Universidad de Costa Rica

Laboratorio 5
Arduino: GPIO, Giroscopio, comunicaciones, TinyML

IE-0624 Laboratorio de Microcontroladores

Prof. MSc. Marco Villalta

Isaac Rojas Hernández
B76693
Amy Herrera Mora
B53473

3 de julio de 2023

Índice

1. Introducción

En el presente documento se describe todo el trabajado realizado para el quinto laboratorio del curso Laboratorio de microcontroladores. En él se desea desarrollar un reconocedor de actividad humana. Por lo tanto debe cumplir con los siguientes requisitos

1. Utilizar una placa Discovery STM32F49 o un kit Arduino Nano 33 BLE.
2. Realizar un programa para el microcontrolador que capture la información del giroscopio y se envíe a la computadora por el puerto USB.
3. Realizar un script de Python que guarde la información recibida del giroscopio y que se encuentre etiquetada con el tipo de movimiento que se efectúa, el script debe guardar esta información en un archivo CSV.
4. Debe registrar 3 movimientos (por ejemplo: mover brazo hacia arriba o hacia abajo, mover brazo en círculos, golpe de brazo, estacionario,etc) por un periodo de tiempo (p.e. 5-15 segundos). Eso mientras se ejecuta el programa de registro de giroscopio en conjunto con el script de comunicaciones a la PC.
5. Realizar un script de Python para crear un modelo de TensorFlow lite con lo siguiente:
   - Que cargue la información de los movimientos guardados en el archivo CSV del paso anterior.
   - Configure una red neuronal (p.e. una red 2 capas (una de entrada y una oculta/intermedia) con varias neuronas cada una y 3 salidas (una por cada tipo de movimiento realizado), usando ReLU como función de activación (excepto para la de salida que se recomienda softmax), algoritmo de optimización rmsprop y métrica de perdida mae).
   - Entrene la red con el 60 % de los datos, se utilizar´a el 20 % de los datos para validar y el resto 20 % para .
   - Exporte el modelo obtenido con TensorFlow Lite.
6. Realizar un programa que utilice el modelo construido en el paso anterior para detectar el tipo de movimiento realizado, los movimientos detectados deben de quedar registrados en la PC por lo que se debe realizar la comunicación entre el microcontrolador y la PC.

Para esto, se desarrollo el proceso etapa por etapa. En primer lugar, se implementa el programa para la captura de los seis datos: aceleración en cada eje aX, aY y aZ del giroscopio; y además la obtención de la posición del giroscopio en cada eje. Estos datos se envía mediante el puerto Serial.

En segundo lugar, se toman los archivos CSV para los tres movimientos diferentes descritos en el enunciado y se cargan en la plataforma Google Colab donde se tiene un Jupyter Notebook que contiene el ejercicio inicial de Arduino TinyML. Dicho programa se modifica para que se utilicen los archivos indicados para cada movimiento y se entrena la red neuronal para que así detecte esos tres tipos de patrones. Adicionalmente, se le indica el nombre de cada patrón para que esta inteligencia artificial sepa lo que se esta entrenando.

Una vez logrado el entreno, se exporta a un archivo .h en formato TensorFlow Lite para que sea mas optimizado. Este ultimo ya es el modelo creado que puede utilizarse, y para esto se debe modificar el archivo $imu_classifier.ino$, que viene incluido dentro de los ejemplos de Arduino, y se agrega en el mismo directorio donde se encuentra dicho modelo.

Finalmente, el $imu_classifier.ino$ imprime los tres tipos de movimiento, captura estos mismos seis datos que se obtienen del giroscopio, los procesa por el modelo de red neuronal entrenado y por ultimo entrega un valor incertidumbre en un rango del 0 al 1, donde entre mas cercano se esta del numero 1, mas segura esta la red neuronal de cual es el movimiento que se esta realizando.

Como principales conclusiones se obtuvieron que implementar un modelo de red neuronal que provea datos de predicción e identificación adecuado, requiere de una captura de datos de entrenamiento extenso. También, se logró implementar de forma adecuada un programa para el Arduino Nano BLE 33 Sense con el que se detecte y se predigan 3 tipos de movimiento a través de una red neuronal. Finalmente, el uso de la herramienta de desarrollo de google collab, para desarrollar, instanciar, entrenar y exportar redes neuronales es de suma importancia, ya que en equipos de computación común este proceso puede llegar a ser muy lento o no servir del todo.

2. Nota teórica

2.1. Arduino Nano 33 BLE Sense Lite

Este dispositivo pertenece a las placas de HW+SW libre con microcontrolador programable. La mayoria de estos MCUs pertenecen a la misma familia AVR. Tiene su propio IDE inspirado en Processing y se programa en lenguaje C/C++. Su hardaware se inspira en la placa libre Wiring del 2003 y nace en Italia en el Instituto Ivrea en el ano 2005. En la figura (1) se muestra esta placa. [1]

Figura 1: Placa del microcontrolador Arduino Nano 33 BLE Sense [1]

Dentro de la descripción general para el Arduino Nano BLE 33 Sense Lite TinyML kit se encuentran varias características que se explicaran a continuación de acuerdo a la hoja de datos especifica para este laboratorio. [1]

- Es una placa basada en el microcontrolador nRF52840 de Nordic Semiconductor.
- Es versátil y compacta ya que incluye 11 sensores.
- Esta diseñada para que se puedan desarrollar aplicaciones de bajo consumo y que incluso requieran de conectividad BLE (Bluetooth).
- Posee un giroscopio, acelerómetro y magnetómetro en la placa (IMU LSM9DS1 de 9 ejes).
- Además, tiene un sensor de proximidad, intensidad de luz, color RGB y detección de gestos APDS9960, micrófono digital y cámara OV7675.
- Aunque la versión Lite no presenta el sensor HTS221 de temperatura y humedad, si posee el LPS22HB, el cual es un sensor de presión y temperatura.
- Presenta soporte de MicroPython.

Las especificaciones técnicas para el Arduino Nano BLE 33 Sense Lite también se encuentran descritas en la hoja de datos general para este microcontrolador y se pueden observar en la figura (2).

Board	Name	Arduino Nano 33 BLE Sense Lite
	SKU	ABX00031
Microcontroller	nRF52840	
USB connector	Micro USB	
Pins	Built-in LED Pin	13
	Digital I/O Pins	14
	Analog input pins	8
	PWM pins	5
	External interrupts	All digital pins
Connectivity	Bluetooth®	NINA-B306
	IMU	LSM9DS
	Microphone	MP34DT05
Sensors	Gesture, light, proximity	APDS9960
	Barometric pressure	LPS22HB
	Temperature, humidity	~~HTS221~~
Communication	UART	RX/TX
	I2C	A4 (SDA), A5 (SCL)
	SPI	D11 (COPI), D12 (CIPO), D13 (SCK). Use any GPIO for Chip Select (CS).
Power	I/O Voltage	3.3V
	Input voltage (nominal)	5-18V
	DC Current per I/O Pin	10 mA
Clock speed	Processor	nRF52840 64MHz
Memory	nRF52840	256 KB SRAM, 1MB flash
Dimensions	Weight	5gr
	Width	18 mm
	Length	45 mm

Figura 2: Especificaciones técnicas para el Arduino Nano BLE 33 Sense. [1]

El diagrama de pines para esta placa se puede apreciar en la figura (3), mientras que la topología de la placa se muestra en la figura (4).

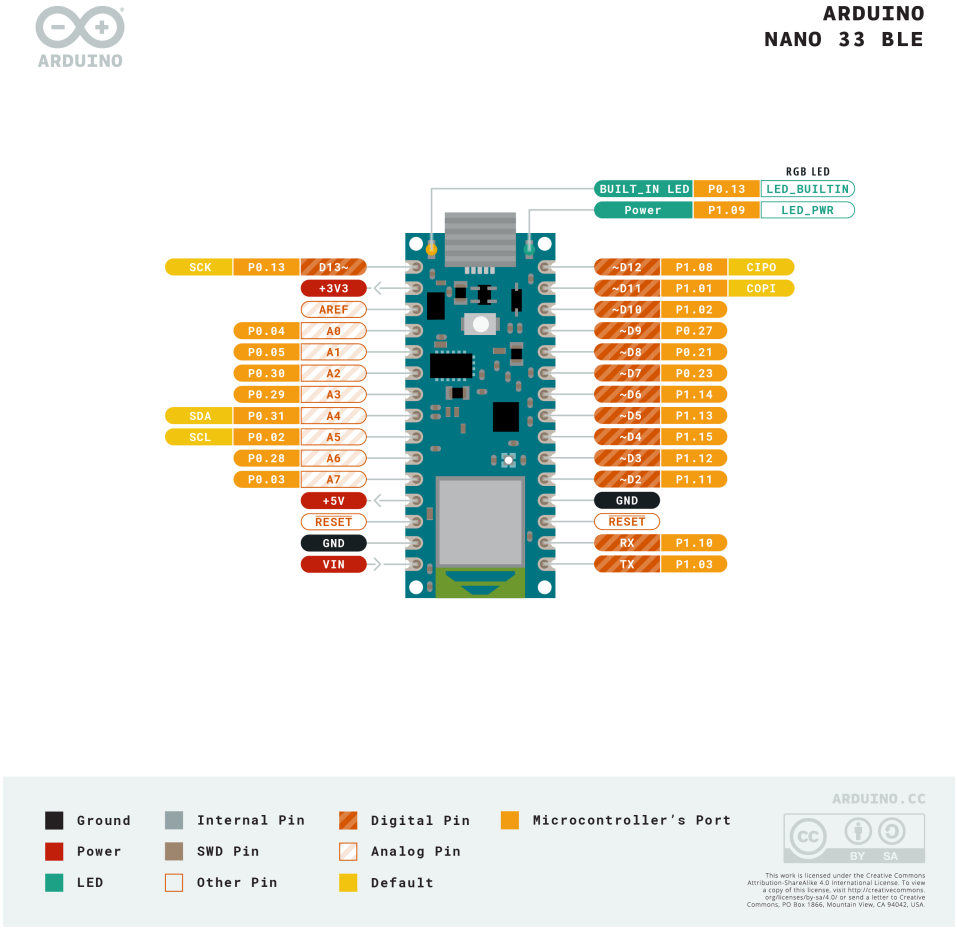


Figura 3: Diagrama de pines para esta placa. [1]

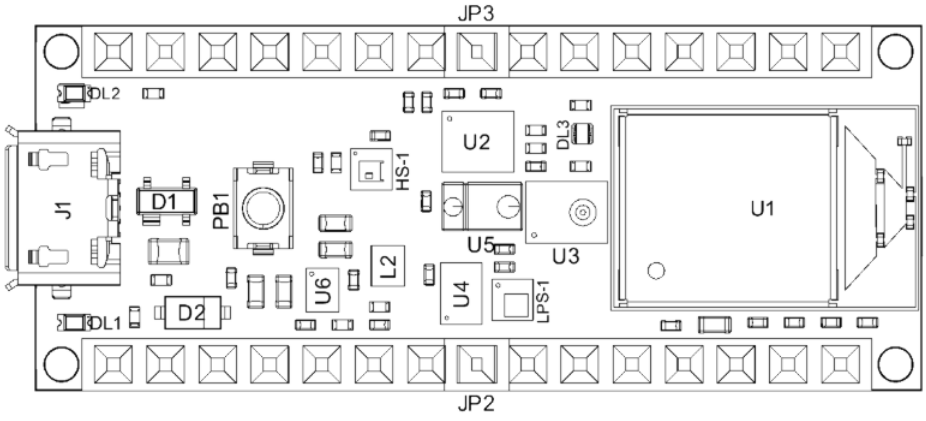


Figura 4: Topología para este microcontrolador. [1]

2.2. Microcontrolador nRF52840

Este microcontrolador es un dispositivo de Nordic Semiconductor, la cual es una compañía que se especializa en conectividad inalámbrica. Es parte de la familia de microcontroladores nRF52. Su diagrama de bloques se puede apreciar en la figura (5). A continuación, se explicaran sus características principales de acuerdo a la hoja de datos del fabricante. [3]

- Arquitectura y rendimiento: El nRF52840 está basado en una arquitectura de 32 bits ARM Cortex-M4. Esto le permite alto rendimiento y capacidad de procesamiento. Además, trabaja a una velocidad de reloj de hasta 64 MHz para poder ejecutar tareas de manera rápida y eficiente.
- Memoria y almacenamiento: Posee una memoria flash integrada de 1 MB para poder almacenar el firmware del dispositivo. También, tiene una memoria SRAM de 256 KB para la ejecución de código y almacenamiento temporal de datos.
- Periféricos: Este dispositivo incluye varios periféricos, como por ejemplo: 48 Puertos GPIO (General Purpose Input/Output) para la interfaz con componentes externos, interfaces de comunicación como UART, I2C y SPI; y convertidores analógico-digital (ADC) con 8 canales para la lectura de sensores.
- Conectividad inalámbrica: El nRF52840 soporta diversos protocolos como Bluetooth Low Energy (BLE) y Bluetooth 5. Además de las características propias del BLE, este dispositivo permite velocidades de datos de 2 Mbps para mejorar la transferencia de datos inalámbrica. Finalmente, posee criptografía acelerada por hardware (ARM TrusZone Cryptocell 310 security subsystem) para garantizar la seguridad de la comunicación inalámbrica
- Alimentación y características energéticas: El diseño del nRF52840 busca ser eficiente en términos de consumo de energía. Para esto, incluye modos de bajo consumo que puedan prolongar la duración de la batería en dispositivos que se alimentan de ella.

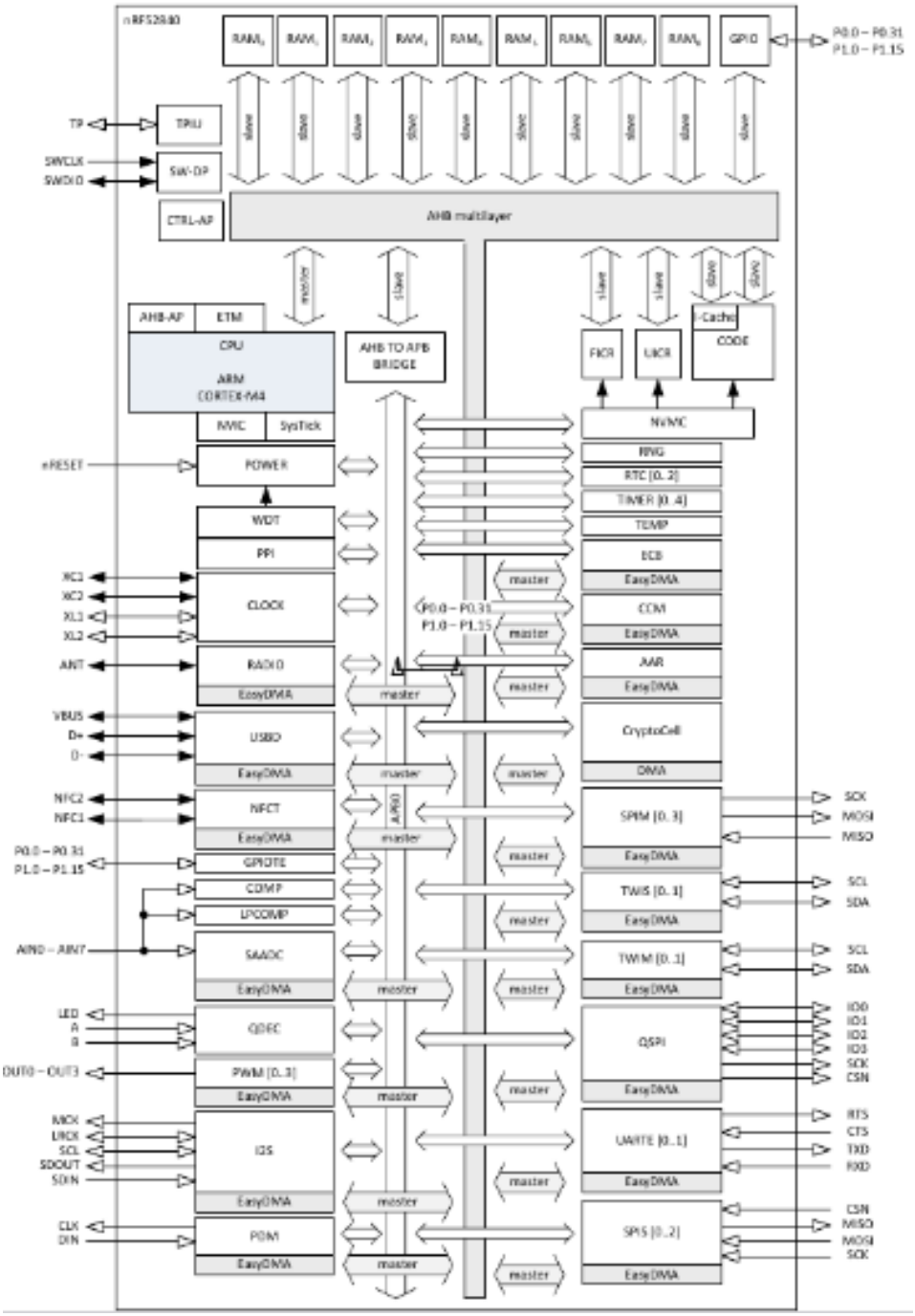


Figura 5: Diagrama de bloques para el nRF52840. [3]

2.3. APDS-9960

El dispositivo APDS-9960 mostrado en la figura (6) es un sensor de detección, reconocimiento de gestos y proximidad desarrollado por Broadcom. Ahora, se explicaran a detalle sus características principales de acuerdo a la hoja de datos para este. [2]

- Reconocimiento de gestos: Este sensor, cuya funcionalidad es controlada mediante una maquina de estados, puede reconocer una variedad de gestos realizados cerca de él tales como deslizamiento hacia arriba, hacia abajo, hacia la derecha, hacia la izquierda, movimientos circulares tanto en sentido de las manecillas del reloj como antihorario.
- Detección de proximidad: Este utiliza un emisor de luz infrarroja y un receptor para detectar objetos cercanos. Con esto puede medir la distancia entre el sensor y cierto objeto en un rango 100 mm aproximadamente.
- Detección de luz ambiente y color: Este dispositivo puede medir la luz ambiental a su alrededor para poder brindar información sobre la intensidad de la luz en el ambiente. Además, este sensor puede detectar y medir el color de la luz ambiente. Esto ultimo permite que sea utilizado en aplicaciones que requieran el uso de RGB. Adicionalmente, cuenta con una función que compensa la luz ambiental para adaptarse a diferentes condiciones de iluminación y garantizar mediciones precisas.
- Interfaz de comunicación: El APDS-9960 se puede comunicar con otros dispositivos por medio de una interfaz I2C (Inter-Integrated Circuit) de dos hilos, lo que facilita su integración en sistemas existentes. Dicha interfaz I2C permite configurar el sensor, recibir datos de detección y controlar estas funciones.

Figura 6: Sensor APDS-9960. [2]

2.4. LPS22HB

Este sensor LPS22HB es un dispositivo de medición de presión y temperatura desarrollado por STMicroelectronics. Es muy preciso y de bajo consumo de energía. Su diagrama de bloques se muestra en la figura (7) y su diagrama de pines en la figura (8). A continuación, se explicaran las características principales descritas en la hoja del fabricante. [5]

- Medición de presión: Es un sensor piezo-resistivo utilizado para medir la presión atmosférica, de hecho, puede medir la presión en un rango desde 260 hasta 1260 hectopascales (hPa). Su alta resolución permite una precisión de hasta ±0.1 hPa en condiciones normales.
- Medición de temperatura: Este dispositivo también contiene un termómetro de alta precisión para medir la temperatura ambiente. Con el puede brindar lecturas de temperatura en un rango desde $-40°$ hasta $+85°$ con una precisión de aproximadamente ±0.5 grados Celsius.
- Interfaz de comunicación y características de configuración: Este sensor también se comunica con otros dispositivos a través de la interfaz I2C (Inter-Integrated Circuit) de dos hilos, ya que permite una fácil integración con otros microcontroladores y sistemas embebidos. Adicionalmente, el LPS22HB proporciona registros internos para configurar y controlar diferentes modos de funcionamiento, resolución y frecuencia de muestreo.
- Baja potencia y modo de bajo consumo: Este sensor LPS22HB incluye un modo de bajo consumo que permite reducir aún más el consumo de energía cuando el sensor no está en uso activo. Además, su tensión de operación se encuentra en el rango desde 1.7 hasta 3.6 V.

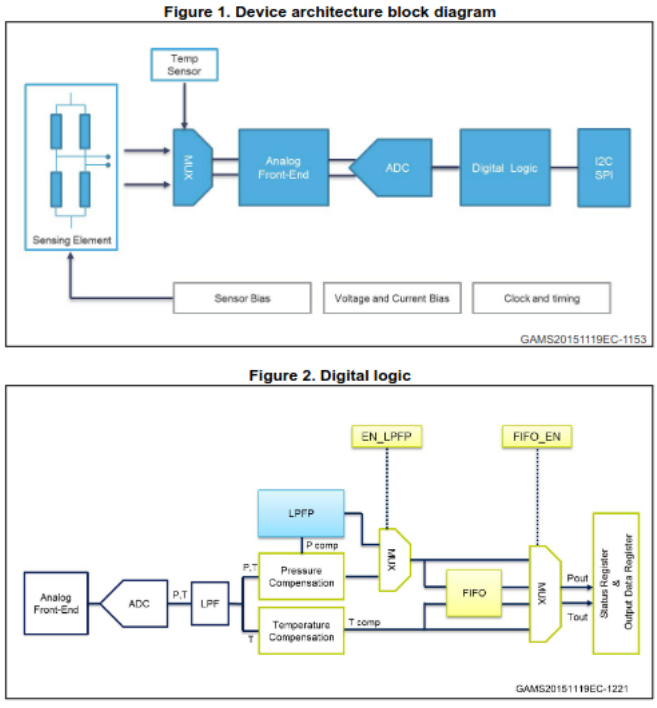


Figura 7: Diagrama de bloques para el sensor LPS22HB. [5]

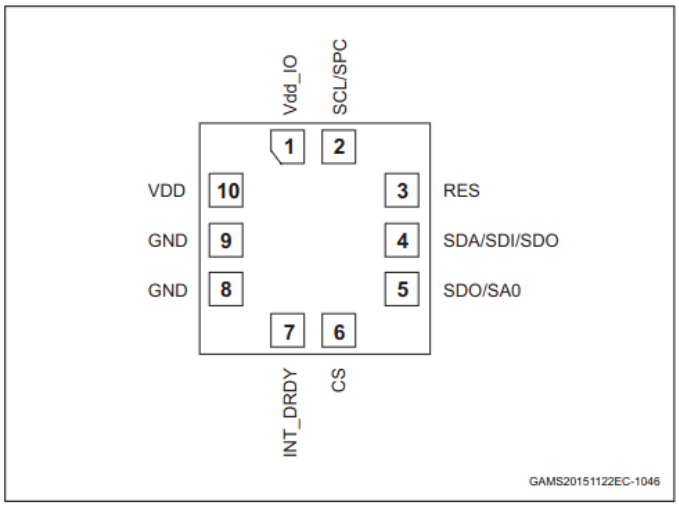


Figura 8: Diagrama de pines para el sensor LPS22HB. [5]

2.5. TensorFlow y TensorFlow Lite

TensorFlow es una biblioteca de Machine Learning de código abierto desarrollada por Google. Permite proporcionar un conjunto de herramientas y mas bibliotecas para desarrollar modelos de aprendizaje automático en muchas plataformas. Es compatible con diversos lenguajes y además con una gran cantidad de aplicaciones desde el procesamiento de imágenes hasta la detección de objetos. Esta utiliza un modelo de grafo de flujo de datos para así poder definir, entrenar y ejecutar modelos de Machine Learning. [6]

Por otro lado, TensorFlow Lite es una versión optimizada, y como su nombre lo indica, mas ligera que TensorFlow. Se diseñó para ayudar a los desarrolladores a ejecutar sus modelos en dispositivos móviles y sistemas integrados que tienen recursos limitados. Brinda una biblioteca mas agilizada y un intérprete que permite una mejor implementación de modelos de aprendizaje automático en estos dispositivos. Además de que utiliza una API mas sencilla y eficiente. En la figura (9) se muestra la arquitectura para esta versión, y en la figura (10) su flujo de trabajo. [6]

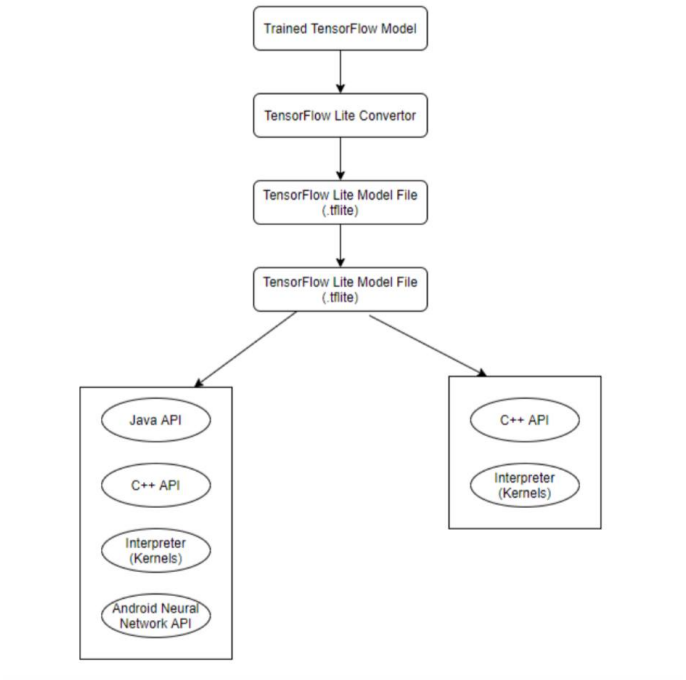


Figura 9: Arquitectura para TensorFlow. [6]

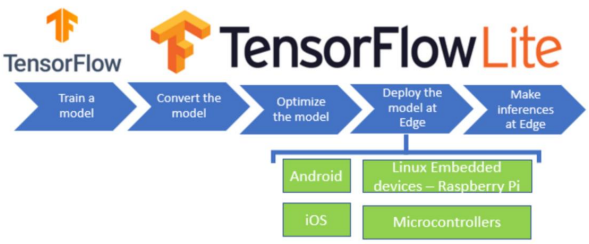


Figura 10: Flujo de trabajo para TensorFlow Lite. [6]

2.6. TinyML

Este concepto de TinyML se refiere a Machine Learning en dispositivos que presentan limitaciones y muy poca potencia tales como: microcontroladores, sistemas embebidos y dispositivos IoT (Internet de las cosas). Es un campo novedoso de las aplicaciones que incluyen algoritmos, hardware y software capaces de analizar datos en dispositivos con sensores. Seguidamente, se detallaran sus principales características. [7]

- Aplicaciones de TinyML: Es utilizada en muchísimas aplicaciones donde el procesamiento de datos y la toma de decisiones en tiempo real son primordiales tales como: reconocimiento de voz, detección de gestos, monitoreo de salud, sistemas de seguridad, entre otros. Con esta nueva tecnología de aprendizaje automático para dispositivos de borde, se logra menor latencia, mayor privacidad de datos y una capacidad mejorada para que pueda funcionar en entornos con conectividad limitada o incluso desconectados.

- Implementación de modelos mas agilizados: Usualmente, los modelos de Machine Learning son grandes y requieren gran cantidad de recursos computacionales para ejecutarse. Para TinyML, se busca que dichos modelos sean más ligeros y optimizados para que puedan adaptarse a las limitaciones de los dispositivos, pero sin alterar mucho la precisión para estas aplicaciones.

2.7. Google Colab

Google Colab es un entorno de desarrollo que permite escribir, ejecutar y colaborar en lenguaje de programación Python. Dicha plataforma es gratuita y ofrece recursos como Jupyter notebooks útiles para Machine Learning, análisis de datos y educación en general. Esto facilita el acceso a recursos potentes de procesamiento sin necesidad de configurarlos localmente. [6]

Este entorno también permite la integración con otros servicios de Google para facilitar la importación y exportación de datos hacia Google Drive y Google Cloud Storage. Además, los notebooks pueden ser compartidos con otros usuarios para que haya colaboración en tiempo real y una de las ventajas es que solo se necesita tener una cuenta en Google. [6]

2.8. HAR

Human Activity Recognition, por sus siglas en ingles, es un método de clasificación de series de tiempo. Su funcionamiento se basa en la predicción del movimiento de una persona según los datos obtenidos de sensores. Para esto, se requieren protocolos de procesamiento de señales complejos y avanzados. Los trabajos mas recientes sobre esta tecnología están basados en el aprendizaje profundo como CNN y RNN, y han mostrado buenos resultados. [6]

2.9. Tabla de componentes para el desarrollo del proyecto

Componentes	**Precio**
Arduino Nano BLE 33 Sense Lite TinyML kit	45 350 colones
Total	**45 350 colones**

Tabla 1: Componentes necesarios para el desarrollo del laboratorio

3. Desarrollo/Análisis de resultados

3.1. Desarrollo

Para poder desarrollar el programa que contenga todas las funcionalidades descritas en el enunciado del laboratorio, fue necesario seguir el diagrama de flujo mostrado en la figura (11).

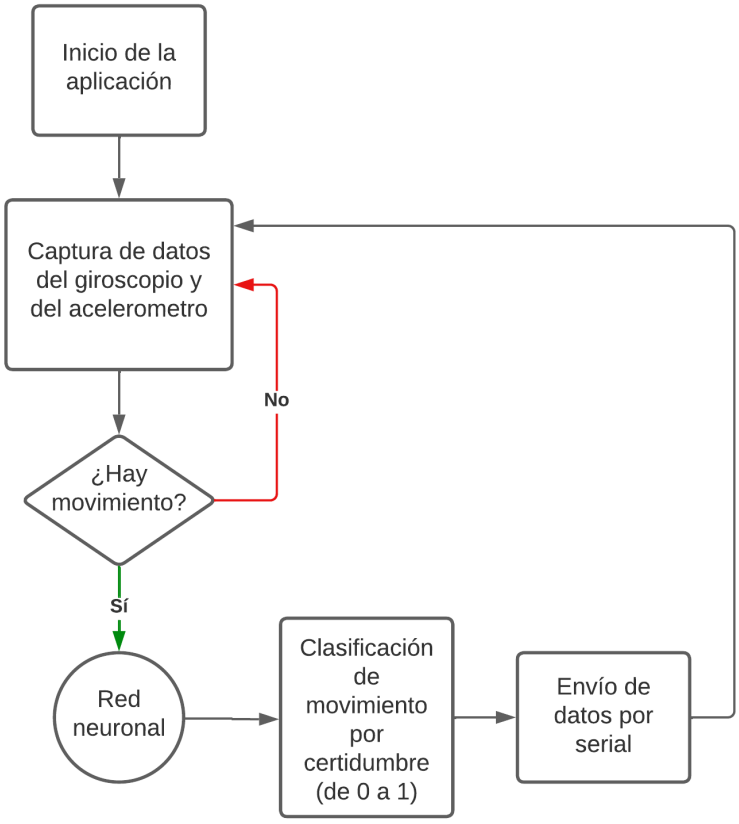


Figura 11: Diagrama de flujos del clasificador de movimiento (Creación propia)

Seguidamente, se explicara el proceso etapa por etapa para este Laboratorio 5. En primer lugar, se implementa el programa que se muestra en el ejemplo. Este captura seis datos: aceleración en cada eje aX, aY y aZ del giroscopio y además captura la posición del giroscopio en cada eje. Estos datos se envía mediante el puerto Serial. En el código 3.1 se muestra.

```
#include <Arduino_LSM9DS1.h>

const float accelerationThreshold = 2.5; // threshold of significant in G's
const int numSamples = 119;

int samplesRead = numSamples;

void setup() {
  Serial.begin(9600);
  while (!Serial);
```

```
11
12    if (!IMU.begin()) {
13      Serial.println("Failed to initialize IMU!");
14      while (1);
15    }
16    //Serial.println("aX,aY,aZ,gX,gY,gZ");
17  }
18
19  void loop() {
20    float aX, aY, aZ, gX, gY, gZ;
21
22    while (samplesRead == numSamples) {
23      if (IMU.accelerationAvailable()) {
24
25        IMU.readAcceleration(aX, aY, aZ);
26
27        float aSum = fabs(aX) + fabs(aY) + fabs(aZ);
28
29        if (aSum >= accelerationThreshold) {
30          // reset the sample read count
31          samplesRead = 0;
32          break;
33        }
34      }
35    }
36
37    while (samplesRead < numSamples) {
38      if (IMU.accelerationAvailable() && IMU.gyroscopeAvailable()) {
39        // read the acceleration and gyroscope data
40        IMU.readAcceleration(aX, aY, aZ);
41        IMU.readGyroscope(gX, gY, gZ);
42
43        samplesRead++;
44
45        // print the data in CSV format
46        Serial.print(aX, 3);
47        Serial.print(',');
48        Serial.print(aY, 3);
49        Serial.print(',');
50        Serial.print(aZ, 3);
51        Serial.print(',');
52        Serial.print(gX, 3);
53        Serial.print(',');
54        Serial.print(gY, 3);
55        Serial.print(',');
56        Serial.print(gZ, 3);
57        Serial.println();
58
59        if (samplesRead == numSamples) {
60          // add an empty line if it's the last sample
61          Serial.println();
62        }
63      }
64    }
65  }
```

Listing 1: Programa de captura de datos cargado en el Arduino basado en los ejemplos del IDE.

El puerto Serial recibe estos seis datos y para esto se implementa el script de Python que se muestra en el código 1. Dicho programa primero imprime un header para el archivo CSV, ya que si se realiza mediante el Arduino habría que darle Reset constantemente porque este dispositivo siempre esta tomando datos. Luego, captura los seis datos aX, aY, aZ, gX, gY y gZ para guardarlos de forma repetitiva hasta alcanzar el número de muestras que se quiere, que en este caso se capturaron 3000 sets de datos por cada movimiento.

```
import sys
import serial
import time
import csv

#Encabezado del documento para sustituir el del arduino que no siempre sale bien
header = ['aX', 'aY', 'aZ', 'gX', 'gY', 'gZ']
#Determinamos la cantidad de datos que se quieren muestrear
samples = 3000
#Flag en 0 para el conteo de datos
counter = 0
#Para poder determinar los datos que se est n capturando
filename = sys.argv[1]

#En caso de que no se conecte de forma adecuada se tiene un manejo de errores:
try:
    ser = serial.Serial(port = '/dev/ttyACM0', baudrate=9600, timeout=1)
    print("Conectado!")
except serial.SerialException as e:
    print("Fallo al conectar con la placa en el puerto: " + str(e))
    sys.exit(1)

#f = open(filename, 'w', encoding='UTF8')
with open(filename, 'w', encoding='UTF8') as f:
    writer = csv.writer(f)
    print(header)
    writer.writerow(header)

    while(counter < samples):

        data = ser.readline().decode('utf-8').replace('\r', "").replace('\n', "")
        data = data.split(',')

        if(len(data) == 6):
            writer.writerow(data)
            print(data) #  nicamente   para poder ver los datos en consola mientras se
   capturan
            counter+=1
ser.close()
```

Listing 2: Script en Python para la captura de datos.

En segundo lugar, se toman los archivos CSV para los tres movimientos diferentes descritos en el enunciado y se cargan en la plataforma Google Colab donde se tiene un Jupyter Notebook que contiene el ejercicio inicial de Arduino TinyML. Este archivo se modifica para que se utilice los archivos indicados para cada movimiento y se entrena la red neuronal para que con esto detecte esos tres tipos de patrones. Los datos se organizan en dataframes y después de instanciar dicho modelo con lo mostrado en 2, este se entrena con una parte de los datos, como se muestra en el código 3, porque otra parte es utilizada para verificar dicho entrenamiento y corregir. Adicionalmente, se le indica el nombre de cada patrón para que esta inteligencia artificial sepa lo que se esta entrenando.

```
# build the model and train it
model = tf.keras.Sequential()
model.add(tf.keras.layers.Dense(50, activation='relu')) # relu is used for
    performance
model.add(tf.keras.layers.Dense(15, activation='relu'))
model.add(tf.keras.layers.Dense(NUM_GESTURES, activation='softmax')) # softmax is
    used, because we only expect one gesture to occur per input
model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])
history = model.fit(inputs_train, outputs_train, epochs=600, batch_size=1,
    validation_data=(inputs_validate, output))
```

Listing 3: Instancia del modelo para entrenarlo.

```
# Randomize the order of the inputs, so they can be evenly distributed for training,
    testing, and validation
# https://stackoverflow.com/a/37710486/2020087
num_inputs = len(inputs)
randomize = np.arange(num_inputs)
np.random.shuffle(randomize)

# Swap the consecutive indexes (0, 1, 2, etc) with the randomized indexes
inputs = inputs[randomize]
outputs = outputs[randomize]

# Split the recordings (group of samples) into three sets: training, testing and
    validation
TRAIN_SPLIT = int(0.6 * num_inputs)
TEST_SPLIT = int(0.2 * num_inputs + TRAIN_SPLIT)

inputs_train, inputs_test, inputs_validate = np.split(inputs, [TRAIN_SPLIT,
    TEST_SPLIT])
outputs_train, outputs_test, outputs_validate = np.split(outputs, [TRAIN_SPLIT,
    TEST_SPLIT])

print("Data set randomization and splitting complete.")
```

Listing 4: Partición de los datos para el proceso de entrenamiento.

En tercer lugar, una vez que se entrena por completo la red neuronal, se implementa una exportación a un archivo .h en ese mismo archivo como se observa en el codigo 4. Además, este se debe exportar en formato TensorFlow Lite, ya que en formato TensorFlow se obtiene un archivo mucho mas pesado que no se podría cargar en un Arduino y que además tiene un nivel de abstracción y complejidad mas alto. Este ultimo ya es el modelo creado que puede utilizarse, y para esto se debe modificar

el archivo $imu_classifier.ino$, que viene incluido dentro de los ejemplos de Arduino. Este se agrega en el mismo directorio donde se encuentra dicho modelo para que pueda utilizarlo con un #include.

```
# Convert the model to the TensorFlow Lite format without quantization
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()

# Save the model to disk
open("gesture_model.tflite", "wb").write(tflite_model)

import os
basic_model_size = os.path.getsize("gesture_model.tflite")
print("Model is %d bytes" % basic_model_size)
```

Listing 5: Exportación del modelo en formato TensorFlow Lite.

Finalmente, el $imu_classifier.ino$ imprime los tres tipos de movimiento, captura estos mismos seis datos que se obtienen del giroscopio, los procesa por el modelo de red neuronal entrenado y por ultimo esta entrega un valor incertidumbre en un rango del 0 al 1, donde entre mas cercano se esta del numero 1, mas segura esta la red neuronal de cual es el movimiento que se esta realizando.

3.2. Compilación

Para lograr su compilación, se utilizo el Arduino IDE 2.0.1. El modelo ya entrenado, al exportarse como un archivo .h con el nombre modelo.h y en formato TensorFlow Lite, se incluye en el ejemplo de captura de datos tal y como se muestra en el código 3.2.

```
#include <Arduino_LSM9DS1.h>
#include <TensorFlowLite.h>
#include <tensorflow/lite/micro/all_ops_resolver.h>
#include <tensorflow/lite/micro/micro_interpreter.h>
#include <tensorflow/lite/schema/schema_generated.h>
#include "model.h"
```

Listing 6: Inclusión del modelo entrenado para que se utilice en el programa de Arduino..

Con esto se garantiza que al compilar el archivo .ino y se cargue en la placa, se realicen las predicciones respectivas para los datos capturados.

3.3. Análisis de Resultados

Para poder analizar los resultados del laboratorio actual, se compiló el programa mostrado anteriormente y se subió al Arduino Nano BLE 33 Sense, con el modelo de red neuronal ya implementado. Luego se realizaron movimientos para ver si concordaban con lo que predecía la red neuronal. El funcionamiento completo se demostró en el siguiente link:

- https://youtu.be/JrPSLNCJgBw

El primer movimiento realizado fue el movimiento de golpe, en este caso se utilizó el movimiento de un "puñetazo", el resultado obtenido cuando se realizaba el movimiento fue el siguiente:

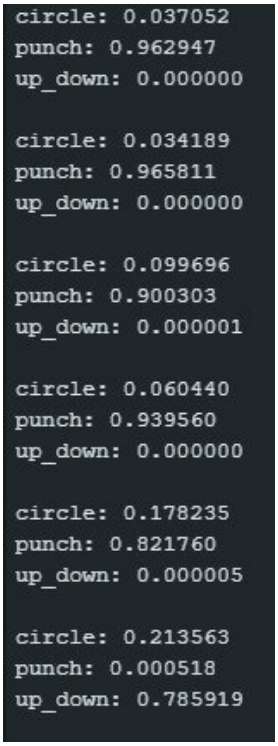


Figura 12: Resultados de movimiento de golpe

Como se puede observar en la figura (12), si se hacen varios movimientos la predicción se mantiene constante, lo que despliega el programa en consola son los tres movimientos para los que se entrenó la red a detectar y a la derecha la certidumbre entre 0 y 1, donde 1 es 100 % certero, del movimiento que se realizó. En este caso al dar golpes la red detecta por encima de 0.9 de certidumbre, mientras que después de realizar el movimiento el porcentaje va cayendo de forma abrupta porque los datos que recibe la red neuronal son de estática.
En el caso de los movimientos circular y arriba-abajo, es un poco más fácil confundirlos para la red neuronal, ya que si se realiza mal el movimiento circular el patrón de los datos llega a ser muy similar entre sí. En la figura (13), se apreciar que la certidumbre usualmente es bastante alta, lo que confirma que la red fue entrenada de forma adecuada. Para el movimiento de arriba-abajo de los brazos, al ser un movimiento bastante sencillo y prácticamente lineal el nivel de certidumbre que presenta el modelo cada vez que se realiza dicho movimiento es bastante alto, usualmente por encima de 0.97, como se puede ver en la figura (14).

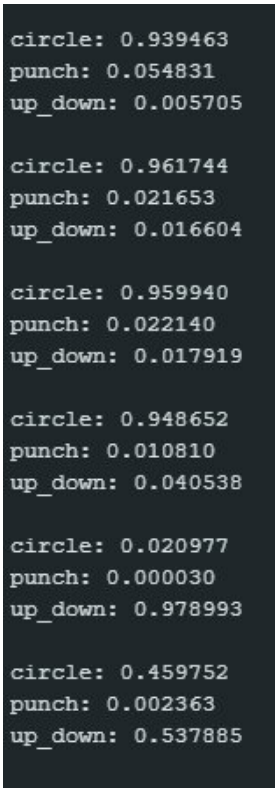


Figura 13: Resultados de movimiento circular

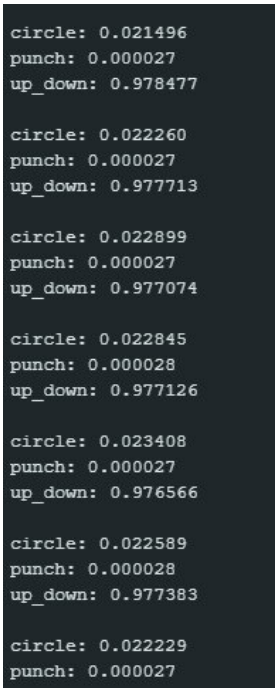


Figura 14: Resultados de movimiento arriba-abajo

Gracias a la gran cantidad de datos capturados, además de la correcta implementación y modificación del código del ejemplo provisto por Arduino, se puede concluir que la implementación de este

laboratorio fue correcta, cumpliendo con el objetivo de detectar con 3 distintos tipos de movimiento del cuerpo humano.

4. Conclusiones y recomendaciones

- Implementar un modelo de red neuronal que provea datos de predicción e identificación adecuado, requiere de una captura de datos de entrenamiento extenso.
- Fue posible ejecutar de forma adecuada un programa para el Arduino Nano BLE 33 Sense con el que se detecte y se predigan 3 tipos de movimiento a través de una red neuronal.
- Se logró implementar, entrenar y exportar una red neuronal con las bibliotecas tensorflow y tensorflow lite.
- El uso de la herramienta de desarrollo de google collab, para desarrollar, instanciar, entrenar y exportar redes neuronales es de suma importancia, ya que en equipos de computación común este proceso puede llegar a ser muy lento o no servir del todo.
- Utilizar bibliotecas reconocidas es una de las mejores formas de abordar una implementación de inteligencia artificial, ya que la documentación disponible y los experimentos previamente hechos ayudan a entender mejor lo que se está realizando.

5. GIT

El código fuente del presente laboratorio 5 se encuentra disponible en el siguiente enlace: https://github.com/mimiherrera/IE0624-Lab-de-Microcontroladores/tree/Lab5

Bibliografía

1. Arduino Nano 33 BLE Sense Datasheet. (2022). Retrieved from Arduino website: https://docs.arduino.cc/resources/datasheets/ABX00031-datasheet.pdf [Accesado: Junio 16, 2023].

2. Broadcom Limited. (2016). APDS-9960 Proximity, Light, RGB, and Gesture Sensor. Retrieved from Broadcom Limited website: https://docs.broadcom.com/doc/AV02-4191EN [Accesado: Junio 16, 2023].

3. Nordic Semiconductor. (2018). nRF52840 Product Specification v1.4. Retrieved from Nordic Semiconductor website: https://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.1.pdf [Accesado: Junio 16, 2023].

4. STMicroelectronics. (2022). LSM9DS1 Datasheet. Retrieved from STMicroelectronics website: https://www.st.com/resource/en/datasheet/lsm9ds1.pdf [Accesado: Junio 16, 2023].

5. STMicroelectronics. (2022). LPS22HB MEMS pressure sensor: 260-1260 hPa absolute digital output barometer. Retrieved from STMicroelectronics website: https://www.st.com/resource/en/datasheet/lps22hb.pdf [Accesado: Junio 16, 2023].

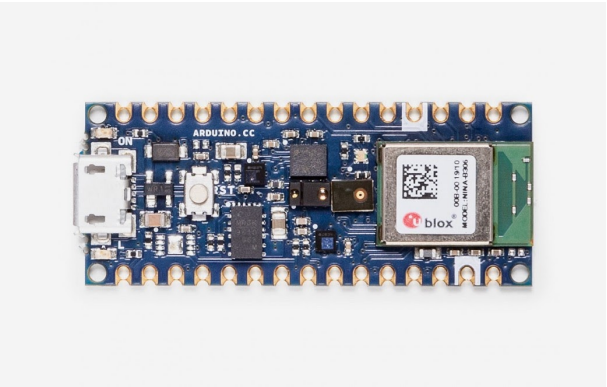
6. M. M. V. Fallas, *TensorFlow Lite-HAR*, IE-0624 Laboratorio de Microcontroladores [Accesado: Junio 18, 2023].

7. M. M. V. Fallas, *Introducción a ML con MCU*, IE-0624 Laboratorio de Microcontroladores [Accesado: Junio 18, 2023].

6. Apéndices

A. Apéndice: Primeras cinco páginas de la hoja de datos Nano33

Product Reference Manual
SKU: ABX00031



Description

Nano 33 BLE Sense is a miniature sized module containing a NINA B306 module, based on Nordic nRF52480 and containing a Cortex M4F, a crypto chip which can securely store certificates and pre shared keys and a 9 axis IMU. The module can either be mounted as a DIP component (when mounting pin headers), or as a SMT component, directly soldering it via the castellated pads

Target areas:

Maker, enhancements, IoT application

Features

- **NINA B306 Module**
  - **Processor**
    - 64 MHz Arm® Cortex-M4F (with FPU)
    - 1 MB Flash + 256 KB RAM
  - **Bluetooth® 5 multiprotocol radio**
    - 2 Mbps
    - CSA #2
    - Advertising Extensions
    - Long Range
    - +8 dBm TX power
    - -95 dBm sensitivity
    - 4.8 mA in TX (0 dBm)
    - 4.6 mA in RX (1 Mbps)
    - Integrated balun with 50 Ω single-ended output
    - IEEE 802.15.4 radio support
    - Thread
    - Zigbee
  - **Peripherals**
    - Full-speed 12 Mbps USB
    - NFC-A tag
    - Arm CryptoCell CC310 security subsystem
    - QSPI/SPI/TWI/I²S/PDM/QDEC
    - High speed 32 MHz SPI
    - Quad SPI interface 32 MHz
    - EasyDMA for all digital interfaces
    - 12-bit 200 ksps ADC
    - 128 bit AES/ECB/CCM/AAR co-processor
- **LSM9DS1** (9 axis IMU)
  - 3 acceleration channels, 3 angular rate channels, 3 magnetic field channels
  - ±2/±4/±8/±16 g linear acceleration full scale
  - ±4/±8/±12/±16 gauss magnetic full scale
  - ±245/±500/±2000 dps angular rate full scale
  - 16-bit data output
- **LPS22HB** (Barometer and temperature sensor)
  - 260 to 1260 hPa absolute pressure range with 24 bit precision
  - High overpressure capability: 20x full-scale
  - Embedded temperature compensation
  - 16-bit temperature data output
  - 1 Hz to 75 Hz output data rateInterrupt functions: Data Ready, FIFO flags, pressure thresholds
- **HTS221** (relative humidity sensor)
  - 0-100% relative humidity range
  - High rH sensitivity: 0.004% rH/LSB
  - Humidity accuracy: ± 3.5% rH, 20 to +80% rH
  - Temperature accuracy: ± 0.5 °C,15 to +40 °C
  - 16-bit humidity and temperature output data

- **APDS-9960** (Digital proximity, Ambient light, RGB and Gesture Sensor)
  - Ambient Light and RGB Color Sensing with UV and IR blocking filters
  - Very high sensitivity – Ideally suited for operation behind dark glass
  - Proximity Sensing with Ambient light rejection
  - Complex Gesture Sensing
- **MP34DT05** (Digital Microphone)
  - AOP = 122.5 dbSPL
  - 64 dB signal-to-noise ratio
  - Omnidirectional sensitivity
  - –26 dBFS ± 3 dB sensitivity
- **ATECC608A** (Crypto Chip)
  - Cryptographic co-processor with secure hardware based key storage
  - Protected storage for up to 16 keys, certificates or data
  - ECDH: FIPS SP800-56A Elliptic Curve Diffie-Hellman
  - NIST standard P256 elliptic curve support
  - SHA-256 & HMAC hash including off-chip context save/restore
  - AES-128 encrypt/decrypt, galois field multiply for GCM
- **MPM3610** DC-DC
  - Regulates input voltage from up to 21V with a minimum of 65% efficiency @minimum load
  - More than 85% efficiency @12V

Contents

1 The Board

As all Nano form factor boards, Nano 33 BLE Sense does not have a battery charger but can be powered through USB or headers.

NOTE: Arduino Nano 33 BLE Sense only supports 3.3V I/Os and is **NOT** 5V tolerant so please make sure you are not directly connecting 5V signals to this board or it will be damaged. Also, as opposed to Arduino Nano boards that support 5V operation, the 5V pin does NOT supply voltage but is rather connected, through a jumper, to the USB power input.

1.1 Ratings

1.1.1 Recommended Operating Conditions

Symbol	Description	Min	Max
	Conservative thermal limits for the whole board:	-40 °C (40 °F)	85°C (185 °F)

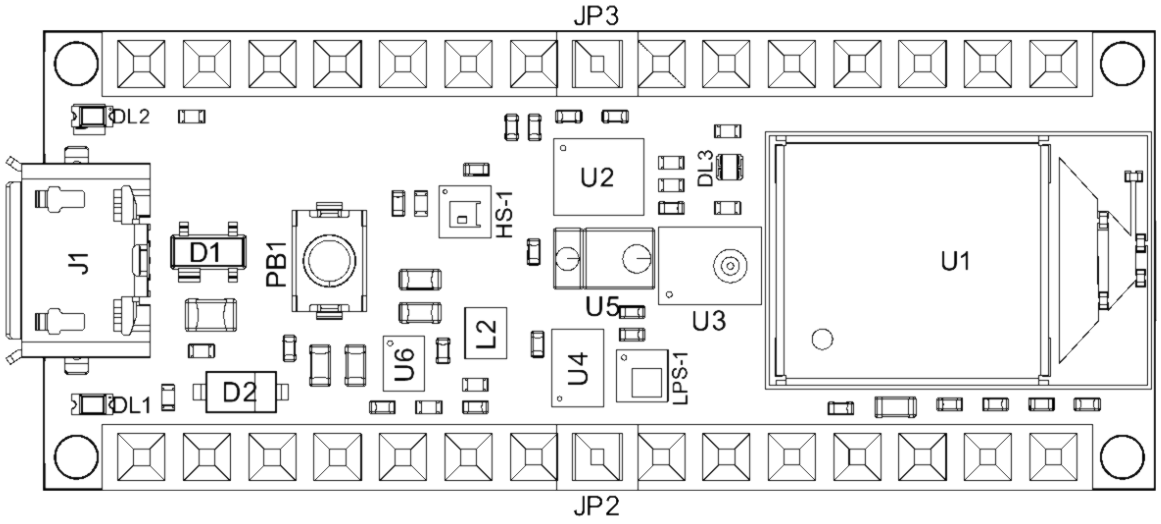
1.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
PBL	Power consumption with busy loop		TBC		mW
PLP	Power consumption in low power mode		TBC		mW
PMAX	Maximum Power Consumption		TBC		mW

2 Functional Overview

2.1 Board Topology

Top:



Board topology top

Ref.	Description	Ref.	Description
U1	NINA-B306 Module Bluetooth® Low Energy 5.0 Module	U6	MP2322GQH Step Down Converter
U2	LSM9DS1TR Sensor IMU	PB1	IT-1185AP1C-160G-GTR Push button
U3	MP34DT06JTR Mems Microphone	HS-1	HTS221 Humidity Sensor
U4	ATECC608A Crypto chip	DL1	Led L

B. Apéndice: Primeras cinco páginas de la hoja de datos APDS-9960

APDS-9960

Digital Proximity, Ambient Light, RGB and Gesture Sensor

AVAGO TECHNOLOGIES

Data Sheet

Lead (Pb) Free
RoHS 6 fully
compliant

Description

The APDS-9960 device features advanced Gesture detection, Proximity detection, Digital Ambient Light Sense (ALS) and Color Sense (RGBC). The slim modular package, L 3.94 × W 2.36 × H 1.35 mm, incorporates an IR LED and factory calibrated LED driver for drop-in compatibility with existing footprints.

Gesture detection

Gesture detection utilizes four directional photodiodes to sense reflected IR energy (sourced by the integrated LED) to convert physical motion information (i.e. velocity, direction and distance) to a digital information. The architecture of the gesture engine features automatic activation (based on Proximity engine results), ambient light subtraction, cross-talk cancelation, dual 8-bit data converters, power saving inter-conversion delay, 32-dataset FIFO, and interrupt-driven I^2C-bus communication. The gesture engine accommodates a wide range of mobile device gesturing requirements: simple UP-DOWN-RIGHT-LEFT gestures or more complex gestures can be accurately sensed. Power consumption and noise are minimized with adjustable IR LED timing.

Description continued on next page...

Applications

- Gesture Detection
- Color Sense
- Ambient Light Sensing
- Cell Phone Touch Screen Disable
- Mechanical Switch Replacement

Ordering Information

Part Number	Packaging	Quantity
APDS-9960	Tape & Reel	5000 per reel

Features

- Ambient Light and RGB Color Sensing, Proximity Sensing, and Gesture Detection in an Optical Module
- Ambient Light and RGB Color Sensing
  - UV and IR blocking filters
  - Programmable gain and integration time
  - Very high sensitivity – Ideally suited for operation behind dark glass
- Proximity Sensing
  - Trimmed to provide consistent reading
  - Ambient light rejection
  - Offset compensation
  - Programmable driver for IR LED current
  - Saturation indicator bit
- Complex Gesture Sensing
  - Four separate diodes sensitive to different directions
  - Ambient light rejection
  - Offset compensation
  - Programmable driver for IR LED current
  - 32 dataset storage FIFO
  - Interrupt driven I^2C-bus communication
- I^2C-bus Fast Mode Compatible Interface
  - Data Rates up to 400 kHz
  - Dedicated Interrupt Pin
- Small Package L 3.94 × W 2.36 × H 1.35 mm

Description (Cont.)

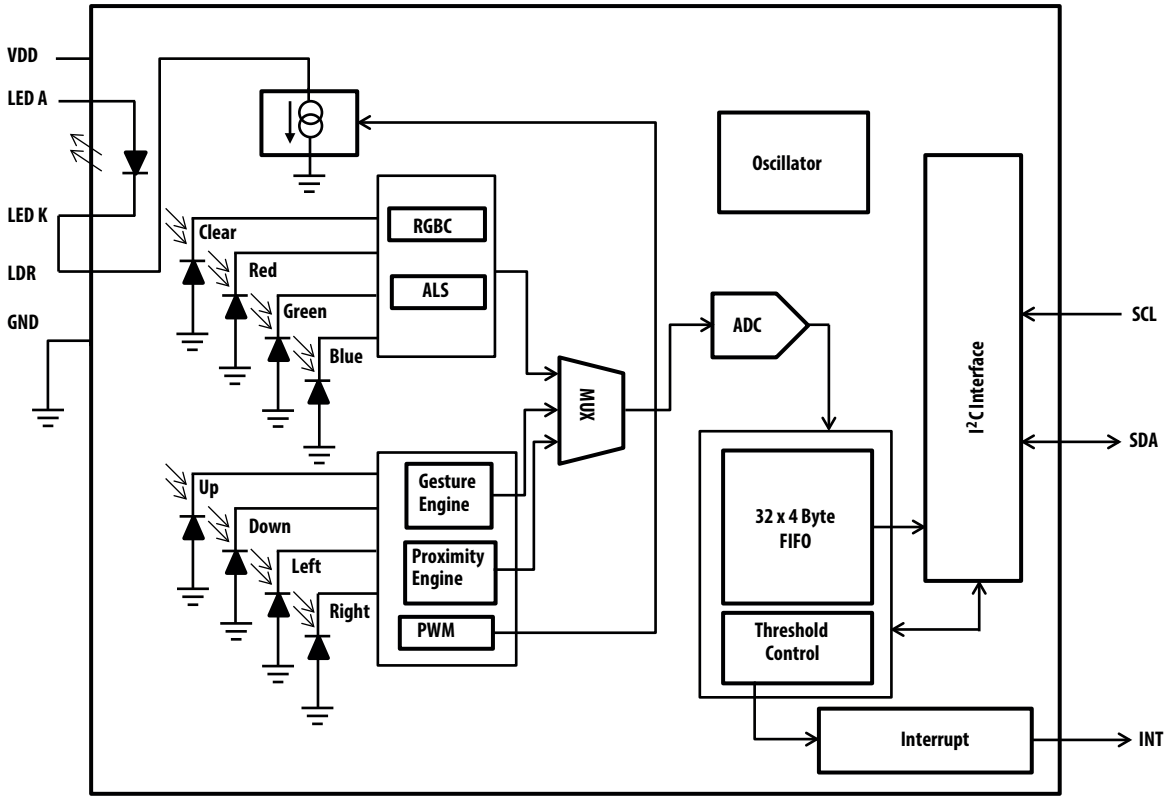
Proximity detection

The Proximity detection feature provides distance measurement (E.g. mobile device screen to user's ear) by photodiode detection of reflected IR energy (sourced by the integrated LED). Detect/release events are interrupt driven, and occur whenever proximity result crosses upper and/or lower threshold settings. The proximity engine features offset adjustment registers to compensate for system offset caused by unwanted IR energy reflections appearing at the sensor. The IR LED intensity is factory trimmed to eliminate the need for end-equipment calibration due to component variations. Proximity results are further improved by automatic ambient light subtraction.

Color and ALS detection

The Color and ALS detection feature provides red, green, blue and clear light intensity data. Each of the R, G, B, C channels have a UV and IR blocking filter and a dedicated data converter producing16-bit data simultaneously. This architecture allows applications to accurately measure ambient light and sense color which enables devices to calculate color temperature and control display backlight.

Functional Block Diagram



I/O Pins Configuration

Pin	Name	Type	Description
1	SDA	I/O	I^2C serial data I/O terminal - serial data I/O for I^2C-bus
2	INT	O	Interrupt - open drain (active low)
3	LDR		LED driver input for proximity IR LED, constant current source LED driver
4	LEDK		LED Cathode, connect to LDR pin when using internal LED driver circuit
5	LEDA		LED Anode, connect to V_{LEDA} on PCB
6	GND		Power supply ground. All voltages are referenced to GND
7	SCL	I	I^2C serial clock input terminal - clock signal for I^2C serial data
8	V_{DD}		Power supply voltage

Absolute Maximum Ratings over operating free-air temperature range (unless otherwise noted)*

Parameter	Symbol	Min	Max	Units	Conditions
Power supply voltage [1]	V_{DD}		3.8	V	
Input voltage range	V_{IN}	-0.5	3.8	V	
Output voltage range	V_{OUT}	-0.3	3.8	V	
Storage temperature range	T_{stg}	-40	85	°C	

* Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

Note 1. All voltages are with respect to GND.

Recommended Operating Conditions

Parameter	Symbol	Min	Typ	Max	Units
Operating ambient temperature	T_A	-30		85	°C
Power supply voltage	V_{DD}	2.4	3.0	3.6	V
Supply voltage accuracy, V_{DD} total error including transients		-3		+3	%
LED supply voltage	V_{LEDA}	3.0		4.5	V

Operating Characteristics, V_{DD} = 3 V, T_A = 25 °C (unless otherwise noted)

Parameter	Symbol	Min	Typ	Max	Units	Test Conditions
IDD supply current [1]	I_{DD}		200	250	µA	Active ALS state PON = AEN = 1, PEN = 0
			790			Proximity, LDR pulse ON, PPulse = 8 (I_{LDR} not included)
			790			Gesture, LDR pulse ON, GPulse = 8 (I_{LDR} not included)
			38			Wait state PON = 1, AEN = PEN = 0
			1.0	10.0		Sleep state [2]
V_{OL} INT, SDA output low voltage	V_{OL}	0		0.4	V	3 mA sink current
I_{LEAK} leakage current, SDA, SCL, INT pins	I_{LEAK}	–5		5	µA	
I_{LEAK} leakage current, LDR P\pin	I_{LEAK}	–10		10	µA	
SCL, SDA input high voltage, V_{IH}	V_{IH}	1.26		V_{DD}	V	
SCL, SDA input low voltage, V_{IL}	V_{IL}			0.54	V	

Notes

1. Values are shown at the VDD pin and do not include current through the IR LED.
2. Sleep state occurs when PON = 0 and I2C bus is idle. If Sleep state has been entered as the result of operational flow, SAI = 1, PON will be high.

Optical Characteristics, V_{DD} = 3 V, T_A = 25 °C, AGAIN = 16×, AEN = 1 (unless otherwise noted)

Parameter	Red Channel		Green Channel		Blue Channel		Units	Test Conditions
	Min	Max	Min	Max	Min	Max		
Irradiance responsivity [1]	0	15	10	42	57	100	%	λ_D = 465 nm [2]
	4	25	54	85	10	45		λ_D = 525 nm [3]
	64	120	0	14	3	29		λ_D = 625 nm [4]

Notes:
1. The percentage shown represents the ratio of the respective red, green, or blue channel value to the clear channel value.
2. The 465 nm input irradiance is supplied by an InGaN light-emitting diode with the following characteristics: dominant wavelength λ_D = 465 nm, spectral halfwidth $\Delta\lambda_{½}$ = 22 nm.
3. The 525 nm input irradiance is supplied by an InGaN light-emitting diode with the following characteristics: dominant wavelength λ_D = 525 nm, spectral halfwidth $\Delta\lambda_{½}$ = 35 nm.
4. The 625 nm input irradiance is supplied by a AlInGaP light-emitting diode with the following characteristics: dominant wavelength λ_D = 625 nm, spectral halfwidth $\Delta\lambda_{½}$ = 15 nm.

RGBC Characteristics, V_{DD} = 3 V, T_A = 25 °C, AGAIN = 16×, AEN = 1 (unless otherwise noted)

Parameter	Min	Typ	Max	Units	Test Conditions
Dark ALS count value		0	3	counts	E_e = 0, AGAIN = 64×, ATIME = 0×DB (100 ms)
ADC integration time step size		2.78		ms	ATIME = 0×FF
ADC number of integration steps	1		256	steps	
Full scale ADC counts per step			1025	counts	
Full scale ADC count value			65535	counts	ATIME = 0×C0 (175 ms)
Gain scaling, relative to 1× gain setting	3.6	4	4.4		4×
	14.4	16	17.6		16×
	57.6	64	70.4		64×
Clear channel irradiance responsivity	18.88	23.60	28.32	counts/(µW/cm2)	Neutral white LED, λ = 560 nm

Proximity Characteristics, V_{DD} = 3 V, T_A = 25 °C, PEN = 1 (unless otherwise noted)

Parameter	Min	Typ	Max	Units	Test Conditions
ADC conversion time step size		696.6		µs	
ADC number of integration steps		1		steps	
Full scale ADC counts			255	counts	
LED pulse count [1]	1		64	pulses	
LED pulse width – LED on time [2]		4		µs	PPLEN = 0
		8			PPLEN = 1
		16			PPLEN = 2
		32			PPLEN = 3
LED drive current [3]		100		mA	LDRIVE = 0
		50			LDRIVE = 1
		25			LDRIVE = 2
		12.5			LDRIVE = 3
LED boost [3]		100		%	LED_BOOST = 0
		150			LED_BOOST = 1
		200			LED_BOOST = 2
		300			LED_BOOST = 3
Proximity ADC count value, no object [4]		10	25	counts	V_{LEDA} = 3 V, LDRIVE = 100 mA, PPULSE = 8, PGAIN = 4x, PPLEN = 8 µs, LED_BOOST = 100%, open view (no glass) and no reflective object above the module.

Table continued on next page...

Proximity Characteristics, V_{DD} = 3 V, T_A = 25 °C, PEN = 1 (unless otherwise noted) (continued)

Parameter	Min	Typ	Max	Units	Test Conditions
Proximity ADC count value, 100 mm distance object [5, 6]	96	120	144	counts	Reflecting object – 73 mm × 83 mm Kodak 90% grey card, 100 mm distance, V_{LEDA} = 3 V, LDRIVE = 100 mA, PPULSE = 8, PGAIN = 4x, PPLEN = 8 µs, LED_BOOST = 100%, open view (no glass) above the module.

Notes:
1. This parameter is ensured by design and characterization and is not 100% tested. 8 pulses are the recommended driving conditions. For other driving conditions, contact Avago Field Sales.
2. Value may be as much as 1.36 µs longer than specified.
3. Value is factory-adjusted to meet the Proximity count specification. Considerable variation (relative to the typical value) is possible after adjustment. LED BOOST increases current setting (as defined by LDRIVE or GLDRIVE). For example, if LDRIVE = 0 and LED BOOST = 100%, LDR current is 100 mA.
4. Proximity offset value varies with power supply characteristics and noise.
5. ILEDA is factory calibrated to achieve this specification. Offset and crosstalk directly sum with this value and is system dependent.
6. No glass or aperture above the module. Tested value is the average of 5 consecutive readings.

Gesture Characteristics, V_{DD} = 3 V, T_A = 25 °C, GEN = 1 (unless otherwise noted)

Parameter	Min	Typ	Max	Units	Test Conditions
ADC conversion time step size [1]		1.39		ms	
LED pulse count [2]	1		64	pulses	
LED pulse width – LED on time [3]		4		µs	GPLEN = 0
		8			GPLEN = 1
		12			GPLEN = 2
		16			GPLEN = 3
LED drive current [4]		100		mA	GLDRIVE = 0
		50			GLDRIVE = 1
		25			GLDRIVE = 2
		12.5			GLDRIVE = 3
LED boost [4]		100		%	LED_BOOST = 0
		150			LED_BOOST = 1
		200			LED_BOOST = 2 [5]
		300			LED_BOOST = 3 [5]
Gesture ADC count value, no object [6]		10	25	counts	V_{LEDA} = 3 V, GLDRIVE = 100 mA, GPULSE = 8, GGAIN = 4x, GPLEN = 8 µs, LED_BOOST = 100%, open view (no glass) and no reflective object above the module, sum of UP & DOWN photodiodes.
Gesture ADC count value [7, 8]	96	120	144	counts	Reflecting object – 73 mm × 83 mm Kodak 90% grey card, 100 mm distance, V_{LEDA} = 3 V, GLDRIVE = 100 mA, GPULSE = 8, GGAIN = 4x, GPLEN = 8 µs, LED_BOOST = 100%, open view (no glass) above the module, sum of UP & DOWN photodiodes.
Gesture wait step size		2.78		ms	GTIME = 0x01

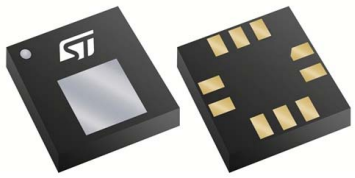
Notes:
1. Each U/D or R/L pair requires a conversion time of 696.6µs. For all four directions the conversion requires twice as much time.
2. This parameter ensured by design and characterization and is not 100% tested. 8 pulses are the recommended driving conditions. For other driving conditions, contact Avago Field Sales.
3. Value may be as much as 1.36 µs longer than specified.
4. Value is factory-adjusted to meet the Gesture count specification. Considerable variation (relative to the typical value) is possible after adjustment.
5. When operating at these LED drive conditions, it is recommended to separate the VDD and VLEDA supplies.
6. Gesture offset value varies with power supply characteristics and noise.
7. ILEDA is factory calibrated to achieve this specification. Offset and crosstalk directly sum with this value and is system dependent.
8. No glass or aperture above the module. Tested value is the average of 5 consecutive readings.

C. Apéndice: Primeras cinco páginas de la hoja de datos LPS22HB

LPS22HB

MEMS nano pressure sensor: 260-1260 hPa absolute digital output barometer

Datasheet - production data



HLGA-10L
(2.0 x 2.0 x 0.76 mm)

Features

- 260 to 1260 hPa absolute pressure range
- Current consumption down to 3 µA
- High overpressure capability: 20x full-scale
- Embedded temperature compensation
- 24-bit pressure data output
- 16-bit temperature data output
- ODR from 1 Hz to 75 Hz
- SPI and I²C interfaces
- Embedded FIFO
- Interrupt functions: Data Ready, FIFO flags, pressure thresholds
- Supply voltage: 1.7 to 3.6 V
- High shock survivability: 22,000 *g*
- Small and thin package
- ECOPACK® lead-free compliant

Applications

- Altimeters and barometers for portable devices
- GPS applications
- Weather station equipment
- Sport watches

Description

The LPS22HB is an ultra-compact piezoresistive absolute pressure sensor which functions as a digital output barometer. The device comprises a sensing element and an IC interface which communicates through I^2C or SPI from the sensing element to the application.

The sensing element, which detects absolute pressure, consists of a suspended membrane manufactured using a dedicated process developed by ST.

The LPS22HB is available in a full-mold, holed LGA package (HLGA). It is guaranteed to operate over a temperature range extending from -40 °C to +85 °C. The package is holed to allow external pressure to reach the sensing element.

Table 1. Device summary

Order code	Temperature range [°C]	Package	Packing
LPS22HBTR	-40 to +85°C	HLGA-10L	Tape and reel

Contents

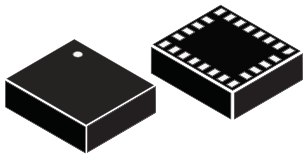
List of tables

D. Apéndice: Primeras cinco páginas de la hoja de datos LSM9DS

LSM9DS1

iNEMO inertial module: 3D accelerometer, 3D gyroscope, 3D magnetometer

Datasheet - production data



LGA-24L (3.5x3x1.0 mm)

Features

- 3 acceleration channels, 3 angular rate channels, 3 magnetic field channels
- ±2/±4/±8/±16 *g* linear acceleration full scale
- ±4/±8/±12/±16 gauss magnetic full scale
- ±245/±500/±2000 dps angular rate full scale
- 16-bit data output
- SPI / I^2C serial interfaces
- Analog supply voltage 1.9 V to 3.6 V
- “Always-on” eco power mode down to 1.9 mA
- Programmable interrupt generators
- Embedded temperature sensor
- Embedded FIFO
- Position and motion detection functions
- Click/double-click recognition
- Intelligent power saving for handheld devices
- ECOPACK®, RoHS and “Green” compliant

Applications

- Indoor navigation
- Smart user interfaces
- Advanced gesture recognition
- Gaming and virtual reality input devices
- Display/map orientation and browsing

Description

The LSM9DS1 is a system-in-package featuring a 3D digital linear acceleration sensor, a 3D digital angular rate sensor, and a 3D digital magnetic sensor.

The LSM9DS1 has a linear acceleration full scale of ±2*g*/±4*g*/±8/±16 *g,* a magnetic field full scale of ±4/±8/±12/±16 gauss and an angular rate of ±245/±500/±2000 dps.

The LSM9DS1 includes an I^2C serial bus interface supporting standard and fast mode (100 kHz and 400 kHz) and an SPI serial standard interface.

Magnetic, accelerometer and gyroscope sensing can be enabled or set in power-down mode separately for smart power management.

The LSM9DS1 is available in a plastic land grid array package (LGA) and it is guaranteed to operate over an extended temperature range from -40 °C to +85 °C.

Table 1. Device summary

Part number	Temperature range [°C]	Package	Packing
LSM9DS1	-40 to +85	LGA-24L	Tray
LSM9DS1TR	-40 to +85	LGA-24L	Tape and reel

Contents

E. Apéndice: Primeras cinco páginas de la hoja de datos nRF52840

nRF52840

Product Specification

v1.1

NORDIC® SEMICONDUCTOR

Feature list

Features:

- *Bluetooth*® 5, IEEE 802.15.4-2006, 2.4 GHz transceiver
  - -95 dBm sensitivity in 1 Mbps *Bluetooth*® low energy mode
  - -103 dBm sensitivity in 125 kbps *Bluetooth*® low energy mode (long range)
  - -20 to +8 dBm TX power, configurable in 4 dB steps
  - On-air compatible with nRF52, nRF51, nRF24L, and nRF24AP Series
  - Supported data rates:
    - *Bluetooth*® 5: 2 Mbps, 1 Mbps, 500 kbps, and 125 kbps
    - IEEE 802.15.4-2006: 250 kbps
    - Proprietary 2.4 GHz: 2 Mbps, 1 Mbps
  - Single-ended antenna output (on-chip balun)
  - 128-bit AES/ECB/CCM/AAR co-processor (on-the-fly packet encryption)
  - 4.8 mA peak current in TX (0 dBm)
  - 4.6 mA peak current in RX
  - RSSI (1 dB resolution)
- ARM® Cortex®-M4 32-bit processor with FPU, 64 MHz
  - 212 EEMBC CoreMark score running from flash memory
  - 52 µA/MHz running CoreMark from flash memory
  - Watchpoint and trace debug modules (DWT, ETM, and ITM)
  - Serial wire debug (SWD)
- Rich set of security features
  - ARM® TrustZone® Cryptocell 310 security subsystem
    - NIST SP800-90A and SP800-90B compliant random number generator
    - AES-128: ECB, CBC, CMAC/CBC-MAC, CTR, CCM/CCM*
    - Chacha20/Poly1305 AEAD supporting 128- and 256-bit key size
    - SHA-1, SHA-2 up to 256 bits
    - Keyed-hash message authentication code (HMAC)
    - RSA up to 2048-bit key size
    - SRP up to 3072-bit key size
    - ECC support for most used curves, among others P-256 (secp256r1) and Ed25519/Curve25519
    - Application key management using derived key model
  - Secure boot ready
    - Flash access control list (ACL)
    - Root-of-trust (RoT)
    - Debug control and configuration
    - Access port protection (CTRL-AP)
  - Secure erase
- Flexible power management
  - 1.7 V to 5.5 V supply voltage range
  - On-chip DC/DC and LDO regulators with automated low current modes
  - 1.8 V to 3.3 V regulated supply for external components
  - Automated peripheral power management
  - Fast wake-up using 64 MHz internal oscillator
  - 0.4 µA at 3 V in System OFF mode, no RAM retention
  - 1.5 µA at 3 V in System ON mode, no RAM retention, wake on RTC
- 1 MB flash and 256 kB RAM
- Advanced on-chip interfaces
  - USB 2.0 full speed (12 Mbps) controller
  - QSPI 32 MHz interface
  - High-speed 32 MHz SPI
  - Type 2 near field communication (NFC-A) tag with wake-on field
    - Touch-to-pair support
  - Programmable peripheral interconnect (PPI)
  - 48 general purpose I/O pins
  - EasyDMA automated data transfer between memory and peripherals
- Nordic SoftDevice ready with support for concurrent multi-protocol
- 12-bit, 200 ksps ADC - 8 configurable channels with programmable gain
- 64 level comparator
- 15 level low-power comparator with wake-up from System OFF mode
- Temperature sensor
- 4x 4-channel pulse width modulator (PWM) unit with EasyDMA
- Audio peripherals: I2S, digital microphone interface (PDM)
- 5x 32-bit timer with counter mode
- Up to 4x SPI master/3x SPI slave with EasyDMA
- Up to 2x I2C compatible 2-wire master/slave
- 2x UART (CTS/RTS) with EasyDMA
- Quadrature decoder (QDEC)
- 3x real-time counter (RTC)
- Single crystal operation
- Package variants
  - aQFN™ 73 package, 7 x 7 mm
  - WLCSP93 package, 3.544 x 3.607 mm

Applications:

- Advanced computer peripherals and I/O devices
  - Mouse
  - Keyboard
  - Multi-touch trackpad
- Advanced wearables
  - Health/fitness sensor and monitor devices
  - Wireless payment enabled devices
- Internet of things (IoT)
  - Smart home sensors and controllers
  - Industrial IoT sensors and controllers
- Interactive entertainment devices
  - Remote controls
  - Gaming controllers

Contents