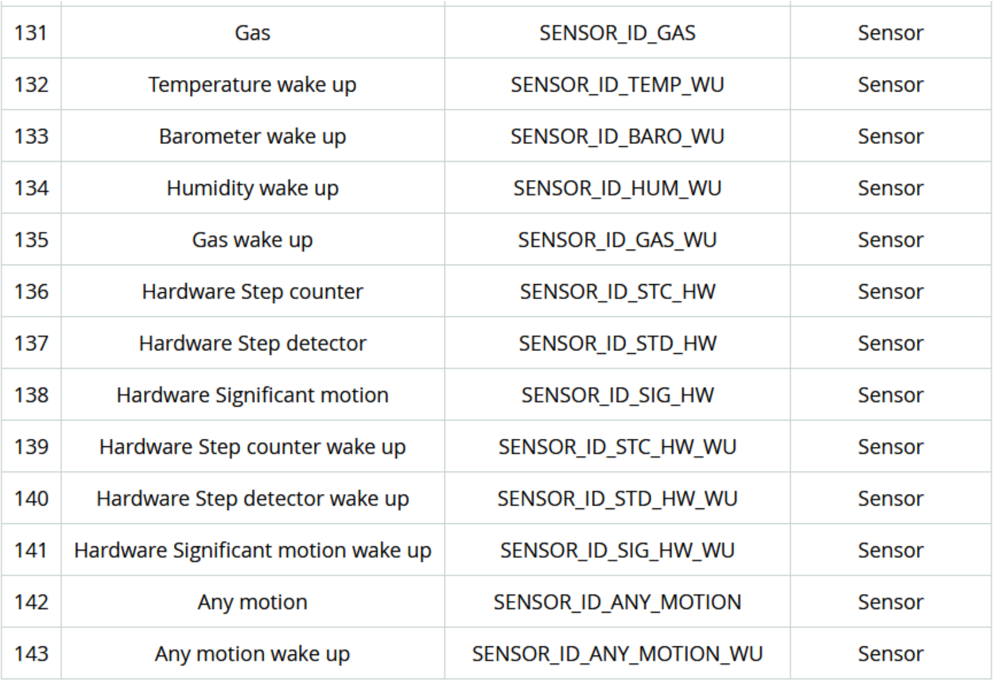


| ID | Description | SENSOR_ID MACRO | Class |
|---|---|---|---|
| 1 | Accelerometer passthrough | SENSOR_ID_ACC_PASS | SensorXYZ |
| 3 | Accelerometer uncalibrated | SENSOR_ID_ACC_RAW | SensorXYZ |
| 4 | Accelerometer corrected | SENSOR_ID_ACC | SensorXYZ |
| 5 | Accelerometer offset | SENSOR_ID_ACC_BIAS | SensorXYZ |
| 6 | Accelerometer corrected wake up | SENSOR_ID_ACC_WU | SensorXYZ |
| 7 | Accelerometer uncalibrated wake up | SENSOR_ID_ACC_RAW_WU | SensorXYZ |
| 10 | Gyroscope passthrough | SENSOR_ID_GYRO_PASS | SensorXYZ |
| 12 | Gyroscope uncalibrated | SENSOR_ID_GYRO_RAW | SensorXYZ |
| 13 | Gyroscope corrected | SENSOR_ID_GYRO | SensorXYZ |
| 14 | Gyroscope offset | SENSOR_ID_GYRO_BIAS | SensorXYZ |
| 15 | Gyroscope wake up | SENSOR_ID_GYRO_WU | SensorXYZ |
| 16 | Gyroscope uncalibrated wake up | SENSOR_ID_GYRO_RAW_WU | SensorXYZ |
| 19 | Magnetometer passthrough | SENSOR_ID_MAG_PASS | SensorXYZ |
| 21 | Magnetometer uncalibrated | SENSOR_ID_MAG_RAW | SensorXYZ |
| 22 | Magnetometer corrected | SENSOR_ID_MAG | SensorXYZ |
| 23 | Magnetometer offset | SENSOR_ID_MAG_BIAS | SensorXYZ |
| 24 | Magnetometer wake up | SENSOR_ID_MAG_WU | SensorXYZ |
| 25 | Magnetometer uncalibrated wake up | SENSOR_ID_MAG_RAW_WU | SensorXYZ |
| 28 | Gravity vector | SENSOR_ID_GRA | SensorXYZ |
| 29 | Gravity vector wake up | SENSOR_ID_GRA_WU | SensorXYZ |
| 31 | Linear acceleration | SENSOR_ID_LACC | SensorXYZ |
| 32 | Linear acceleration wake up | SENSOR_ID_LACC_WU | SensorXYZ |
| 34 | Rotation vector | SENSOR_ID_RV | SensorQuaternion |
| 35 | Rotation vector wake up | SENSOR_ID_RV_WU | SensorQuaternion |
| 37 | Game rotation vector | SENSOR_ID_GAMERV | SensorQuaternion |
| 38 | Game rotation vector wake up | SENSOR_ID_GAMERV_WU | SensorQuaternion |
| 40 | Geomagnetic rotation vector | SENSOR_ID_GEORV | SensorQuaternion |
| 41 | Geomagnetic rotation vector wake up | SENSOR_ID_GEORV_WU | SensorQuaternion |
| 43 | Orientation | SENSOR_ID_ORI | SensorOrientation |
| 44 | Orientation wake up | SENSOR_ID_ORI_WU | SensorOrientation |
| 48 | Tilt detector | SENSOR_ID_TILT_DETECTOR | Sensor |
| 50 | Step detector | SENSOR_ID_STD | Sensor |
| 52 | Step counter | SENSOR_ID_STC | Sensor |
| 53 | Step counter wake up | SENSOR_ID_STC_WU | Sensor |

# Getting Started
## Read Sensors Data



| | | | |
|---|---|---|---|
| 55 | Significant motion | SENSOR_ID_SIG | Sensor |
| 57 | Wake gesture | SENSOR_ID_WAKE_GESTURE | Sensor |
| 59 | Glance gesture | SENSOR_ID_GLANCE_GESTURE | Sensor |
| 61 | Pickup gesture | SENSOR_ID_PICKUP_GESTURE | Sensor |
| 63 | Activity recognition | SENSOR_ID_AR | SensorActivity |
| 67 | Wrist tilt gesture | SENSOR_ID_WRIST_TILT_GESTURE | Sensor |
| 69 | Device orientation | SENSOR_ID_DEVICE_ORI | SensorOrientation |
| 70 | Device orientation wake up | SENSOR_ID_DEVICE_ORI_WU | Sensor |
| 75 | Stationary detect | SENSOR_ID_STATIONARY_DET | Sensor |
| 77 | Motion detect | SENSOR_ID_MOTION_DET | Sensor |
| 91 | Accelerometer offset wake up | SENSOR_ID_ACC_BIAS_WU | SensorXYZ |
| 92 | Gyroscope offset wake up | SENSOR_ID_GYRO_BIAS_WU | SensorXYZ |
| 93 | Magnetometer offset wake up | SENSOR_ID_MAG_BIAS_WU | SensorXYZ |
| 94 | Step detector wake up | SENSOR_ID_STD_WU | Sensor |
| 115 | BSEC data | SENSOR_ID_BSEC | SensorBSEC |
| 128 | Temperature | SENSOR_ID_TEMP | Sensor |
| 129 | Barometer | SENSOR_ID_BARO | Sensor |
| 130 | Humidity | SENSOR_ID_HUM | Sensor |

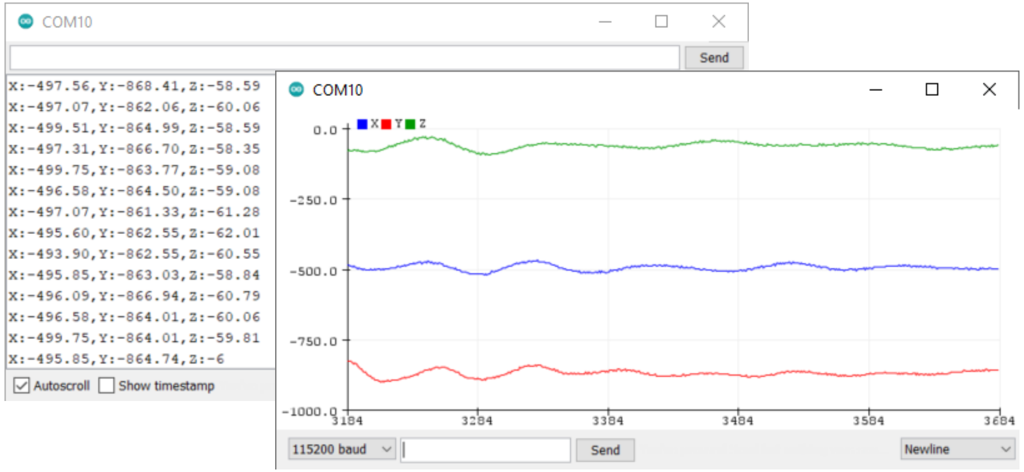


| | | | |
|---|---|---|---|
| 131 | Gas | SENSOR_ID_GAS | Sensor |
| 132 | Temperature wake up | SENSOR_ID_TEMP_WU | Sensor |
| 133 | Barometer wake up | SENSOR_ID_BARO_WU | Sensor |
| 134 | Humidity wake up | SENSOR_ID_HUM_WU | Sensor |
| 135 | Gas wake up | SENSOR_ID_GAS_WU | Sensor |
| 136 | Hardware Step counter | SENSOR_ID_STC_HW | Sensor |
| 137 | Hardware Step detector | SENSOR_ID_STD_HW | Sensor |
| 138 | Hardware Significant motion | SENSOR_ID_SIG_HW | Sensor |
| 139 | Hardware Step counter wake up | SENSOR_ID_STC_HW_WU | Sensor |
| 140 | Hardware Step detector wake up | SENSOR_ID_STD_HW_WU | Sensor |
| 141 | Hardware Significant motion wake up | SENSOR_ID_SIG_HW_WU | Sensor |
| 142 | Any motion | SENSOR_ID_ANY_MOTION | Sensor |
| 143 | Any motion wake up | SENSOR_ID_ANY_MOTION_WU | Sensor |

# Getting Started
## Read Sensors Data

```
#include "Arduino_BHY2.h"

#define ACCEL_FS8G_CONV_FACTOR 0.24414

// Create a reference to the accel sensor
SensorXYZ accel(SENSOR_ID_ACC);

void setup() {
  // Setup the serial communication
  Serial.begin(115200);
  while(!Serial);

  // Setup the BHY and the sensors of interest
  BHY2.begin();
  accel.begin();
}

void loop() {
  // Update function should be continuously polled
  BHY2.update();
```

```
  // Print data
  Serial.print(String("X:") + String(accel.x() *
ACCEL_FS8G_CONV_FACTOR));
  Serial.print(",");
  Serial.print(String("Y:") + String(accel.y() *
ACCEL_FS8G_CONV_FACTOR));
  Serial.print(",");
  Serial.println(String("Z:") + String(accel.z() *
ACCEL_FS8G_CONV_FACTOR));
}
```



```
X:-497.56,Y:-868.41,Z:-58.59
X:-497.07,Y:-862.06,Z:-60.06
X:-499.51,Y:-864.99,Z:-58.59
X:-497.31,Y:-866.70,Z:-58.35
X:-499.75,Y:-863.77,Z:-59.08
X:-496.58,Y:-864.50,Z:-59.08
X:-497.07,Y:-861.33,Z:-61.28
X:-495.60,Y:-862.55,Z:-62.01
X:-493.90,Y:-862.55,Z:-60.55
X:-495.85,Y:-863.03,Z:-58.84
X:-496.09,Y:-866.94,Z:-60.79
X:-496.58,Y:-864.01,Z:-60.06
X:-499.75,Y:-864.01,Z:-59.81
X:-495.85,Y:-864.74,Z:-6
```

☑ Autoscroll ☐ Show timestamp

115200 baud

Newline

# Exercise for you

## Sensor Data on LED

# Exercise for you
## Sensor Data on LED

- Task:
  show the data you prefer on the RGB LED

- Including Nicla library

```
#include <Nicla_System.h>
```

- In the setup() function

```
// Setup the Nicla library and the LED
nicla::begin();
nicla::leds.begin();
// Configure the built-in LED as output
pinMode(LED_BUILTIN, OUTPUT);
```

- In the loop() function

```
// Setting LED color:
nicla::leds.setColor(cyan);
// function input can be a single keyword
(off, red, green, blue, yellow. Magenta, cyan)
// or also the three RGB values from 0 to 255 for red, green and blue
nicla::leds.setColor(red_val, green_val, blue_val);
```

- Extra: how not to run the loop as fast as possible:

```
void loop(){
        static auto lastCheck = millis();
        BHY2.update();

        If (millis() - lastCheck >= 150){
                // your code here
                lastCheck = millis();
        }
}
```

# Exercise – An example
## Gravity on RGB LED

```
#include "Arduino_BHY2.h"
#include <Nicla_System.h>

#define DEBUG
#define ACCEL_FS8G_CONV_FACTOR 0.24414

// Create the structure which will contain time and data
struct Data {
  float ts;
  float x;
  float y;
  float z;
};

// Create a reference to the accel sensor
SensorXYZ accel(SENSOR_ID_ACC);
```

```
void setup() {
  // Setup the serial communication
  Serial.begin(115200);
  while(!Serial);

  // Setup the BHY and the sensors of interest
  BHY2.begin();
  accel.begin();

  // Setup the Nicla library and the LED
  nicla::begin();
  nicla::leds.begin();

  // Configure the built-in LED as output
  pinMode(LED_BUILTIN, OUTPUT);
}
```

# Exercise – An example
## Gravity on RGB LED

```
void loop() {
 // Static time, stored only the first iteration
 static auto now = millis();

 // Update function should be continuously polled
 BHY2.update();

 // Do something after every 250 ms
 if (millis() - now >= 250) {
  now = millis();

  // Read the accel content
  Data data { now, accel.x(), accel.y(), accel.z() };
```

```
// Color logic:
  // - Ratio of each axis to accel magnitude mapped to (0, 255) range
  float magnitude = sqrt(pow(data.x, 2) + pow(data.y, 2) + pow(data.z, 2));
  short red = abs(data.x)/magnitude * 255;
  short green = abs(data.y)/magnitude * 255;
  short blue = abs(data.z)/magnitude * 255;
  nicla::leds.setColor(red, green, blue);

#ifdef DEBUG // Print debug messages
  Serial.println(String("time [ms]: ") + String(data.ts));
  Serial.println(String("acceleration [mg]:"));
  Serial.println(String("\tX: ") + String(data.x * ACCEL_FS8G_CONV_FACTOR));
  Serial.println(String("\tY: ") + String(data.y * ACCEL_FS8G_CONV_FACTOR));
  Serial.println(String("\tZ: ") + String(data.z * ACCEL_FS8G_CONV_FACTOR));
#endif
 }
}
```

# Guided exercise

# Sensor Data Leveraging

How can we fully leverage our sensors?

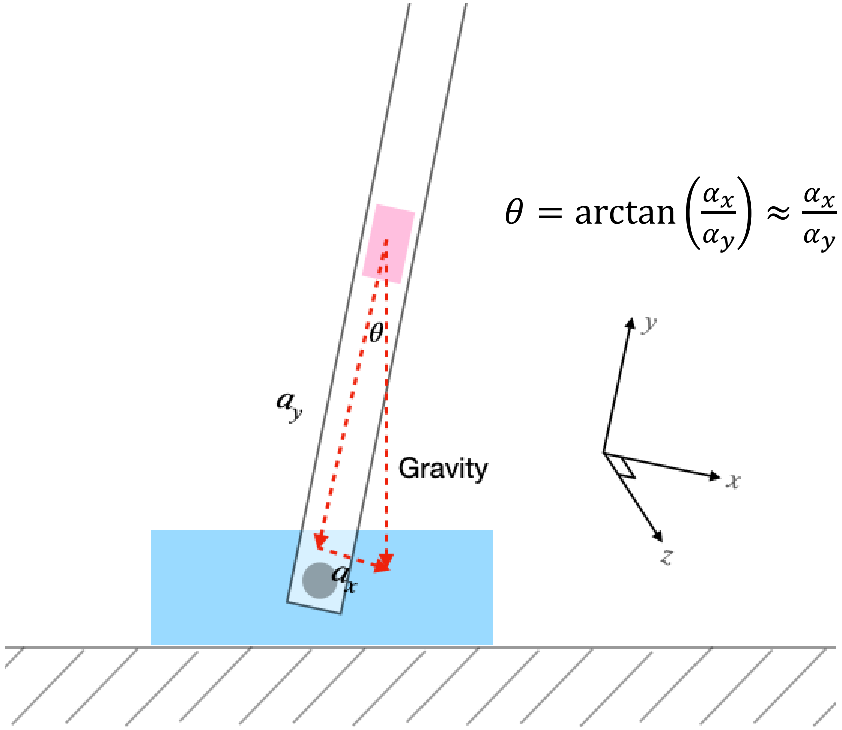
# Guided exercise
## Data Fusion

Combining two different measurement sources of the same information to improve its accuracy.

| Target | Pitch angle |
|---|---|
| Source n.1 | Accelerometer |
| Source n.2 | Gyroscope |

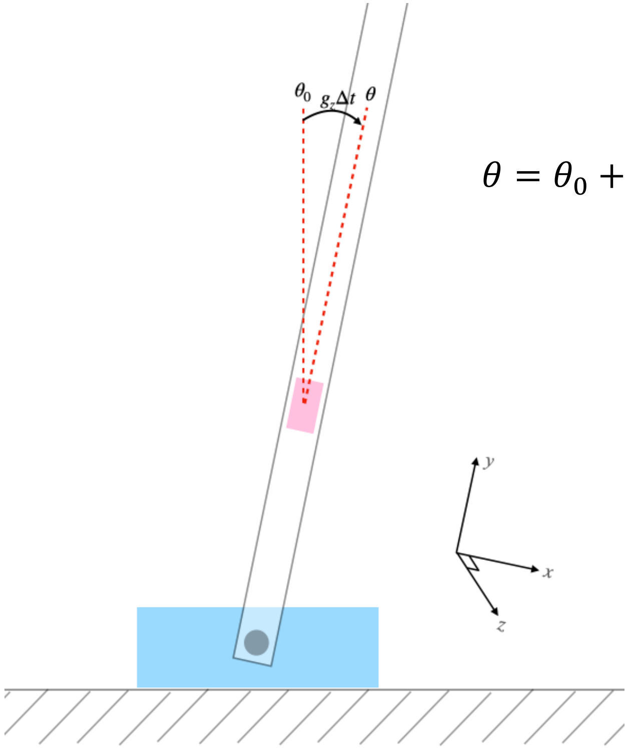
# Guided exercise
## Data Fusion

### Accelerometer as inclinometer



$$\theta = \arctan\left(\frac{\alpha_x}{\alpha_y}\right) \approx \frac{\alpha_x}{\alpha_y}$$

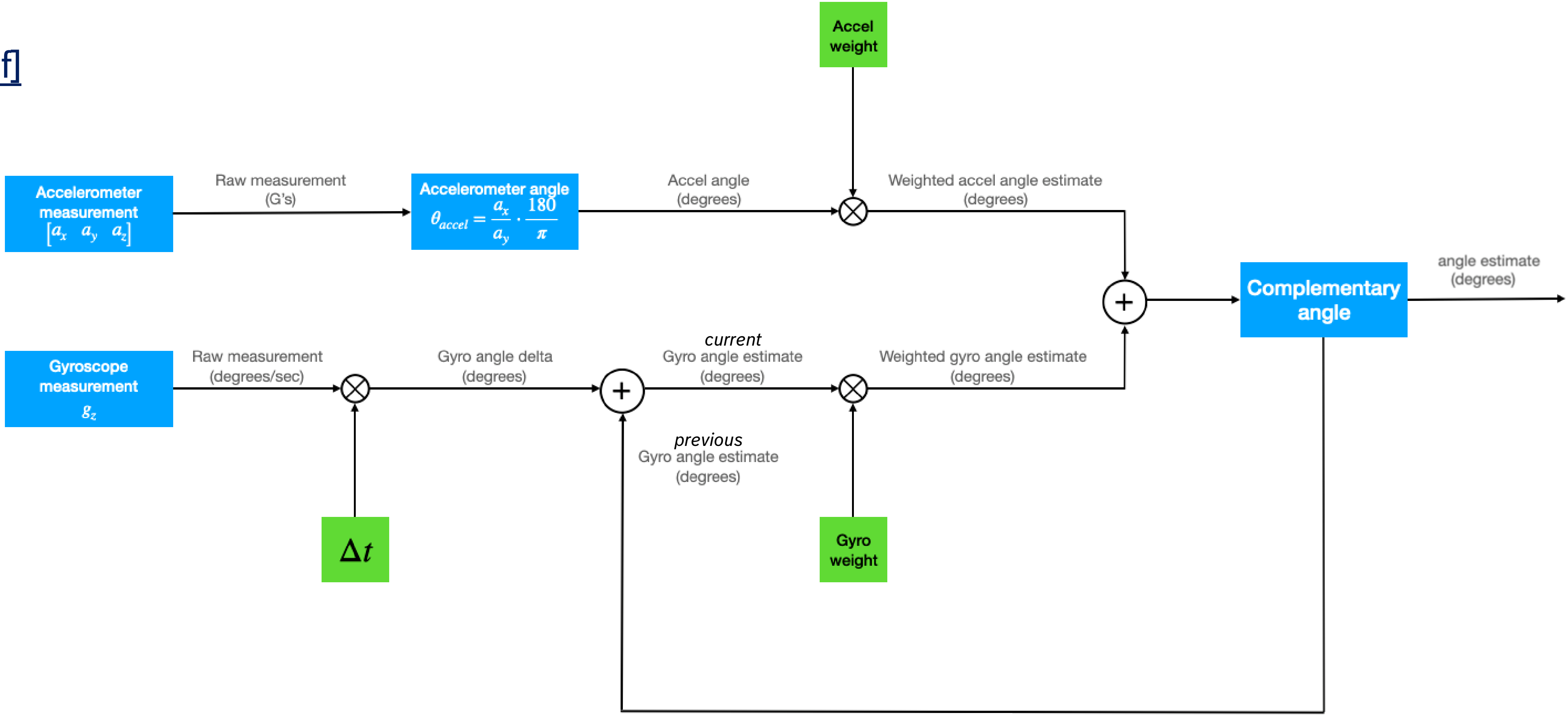
### Gyroscope output integration



$$\theta = \theta_0 + g_z \Delta t$$

# Guided exercise
## Complementary Filter

[Ref]



Accelerometer measurement
$[a_x \;\; a_y \;\; a_z]$
Raw measurement (G's)
Accelerometer angle
$\theta_{accel} = \frac{a_x}{a_y} \cdot \frac{180}{\pi}$
Accel angle (degrees)
Accel weight
Weighted accel angle estimate (degrees)
Complementary angle
angle estimate (degrees)
Gyroscope measurement
$g_z$
Raw measurement (degrees/sec)
$\Delta t$
Gyro angle delta (degrees)
*current* Gyro angle estimate (degrees)
*previous* Gyro angle estimate (degrees)
Gyro weight
Weighted gyro angle estimate (degrees)

# Guided exercise

## Complementary Filter - pt1

```
#include "Arduino_BHY2.h"
#include "math.h"
#define ACCEL_LSB2MG_FS8 0.24414
#define GYRO_LSB2DPS_FS2000 0.06104

// Reference to accel and gyro sensors
SensorXYZ accel(SENSOR_ID_ACC);
SensorXYZ gyro(SENSOR_ID_GYRO);

float now = 0;
float pitch_accel = 0;
float pitch_gyro = 0;
float pitch_rawgyro = 0;
float pitch_filt = 0;
float dt = 0;
const float alpha = 0.98;
```

```
void setup() {
  // Setup serial communication
  Serial.begin(115200);
  while(!Serial);

  // Start BHY and sensors
  BHY2.begin();
  accel.begin();
  gyro.begin();
}
```

# Guided exercise
## Complementary Filter - pt2

```
void loop() {
  // Update function should be continuously polled
  BHY2.update();

  // Update delta time [s]
  dt = (millis() - now)/1000;
  now = millis();


  // Update pitch from single sources
  pitch_accel = atan2(-accel.x(), sqrt(pow(accel.y(), 2) + pow(accel.z(), 2))) * 180/PI;
  pitch_rawgyro -= dt * gyro.y() * GYRO_LSB2DPS_FS2000;


 // Pitch from complementary filter
  pitch_gyro = pitch_filt - dt * gyro.y() * GYRO_LSB2DPS_FS2000;
  pitch_filt = alpha * pitch_gyro + (1 - alpha) * pitch_accel;

// Print messages
  Serial.print(String("Accel:") + String(pitch_accel));
  Serial.print(",");
  Serial.print(String("Gyro:") + String(pitch_rawgyro));
  Serial.print(",");
  Serial.println(String("Filter:") + String(pitch_filt));
}
```

# Guided exercise

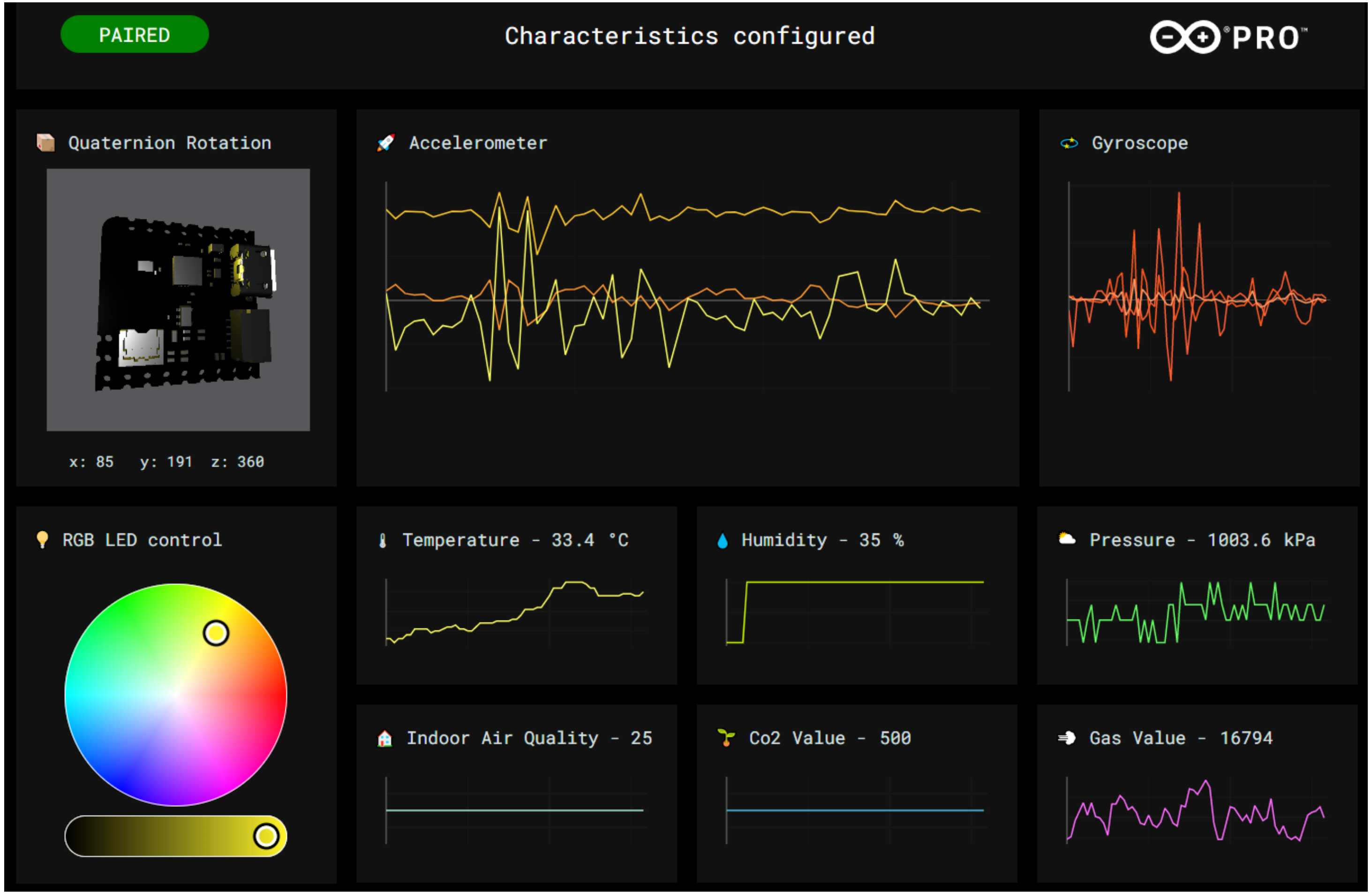
## Orientation Virtual Sensor

```
#include "Arduino_BHY2.h"
#include "math.h"
#define ACCEL_LSB2MG_FS8 0.24414
#define GYRO_LSB2DPS_FS2000 0.06104

// Create a reference to accel and gyro sensors
SensorXYZ accel(SENSOR_ID_ACC);
SensorXYZ gyro(SENSOR_ID_GYRO);
SensorOrientation orientation(SENSOR_ID_ORI);

float now = 0;
float pitch_accel = 0.0;
float pitch_gyro = 0.0;
float dt = 0;

void setup() {
  // Setup the serial communication
  Serial.begin(115200);
  while(!Serial);

  // Setup the BHY and the sensors of interest
  BHY2.begin();
  accel.begin();
  gyro.begin();
  orientation.begin();
  BHY2.configureSensor(SENSOR_ID_ACC, 100, 1);
  BHY2.configureSensor(SENSOR_ID_GYRO, 100, 1);
}
```

```
void loop() {
  // Update function should be continuously polled
  BHY2.update();

  // Update delta time [s]
  dt = (millis() - now)/1000;
  now = millis();

  pitch_accel = atan2(accel.x(), sqrt(pow(accel.y(), 2) + pow(accel.z(), 2))) * 180/PI;
  pitch_gyro += dt * gyro.y() * GYRO_LSB2DPS_FS2000;

  // Print messages
  Serial.print(String("Accel:") + String(pitch_accel));
  Serial.print(",");
  Serial.print(String("Gyro:") + String(pitch_gyro));
  Serial.print(",");
  // WATCHOUT! Due to a bug in the current release, roll and pitch are currently exchanged
  Serial.println(String("Orientation:") + String(orientation.roll()));
}
```

# WebBLE Dashboard

- Arduino Nicla Sense ME - Web BLE test in Chrome browser



Characteristics configured

Quaternion Rotation

x: 85 y: 191 z: 360

Accelerometer

Gyroscope

RGB LED control

Temperature - 33.4 °C

Humidity - 35 %

Pressure - 1003.6 kPa

Indoor Air Quality - 25

Co2 Value - 500

Gas Value - 16794

# University program
## Joining Bosch Sensortec

### Internship

6 Months paid internship for Bachelor and Master Students

### Master thesis

6 Months paid Master-thesis in collaboration with your University

### PhD Program

3 years program in collaboration with a University

### Direct Entry

Start your Career as an Engineer after your Master Degree

# Thank you!

## GET IN TOUCH WITH US

**Francesco Sechi**
**Leonardo Gaffuri Pagani**

@BoschMEMS

www.bosch-sensortec.com

community.bosch-sensortec.com

linkedin.com/company/bosch-sensortec

youtube.com/user/BoschSensortec

Bosch Automotive Electronics

Bosch Sensortec

Bosch Italia