

Revisiones parciales 7.5/15pts

Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0624 Laboratorio de Microcontroladores
II ciclo 2022

-Revisiones parciales:Atrasados e

Proyecto de microcontroladores

Sistema de seguridad de detección de caras con Arduino
Nano 33 BLE Sense

Luis Javier Herrera B93840
Juan Ignacio Montealegre Salazar B95001

Profesor: Marco Villalta Fallas

Miércoles 7 de diciembre del 2022

Índice

Índice de figuras	III
Índice de tablas	IV
1. Resumen	1
2. Objetivos y alcances	1
2.1. Objetivo General	1
2.2. Objetivos Específicos	1
3. Justificación	1
4. Nota teórica	2
4.1. Placa Arduino Nano 33 BLE Sense	2
4.2. Características principales	3
4.2.1. Procesador	3
4.2.2. Módulo con cámara OV7576	5
4.3. TinyML	6
4.3.1. TensorFlow Lite	6
4.3.2. Clasificador en cascada	7
4.3.3. Análisis de componentes principales	8
5. Desarrollo y Análisis de Resultados	9
5.1. Desarrollo del circuito	9
5.2. Desarrollo del programa	9
5.2.1. Recopilación de datos	9
5.2.2. Creación del modelo y entrenamiento	11
5.2.3. Exportación del modelo	14
5.3. Repositorio de Git	14
5.4. Funcionalidad del circuito y del programa	14
6. Conclusiones y Recomendaciones	17
7. Referencias	18
8. Anexos	19

Índice de figuras

1.	Placa Arduino Nano 33 BLE Sense [1].	2
2.	Topología general de la placa [1].	3
3.	Diagrama de pines de ARM Cortex-M4 [1].	4
4.	Tabla de registros de procesador ARM Cortex-M4 [1].	4
5.	Tabla de registros de procesador ARM Cortex-M4 [1].	5
6.	Cámara OV7675 [2].	6
7.	Funcionamiento del TinyML con TensorFlowLite [3].	6
8.	Características de HAAR para el reconocimiento de rostros [4].	7
9.	Círculo empleado en el proyecto. Elaboración propia.	9
10.	Rostro resaltado al aplicar modelo de algoritmo de Viola Jones. Elaboración propia.	10
11.	Set de datos de imágenes con rostro visible. Elaboración propia.	10
12.	Set de datos de imágenes con rostro portando mascarilla. Elaboración propia.	11
13.	Detección de cara utilizando modelo PCA. Elaboración propia.	11
14.	Detección de mascarilla utilizando modelo PCA. Elaboración propia.	12
15.	Resultado de segundo intento de modelo. Elaboración propia.	12
16.	Especificaciones de modelo en EdgeImpulse. Elaboración propia.	13
17.	Resultado de modelo generado con EdgeImpulse. Elaboración propia.	13
18.	Detección de mascarilla. Elaboración propia.	14
19.	Detección de rostro. Elaboración propia.	15
20.	Caso en el que no se detecta ni un rostro ni una mascarilla. Elaboración propia.	15
21.	Salida de monitor serial al detectar una mascarilla. Elaboración propia.	16
22.	Salida de monitor serial al detectar un rostro. Elaboración propia.	16

Índice de tablas

1. Resumen

En el presente proyecto del curso Laboratorio de Microcontroladores se presenta una aplicación de detección de rostros humanos utilizando Machine Learning y el microcontrolador Arduino Nano 33 BLE Sense. Para el proceso de entrenamiento se utilizaron muestras en forma de imágenes de los integrantes del proyecto y con ayuda de la librería Open CV se recortaron las imágenes para obtener solamente la porción de la cara necesaria. Posteriormente, se intentó elaborar el modelo de aprendizaje con diferentes enfoques, intentando como primer método un clasificador en cascada y análisis PCA. Debido a complicaciones que se tuvieron exportando el modelo creado al microcontrolador se optó por la opción de utilizar la herramienta Edge Impulse para entrenar y crear un modelo predeterminado para el procesamiento y clasificación de imágenes a base neuronal densa. El rendimiento del modelo fue bastante bueno obteniendo porcentajes de precisión de hasta el 94.36 % en la etapa de pruebas. Finalmente, se exportó el modelo como una librería para Arduino y se cargó directamente al microcontrolador. Además, se utilizaron los LEDs incorporados en el microcontrolador para indicar la inferencia realizada por el modelo. A pesar de no cumplir con la totalidad de los objetivos, se logró realizar un programa basado en Machine Learning funcional para el reconocimiento de rostros humanos así como profundizar en conceptos relacionados con el tema y con la aplicación de detección de rostros.

Palabras clave: *Arduino, Machine Learning, sistemas de seguridad, reconocimiento facial*

2. Objetivos y alcances

2.1. Objetivo General

- Crear un sistema de seguridad funcional para la detección de rostros humanos y vestimentas utilizando el microcontrolador Arduino Nano 33 BLE Sense.

2.2. Objetivos Específicos

- Lograr detectar un rostro humano por medio de la utilización de machine learning haciendo uso de un microcontrolador.
- Implementar un sistema de comunicación por medio de IoT para el envío de datos en caso de detectar un rostro humano.
- Profundizar conceptos abordados a lo largo del curso por medio de la investigación y práctica.

3. Justificación

Los sistemas de seguridad se utilizan ya sea en viviendas propias o en edificaciones de toda índole con el objetivo de mantener la propiedad segura ante la posibilidad de un robo. Estos pueden estar compuestos por cámaras, sensores de movimiento, cerraduras, alarmas, entre otros componentes. Conforme la tecnología ha ido avanzando en las últimas décadas, los sistemas de seguridad se han ido modernizando para realizar tareas mucho más complejas y más eficientes. Llegando entonces a hacer uso de los últimos avances de la tecnología como los son el Machine Learning y el IoT para establecer una conexión más directa con el usuario y a la misma vez ampliar su rango de aplicaciones posibles.

El proyecto que nos ocupa se plantea entonces elaborar una sección de uno de estos sistemas de seguridad modernos en la cual se utilicen los conceptos vistos en el curso. Con el propósito de de meramente poner en práctica las habilidades aprendidas durante el curso así como elementos nuevos. Para la realización del proyecto se va a utilizar la cámara OV7675 junto con el microcontrolador Arduino Nano 33 BLE Sense como componentes principales. Lo anteriormente mencionado con el fin de crear un sistema de seguridad simplificado en donde, utilizando la imagen capturada por la cámara, se detecte en tiempo real un rostro humano y detectar si este porta algún tipo de vestimenta como una mascarilla o gorra que no permita visualizar bien su rostro. La aplicación elegida se puede considerar de relevancia ya que en los bancos y otros establecimientos de importancia se requiere poder tener una imagen clara del rostro de los clientes. La principal limitante del proyecto es la complejidad que este pueda abarcar ya que desarrollar un algoritmo de reconocimiento facial funcional puede ser bastante difícil. Debido a esto se propone entonces la tarea principal de lograr un reconocimiento de rostros humanos y objetos como mascarillas y gorras. Por último, con la elaboración de este proyecto se busca expandir el aprendizaje con los temas relacionados a los microcontroladores y sus diversas aplicaciones como los es el aprendizaje autónomo en los microcontroladores. Así también, para poder entender a grandes rasgos diferentes maneras de modernizar y automatizar los sistemas de seguridad actuales.

4. Nota teórica IOT -2.5

En esta sección se describe el microcontrolador utilizado en esta ocasión, así como cada periférico utilizado (registros e instrucciones/funciones), componente electrónico complementario y el diseño del circuito final. El proceso de pruebas realizadas para llegar al circuito final se abarca en la sección "Desarrollo y Análisis de Resultados".

4.1. Placa Arduino Nano 33 BLE Sense

El MCU conocido como Arduino Nano 33 BLE Sense es una placa de tamaño reducido la cual contiene un módulo NINA B306 que se basa en Nordic nRF52480 y que posee un chip criptográfico con la capacidad de almacenar de forma segura certificados así como claves compartidas y una IMU de 9 ejes de operación. Este chip criptográfico se denomina Cortex M4F [1]. La placa en sí se puede montar como un componente de tipo DIP (cuando se montan los cabezales de clavija) o también como un componente STM (soldándolo directamente a los *pads* en la palca).

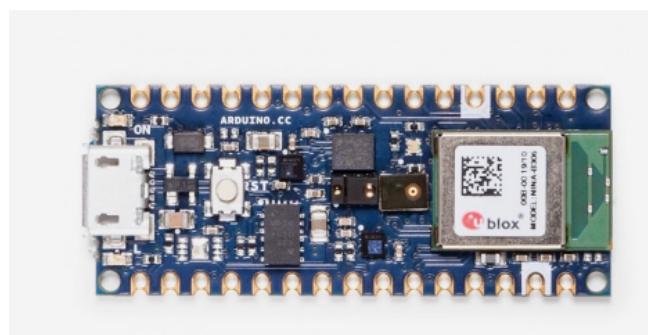


Figura 1: Placa Arduino Nano 33 BLE Sense [1].

4.2. Características principales

La placa Arduino Nano 33 BLE Sense cuenta con una variedad de aplicaciones principalmente al desarrollo, mejoras y usos en IoT. Para lograr esto, este MCU cuenta con una amplia lista de periféricos dentro de los que se encuentran los siguientes:

- Módulo NINA B306 (en este módulo se encuentra el procesador 64 MHz Arm® Cortex-M4F con FPU)
- LSM9DS1 (IMU de 9 ejes)
- LPS22HB (barómetro y sensor de temperatura)
- HTS221 (sensor de humedad relativa)
- APDS-9960 (sensor de proximidad digital, luz ambiente, RGB y gestos)
- MP34DT05 (micrófono digital)
- ATECC608A (Crypto Chip)
- MPM3610 (convertidor DC-DC)

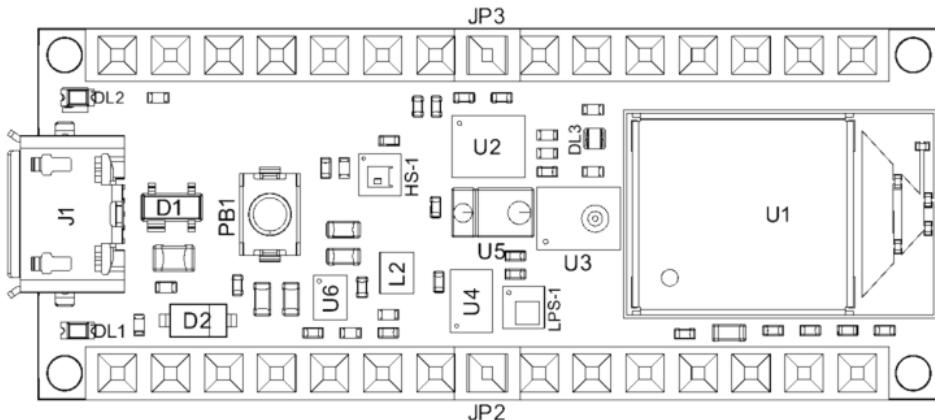


Figura 2: Topología general de la placa [1].

4.2.1. Procesador

El procesador ARM Cortex-M4 con unidad de punto flotante (FPU) tiene un conjunto de instrucciones de 32 bits que implementa un superconjunto de instrucciones de 16 y 32 bits para maximizar la densidad del código y la actuación. Este procesador pone en práctica métodos que le permiten aumentar el rendimiento mientras se logra el máximo ahorro de energía el procesamiento de señales. Dentro de estas técnicas se encuentran: instrucciones de procesamiento digital de señales, instrucciones de acumulación y multiplicación de un solo ciclo, división de hardware, instrucciones de datos múltiples de instrucción única de 8 y 16 bits y una unidad de punto flotante de precisión única [5].

El *ARM Cortex Microcontroller Software Interface Standard* (CMSIS) es una capa de abstracción de hardware para la familia de procesadores ARM Cortex y es implementando y disponible en el m4 CPU. La velocidad de reloj de la CPU es de 64 MHz y contiene 1 MB de

memoria flash y 256 kB de RAM que se pueden usar para el código y el almacenamiento de datos [5]. A continuación se muestra el diagrama de pines del procesador:

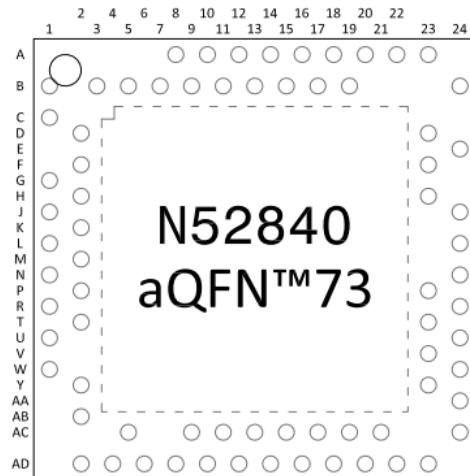


Figura 3: Diagrama de pines de ARM Cortex-M4 [1].

Además la tabla completa de registros del procesador se puede observar las Figuras 4 y 5.

Register	Offset	Description
CODEPAGESIZE	0x010	Code memory page size
CODESIZE	0x014	Code memory size
DEVICEID[0]	0x060	Device identifier
DEVICEID[1]	0x064	Device identifier
ER[0]	0x080	Encryption root, word 0
ER[1]	0x084	Encryption root, word 1
ER[2]	0x088	Encryption root, word 2
ER[3]	0x08C	Encryption root, word 3
IR[0]	0x090	Identity Root, word 0
IR[1]	0x094	Identity Root, word 1
IR[2]	0x098	Identity Root, word 2
IR[3]	0x09C	Identity Root, word 3
DEVICEADDRTYPE	0x0A0	Device address type
DEVICEADDR[0]	0x0A4	Device address 0
DEVICEADDR[1]	0x0A8	Device address 1
INFO.PART	0x100	Part code
INFO.VARIANT	0x104	Build code (hardware version and production configuration)
INFO.PACKAGE	0x108	Package option
INFO.RAM	0x10C	RAM variant
INFO.FLASH	0x110	Flash variant
INFO.UNUSED8[0]	0x114	Reserved
INFO.UNUSED8[1]	0x118	Reserved
INFO.UNUSED8[2]	0x11C	Reserved
PRODTEST[0]	0x350	Production test signature 0
PRODTEST[1]	0x354	Production test signature 1
PRODTEST[2]	0x358	Production test signature 2
TEMP.A0	0x404	Slope definition A0
TEMP.A1	0x408	Slope definition A1
TEMP.A2	0x40C	Slope definition A2
TEMP.A3	0x410	Slope definition A3
TEMP.A4	0x414	Slope definition A4
TEMP.A5	0x418	Slope definition A5
TEMP.B0	0x41C	Y-intercept B0

Figura 4: Tabla de registros de procesador ARM Cortex-M4 [1].

Register	Offset	Description
TEMP.B1	0x420	Y-intercept B1
TEMP.B2	0x424	Y-intercept B2
TEMP.B3	0x428	Y-intercept B3
TEMP.B4	0x42C	Y-intercept B4
TEMP.B5	0x430	Y-intercept B5
TEMP.T0	0x434	Segment end T0
TEMP.T1	0x438	Segment end T1
TEMP.T2	0x43C	Segment end T2
TEMP.T3	0x440	Segment end T3
TEMP.T4	0x444	Segment end T4
NFC.TAGHEADER0	0x450	Default header for NFC tag. Software can read these values to populate NFCID1_3RD_LAST, NFCID1_2ND_LAST, and NFCID1_LAST.
NFC.TAGHEADER1	0x454	Default header for NFC tag. Software can read these values to populate NFCID1_3RD_LAST, NFCID1_2ND_LAST, and NFCID1_LAST.
NFC.TAGHEADER2	0x458	Default header for NFC tag. Software can read these values to populate NFCID1_3RD_LAST, NFCID1_2ND_LAST, and NFCID1_LAST.
NFC.TAGHEADER3	0x45C	Default header for NFC tag. Software can read these values to populate NFCID1_3RD_LAST, NFCID1_2ND_LAST, and NFCID1_LAST.
TRNG90B.BYTES	0xC00	Amount of bytes for the required entropy bits
TRNG90B.RCCUTOFF	0xC04	Repetition counter cutoff
TRNG90B.APCUTOFF	0xC08	Adaptive proportion cutoff
TRNG90B.STARTUP	0xC0C	Amount of bytes for the startup tests
TRNG90B.ROSC1	0xC10	Sample count for ring oscillator 1
TRNG90B.ROSC2	0xC14	Sample count for ring oscillator 2
TRNG90B.ROSC3	0xC18	Sample count for ring oscillator 3
TRNG90B.ROSC4	0xC1C	Sample count for ring oscillator 4

Figura 5: Tabla de registros de procesador ARM Cortex-M4 [1].

4.2.2. Modulo con cámara OV7576

El módulo con cámara OV7576 cuenta con una cámara CMOS que se utiliza comúnmente para aplicaciones como juguetes, teléfonos celulares, multimedia PC y cámaras estáticas [2]. Entre sus principales características están:

- soporte para tamaños de imagen: VGA (640x480), QVGA (320x240) y QQVGA (160x120)
- compatibilidad con formatos de salida: YUV4:2:2, Raw RGB, UIT656, RGB565
- interfaz de salida paralela de puerto de video digital (DVP)
- bucle de bloqueo de fase en chip
- regulador integrado de 1,5 V para núcleo
- capaz de mantener los valores de registro al apagarse
- controles programables para velocidad de cuadro, espejo y volteo, AEC/AGC y ventanas

Además, algunas especificaciones clave:

- **tamaño de matriz activa:** 640x480
- **fuente de alimentación:** analógico: 2,6 – 3,0 V, núcleo: 1.5V DC + 5% (regulador interno), E/S: 1,71 – 3,0 V
- **requerimientos de energía:** activo: 98 mW, en espera: 60 W
- **rango de temperatura:** en funcionamiento: -30°C a 70°C, imagen estable: 0°C a 50°C
- **formatos de salida:** YUV422, Raw RGB, ITU656, RGB565
- **tamaño de la lente:** 1/9"

- **ángulo del rayo principal de la lente:** 21°
- **frecuencia de reloj de entrada:** de 1,5 a 27 MHz

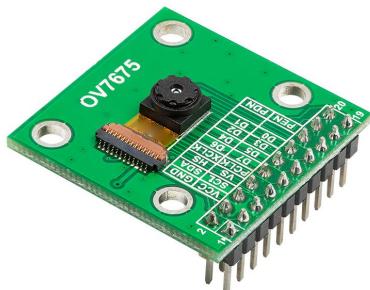


Figura 6: Cámara OV7675 [2].

4.3. TinyML

El Tiny Machine Learning (TinyML) se define ampliamente como un campo de crecimiento exponencial en tecnologías y aplicaciones de aprendizaje automático que incluye hardware (circuitos integrados y dedicados), algoritmos y software capaces de realizar análisis de datos de sensores en dispositivos en las áreas de visión, biomédicos, audio, IMU, entre otros [6]. Estos se caracterizan por consumir potencia extremadamente baja, típicamente en el rango de mW y hasta menor. Gracias a los anteriormente mencionado permite una variedad de casos de uso continuamente encendidos y enfocado a dispositivos que funcionan con baterías.

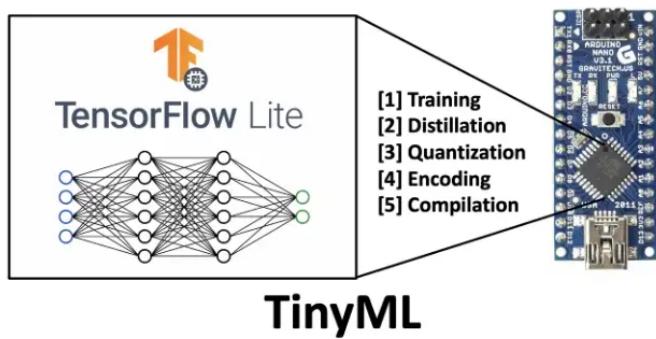


Figura 7: Funcionamiento del TinyML con TensorFlowLite [3].

4.3.1. TensorFlow Lite

TensorFlow es una biblioteca de software de código abierto para inteligencia artificial y aprendizaje automático con redes neuronales profundas [7]. TensorFlow fue desarrollado por Google Brain para uso interno en la empresa y de código abierto desde el 2015.

TensorFlow Lite es un marco de aprendizaje profundo de código abierto diseñado para la inferencia en el dispositivo. Esta librería proporciona un conjunto de herramientas que permite el aprendizaje automático en el dispositivo al dar la posibilidad de que los desarrolladores ejecuten sus modelos entrenados en dispositivos y computadoras móviles, integrados y de IoT [7]. Es compatible con plataformas como Linux, sistemas operativos de dispositivos móviles y MCU. TensorFlow Lite está especialmente optimizado para el aprendizaje automático en el dispositivo (Edge ML). Como modelo Edge ML, es adecuado para la implementación en dispositivos de borde con recursos limitados.

4.3.2. Clasificador en cascada

La detección de objetos utilizando clasificadores en cascada basados en características HAAR es un método eficaz de detección de objetos. Se enfoca en el aprendizaje automático en el que una función de cascada se entrena a partir de muchas imágenes positivas y negativas. Posteriormente, se utiliza para detectar objetos en otras imágenes. Para su implementación en la detección de rostros, el algoritmo necesita muchas imágenes de rostros e imágenes sin rostros para entrenar al clasificador [4]. Después, se tiene que extraer características de él. Para esto, se utilizan las características de HAAR que se aprecian en la Figura 8. Cada característica es un valor único obtenido al restar la suma de píxeles debajo del rectángulo blanco de la suma de píxeles debajo del rectángulo negro.

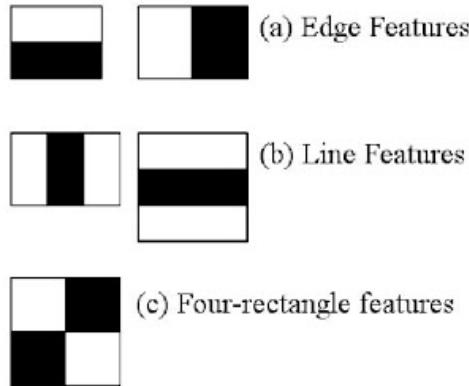


Figura 8: Características de HAAR para el reconocimiento de rostros [4].

Todos los tamaños y ubicaciones posibles de cada núcleo se utilizan para calcular muchas funciones. Para cada cálculo de características, se necesita encontrar la suma de los píxeles debajo de los rectángulos blanco y negro. Se aplica cada una de las funciones en todas las imágenes de entrenamiento. Para cada característica, encuentra el mejor umbral en el que clasificará las caras en positivas y negativas. Se seleccionan las características con una tasa de error mínima, lo que significa que son las características que clasifican con mayor precisión las imágenes de rostros y no rostros. Pero, en una imagen la mayor parte de la imagen es una región sin cara [4]. Por lo tanto, es una mejor idea tener un método simple para verificar si una ventana no es una región de la cara. Si no es así, el algoritmo desecha la ventana de inmediato. En su lugar, se concentra más en las regiones donde puede haber una cara.

Para lo anteriormente mencionado se propone el concepto de **cascada de clasificadores**. En lugar de aplicar las 6000 funciones en una ventana, las funciones se agrupan en diferentes etapas de clasificadores y se aplican una por una. Si una ventana falla en la primera etapa, se elimina y no se consideran las características restantes de esa ventana. Si pasa, se aplica la

segunda etapa de características y continúe el proceso [4]. La ventana que pasa por todas las etapas es una región frontal lográndose así el objetivo principal.

4.3.3. Análisis de componentes principales

El análisis de componentes principales, o PCA, es un método de reducción de la dimensionalidad que a menudo se usa para reducir la dimensionalidad de grandes conjuntos de datos, transformando un gran conjunto de variables en uno más pequeño que aún contiene la mayor parte de la información en el gran conjunto [8]. La reducción del número de variables de un conjunto de datos naturalmente se produce sacrificando la precisión, pero el truco en la reducción de la dimensionalidad es cambiar un poco de precisión por simplicidad. Porque los conjuntos de datos más pequeños son más fáciles de explorar y visualizar y hacen que el análisis de datos sea mucho más fácil y rápido para los algoritmos de aprendizaje automático sin variables extrañas para procesar [8].

La idea principal del análisis de componentes principales entonces es reducir la cantidad de variables de un conjunto de datos mientras se conserva la mayor cantidad de información posible.

5. Desarrollo y Análisis de Resultados

En esta sección se comenta de forma detallada el desarrollo del proyecto. Primero se explica el desarrollo del circuito como tal, y luego el diseño del programa a partir del programa deseado.

5.1. Desarrollo del circuito

El circuito en este caso corresponde a los componentes presentes en el kit de Tiny Machine Learning de Arduino, el cual la Placa Arduino Nano 33 BLE Sense, una cámara OV7675 y un shield que permite conectar la placa con la cámara e incluye un botón, así como otras salidas. Al acoplarse se tiene el circuito de la siguiente figura.

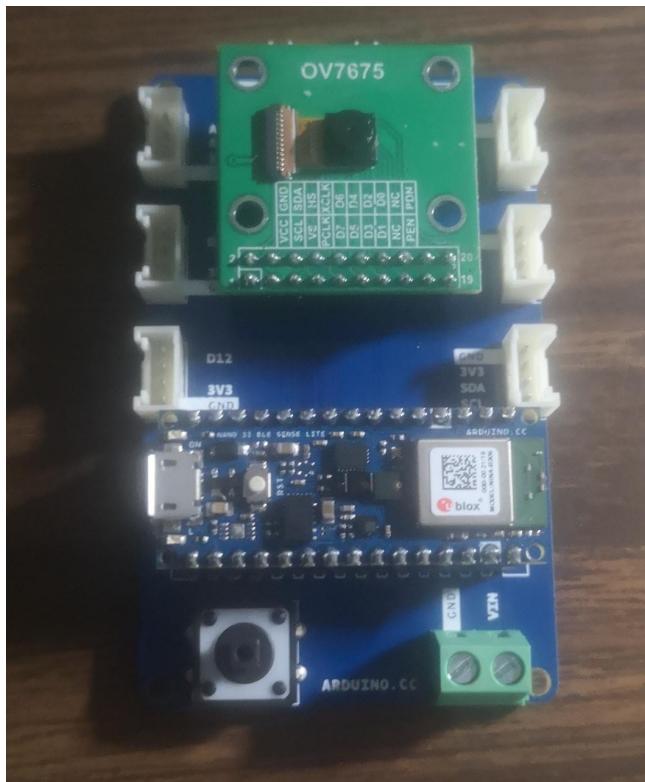


Figura 9: Circuito empleado en el proyecto. Elaboración propia.

5.2. Desarrollo del programa

El desarrollo del programa se centra en la creación de una red neuronal, la cual se puede dividir en los siguientes tres partes:

1. Recopilación de datos
 2. Creación del modelo y entrenamiento
 3. Entrenamiento del modelo

Se procede a explicar lo realizado en cada sección:

5.2.1. Recopilación de datos

Se parte de los datos obtenidos del algoritmo de *Viola Jones*, los cuales permiten, junto con una función de clasificación en cascada, detectar un rostro humano visto de frente en una

imagen. Se hace esto para tomar diversas fotos de solo las caras de una persona y no entrenar el modelo con el resto de la imagen, ya que el objetivo es entrenar para detectar rostros. Al aplicar la función de clasificación en Cascada de OpenCV a una imagen de prueba y resaltar la zona que coincide con el modelo se obtiene la siguiente imagen:

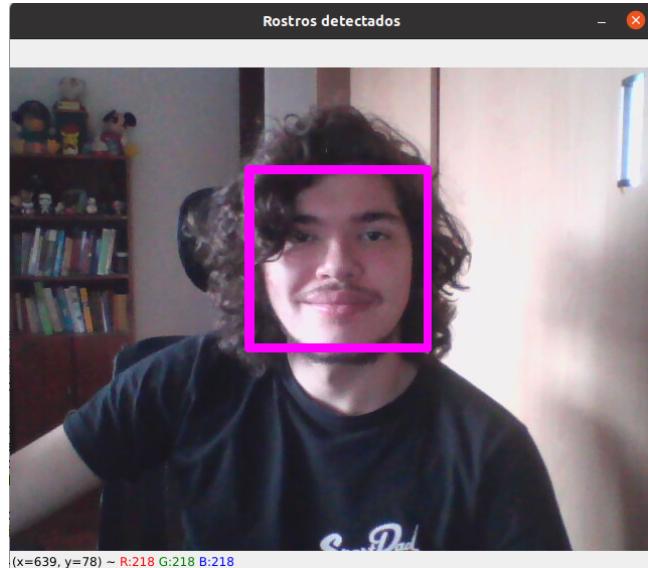


Figura 10: Rostro resaltado al aplicar modelo de algoritmo de Viola Jones. Elaboración propia.

De la imagen anterior luego se guarda una imagen nueva que contiene únicamente lo contenido dentro del cuadro. Esto se realiza para un total de 2000 imágenes captadas por la cámara de video de la computadora en uso, donde 1000 fueron tomadas con el rostro visible y las otras 1000 fueron tomadas utilizando una mascarilla.

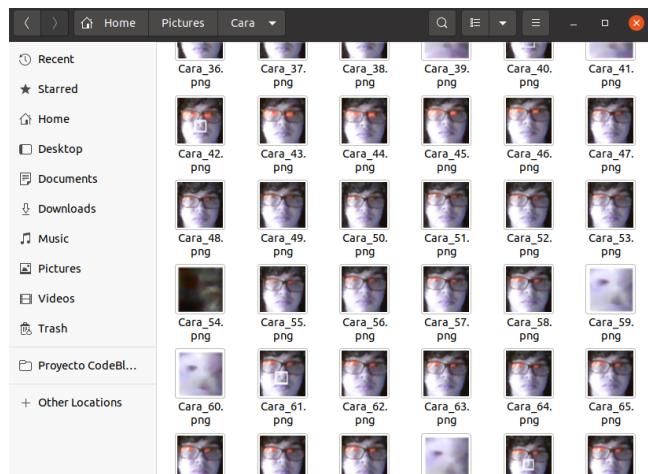


Figura 11: Set de datos de imágenes con rostro visible. Elaboración propia.

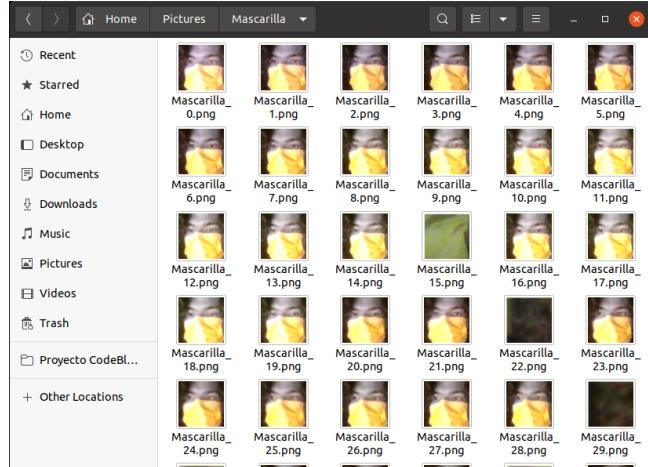


Figura 12: Set de datos de imágenes con rostro portando mascarilla. Elaboración propia.

De esta forma se tienen ya los sets de datos necesarios para entrenar el modelo.

5.2.2. Creación del modelo y entrenamiento

Con respecto a la creación del modelo, primero se buscar desarrollar una red neuronal basada en el análisis principal de componentes, ya que es un método simple que genera modelos ligeros y muy precisos. Al generar este modelo, entrenarlo y probarlo con los datos captados por la cámara de la computadora se obtienen resultados satisfactorios, ya que se logran detectar mascarillas tal y como se muestra en las siguientes figuras.

Esto no tier



Figura 13: Detección de cara utilizando modelo PCA. Elaboración propia.



Figura 14: Detección de mascarilla utilizando modelo PCA. Elaboración propia.

Sin embargo, no fue posible exportar este modelo a la placa Arduino, por lo que se opta por crear el modelo desde Tensorflow definiendo las neuronas y capas de un modelo. Este método, además de demandar mucho tiempo, no resultó exitoso, tal y como se muestra en la siguiente figura:

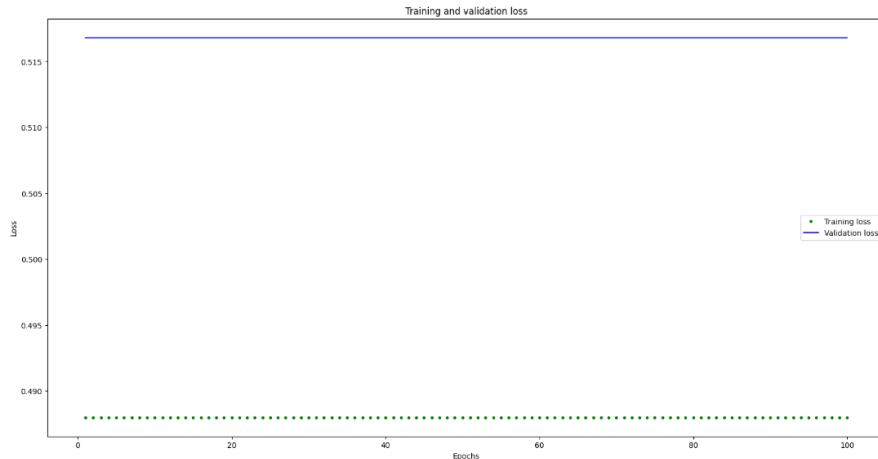


Figura 15: Resultado de segundo intento de modelo. Elaboración propia.

Finalmente, se opta por crear el modelo utilizando la herramienta de EdgeImpulse. Esta herramienta facilita la creación de una red neuronal por su interfaz gráfica. Además, permite exportar el modelo directamente a un microcontrolador. En las siguientes figuras se muestran las especificaciones indicadas para crear la red neuronal (impulso), así como los resultados del entrenamiento.

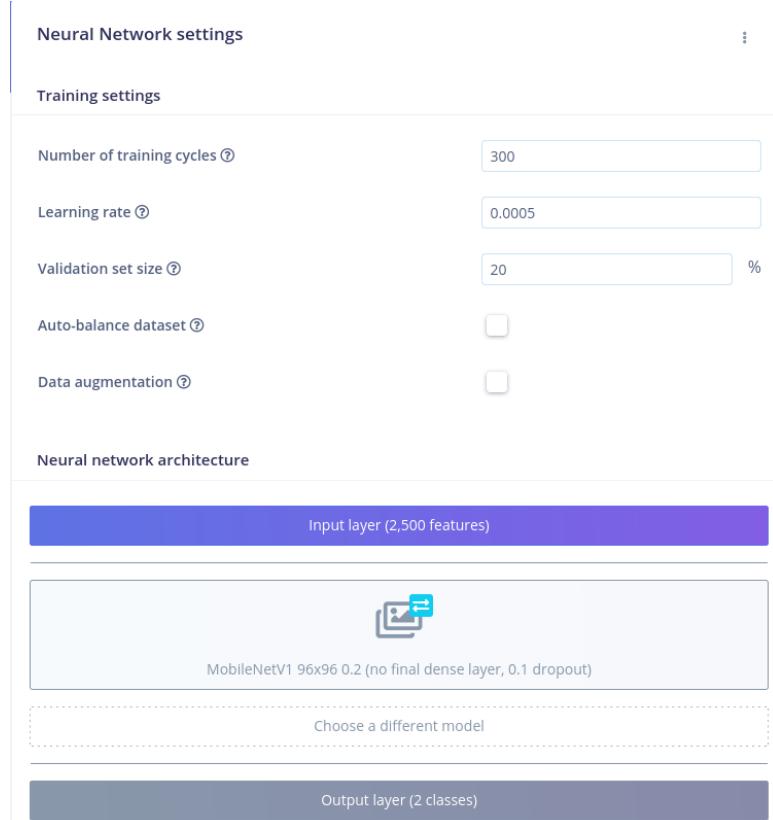


Figura 16: Especificaciones de modelo en EdgeImpulse. Elaboración propia.



Figura 17: Resultado de modelo generado con EdgeImpulse. Elaboración propia.

5.2.3. Exportación del modelo

Una vez que se tiene el modelo ya generado y entrenado, en el caso de EdgeImpulse, es posible exportar una librería para la placa deseada que contiene la red neuronal desarrollada y además posee un ejemplo de prueba. Se parte de este ejemplo para añadir las funciones adicionales deseadas, las cuales corresponden a encender una luz en específico para la predicción realizada. Se establecen los siguientes colores:

- Morado: predicción de más de 60 % de un rostro.
- Amarillo: predicción de más de 60 % de una mascarilla.
- Celeste: no se cumple ninguno de los casos anteriores

Ya con esto implementado se concluye el programa del proyecto.

5.3. Repositorio de Git

Este proyecto se trabajó en un repositorio de Git para poder tener un control de las diversas versiones de todos los archivos que componen el proyecto. Se utilizó la plataforma de GitHub y el repositorio se encuentran en este link. Este proyecto se encuentra en el directorio “Proyecto”.

5.4. Funcionalidad del circuito y del programa

La funcionalidad del circuito se comprueba al probar el funcionamiento del circuito cuando la cámara se coloca frente a un rostro sin mascarilla y ante uno portando una mascarilla, así como el resultado cuando lo que observa la cámara no logra una asociatividad mayor al 60 % con alguno de los casos mencionados. Debido a que es complicado mostrar imágenes donde el Arduino hace la predicción directamente, se adjuntan imágenes de las luces que enciende la placa ante cada uno de los casos mencionados.



Figura 18: Detección de mascarilla. Elaboración propia.

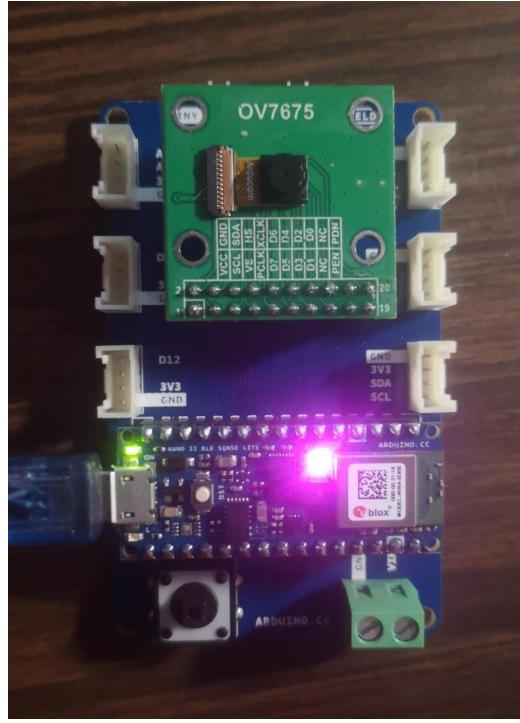


Figura 19: Detección de rostro. Elaboración propia.

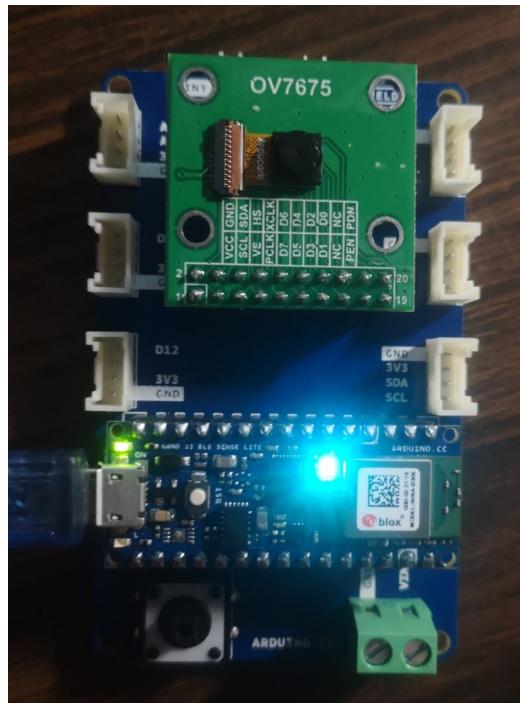


Figura 20: Caso en el que no se detecta ni un rostro ni una mascarilla. Elaboración propia.

En las siguientes imágenes se muestran ejemplos de donde se predice que se detecta una mascarilla y un rostro respectivamente.

```

Starting inferencing in 2 seconds...
Taking photo...
Predictions (DSP: 3 ms., Classification: 175 ms., Anomaly: 0 ms.):
  Cara: 0.06250
  Mascarilla: 0.93750

Starting inferencing in 2 seconds...
Taking photo...
Predictions (DSP: 3 ms., Classification: 175 ms., Anomaly: 0 ms.):
  Cara: 0.00781
  Mascarilla: 0.99219

```

Figura 21: Salida de monitor serial al detectar una mascarilla. Elaboración propia.

```

Starting inferencing in 2 seconds...
Taking photo...
Predictions (DSP: 3 ms., Classification: 175 ms., Anomaly: 0 ms.):
  Cara: 0.97656
  Mascarilla: 0.02344

Starting inferencing in 2 seconds...
Taking photo...
Predictions (DSP: 3 ms., Classification: 175 ms., Anomaly: 0 ms.):
  Cara: 0.99219
  Mascarilla: 0.00781

```

Figura 22: Salida de monitor serial al detectar un rostro. Elaboración propia.

Es importante mencionar que la distancia a la que se debe colocar la cámara de la persona es aproximadamente 22 cm, valor que se encontró luego de múltiples pruebas. Aplicando esta distancia y estando en un ambiente con luz controlada el desempeño del modelo es muy bueno, ya que las predicciones realizadas son muy certeras. Sin embargo, el modelo solo se entrenó con los rasgos faciales de una persona, por lo que presenta dificultad en realizar buenas predicciones con otras personas. Esto se puede solucionar entrenando el modelo con sets de datos más variados.

6. Conclusiones y Recomendaciones

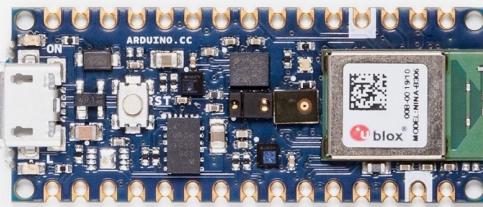
En la elaboración del proyecto se llegaron a conocer las características de lo que conlleva el desarrollo de una red neuronal y algunas formas diferentes que existen para hacerlo, así como la forma en que se puede exportar una red neuronal a un microcontrolador. A pesar de no cumplir con la totalidad de los objetivos propuestos, se logró desarrollar un programa basado en Machine Learning para el reconocimiento fácil básico y funcional aplicado a microcontroladores. Un gran aspecto positivo del proyecto consiste en que se profundizaron conceptos de Machine Learning enfocados en la detección de rostros como lo fueron el de PCA, análisis en cascadas y utilización y manejo de herramienta EdgeImpulse. Un objetivo que no se logró alcanzar, aparte de no poder detectar múltiples objetos como gorras, consiste en la propuesta inicial de enviar los datos a un servidor IoT. Este objetivo, al delimitar mejor el proyecto, resultó no ser relevante para el desarrollo del mismo. Con esto a un lado se lograron satisfacer los objetivos más importantes del proyecto.

Como recomendaciones se tiene llevar un mejor manejo del tiempo, ya que los temas relacionados a Inteligencia Artificial suelen ser muy complejos y demandan mucho tiempo. Además, en un futuro proyecto se intentaría exportar una red neuronal creada sin el uso de programas de por medio como EdgeImpulse, ya que este hace la mayoría de los pasos y el aprendizaje obtenido suele ser menor. Finalmente, se recomienda un estudio de los componentes a utilizar para asegurarse que puedan soportar la red neuronal planteada, así como definir mejor los alcances del proyecto para determinar si se requieren más componentes.

7. Referencias

- [1] Arduino. Arduino nano 33 ble sense. Datasheet SKU: ABX00031, Arduino, feb 2022.
- [2] OmniVision. Ov7675/ov7175 datasheet. Datasheet version 2.0, OmniVision, mar 2009.
- [3] Matthew Stewart. Tiny machine learning: The next ai revolution. Disponible en: <https://towardsdatascience.com/tiny-machine-learning-the-next-ai-revolution-495c26463868>, 2016.
- [4] OpenCV. Cascade classifier. Disponible en: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifizr.html, 2022.
- [5] Nordic Semiconductor. nrf52840 product specification. Datasheet v1.1, Nordic Semiconductor, feb 2019.
- [6] TinyML Foundation. About tinyml foundation. Disponible en: <https://www.tinyml.org/about/>, 2022.
- [7] Gaudenz Boesch. Tensorflow lite – real-time computer vision on edge devices. Disponible en: <https://viso.ai/edge-ai/tensorflow-lite/>, 2022.
- [8] Zakaria Jaadi. A step-by-step explanation of principal component analysis (pca). Disponible en: <https://builtin.com/data-science/step-step-explanation-principal-component-analysis>, 2022.

8. Anexos



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	5
1.1 Ratings	5
1.1.1 Recommended Operating Conditions	5
1.2 Power Consumption	5
2 Functional Overview	5
2.1 Board Topology	5
2.2 Processor	6
2.3 Crypto	6
2.4 IMU	7
2.5 Barometer and Temperature Sensor	7
2.6 Relative Humidity and Temperature Sensor	7
2.7 Digital Proximity, Ambient Light, RGB and Gesture Sensor	7
2.7.1 Gesture Detection	7
2.7.2 Proximity Detection	7
2.7.3 Color and ALS Detection	8
2.8 Digital Microphone	8
2.9 Power Tree	8
3 Board Operation	9
3.1 Getting Started - IDE	9
3.2 Getting Started - Arduino Web Editor	9
3.3 Getting Started - Arduino IoT Cloud	9
3.4 Sample Sketches	9
3.5 Online Resources	9
3.6 Board Recovery	9
4 Connector Pinouts	9
4.1 USB	10
4.2 Headers	10
4.3 Debug	11
5 Mechanical Information	11
5.1 Board Outline and Mounting Holes	11
6 Certifications	12
6.1 Declaration of Conformity CE DoC (EU)	12
6.2 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021	12
6.3 Conflict Minerals Declaration	13
7 FCC Caution	13
8 Company Information	14
9 Reference Documentation	14
10 Revision History	14



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)

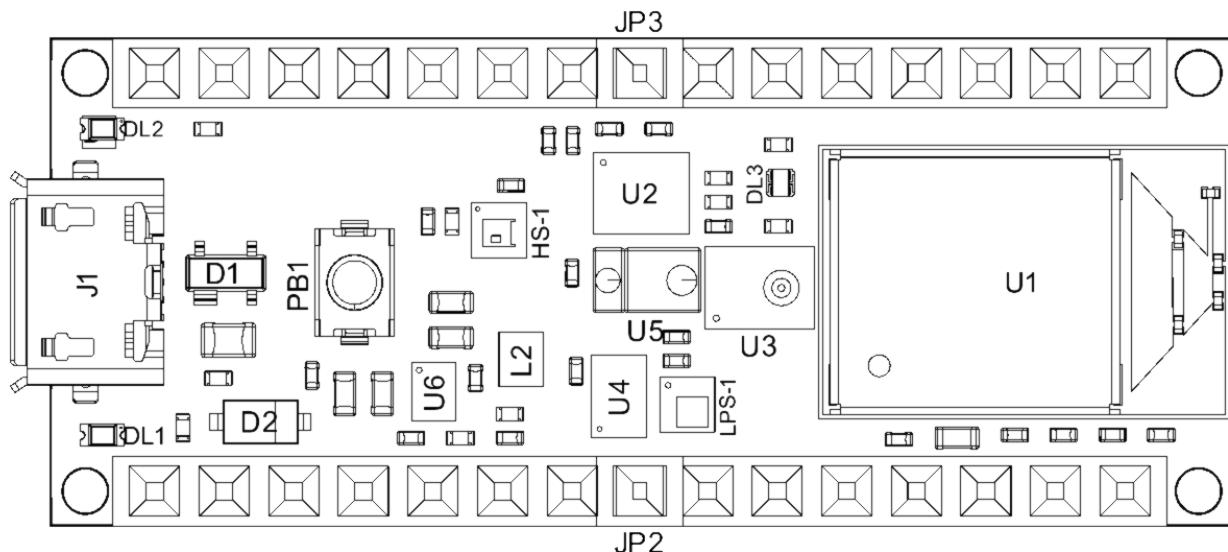
1.2 Power Consumption

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

nRF52840

Product Specification

v1.1



Feature list

Features:	
<ul style="list-style-type: none">• Bluetooth® 5, IEEE 802.15.4-2006, 2.4 GHz transceiver<ul style="list-style-type: none">• -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:<ul style="list-style-type: none">• 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<ul style="list-style-type: none">• 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<ul style="list-style-type: none">• ARM® TrustZone® Cryptocell 310 security subsystem<ul style="list-style-type: none">• 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<ul style="list-style-type: none">• Flash access control list (ACL)• Root-of-trust (RoT)• Debug control and configuration• Access port protection (CTRL-AP)• Secure erase	<ul style="list-style-type: none">• Flexible power management<ul style="list-style-type: none">• 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<ul style="list-style-type: none">• 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<ul style="list-style-type: none">• 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<ul style="list-style-type: none">• 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

4 Core components

4.1 CPU

The ARM® Cortex-M4 processor with floating-point unit (FPU) has a 32-bit instruction set (Thumb®-2 technology) that implements a superset of 16- and 32-bit instructions to maximize code density and performance.

This processor implements several features that enable energy-efficient arithmetic and high-performance signal processing, including:

- Digital signal processing (DSP) instructions
- Single-cycle multiply and accumulate (MAC) instructions
- Hardware divide
- 8- and 16-bit single instruction multiple data (SIMD) instructions
- Single-precision floating-point unit (FPU)

The ARM® Cortex® Microcontroller Software Interface Standard (CMSIS) hardware abstraction layer for the ARM® Cortex® processor series is implemented and available for the M4 CPU.

Real-time execution is highly deterministic in thread mode, to and from sleep modes, and when handling events at configurable priority levels via the nested vectored interrupt controller (NVIC).

Executing code from flash will have a wait state penalty on the nRF52 series. An instruction cache can be enabled to minimize flash wait states when fetching instructions. For more information on cache, see [Cache](#) on page 26. The section [Electrical specification](#) on page 20 shows CPU performance parameters including wait states in different modes, CPU current and efficiency, and processing power and efficiency based on the CoreMark® benchmark.

The ARM system timer (SysTick) is present on nRF52840. The SysTick's clock will only tick when the CPU is running or when the system is in debug interface mode.

4.1.1 Floating point interrupt

The floating point unit (FPU) may generate exceptions when used due to e.g. overflow or underflow, which in turn will trigger the FPU interrupt.

See [Instantiation](#) on page 23 for more information about the exceptions triggering the FPU interrupt.

To clear the IRQ (interrupt request) line when an exception has occurred, the relevant exception bit within the floating-point status and control register (FPSCR) needs to be cleared. For more information about the FPSCR or other FPU registers, see [Cortex-M4 Devices Generic User Guide](#).

4.1.2 CPU and support module configuration

The ARM® Cortex®-M4 processor has a number of CPU options and support modules implemented on the device.

Option / Module	Description	Implemented
Core options		
NVIC	Nested vector interrupt controller	48 vectors
PRIORITIES	Priority bits	3
WIC	Wakeup interrupt controller	NO
Endianness	Memory system endianness	Little endian
Bit-banding	Bit banded memory	NO
DWT	Data watchpoint and trace	YES
SysTick	System tick timer	YES
Modules		
MPU	Memory protection unit	YES
FPU	Floating-point unit	YES
DAP	Debug access port	YES
ETM	Embedded trace macrocell	YES
ITM	Instrumentation trace macrocell	YES
TPIU	Trace port interface unit	YES
ETB	Embedded trace buffer	NO
FPB	Flash patch and breakpoint unit	YES
HTM	AMBA™ AHB trace macrocell	NO

4.1.3 Electrical specification

4.1.3.1 CPU performance

The CPU clock speed is 64 MHz. Current and efficiency data is taken when in System ON and the CPU is executing the CoreMark™ benchmark. It includes power regulator and clock base currents. All other blocks are IDLE.

Symbol	Description	Min.	Typ.	Max.	Units
W _{FLASH}	CPU wait states, running CoreMark from flash, cache disabled			2	
W _{FLASHCACHE}	CPU wait states, running CoreMark from flash, cache enabled			3	
W _{RAM}	CPU wait states, running CoreMark from RAM			0	
CM _{FLASH}	CoreMark, running CoreMark from flash, cache enabled		212		CoreMark
CM _{FLASH/MHz}	CoreMark per MHz, running CoreMark from flash, cache enabled		3.3		CoreMark/ MHz
CM _{FLASH/mA}	CoreMark per mA, running CoreMark from flash, cache enabled, DCDC 3V		64		CoreMark mA

4.2 Memory

The nRF52840 contains 1 MB of flash and 256 kB of RAM that can be used for code and data storage.

The CPU and peripherals with EasyDMA can access memory via the AHB multilayer interconnect.

The CPU is also able to access peripherals via the AHB multilayer interconnect, as illustrated in [Memory layout](#) on page 21.

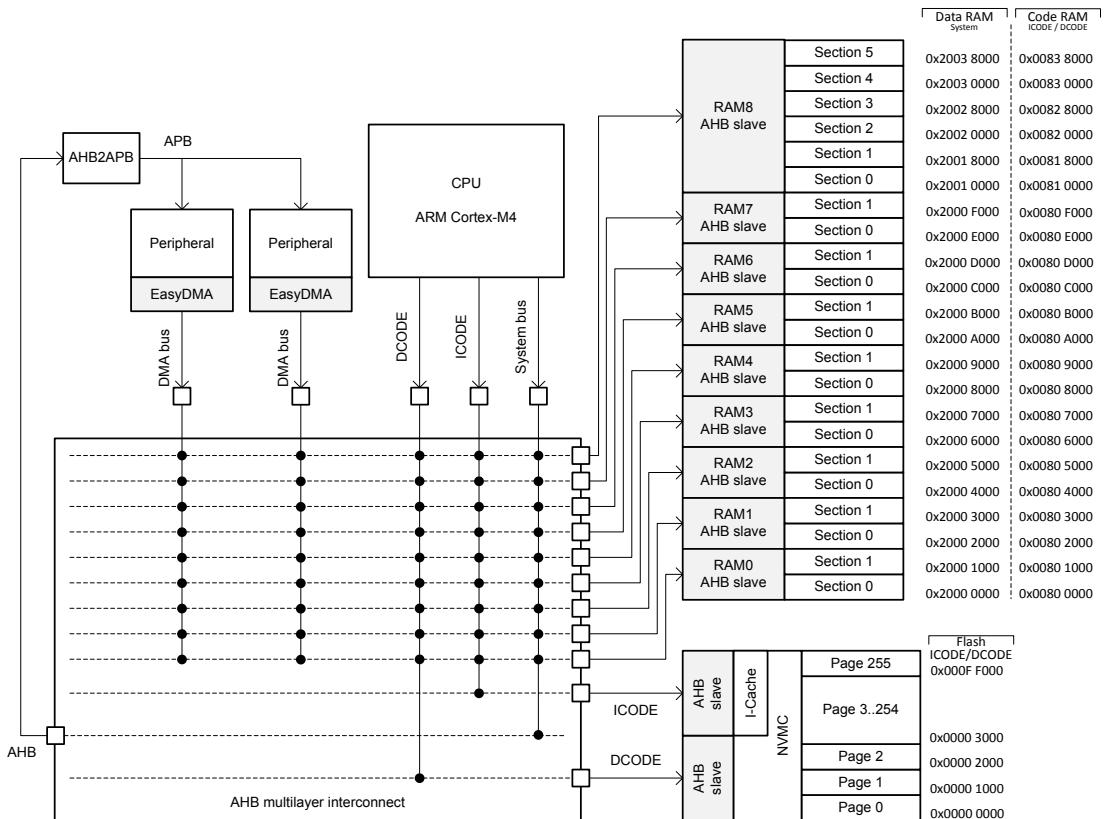


Figure 2: Memory layout

See [AHB multilayer](#) on page 49 and [EasyDMA](#) on page 46 for more information about the AHB multilayer interconnect and the EasyDMA.

The same physical RAM is mapped to both the Data RAM region and the Code RAM region. It is up to the application to partition the RAM within these regions so that one does not corrupt the other.

4.2.1 RAM - Random access memory

The RAM interface is divided into 9 RAM AHB slaves.

RAM AHB slave 0-7 is connected to 2x4 kB RAM sections each and RAM AHB slave 8 is connected to 6x32 kB sections, as shown in [Memory layout](#) on page 21.

Each of the RAM sections have separate power control for System ON and System OFF mode operation, which is configured via RAM register (see the [POWER — Power supply](#) on page 61).

4.2.2 Flash - Non-volatile memory

The flash can be read an unlimited number of times by the CPU, but it has restrictions on the number of times it can be written and erased and also on how it can be written.

Writing to flash is managed by the non-volatile memory controller (NVMC), see [NVMC — Non-volatile memory controller](#) on page 24.

The flash is divided into 256 pages of 4 kB each that can be accessed by the CPU via both the ICODE and DCODE buses as shown in [Memory layout](#) on page 21.

4.2.3 Memory map

The complete memory map for the nRF52840 is shown in [Memory map](#) on page 22. As described in [Memory](#) on page 20, Code RAM and Data RAM are the same physical RAM.

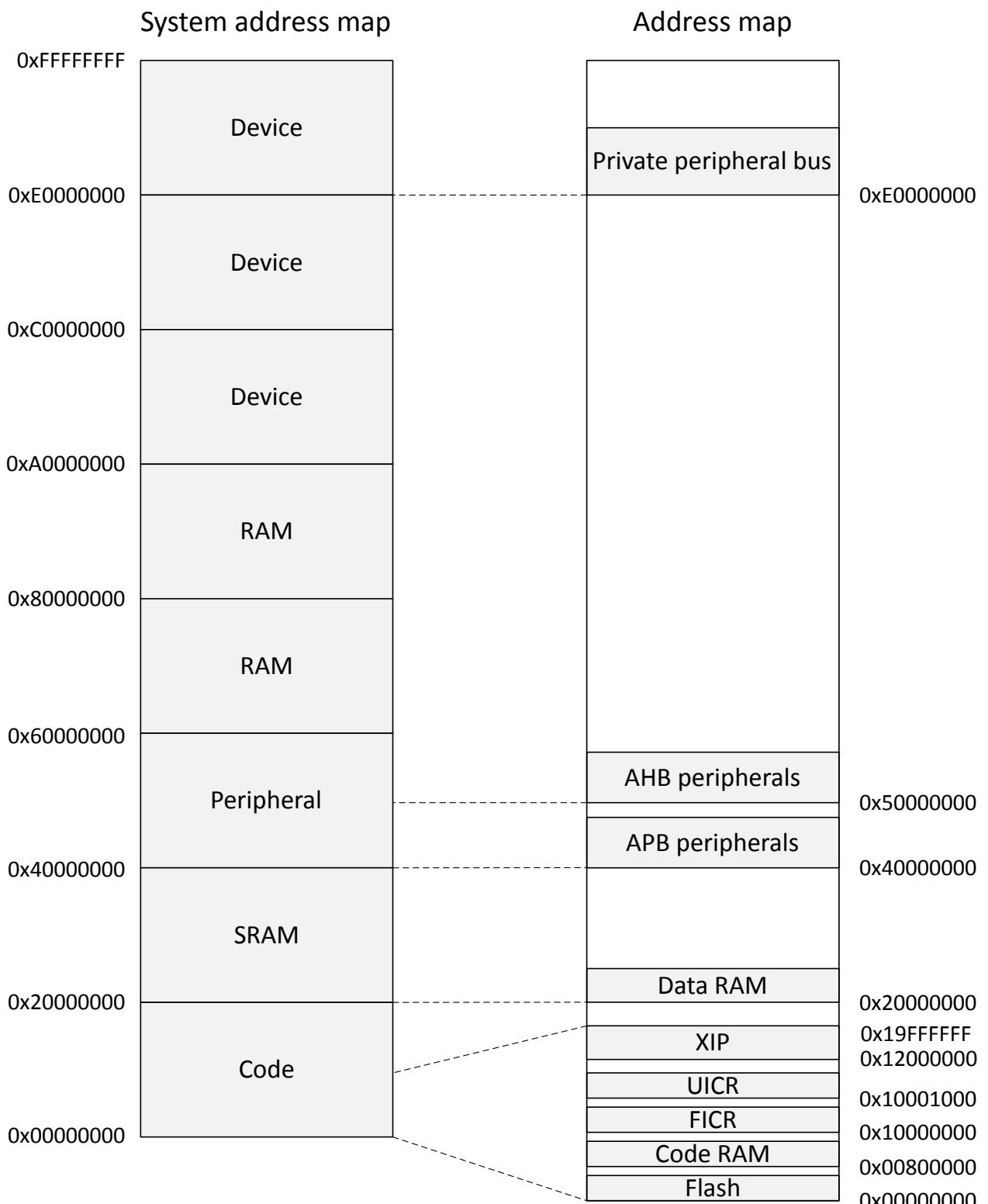


Figure 3: Memory map

4.2.4 Instantiation

ID	Base address	Peripheral	Instance	Description	
0	0x40000000	CLOCK	CLOCK	Clock control	
0	0x40000000	POWER	POWER	Power control	
0	0x50000000	GPIO	GPIO	General purpose input and output	Deprecated
0	0x50000000	GPIO	P0	General purpose input and output, port 0	
0	0x50000300	GPIO	P1	General purpose input and output, port 1	
1	0x40001000	RADIO	RADIO	2.4 GHz radio	
2	0x40002000	UART	UART0	Universal asynchronous receiver/transmitter	Deprecated
2	0x40002000	UARTE	UARTE0	Universal asynchronous receiver/transmitter with EasyDMA, unit 0	
3	0x40003000	SPI	SPI0	SPI master 0	Deprecated
3	0x40003000	SPIM	SPIMO	SPI master 0	
3	0x40003000	SPIS	SPISO	SPI slave 0	
3	0x40003000	TWI	TWIO	Two-wire interface master 0	Deprecated
3	0x40003000	TWIM	TWIMO	Two-wire interface master 0	
3	0x40003000	TWIS	TWISO	Two-wire interface slave 0	
4	0x40004000	SPI	SPI1	SPI master 1	Deprecated
4	0x40004000	SPIM	SPIM1	SPI master 1	
4	0x40004000	SPIS	SPIS1	SPI slave 1	
4	0x40004000	TWI	TWI1	Two-wire interface master 1	Deprecated
4	0x40004000	TWIM	TWIM1	Two-wire interface master 1	
4	0x40004000	TWIS	TWIS1	Two-wire interface slave 1	
5	0x40005000	NFCT	NFCT	Near field communication tag	
6	0x40006000	GPIOTE	GPIOTE	GPIO tasks and events	
7	0x40007000	SAADC	SAADC	Analog to digital converter	
8	0x40008000	TIMER	TIMER0	Timer 0	
9	0x40009000	TIMER	TIMER1	Timer 1	
10	0x4000A000	TIMER	TIMER2	Timer 2	
11	0x4000B000	RTC	RTC0	Real-time counter 0	
12	0x4000C000	TEMP	TEMP	Temperature sensor	
13	0x4000D000	RNG	RNG	Random number generator	
14	0x4000E000	ECB	ECB	AES electronic code book (ECB) mode block encryption	
15	0x4000F000	AAR	AAR	Accelerated address resolver	
15	0x4000F000	CCM	CCM	AES counter with CBC-MAC (CCM) mode block encryption	
16	0x40010000	WDT	WDT	Watchdog timer	
17	0x40011000	RTC	RTC1	Real-time counter 1	
18	0x40012000	QDEC	QDEC	Quadrature decoder	
19	0x40013000	COMP	COMP	General purpose comparator	
19	0x40013000	LPCOMP	LPCOMP	Low power comparator	
20	0x40014000	EGU	EGU0	Event generator unit 0	
20	0x40014000	SWI	SWI0	Software interrupt 0	
21	0x40015000	EGU	EGU1	Event generator unit 1	
21	0x40015000	SWI	SWI1	Software interrupt 1	
22	0x40016000	EGU	EGU2	Event generator unit 2	
22	0x40016000	SWI	SWI2	Software interrupt 2	
23	0x40017000	EGU	EGU3	Event generator unit 3	
23	0x40017000	SWI	SWI3	Software interrupt 3	
24	0x40018000	EGU	EGU4	Event generator unit 4	
24	0x40018000	SWI	SWI4	Software interrupt 4	
25	0x40019000	EGU	EGU5	Event generator unit 5	
25	0x40019000	SWI	SWI5	Software interrupt 5	

ID	Base address	Peripheral	Instance	Description	
26	0x4001A000	TIMER	TIMER3	Timer 3	
27	0x4001B000	TIMER	TIMER4	Timer 4	
28	0x4001C000	PWM	PWM0	Pulse width modulation unit 0	
29	0x4001D000	PDM	PDM	Pulse Density modulation (digital microphone) interface	
30	0x4001E000	ACL	ACL	Access control lists	
30	0x4001E000	NVMC	NVMC	Non-volatile memory controller	
31	0x4001F000	PPI	PPI	Programmable peripheral interconnect	
32	0x40020000	MWU	MWU	Memory watch unit	
33	0x40021000	PWM	PWM1	Pulse width modulation unit 1	
34	0x40022000	PWM	PWM2	Pulse width modulation unit 2	
35	0x40023000	SPI	SPI2	SPI master 2	Deprecated
35	0x40023000	SPIM	SPIM2	SPI master 2	
35	0x40023000	SPIS	SPIS2	SPI slave 2	
36	0x40024000	RTC	RTC2	Real-time counter 2	
37	0x40025000	I2S	I2S	Inter-IC sound interface	
38	0x40026000	FPU	FPU	FPU interrupt	
39	0x40027000	USBD	USBD	Universal serial bus device	
40	0x40028000	UARTE	UARTE1	Universal asynchronous receiver/transmitter with EasyDMA, unit 1	
41	0x40029000	QSPI	QSPI	External memory interface	
42	0x5002A000	CC_HOST_RGF	CC_HOST_RGF	Host platform interface	
42	0x5002A000	CRYPTOCELL	CRYPTOCELL	CryptoCell subsystem control interface	
45	0x4002D000	PWM	PWM3	Pulse width modulation unit 3	
47	0x4002F000	SPIM	SPIM3	SPI master 3	
N/A	0x10000000	FICR	FICR	Factory information configuration	
N/A	0x10001000	UICR	UICR	User information configuration	

Table 3: Instantiation table

4.3 NVMC — Non-volatile memory controller

The non-volatile memory controller (NVMC) is used for writing and erasing of the internal flash memory and the UICR (user information configuration registers).

The [CONFIG](#) on page 27 is used to enable the NVMC for writing (CONFIG.WEN = Wen) and erasing (CONFIG.WEN = Een). The user must make sure that writing and erasing are not enabled at the same time. Having both enabled at the same time may result in unpredictable behavior.

The CPU must be halted before initiating a NVMC operation from the debug system.

4.3.1 Writing to flash

When write is enabled, full 32-bit words can be written to word-aligned addresses in the flash.

As illustrated in [Memory](#) on page 20, the flash is divided into multiple pages. The same 32-bit word in the flash can only be written n_{WRITE} number of times before a page erase must be performed.

The NVMC is only able to write 0 to bits in the flash that are erased (set to 1). It cannot rewrite a bit back to 1. Only full 32-bit words can be written to flash using the NVMC interface. To write less than 32 bits, write the data as a full 32-bit word and set all the bits that should remain unchanged in the word to 1. Note that the restriction on the number of writes (n_{WRITE}) still applies in this case.

Only word-aligned writes are allowed. Byte or half-word-aligned writes will result in a hard fault.

The time it takes to write a word to flash is specified by t_{WRITE} . The CPU is halted if the CPU executes code from the flash while the NVMC is writing to the flash.

7 Hardware and layout

7.1 Pin assignments

This section describes the pin assignment and the pin functions.

This device provides flexibility when it comes to routing and configuration of the GPIO pins. However, some pins have recommendations for how the pin should be configured or what it should be used for. See [aQFN73 ball assignments](#) on page 576 and [WLCSP ball assignments](#) on page 579 for more information about this.

7.1.1 aQFN73 ball assignments

The ball assignment table and figure describe the assignments for this variant of the chip.

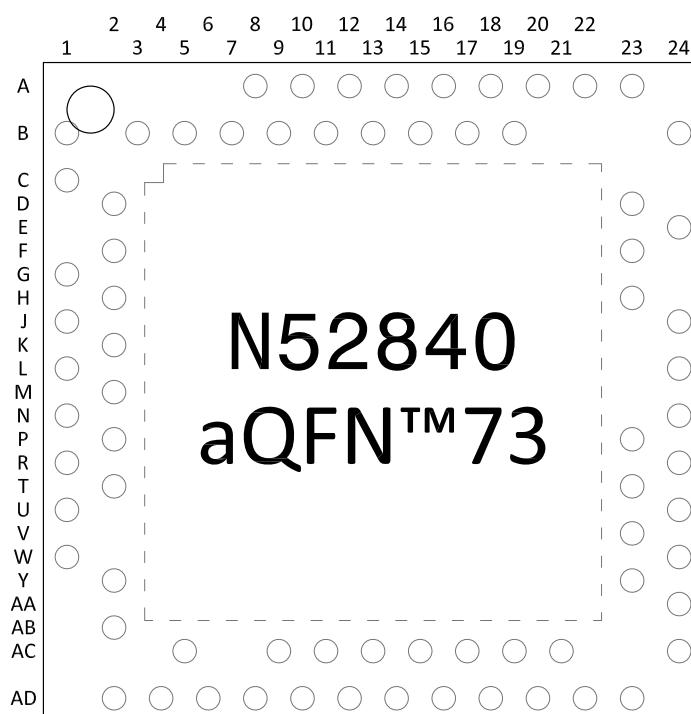


Figure 206: aQFN™ 73 ball assignments, top view

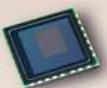


OV7675/OV7175

datasheet

PRODUCT SPECIFICATION

1/9" CMOS VGA (640x480) image sensor
with OmniPixel3-HS™ technology



applications

- cellular phones
- toys
- PC multimedia
- digital still cameras

ordering information

- **OV07675-A23A** (color, lead-free)
23-pin CSP3
- **OV07175-A23A** (b&w, lead-free)
23-pin CSP3

features

- support for image sizes: VGA (640x480), QVGA (320x240) and QQVGA (160x120)
- support for output formats: YUV4:2:2, Raw RGB, ITU656, RGB565
- digital video port (DVP) parallel output interface
- on-chip phase lock loop (PLL)
- built-in 1.5V regulator for core
- capable of maintaining register values at power down
- programmable controls for frame rate, mirror and flip, AEC/AGC, and windowing

- support for horizontal and vertical sub-sampling
- automatic image control functions: automatic exposure control (AEC), automatic white balance (AWB) and automatic black level calibration (ABLC)
- image quality controls: defect pixel correction and lens shading correction
- support for black sun cancellation
- standard serial SCCB interface
- parallel I/O tri-state configurability and programmable polarity
- module size: 6 mm x 6 mm

key specifications

- **active array size:** 640x480
- **power supply:**
analog: 2.6 ~ 3.0V
core: 1.5V DC \pm 5% (internal regulator)
I/O: 1.71 ~ 3.0V
- **power requirements:**
active: 98 mW
standby: 60 μ W
- **temperature range:**
operating: -30°C to 70°C (see [table 8-2](#))
stable image: 0°C to 50°C (see [table 8-2](#))
- **output formats:** YUV422, Raw RGB, ITU656, RGB565
- **lens size:** 1/9"
- **lens chief ray angle:** 21° (see [figure 10-2](#))
- **input clock frequency:** 1.5 ~ 27 MHz
(see [table 8-5](#))
- **scan mode:** progressive
- **maximum image transfer rate:** (see [table 2-1](#) for details)
- **sensitivity:** 1800 mV/(Lux-sec)
- **shutter:** rolling shutter
- **S/N ratio:** 38 dB
- **dynamic range:** 71 dB
- **maximum exposure interval:** 510 x t_{ROW}
- **pixel size:** 2.5 μ m x 2.5 μ m
- **dark current:** 10 mV/sec @ 60°C
- **well capacity:** 12 Ke $^-$
- **fixed pattern noise (FPN):** 1% of V_{PEAK-TO-PEAK}
- **image area:** 1640 μ m x 1220 μ m
- **package dimensions:** 2815 x 2825 μ m