Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE0624 - Laboratorio de Microcontroladores

Detección de Anomalías en el Latido Cardíaco

Prof. MSc. Marco Villalta Fallas

Por: Andrés Artavia Solano - B90789

II - 2024

Índice

1. Introducción

El monitoreo de la salud cardíaca es fundamental para la detección temprana de anomalías como arritmias, las cuales pueden ser indicativas de condiciones potencialmente graves. El análisis del intervalo R-R, que mide el tiempo entre latidos consecutivos, es una técnica ampliamente utilizada en la cardiología para evaluar la regularidad del ritmo cardíaco y detectar irregularidades. La implementación de sistemas de bajo costo para este propósito es una necesidad creciente, especialmente en aplicaciones portátiles y de monitoreo continuo.

En este proyecto, se desarrolla un sistema basado en Arduino para la detección de anomalías cardíacas mediante el análisis del intervalo R-R y el uso de un modelo de aprendizaje automático. El sistema captura señales analógicas de un sensor cardíaco, las filtra para reducir el ruido y detecta los picos que representan los latidos. A partir de estos picos, calcula los intervalos R-R, los normaliza y los alimenta a un modelo de TensorFlow Lite para clasificar los latidos como normales o anómalos. Finalmente, la clasificación es representada visualmente a través de LEDs, facilitando una interpretación inmediata.

El diseño modular del sistema y su capacidad para operar en tiempo real lo convierten en una herramienta práctica y accesible para el monitoreo continuo de la salud cardíaca. Este enfoque no solo ofrece una solución eficaz para detectar irregularidades en el ritmo cardíaco, sino que también abre la puerta a futuras aplicaciones en dispositivos portátiles y tecnologías de monitoreo remoto, mejorando significativamente la capacidad de respuesta ante eventos cardíacos críticos.

Nota: El código fuente se puede encontrar en la rama "main"del repositorio, en la carpeta "proyecto". `https://github.com/artavias/IE0624/tree/main`

2. Alcances

El proyecto tiene como alcances la detección de latidos cardíacos en tiempo real, clasificándolos como normales o anómalos mediante un modelo de aprendizaje automático entrenado con datos clínicos confiables. Se plantea como un dispositivo portátil, de bajo costo y fácil de usar, ideal para entornos domésticos o con recursos limitados gracias al procesamiento local de los datos directamente en el Arduino Nano 33 BLE, sin depender de una conexión a internet. Además, cuenta con una interfaz sencilla que facilita su comprensión para usuarios sin conocimientos técnicos avanzados. El dispositivo integrará señales en tiempo real desde el sensor AD8232, digitalizándolas para su análisis, y ofrecerá la posibilidad de futuras extensiones, como el almacenamiento de datos o la conexión con aplicaciones móviles para monitoreo remoto. Si bien los datos del sensor no son 100 % precisos y contienen mucho ruido, el sistema puede funcionar para alentar a realizarse un chequeo medico profesional al detectar alguna anomalia consistente en el latido cardíaco de una persona y de esta manera detectar tempranamente afecciones al corazón.

3. Justificación

Este proyecto busca desarrollar un sistema accesible y eficiente para la detección en tiempo real de anomalías en los latidos cardíacos, utilizando un Arduino Nano 33 BLE, un sensor AD8232 y un modelo de aprendizaje automático entrenado con la base de datos MIT-BIH Arrhythmia. Las enfermedades cardiovasculares son una de las principales causas de mortalidad mundial, y un diagnóstico temprano puede salvar vidas. Sin embargo, los sistemas tradicionales de monitoreo suelen ser costosos y de difícil acceso, especialmente en comunidades con menos recursos.

La propuesta combina tecnología asequible y portátil con inteligencia artificial para crear una solución innovadora que identifique patrones anormales de manera precisa y rápida.

4. Nota Teórica

A continuación se muestra un diagrama de pines del Arduino Nano 33 BLE:

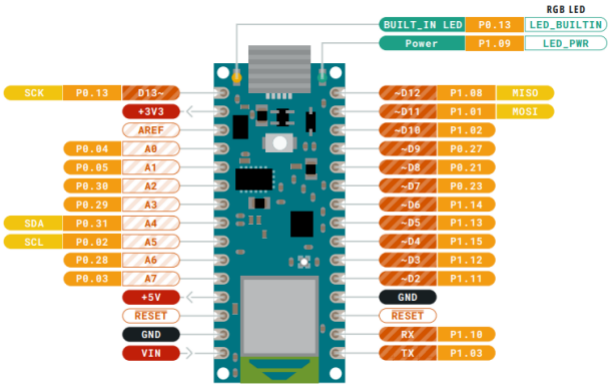


Figura 1: Diagrama de pines. [1]

4.1. Arduino Nano 33 BLE

El Arduino Nano 33 BLE esta diseñada para facilitar la creación de aplicaciones de Internet de las Cosas (IoT), dispositivos portátiles y sistemas embebidos que requieren conectividad inalámbrica y capacidades avanzadas de procesamiento.

El arduino Nano 33 BLE se basa en un microcontrolador de 32 bits ARM Cortex-M4 a 64 MHz. Ofrece una arquitectura eficiente y poderosa que permite el manejo de tareas complejas, como procesamiento de datos en tiempo real y ejecución de modelos de aprendizaje automático optimizados. Soporta comunicación BLE, lo que le permite conectar de manera eficiente dispositivos inalámbricos de bajo consumo, una capacidad clave para aplicaciones que requieren transmisión de datos a smartphones, computadoras u otros dispositivos IoT. Esta placa tiene un sensor de inercia que combina un acelerómetro, un giroscopio y un magnetómetro, proporcionando una solución completa para la detección y el seguimiento del movimiento en tres dimensiones. La integración de estos sensores en una única placa es fundamental para aplicaciones de Reconocimiento de Actividad Humana (HAR), ya que permite medir con precisión la aceleración lineal, la rotación y la orientación en tiempo real. Sin embargo para este laboratorio solo se utilizara el giroscopio el cual registra la velocidad angular, permitiendo detectar rotaciones y movimientos angulares. Magnetómetro: Proporciona datos sobre el campo magnético ambiental, facilitando la orientación espacial del dispositivo.

A diferencia de otras placas Arduino, el Nano 33 BLE cuenta con una capacidad de procesamiento mucho más alta, lo que le permite ejecutar modelos de aprendizaje automático ligeros, como aquellos generados por TensorFlow Lite, directamente en el dispositivo. Esto abre la puerta a aplicaciones de inteligencia artificial en tiempo real en el "borde"de la red, reduciendo la latencia y la necesidad de enviar datos a un servidor externo para su procesamiento.

En resumen, el Arduino Nano 33 BLE, con su potente microcontrolador y sus sensores, es una plataforma ideal para aplicaciones de inteligencia artificial embebida. En el contexto de un sistema de Reconocimiento de Actividad Humana, su capacidad de procesar datos en tiempo real y ejecutarlos en modelos de aprendizaje automático en el "borde"demuestra su versatilidad y potencial para resolver problemas del mundo real con una solución compacta y de bajo costo. [2]

4.2. TensorFlow Lite y redes neuronales

TensorFlow Lite es una versión optimizada de TensorFlow para dispositivos móviles y sistemas embebidos. Permite realizar inferencias en tiempo real con baja latencia y bajo consumo de recursos, lo que resulta fundamental para aplicaciones de HAR en microcontroladores. Para este laboratorio se exporta un modelo entrenado de TensorFlow Lite desde la plataforma Edge Impulse para luego correr en el Arduino Nano 33 BLE y hacer inferencias en tiempo real sobre el movimiento que se esta realizando.

A continuación se muestra el diagrama de una neurona típica en una red neuronal.

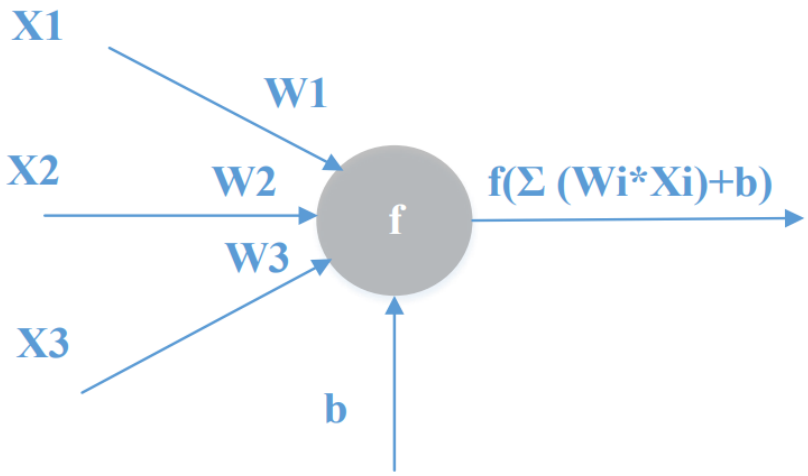


Figura 2: Neurona básica de una red neuronal.[3]

Como se muestra en la figura, la función de cada neurona es realizar el siguiente calculo matemático:

$$output = f\left(\sum (Wi * Xi) + b\right) \tag{1}$$

En donde Wi representa el vector de pesos (weight) y Xi el vector de entradas a la neurona, b seria el sesgo (bias) y la función f es una función de activación como ReLU, tanh entre otras. Estas neuronas se interconectan entre ellas para formar una red neuronal como se muestra a continuación.

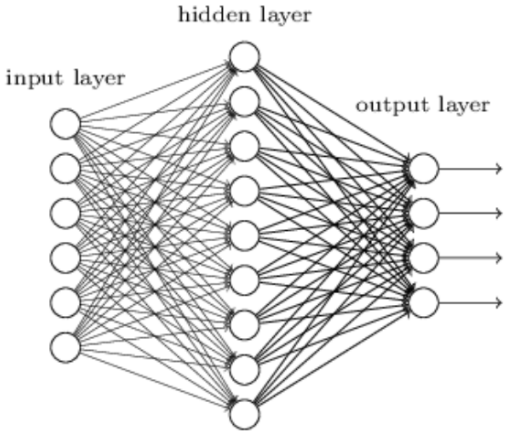


Figura 3: Red Neuronal [3]

Para este laboratorio se entrenó una red neuronal de 3 capas, una capa de entrada y una capa escondida que utilizan una función de activación ReLU. Esta función retorna 0 si el valor de entrada es menor o igual a 0 o retorna el mismo valor de entrada si este es mayor a 0.

4.3. Función de Activación Sigmoide

Para la capa de salida de la red neuronal empleada solo se tiene una neurona, con una función de activación sigmoide σ cuya formula se muestra a continuación:

$$\sigma = \frac{1}{1 + e^{-t}} \tag{2}$$

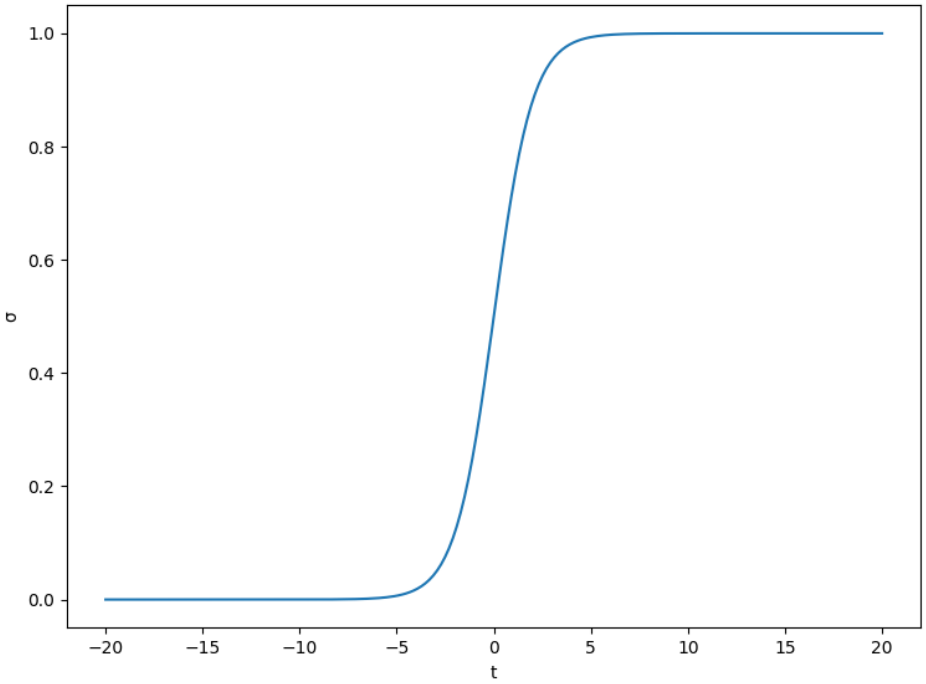


Figura 4: Función Sigmoide. Elaboración propia.

Como se puede notar en la figura 4 la función sigmoide da un resultado de aproximadamente 0 para valores menores a -5 y retorna un resultado de aproximadamente 1 para valores mayores a 5. En el caso de t=0 se tiene que la función sigmoide retorna 0.5. Se tiene entonces que a la salida de la red neuronal vamos a obtener un numero en un rango de [0, 1].

4.4. MIT-BIH Arrythmia

La base de datos MIT-BIH Arrhythmia es una de las colecciones más reconocidas y utilizadas en el ámbito de la investigación médica y el análisis de señales cardíacas. Fue creada por el Laboratorio de Ingeniería Biomédica del Instituto Tecnológico de Massachusetts (MIT) en colaboración con el Hospital Beth Israel (BIH), con el objetivo de proporcionar un recurso estandarizado para el estudio y desarrollo de herramientas de análisis y diagnóstico de electrocardiogramas (ECG).

Este conjunto de datos contiene un total de 48 grabaciones de ECG, cada una con una duración de 30 minutos y registrada en dos canales simultáneos. Las señales fueron muestreadas a una frecuencia de 360 muestras por segundo, utilizando una resolución de 11 bits dentro de

un rango de 10 milivoltios. Estas características garantizan una calidad óptima de las señales para su análisis detallado.

Las grabaciones incluyen anotaciones manuales que identifican más de 100.000 latidos cardíacos, clasificados en diversas categorías, como latidos normales, arritmias ventriculares y supraventriculares, entre otros tipos. Estas anotaciones también especifican la ubicación exacta de cada evento dentro del registro, lo que facilita su uso en el desarrollo y evaluación de algoritmos automáticos de detección y clasificación de latidos.

La base de datos fue diseñada para representar una amplia variedad de condiciones cardíacas, incluyendo ritmos y arritmias poco frecuentes, haciendo de ella un recurso especialmente valioso para el entrenamiento de modelos de aprendizaje automático y la validación de herramientas de diagnóstico. Además, estableció estándares clave en la evaluación de algoritmos de análisis de ECG, convirtiéndose en una referencia global en el campo. [4]

4.5. AD8232

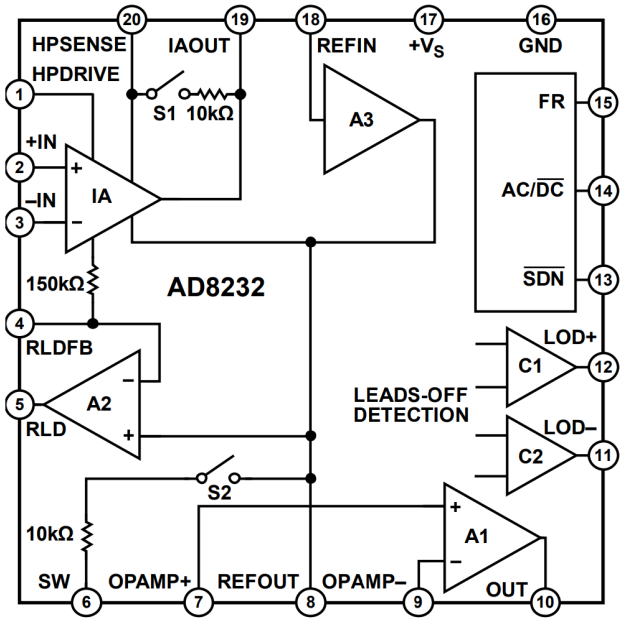


Figura 5: Esquemático del AD8232.

El AD8232 es un módulo especializado en la adquisición de señales de electrocardiograma, diseñado para aplicaciones de monitoreo cardíaco. Este dispositivo se encarga de amplificar y filtrar las señales eléctricas del corazón, que son muy débiles y susceptibles a interferencias, facilitando su análisis en tiempo real mediante microcontroladores.

Una de sus principales características es el amplificador de instrumentación integrado, que proporciona una alta ganancia para amplificar las señales cardíacas, típicamente del orden de microvoltios. Además, cuenta con filtros pasa-altas y pasa-bajas que eliminan el ruido de baja frecuencia, como el causado por movimientos corporales, y de alta frecuencia, como las interferencias de la red eléctrica. Estos filtros están optimizados para capturar las frecuencias características del ECG, que oscilan entre 0.05 Hz y 150 Hz.

El AD8232 también incluye un sistema de referencia interna que genera un voltaje de referencia a la mitad del voltaje de alimentación (Vcc/2), lo que permite procesar la señal del ECG como una señal unipolar. Esto simplifica el diseño del sistema, haciéndolo ideal para dispositivos portátiles. Además, su diseño compacto y ligero lo hace adecuado para aplicaciones móviles y vestibles, como monitores de salud portátiles o dispositivos de fitness.[5]

En este proyecto, el AD8232 desempeña un papel clave al capturar las señales eléctricas del corazón mediante electrodos colocados en la piel. Estas señales se amplifican y filtran antes de ser enviadas al microcontrolador para su digitalización y posterior análisis mediante un modelo de aprendizaje automático. Su capacidad de ofrecer una señal analógica relativamente limpia y confiable es esencial para garantizar el buen desempeño del sistema de detección de anomalías en los latidos cardíacos.

4.6. Intervalo R-R

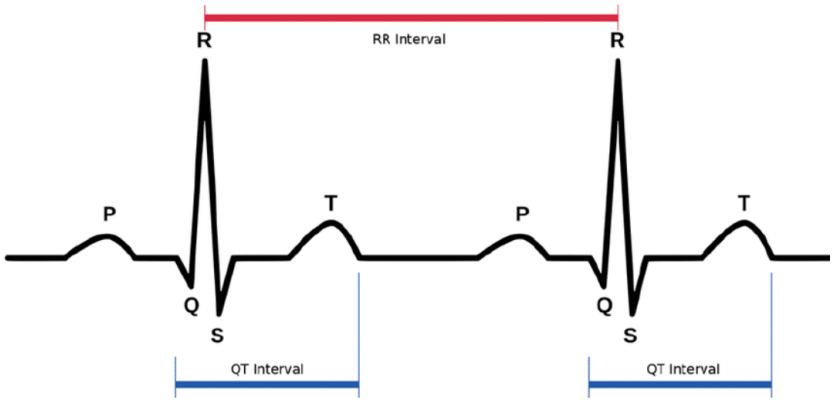


Figura 6: Intervalo R-R [6].

El intervalo R-R es un parámetro fundamental en el análisis de señales de electrocardiograma (ECG). Representa el tiempo entre dos ondas R consecutivas en el ECG, las cuales corresponden al pico máximo de despolarización ventricular durante un ciclo cardíaco. Este intervalo es esencial para evaluar la regularidad de los latidos cardíacos y detectar posibles anomalías.[6]

5. Desarrollo

El primer paso es entrenar el modelo de TensorFlow. Para esto se utilizó el script de Python entrenar_modelo.py. El programa comienza cargando datos de electrocardiogramas de la base de datos MIT-BIH Arrhythmia. Se utiliza la biblioteca wfdb para listar y leer los registros disponibles. Cada registro contiene tanto señales de ECG como anotaciones que identifican eventos específicos en la señal, como los picos R, correspondientes a los latidos del corazón.

Una vez cargados los datos, el programa procede a calcular los intervalos R-R, que son las diferencias temporales entre picos consecutivos. Los intervalos R-R van a ser los datos que se utilizaran para entrenar el modelo.

Estos intervalos se normalizan dividiendo cada valor por el máximo intervalo encontrado, lo que asegura que los datos estén escalados entre 0 y 1. Este paso es crucial para garantizar que la red neuronal reciba datos consistentes y comparables independientemente de las variaciones en la duración de los intervalos.

El procesamiento de los datos incluye el uso de ventanas deslizantes de tamaño fijo, en este caso, de 10 intervalos consecutivos. Para cada ventana, se examinan las anotaciones de los latidos dentro de ese rango. Si todos los latidos son normales (etiquetados como N), la ventana se clasifica como normal. De lo contrario se marca con una anomalía. Los intervalos normalizados y las etiquetas resultantes se almacenan como entradas (x) y salidas (y) para el entrenamiento.

El conjunto de datos se divide en subconjuntos de entrenamiento y prueba, utilizando el 80 % de los datos para entrenamiento y el 20 % para prueba. Además, dado que los datos de ECG suelen estar desbalanceados (es decir, hay muchas más ventanas normales que anómalas), se aplica el método SMOTE (Synthetic Minority Oversampling Technique) para generar ejemplos sintéticos de las clases minoritarias y equilibrar el conjunto de datos.

Con los datos preparados, se construye una red neuronal utilizando TensorFlow Keras. La red consta de tres capas: una capa de entrada con 16 neuronas y una función de activación ReLU, una capa intermedia con 8 neuronas y la misma función de activación, y una capa de salida con una neurona y una función de activación sigmoidal.

El modelo se entrena durante 5 épocas o epochs con un tamaño de lote de 32, reservando el 20 % del conjunto de entrenamiento para validación. Una vez completado el entrenamiento, el modelo es convertido a formato TensorFlow Lite utilizando el conversor integrado de TensorFlow, lo que permite su implementación en dispositivos con recursos limitados. Finalmente, el modelo convertido se guarda en un archivo llamado model.tflite. Este modelo luego es dumpeado a un archivo de C++ utilizando el comando de linux xxd, el cual lo guarda como un arreglo de caracteres hexadecimales en un archivo llamado model.cpp. Se creo posteriormente un archivo de encabezado para poder importar las variables del modelo de manera sencilla. Una parte del codigo de python se muestra a continuación:

```python
db_path = 'mit-bih-arrhythmia-database-1.0.0'
name_rec = '101'
path_rec = os.path.join(db_path, name_rec)

rec_list = wfdb.get_record_list('mitdb')
x =[]
y = []
window_size = 10

for rec in rec_list:
    annotation = wfdb.rdann(os.path.join(db_path, rec), 'atr')
    peaks = annotation.sample
    intervals = np.diff(peaks)
    normalized_intervals = intervals / intervals.max()

    for i in range(len(intervals) - window_size + 1):
        window_annotations = annotation.symbol[i:i + window_size]
        window_intervals = normalized_intervals[i:i + window_size]

        abnomarl_counter = sum(1 for annotation in window_annotations if annotation == 'V')

        if all(annotation == 'N' for annotation in window_annotations) or (abnomarl_counter >= 2):
            x.append(window_intervals)
            y.append(abnomarl_counter >= 2)

x = np.array(x)
y = np.array(y)

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
smote = SMOTE()
x_train_smote, y_train_smote = smote.fit_resample(x_train, y_train)

model = Sequential([Dense(16, input_dim = window_size, activation = 'relu'),
                    Dense(8, activation = 'relu'),
                    Dense(1, activation = 'sigmoid')])

model.compile(optimizer=Adam(learning_rate=0.001), loss='binary_crossentropy', metrics=['accuracy'])

history = model.fit(x_train_smote, y_train_smote, epochs = 5, batch_size = 32, validation_split = 0.2

converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()

with open('model.tflite', 'wb') as file:
    file.write(tflite_model)
```

Figura 7: Codigo de python para entrenar el modelo.

Una vez teniendo el modelo compatible se procedió con el código de arduino en C++. El codigo se puede encontrar en el archivo ´´arrythmia.ino". Este programa analiza una señal analógica recibida a través del pin A0. El propósito del sistema es procesar la señal, detectar picos en ella y realizar inferencias utilizando el modelo. Dependiendo de los resultados, controla dos LEDs en el Arduino.

El sistema lee constantemente valor analógico desde el pin A0. Este valor representa la señal proporcionada por el sensor AD8232. Como estamos interesado en obtener la distancia temporal entre picos, el nivel de tensión no es relevante por lo que podemos trabajar con el dato obtenido de la lectura del pin directamente, sin tener que mapear el valor a un rango de tensión.

Para reducir el ruido y obtener datos más representativos, las muestras se procesan a través de un filtro. Este filtro mantiene un buffer circular de las últimas 10 muestras, calcula su promedio y devuelve un valor filtrado.

Se analiza el valor filtrado para determinar si se ha detectado un pico. Esto se hace mediante una clase específica llamada detector_picos, que compara cada valor con un umbral predefinido de 750. El sistema asegura que los picos sean válidos, es decir, no se detectan picos consecutivos si el valor no baja por debajo del umbral.

Cuando se detecta un pico, se calcula el tiempo transcurrido desde el último pico en milisegundos. Este intervalo se guarda en otro buffer circular que puede almacenar hasta 10 intervalos R-R.

Cuando el buffer de intervalos está lleno se alimentan los datos al modelo de TensorFlow Lite para realizar una inferencia. El modelo predice una salida basada en los intervalos de tiempo entre los picos.

Dependiendo del resultado de la inferencia, el programa controla dos LEDs integrados en el Arduino Nano 33 BLE. Si la salida del modelo es mayor a 0.5, el LED rojo se enciende, de lo contrario, se enciende el LED verde.

```
109  void loop() {
110    static long ultimo_tiempo_pico = millis(); // Tiempo del ultimo pico detectado
111    int muestra = analogRead(A0); // Leer senal analogica del pin A0
112
113    // Filtrar senal con media movil
114    filtro.agregar(muestra);
115    int valor_filtrado = filtro.promedio();
116
117    // Mostrar valor filtrado en el monitor serie
118    Serial.println(valor_filtrado);
119
120    // Detectar picos
121    if (detector_picos.detectar(valor_filtrado)) {
122      long tiempo_actual = millis();
123      int intervalo = tiempo_actual - ultimo_tiempo_pico;
124      ultimo_tiempo_pico = tiempo_actual;
125
126      // Guardar intervalo en el buffer de picos
127      buffer_intervalos_picos.push(intervalo);
128
129      // Si el buffer de intervalos ests lleno realizar inferencia
130      if (buffer_intervalos_picos.isFilled()) {
131        for (int i = 0; i < TAMANO_VENTANA; ++i) {
132          interpreter->input(0)->data.f[i] = static_cast<float>(buffer_intervalos_picos[i]) / 1000.0; // Normalizacion
133        }
134
135        // Ejecutar el modelo
136        if (interpreter->Invoke() == kTfLiteOk) {
137          float prediccion = interpreter->output(0)->data.f[0];
138          encender_leds(prediccion); // Controlar LEDs según la predicción
139        }
140      }
141    }
142
143    delay(10); // Retardo para evitar lecturas excesivas
144  }
```

Figura 8: Función loop() del código del arduino.

6. Video Demostración

Un video de la demostración del proyecto en funcionamiento se puede encontrar en el siguiente enlace: `https://www.youtube.com/shorts/xM2OYDD5qho`

7. Análisis de los resultados

En este caso se tienen conectados los electrodos para medir la onda de electrocardiograma. El graficador serial que se muestra en el video esta utilizando los datos filtrados del sensor AD8232 recibidos por el puerto serial. Como se puede observar, los datos aun poseen bastante ruido. Sin embargo también se puede notar como hay picos que sobresalen muy por encima de los demás datos de forma periódica. Estos son los picos que sobrepasan el umbral de 750 para ser detectados como picos y corresponden a la parte R de la señal de electrocardiograma. El sistema detecta correctamente estos picos y logra tomar los intervalos R-R para alimentar el modelo y así poder indicar que la onda recibida de electrocardiograma es una onda normal con una luz verde en el arduino. Al final del video se desconecta el electrodo de la piel y se puede notar como inmediatamente la luz del arduino cambia a roja. Los datos mandados por el puerto serial fueron capturados mediante un script de python y guardados en archivo csv con el cual se realizó la siguiente gráfica:

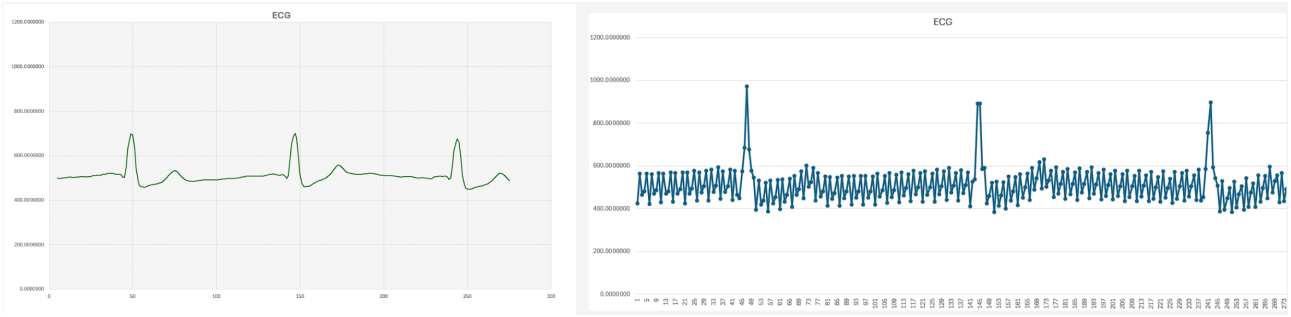


Figura 9: Señal del sensor procesada resultante. A la derecha los datos puntuales y al izquierda una linea de mejor ajuste.

Como se observa en la figura 9 la señal resultante aunque es ruidosa, se puede distinguir visualmente los picos R de la onda. Gracias al procesamiento de señal realizado se pudo extraer exitosamente los intervalos R-R para alimentar al modelo y realizar inferencias exitosas. Se puede notar la semejanza con una onda típica de electrocardiograma 6 e identificar todas las partes de la onda, sobretodo los picos R.

El sistema desarrollado demuestra ser una herramienta efectiva para la detección de anomalías en el ritmo cardíaco. El análisis basado en el intervalo R-R permite identificar irregularidades en el ritmo cardíaco, como lo son las arritmias.

Sin embargo hay un punto importante a resaltar y es que el umbral de 750 se tomo arbitrariamente después de analizar los datos procesados del sensor. Se notó que prácticamente ninguna parte de la señal que no fuera un pico R sobrepasaba este limite y por eso se definió como el umbral. Este valor puede variar de persona a persona y en este caso solo se baso en los datos de una sola persona para definir ese umbral. Es por esto que el sistema es muy propenso a fallar cuando una persona diferente se conecte los electrodos. Pueden existir persona que sobrepasarían el umbral en muchas parte de la onda que no corresponden a picos R así como pueden existir personas cuyas señales resultantes no sobrepasan el umbral, resultando entonces en fallos de las predicciones del modelo. El sistema de esta forma requeriría una calibración cada vez que una persona diferente lo use.

8. Conclusiones

El preprocesamiento de la señal analógica mediante un filtro de promedio de datos en un buffer fue fundamental para reducir el ruido y mejorar la precisión de los datos analizados. Esto aseguró que los valores del intervalo R-R utilizados fueran representativos, lo que a su vez optimizó la eficacia del modelo de aprendizaje automático. El modelo, implementado en TensorFlow Lite, demuestra que es posible realizar inferencias en tiempo real incluso en sistemas embebidos como Arduino.

Además, el sistema integra LEDs como indicadores visuales que permiten al usuario identificar fácilmente si los latidos cardíacos detectados son normales o presentan anomalías. Esta interfaz sencilla mejora la accesibilidad del sistema y su utilidad en contextos prácticos.

La interpretación del intervalo R-R permite detectar patrones anómalos que podrían ser indicativos de problemas cardíacos. Este enfoque destaca por su simplicidad y efectividad al proporcionar información clave para clasificaciones automáticas que de otra forma pueden pasar desapercibidas, resultando en posibles complicaciones de salud en las personas. Se recalca que el sistema no es un sustituto de un profesional con equipo mas adecuado y por lo tanto no es ideal para realizar diagnósticos, si no mas bien es una exploración a la posibilidad de que en un futuro estos diagnósticos de afecciones cardíacas puedan ser automatizados o al menos asistidos por inteligencia artificial.

En conclusión, se demuestra que el procesamiento de datos y las predicciones utilizando modelos de aprendizaje automático son posible en dispositivos con recursos limitados y de costos relativamente bajos como lo es el Arduino Nano 33 BLE para detectar anomalías en el latido cardíaco.

Referencias

[1] Arduino. Arduino nano 33 ble datasheet. https://docs.arduino.cc/resources/datasheets/ABX00030-datasheet.pdf.

[2] Arduino. Nano 33 ble features. https://docs.arduino.cc/hardware/nano-33-ble/#features.

[3] Michael Nielsen. Why are deep neural networks hard to train? http://neuralnetworksanddeeplearning.com/chap5.html, 2019.

[4] Mark RG. Moody GB. The impact of the mit-bih arrhythmia database. IEEE Eng in Med and Biol 20(3):45-50 (May-June 2001). (PMID: 11446209).

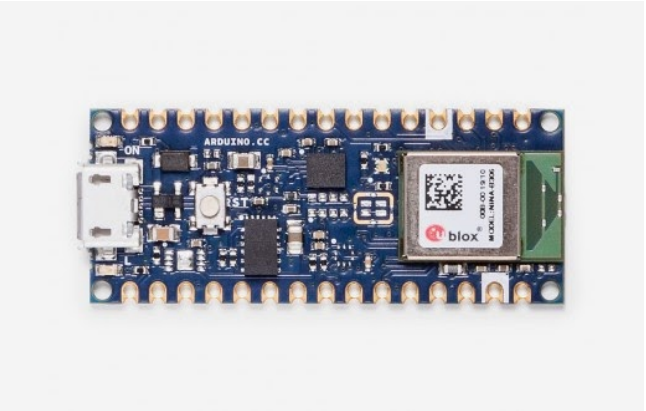
[5] Analag Devices. Ad8232 datasheet. https://www.analog.com/media/en/technical-documentation/data-sheets/ad8232.pdf.

[6] Hugo Parrales. Intervalo qt. https://cerebromedico.com/electrocardiograma/intervalo-qt/.

9. Apéndice

9.1. Hoja de datos del Arduino Nano 33 BLE

Product Reference Manual
SKU: ABX00030



Description

Arduino® Nano 33 BLE is a miniature sized module containing a NINA B306 module, based on Nordic nRF52480 and containing an Arm® Cortex®-M4F 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, basic IoT application scenarios

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
- **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 does not have a battery charger but can be powered through USB or headers.

NOTE: Nano 33 BLE 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 Application Examples

Sound spectrum: Create a sound spectrum to visualize sound frequencies. Connect an Nano 33 BLE and a microphone or amplifier.

Social distancing sensor: Keeping the social distance has become more important than ever to ensure your own, as well as others health. By connecting an Arduino Nano 33 BLE with a sensor and a LED display, you can create a wearable band that alerts you when you get too close to other people.

Healthy plant scanner: Watering your plants isn't always enough to keep them happy. Diseases, lack of sunlight etc. could also be vital factors for unhealthy plants. Keep your plants happy by creating a detector and train it to detect any diseases, all with an Nano 33 BLE.

1.2 Ratings

1.2.1 Recommended Operating Conditions

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

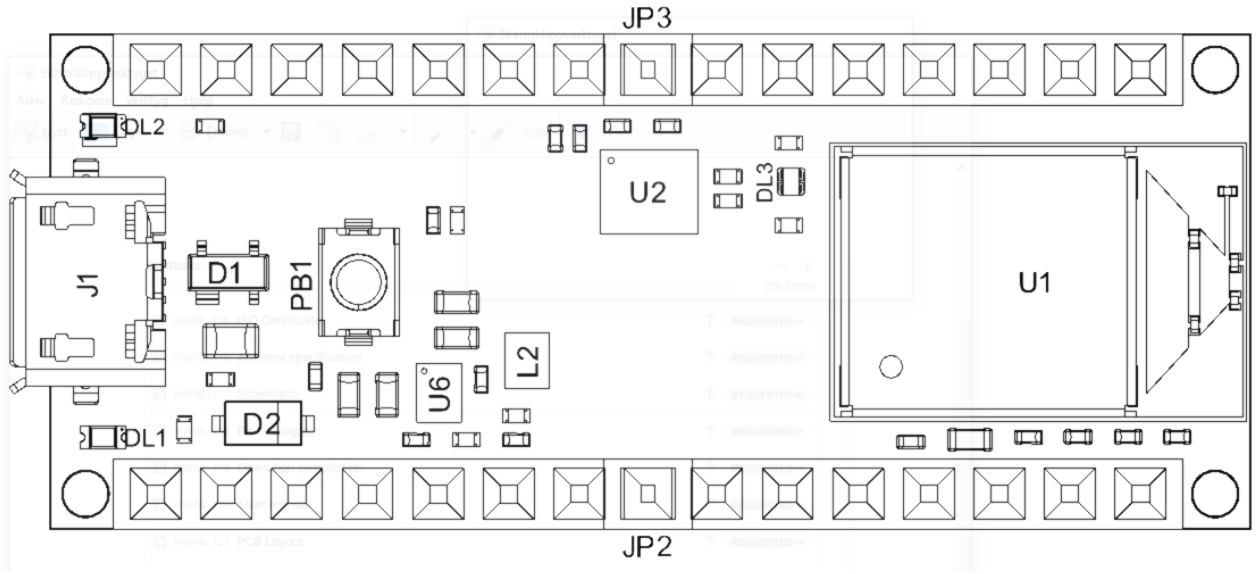
1.3 Power Consumption

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

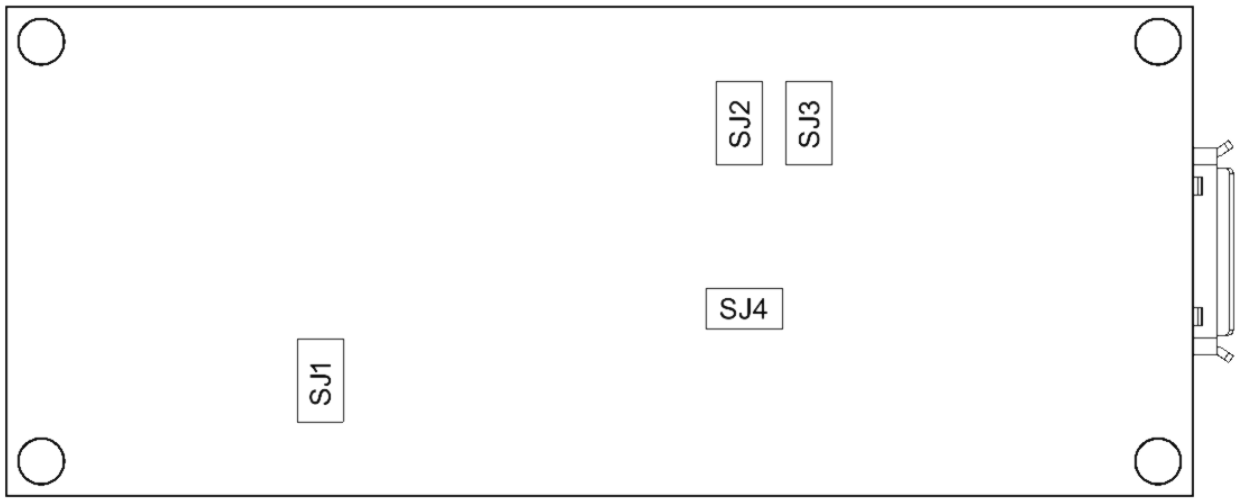
2.1.1 Top



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
DL1	Led L	DL2	Led Power

2.1.2 Bottom



Board topology bottom

Ref.	Description	Ref.	Description
SJ1	VUSB Jumper	SJ2	D7 Jumper
SJ3	D8 Jumper	SJ4	3v3 Jumper

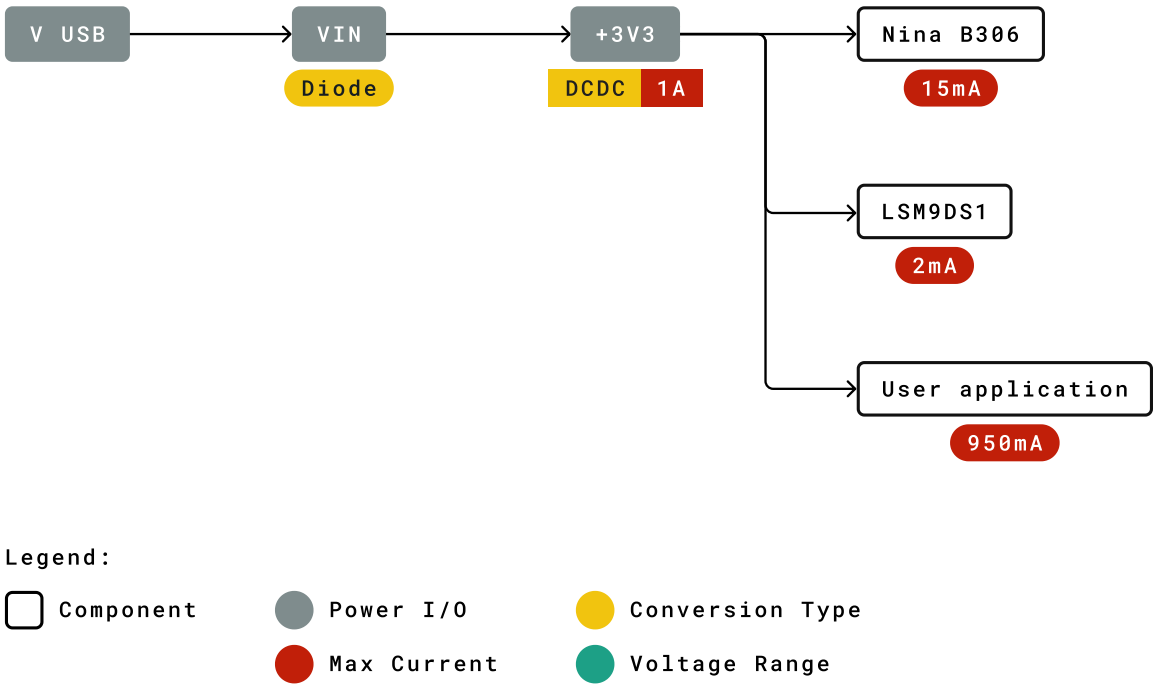
2.2 Processor

The Main Processor is a Arm® Cortex®-M4F running at up to 64MHz. Most of its pins are connected to the external headers, however some are reserved for internal communication with the wireless module and the on-board internal I^2C peripherals (IMU and Crypto).

NOTE: As opposed to other Arduino Nano boards, pins A4 and A5 have an internal pull up and default to be used as an I^2C Bus so usage as analog inputs is not recommended.

2.3 Power Tree

The board can be powered via USB connector, V_{IN} or V_{USB} pins on headers.



Power tree

NOTE: Since V_{USB} feeds V_{IN} via a Schottky diode and a DC-DC regulator specified minimum input voltage is 4.5V the minimum supply voltage from USB has to be increased to a voltage in the range between 4.8V to 4.96V depending on the current being drawn.

3 Board Operation

3.1 Getting Started - IDE

If you want to program your Nano 33 BLE while offline you need to install the Arduino Desktop IDE [1] To connect the Nano 33 BLE to your computer, you'll need a Micro-B USB cable. This also provides power to the board, as indicated by the LED.

3.2 Getting Started - Arduino Cloud Editor

All Arduino boards, including this one, work out-of-the-box on the Arduino Cloud Editor [2], by just installing a simple plugin.

The Arduino Cloud Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

3.3 Getting Started - Arduino Cloud

All Arduino IoT enabled products are supported on Arduino Cloud which allows you to Log, graph and analyze sensor data, trigger events, and automate your home or business.

3.4 Sample Sketches

Sample sketches for the Nano 33 BLE can be found either in the "Examples" menu in the Arduino IDE or in Arduino Docs [4].

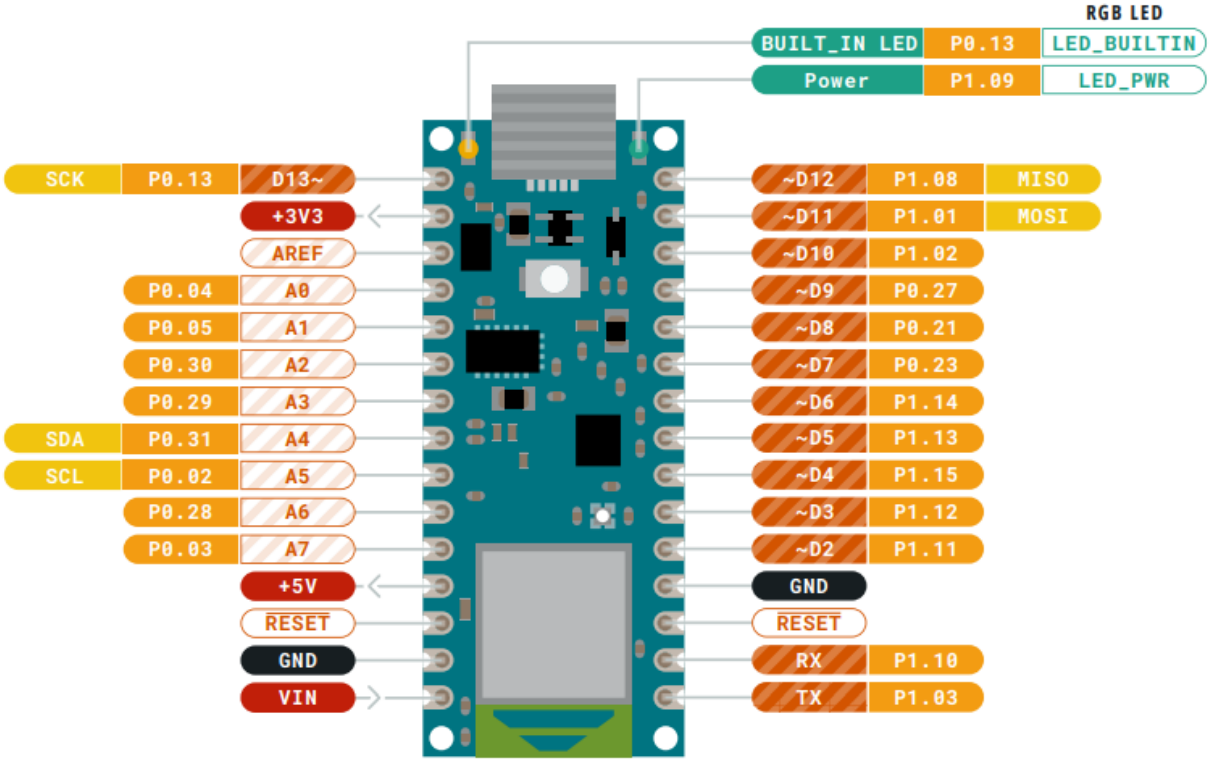
3.5 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on Arduino Project Hub **[5]**, the Arduino Library Reference **[6]** and the online store **[7]** where you will be able to complement your board with sensors, actuators and more

3.6 Board Recover

All Arduino boards have a built-in bootloader which allows flashing the board via USB. In case a sketch locks up the processor and the board is not reachable anymore via USB it is possible to enter bootloader mode by double-tapping the reset button right after power up.

4 Connector Pinouts



Pinout

4.1 USB

Pin	Function	Type	Description
1	VUSB	Power	Power Supply Input. If board is powered via VUSB from header this is an Output **(1)**
2	D-	Differential	USB differential data -
3	D+	Differential	USB differential data +
4	ID	Analog	Selects Host/Device functionality
5	GND	Power	Power Ground

4.2 Headers

The board exposes two 15 pin connectors which can either be assembled with pin headers or soldered through castellated vias.

Pin	Function	Type	Description
1	D13	Digital	GPIO/Built-in LED
2	+3V3	Power Out	Internally generated power output to external devices
3	AREF	Analog	Analog Reference; can be used as GPIO
4	A0/DAC0	Analog	ADC in/DAC out; can be used as GPIO
5	A1	Analog	ADC in; can be used as GPIO
6	A2	Analog	ADC in; can be used as GPIO
7	A3	Analog	ADC in; can be used as GPIO
8	A4/SDA	Analog	ADC in; I2C SDA; Can be used as GPIO **(1)**
9	A5/SCL	Analog	ADC in; I2C SCL; Can be used as GPIO **(1)**
10	A6	Analog	ADC in; can be used as GPIO
11	A7	Analog	ADC in; can be used as GPIO
12	VUSB	Power In/Out	Normally NC; can be connected to VUSB pin of the USB connector by shorting a jumper

Pin	Function	Type	Description
13	RST	Digital In	Active low reset input (duplicate of pin 18)
14	GND	Power	Power Ground
15	VIN	Power In	Vin Power input
16	TX	Digital	USART TX; can be used as GPIO
17	RX	Digital	USART RX; can be used as GPIO
18	RST	Digital	Active low reset input (duplicate of pin 13)
19	GND	Power	Power Ground
20	D2	Digital	GPIO
21	D3/PWM	Digital	GPIO; can be used as PWM
22	D4	Digital	GPIO
23	D5/PWM	Digital	GPIO; can be used as PWM
24	D6/PWM	Digital	GPIO, can be used as PWM
25	D7	Digital	GPIO
26	D8	Digital	GPIO
27	D9/PWM	Digital	GPIO; can be used as PWM
28	D10/PWM	Digital	GPIO; can be used as PWM
29	D11/MOSI	Digital	SPI MOSI; can be used as GPIO
30	D12/MISO	Digital	SPI MISO; can be used as GPIO

4.3 Debug

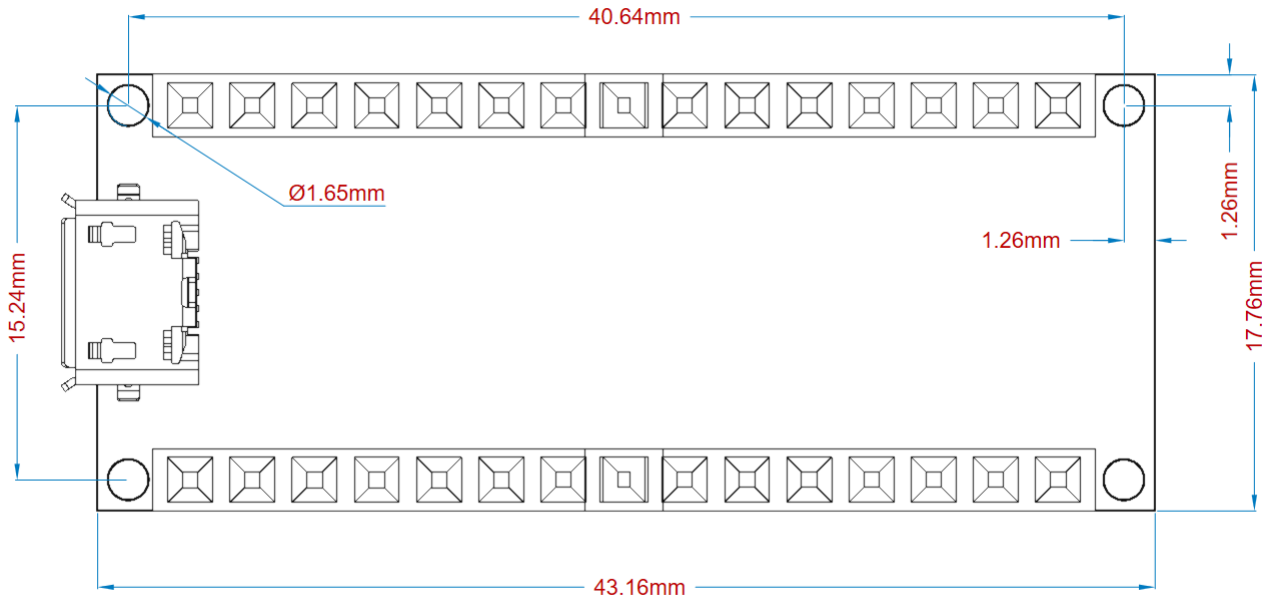
On the bottom side of the board, under the communication module, debug signals are arranged as 3x2 test pads with 100 mil pitch with pin 4 removed. Pin 1 is depicted in Figure 3 – Connector Positions

Pin	Function	Type	Description
1	+3V3	Power Out	Internally generated power output to be used as voltage reference
2	SWD	Digital	nRF52480 Single Wire Debug Data
3	SWCLK	Digital In	nRF52480 Single Wire Debug Clock
5	GND	Power	Power Ground
6	RST	Digital In	Active low reset input
1	+3V3	Power Out	Internally generated power output to be used as voltage reference

5 Mechanical Information

5.1 Board Outline and Mounting Holes

The board measures are mixed between metric and imperial. Imperial measures are used to maintain 100 mil pitch grid between pin rows to allow them to fit a breadboard whereas board length is Metric



Board layout

6 Certifications

6.1 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

6.2 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	Maximum limit (ppm)
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions : No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the

REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.

6.3 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

7 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.

Important: The operating temperature of the EUT can't exceed 85°C and shouldn't be lower than -40°C.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 2014/53/EU. This product is allowed to be used in all EU member states.

Frequency bands	Maximum output power (ERP)
863-870Mhz	5.47 dBm

8 Company Information

Company name	Arduino S.r.l
Company Address	Via Andrea Appiani 25 20900 MONZA Italy

9 Reference Documentation

Reference	Link
Arduino IDE (Desktop)	https://www.arduino.cc/en/software
Arduino Cloud Editor	https://create.arduino.cc/editor
Arduino Cloud Editor - Getting Started	https://docs.arduino.cc/arduino-cloud/guides/editor/
Arduino Documentation	https://docs.arduino.cc
Arduino Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/
Arduino Store	https://store.arduino.cc/
Forum	http://forum.arduino.cc/
SAMD21G18	https://ww1.microchip.com/downloads/aemDocuments/documents/MCU32/ProductDocuments/DataSheets/SAM-D21DA1-Family-Data-Sheet-DS40001882G.pdf
NINA W102	https://content.u-blox.com/sites/default/files/NINA-W10_DataSheet_UBX-17065507.pdf
ECC608	https://ww1.microchip.com/downloads/aemDocuments/documents/SCBU/ProductDocuments/DataSheets/ATECC608A-CryptoAuthentication-Device-Summary-Data-Sheet-DS40001977B.pdf
MPM3610	https://www.monolithicpower.com/pub/media/document/MPM3610_r1.01.pdf
NINA Firmware	https://github.com/arduino/nina-fw
ECC608 Library	https://github.com/arduino-libraries/ArduinoECCX08
LSM6DSL Library	https://github.com/stm32duino/LSM6DSL

10 Revision History

Date	Revision	Changes
25/04/2024	3	Updated link to new Cloud Editor
03/08/2022	2	Reference documentation links updates
21/04/2021	1	General datasheet updates