UNIVERSIDAD DE COSTA RICA

Escuela de Ingeniería Eléctrica

IE-0624: Laboratorio de Microcontroladores

Laboratorio 5

STM32/Arduino: GPIO, Giroscopio, comunicaciones, TinyML Machine Learning

Prof. Marco Villalta

Estudiantes:
Sofía Fonseca Muñoz, B42634

28 de febrero de 2023

Índice

1. Introducción/Resumen

Este proyecto consiste en el aprendizaje y utilización de Machine Learning para microcontroladores. Para ello se utilizó el microcontrolador STM32F429i Discovery kit y algunas herramientas digitales como TensorFlow Lite, Google Colab, Python.

Los datos para alimentar la red neuronal fueron tomados por medio del microcontrolador, ya que es ahí donde se quiere ver el sistema funcionando. Se grabaron 3 movimientos por 15 segundos cada uno: brazo arriba abajo, extensión de codo y círculos con el brazo.

La red neronal fue entrenada através de Google Colab, se utilizó 60 % de los datos para entrenamiento, 20 % para comprobación y 20 % para validación. Una vez hecha se exportó para utilizarla como una biblioteca en un programa para el microcontrolador STM.

El repositorio correspondiente a este laboratorio se puede encontrar en el enlace https://github.com/sofifon/IE0624-2023/tree/main/Lab5_AI.

2. Nota teórica

2.1. Microcontrolador

Para este laboratorio se utilizó la placa STM32F429i Disovery kit. Esta placa permite el desarrollo de aplicaciones fracias a el microprocesador de alto rendimiento STM32F4 que incluye la placa. Es una placa que ha sido desarrollada por medio de software libre por lo que contiene una extensa cantidad de librerias y ejemplos para el desarrollo de proyectos. [1]

La placa cuenta con un procesador STM32F429ZIT6 con una memoria flash de 2Mb y 256 Kb de RAM, una pantalla LCD táctil de 2.4 pulgadas, conexión por mini USB, un giroscopio, 6 LEDs, dos botones, funciones de datos USB, poder por medio de batería ecterna. [1] En la siguiente imagen se puede observar el diagrama de la placa y los componentes que posee en su lado superior:

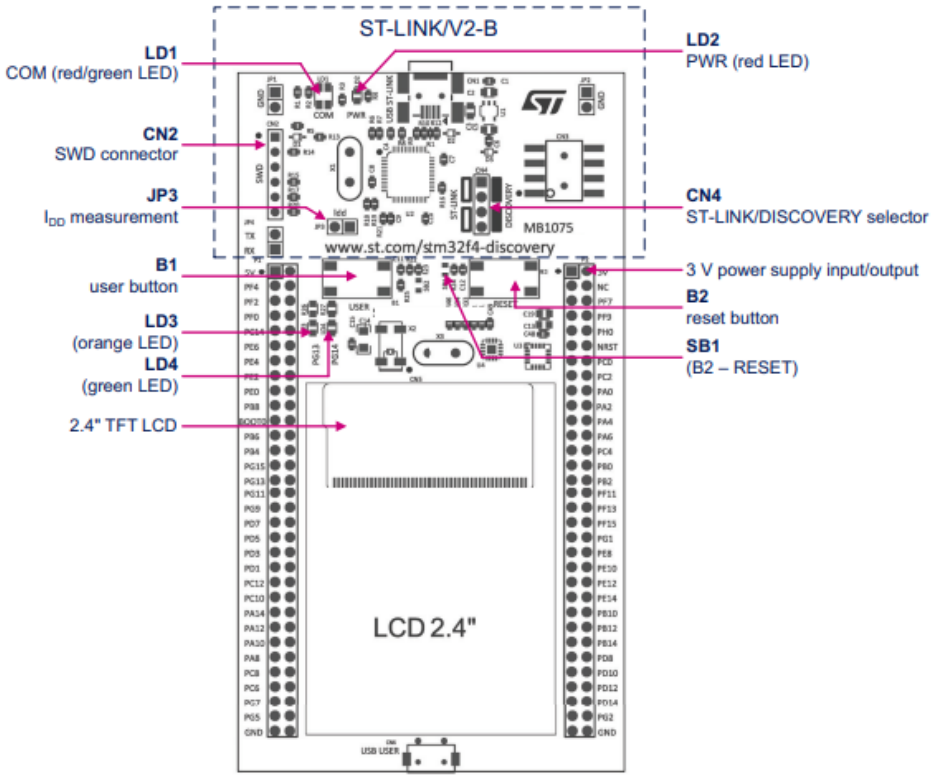


Figura 1: Diagrama frontal de la placa STM32F429 Discovery Kit. Tomada de https://www.st.com/en/evaluation-tools/32f429idiscovery.html

En la siguiente imagen se muestra el lado trasero de la placa donde incluso de puede observar el microcontrolador:

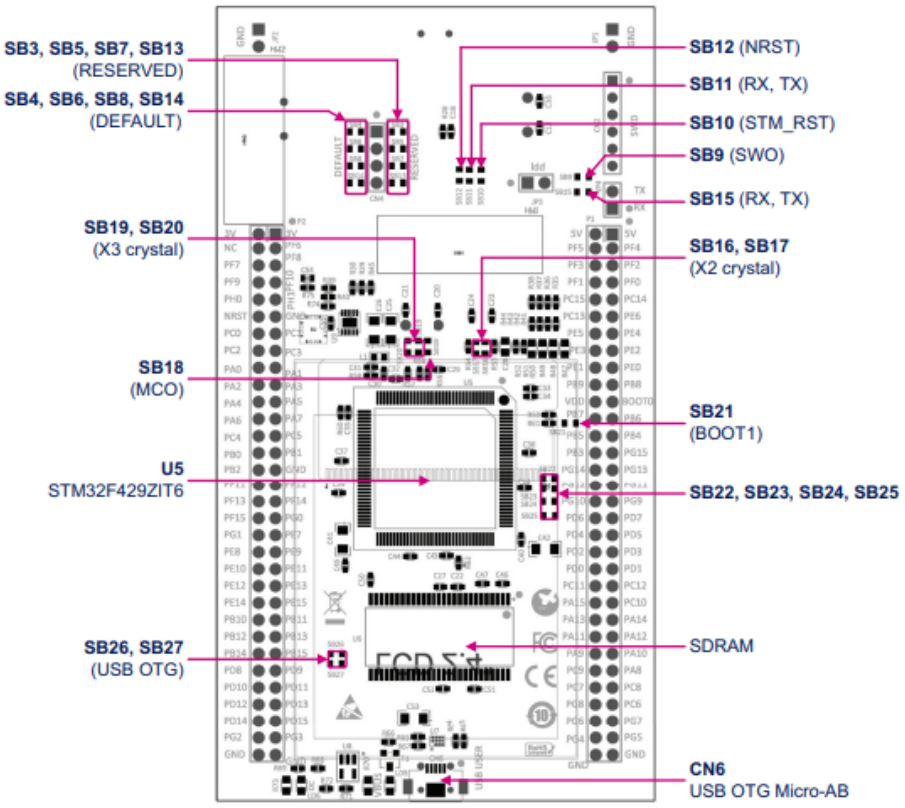


Figura 2: Diagrama trasero de la placa STM32F429 Discovery Kit. Tomada de https://www.st.com/en/evaluation-tools/32f429idiscovery.html

Es importante destacar que muchas de las funcionalidades que logra este kit se deben a la placa externa que contiene el microcontrolador. El microcontrolador en sí tiene buen rendimiento pero lo que se utiliza por ejemplo en este laboratorio se logra gracias a la placa. En el siguiente diagrama de bloques se muestra como se realizaron las conexiones de los componentes.

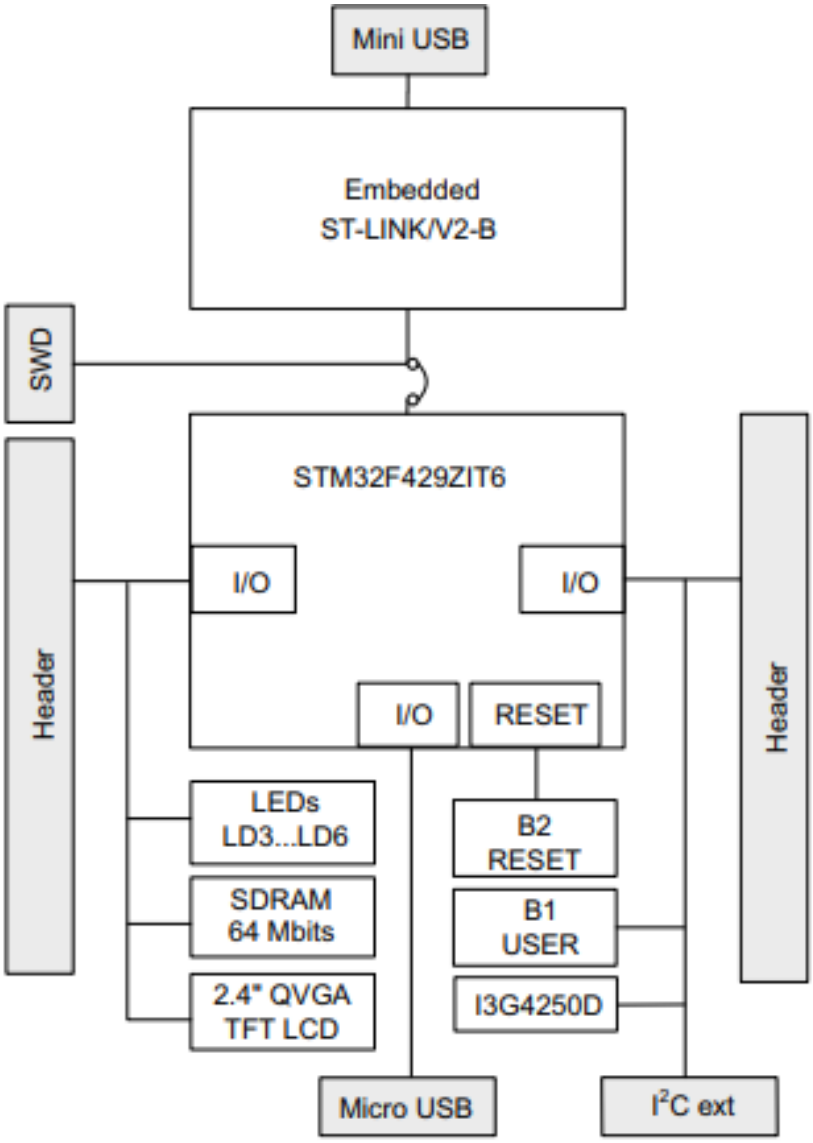


Figura 3: Diagrama de bloques de la placa. Tomada de https://www.st.com/en/evaluation-tools/32f429idiscovery.html

El giroscopio que contiene esta placa es un I3G4250D de bajo poder, el sensor detecta en tres ejes por medio de su elemento. Este dispositivo además contiene una interfaz IC que permite enviar los datos por medio de un protocolo I2C o SPI. Este kit lo controla por medio de la interfaz SPI. [1]

2.1.1. Características eléctricas

Debido a que esta es una placa que contiene todos los coponentes, no se utilizaron las características eléctricas más allá del uso de una fuente de poder externo. Esta debe ser de 5V y existen dos diodos que protegen la placa de otros valores. Los pines de 5V y 3V entregan 5V y 3V respectivamente y el consumo de poder debe ser menor a 100 mA. [1]

2.2. Bibliotecas

A nivel de software, se utilizó la biblioteca libopencm3, la cual es una biblioteca de software libre hecha para la familia de microprocesadores STM. La biblioteca posee sub bibliotecas que facilitan el uso de cada uno de los elmentos de la tarjeta. Además libopencm3 posee una serie de ejemplos que tienen funcionalidades que se necesitaban para este proyecto, por lo que se tomaron algunas de estas como bibliotecas para el desarrollo del proyecto. Acontinuación un alista de las bibliotecas:

- **math.h**: para realizar operaciones matemáticas relacionadas a los ángulos del girospcopio.
- **rcc.h**: para configurar el reloj interno del microcontrolador.
- **gpio.h**: para controlar los elementos externos como los leds y botones.
- **math.h**: para realizar operaciones matemáticas relacionadas a los ángulos del girospcopio.
- **spi.h**: para la conexión del girospcopio.
- **usart.h**: para el envío de la información a la computadora por medio del protocolo USART.

Una vez que el modelo fue creado, se utilizaron bibliotecas de TensorFlow Lite, las cuales están diseñadas para el uso de el modelo de la red neuronal con un microcontrolador, en este caso con el Discovery kit. Además el modelo creado es utilizado dentro de programa final que corre el modelo en el microcontrolador.

2.3. Diseño

Debido a que para este proyecto se está utilizando solamente los compomentes que incluye la placa, no se realizó un diseño físico.

2.4. Componentes electrónicos

Para la creación de este laboratorio se utilizó solamente la placa STM32F429 Discovery kit. Esta placa tiene un valor aproximado de 20385 colones, tomando en cuenta un tipo de cambio de 565 colones equivalentes a 1 USD.

3. Desarrollo/Análisis de resultados

Para el desarrollo de este laboratorio existe una serie de requerimientos para conseguir el objetivo. A continuación una lista de cada uno de ellos:

1. Realizar un programa para el microcontrolador que capture la informacion del giroscopio y se envie a la computadora por el puerto USB.

2. Realizar un script de Python que guarde la informacion recibida del giroscopio y que se encuentre etiquetada con el tipo de movimiento que se efectua, el script debe guardar esta informacion en un archivo CSV.

3. Registrar 3 movimientos (por ejemplo: mover brazo hacia arriba o hacia abajo, mover brazo en circulos, golpe de brazo, estacionario,etc) por un periodo de tiempo (p.e. 5-15 segundos). Eso mientras se ejecuta el programa de registro de giroscopio en conjunto con el script de comunicaciones a la PC.

4. Realizar un script de Python para crear un modelo de TensorFlow lite que cargue la informacion de los movimientos guardados en el archivo CSV del paso anterior, configure una red neuronal (p.e. una red 2 capas (una de entrada y una oculta/intermedia) con varias neuronas cada una y 3 salidas (una por cada tipo de movimiento realizado), usando ReLU como funcion de activación (excepto para la de salida que se recomienda softmax), algoritmo de optimizacion rmsprop y metrica de perdida mae).

5. Entrenar la red con el 60 % de los datos, se utilizará el 20 % de los datos para validar y el resto 20 % para pruebas.

6. Exportar el modelo obtenido con TensorFlow Lite.

7. Realizar un programa que utilice el modelo construido en el paso anterior para detectar el tipo de movimiento realizado, los movimientos detectados deben de quedar registrados en la PC por lo que se debe realizar la comunicación entre el microcontrolador y la PC.

A continuación se va a mostrar tanto la forma en que se implementó así como el resultado obtenido de cada uno de los requirimientos.

3.1. Lectura y almacenamiento de los 3 movimientos

Esta parte consiste en la creación de un programa para el microcontrolador y un script para leer los datos desde la computadora. El programa del microcontrolador, fue basado en el sismógrafo del laboratorio 4. Se modificó de manera que la información de la batería no se compartiera por puerto serial y que los datos no se mostraran en la pantalla del microprocesador. Las instrucciones del gráfico toman mucho tiempo, por lo que se eliminó para que la frecuencia con que se envían los datos por USART fuera más alta.

Por otro lado el script de python para crear los datos fue basado en el hecho para el laboratorio 3, de manera que solo recibiera tres columnas de información y las acomodara. El archivo corresponde a https://github.com/sofifon/IE0624-2023/blob/main/Lab5_AI/src/recorder.py.

Una vez que se tenía los dos pasos anteriores trabajando en conjunto, se hizo una prueba de los datos. Se midió el tiempo de cada movimiento para tener 7 repeticiones en 15 segundo con aproximadamente 1000 muestras. Una vez que se comprobó que se podía realizar de esta manera, se grabaron

los datos con un cronómetro. Los datos fueron levemente modificados para tener exactamente 1000 muestras ya que esto afecta la creación del modelo de TensorFlow.

El resultado de la grabación de los movimientos se encuentra en https://github.com/sofifon/IE0624-2023/tree/main/Lab5_AI/src/data.

3.2. Creación y entrenamiento del modelo

Para realizar el modelo a partir de la red neuronal, se utilizó la herramienta Google Colab, ya que permite acceder a un servidor con más poder de procesamiento para crear y entrenar la red neuronal que lo que la computadora local puede soportar. El proceso que se realizó, fue una adaptación del siguiente tutorial que está enfocado en Tensorflow Lite para microcontroladores https://github.com/arduino/ArduinoTensorFlowLiteTutorials/blob/master/GestureToEmoji/arduino_tinyml_workshop.ipynb

Este se modificó para procesar solamente los 3 datos x, y y z que lee el giroscopio. El primer paso fue revisar los datos capturados en el paso anterior, para asegurarse que existan siete repeticiones del movimiento en cada uno de los datos dados. La manera más sencilla de realizar la observación es por medio de la creación de una gráfica del movimiento a partir de los datos de cada eje. Cada pico corresponde a un movimiento completo. A continuación un ejemplo de la gráfica para la revisión:

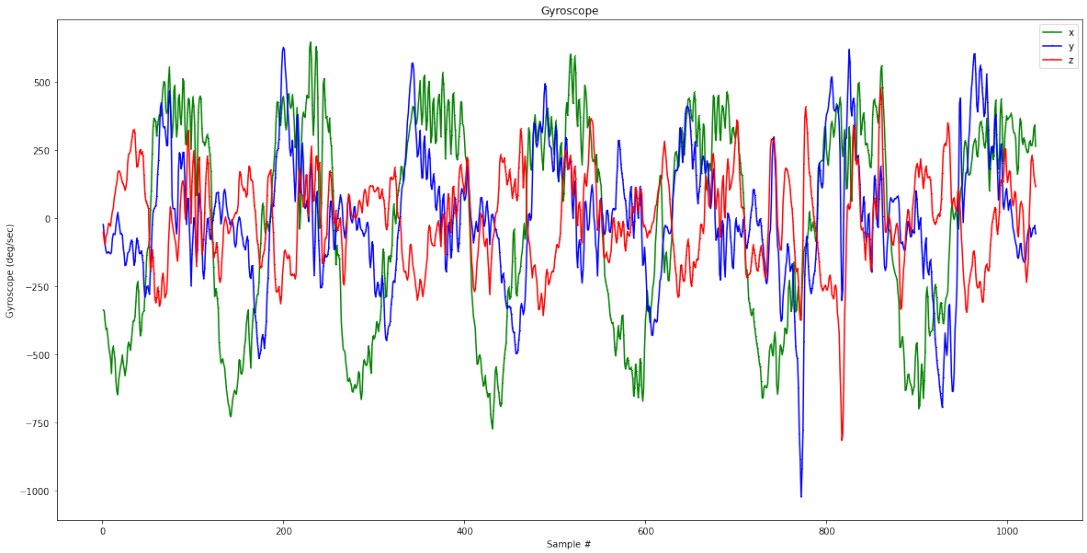


Figura 4: Gráfica del movimiento de brazo de arriba a abajo.

Una vez realizada la observación, se procede a preparar los datos. Se crea una semilla, se nombran los datos por movimiento y se hace una lista del resultado esperado, en este caso detectar cuál movimiento corresponde al set de datos que se está observando.

Cuando los datos están preparados, se mezclan para que haya una distribución aleatoria de datos. Estos son divididos en grupos, un 60 % de los datos se utilizan para entrenar el modelo, un 20 % para probarlo y un 20 % para validarlo.

Se procede entonces a la creación del modelo utilizando keras. Se utiliza relu como función activadora, rmsprop para optimizar y mae para medir las pérdidas. La siguiente imagen muestra la distribución del entrenamiento y la validación.

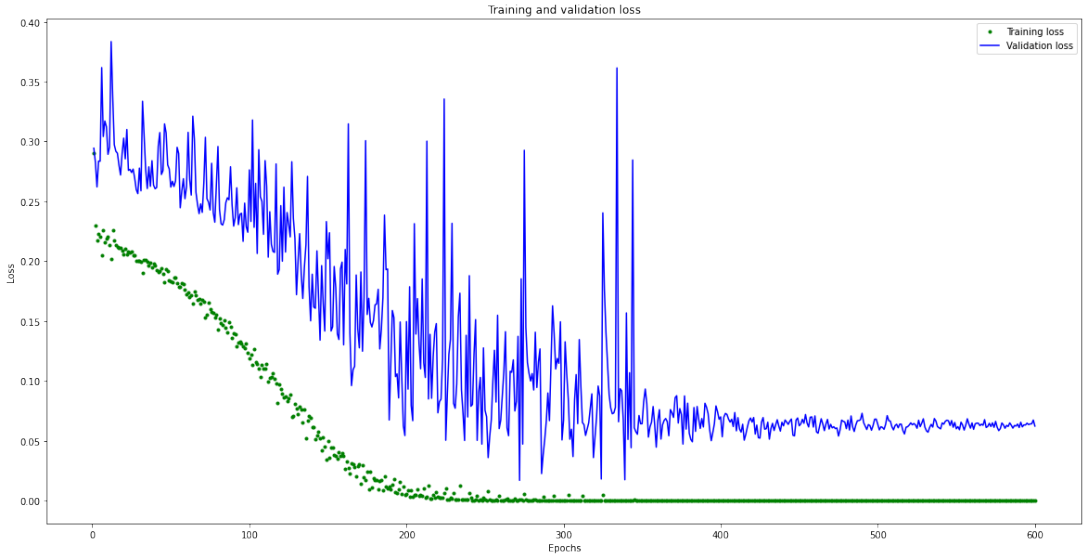


Figura 5: gráfica de pérdida de entrenamiento y validación de la red neuronal. Imagen propia.

Con el modelo listo, se utilizan los datos para validar. La siguiente imagen muestra el resultado esperado y la predicción realizada por el modelo. La precisión del modelo es baja, posiblemente porque la cantidad de datos no son suficientes o la muestra podría estar mejor tomada, pero se requiere mayor cantidad de procesamiento y tiempo para lograrlo.

```
1/1 [==============================] - 0s 34ms/step
predictions =
 [[1.    0.    0.   ]
 [0.104 0.878 0.018]
 [1.    0.    0.   ]
 [0.998 0.002 0.   ]]
actual =
 [[0. 0. 1.]
 [0. 1. 0.]
 [0. 0. 1.]
 [1. 0. 0.]]
```

Figura 6: Comparación de datos esperados y la predicción del modelo. Imagen propia.

El último paso en este proceso es convertir el modelo. Como se utilizó Tensorflow, primero se convierte a un modelo de la versión Lite. Luego se pasa de un archivo de TensorFlow Lite a un archivo de C, para que el microcontrolador lo pueda utilizar.

3.3. Creación del programa para detectar movimientos con STM32f429i

Para este paso, fue necesario importar las bibliotecas de TensorFlowLite para microcontroladores. [3] Además de importar el modelo de la red neuronal para utilizarla como una biblioteca.

Dentro del archivo principal se incluyen la configuración de la red neuronal, para que reconozca cuál es la variable de entrada y cuál la de salida, entre otros. Cuando estos pasos se completan, se crea un buclle que introduce cada una de las lecturas del giroscopio hasta tener el tamaño de la muestra adecuado para la comparación con el modelo.

Este programa corre adecuadamente, sin embargo, fue difícil obtener lecturas correspondientes a los movimientos elegidos. Posiblemente se debe a que el inicio de la muestra no coincide con el inicio del movimiento y no lo puede reconocer. Esto se podría mejorar con una toma de datos más completa, pero como se mencionó adelante, se requiere más tiempo y poder de procesamiento para entrenar la red neuronal.

4. Conclusiones y recomendaciones

- TensorFlow Lite es una herramienta que permite tenr niveles de nivel de complejidad intermedio para procesarlos en un microcontrolador.
- Machine Learning en un microcontrolador permite brindas soluciones para pequeños problemas que tenemos día con día. De manera que se puede utlizar ML sin necesidad de tener un sistema complejo.
- Se puede entrenar una red neuronal en una computadora portátil. Esto desmiente la idea de que se necesitan sistemas complejos para tener Machine Learning.

Referencias

1. ST Life Augmented (2019) Discovery kit with STM32F429ZI MCU. Recuperado de: https://www.st.com/en/evaluation-tools/32f429idiscovery.html#documentation

2. Libopencm3 (2023) Libopencm3 Examples. Recuperado de: https://github.com/libopencm3/libopencm3-examples

3. DigiKey (2021) TinyML: Getting Started with TensorFlow Lite for Microcontrollers. Recuperado de: https://youtu.be/gDFWCxrJruQ

4. Lucha Chiodini (2021) Tiny ML on Arduino. Recuperado de: https://github.com/arduino/ArduinoTensorFlowLiteTutorials/blob/master/GestureToEmoji/arduino_tinyml_workshop.ipynb

5. Apéndice

UM1670
User manual

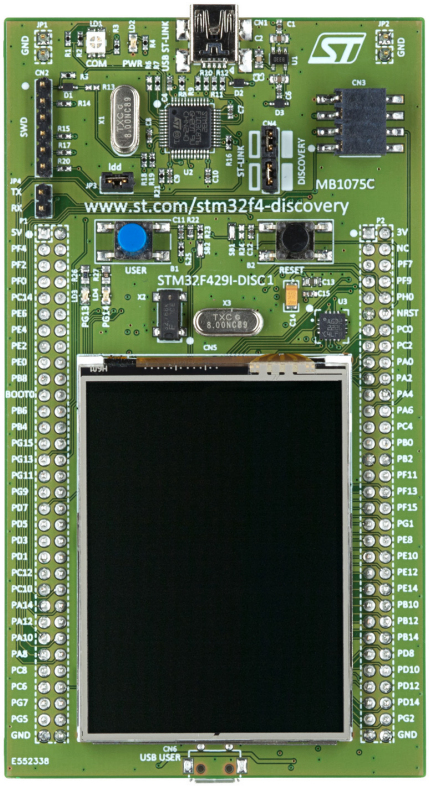
Discovery kit with STM32F429ZI MCU

Introduction

The 32F429IDISCOVERY Discovery kit allows users to easily develop applications with the STMicroelectronics Arm® Cortex®-M4 core-based STM32F429 high-performance microcontroller. It includes an ST-LINK/V2-B embedded debug tool, a 2.4" QVGA TFT LCD, an external 64-Mbit SDRAM, an ST MEMS gyroscope, a USB OTG Micro-AB connector, LEDs and push-buttons.

The board comes with the STM32 comprehensive free software libraries and examples available with the STM32CubeF4 MCU Package, as well as direct access to the Arm® Mbed Enabled™ resources at the http://mbed.org website.

Figure 1. STM32F429 Discovery board



Picture is not contractual.

1 Features

- STM32F429ZIT6 microcontroller featuring 2 Mbytes of Flash memory, 256 Kbytes of RAM in an LQFP144 package
- 2.4" QVGA TFT LCD
- USB OTG with Micro-AB connector
- I3G4250D, ST MEMS motion sensor 3-axis digital output gyroscope
- Six LEDs:
  - LD1 (red/green) for USB communication
  - LD2 (red) for 3.3 V power-on
  - Two user LEDs: LD3 (green), LD4 (red)
  - Two USB OTG LEDs: LD5 (green) V_{BUS} and LD6 (red) OC (over-current)
- Two push-buttons (user and reset)
- 64-Mbit SDRAM
- Extension header for LQFP144 I/Os for a quick connection to the prototyping board and an easy probing
- On-board ST-LINK/V2-B
- USB functions:
  - Debug port
  - Virtual COM port
  - Mass storage
- Mbed Enabled™ (see http://mbed.org)
- Board power supply: through the USB bus or from an external 5 V supply voltage
- Comprehensive free software including a variety of examples, part of STM32CubeF4 MCU Package or STSW-STM32138, for using legacy standard libraries

6 Hardware layout

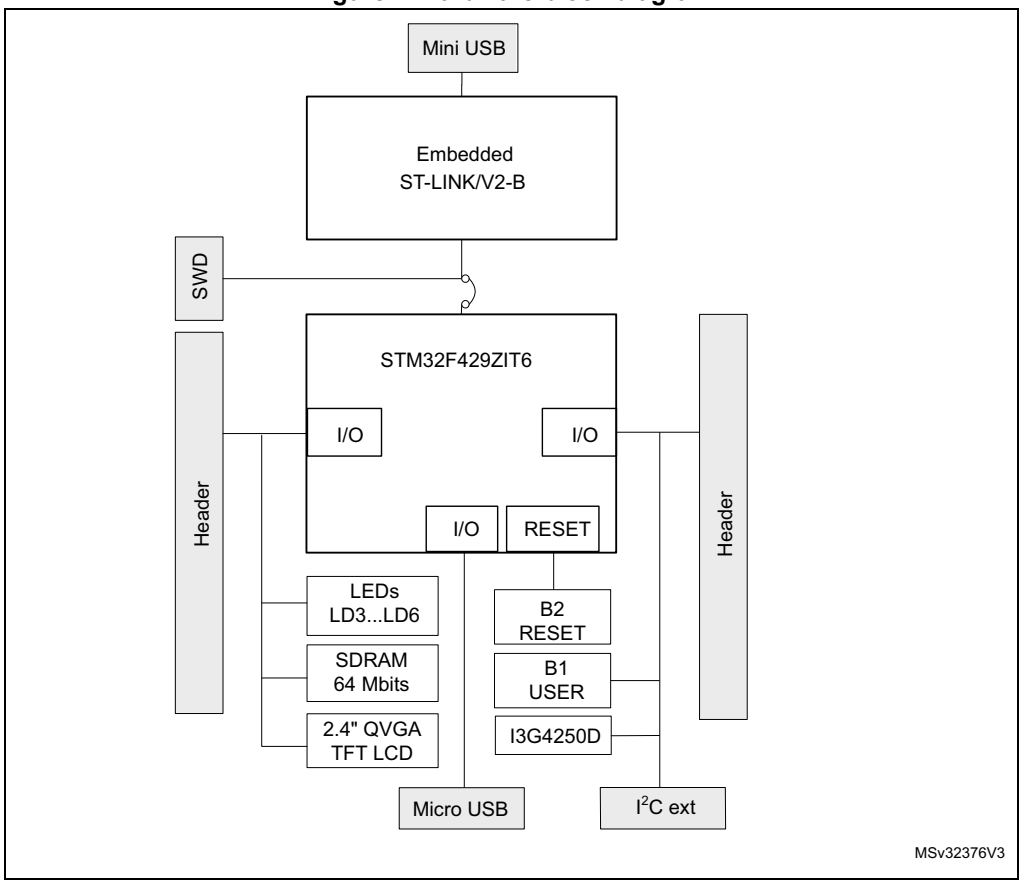
The STM32F429 Discovery board has been designed around the STM32F429ZIT6 microcontroller in a 144-pin LQFP package.

Figure 2 illustrates the connections between the STM32F429ZIT6 and its peripherals (ST-LINK/V2-B, push-buttons, LEDs, USB OTG, ST-MEMS gyroscope, accelerometer, magnetometer, and connectors).

Figure 3 and *Figure 4* show the location of these features on the STM32F429 Discovery board.

Figure 5 shows the mechanical dimensions of the STM32F429 Discovery board.

Figure 2. Hardware block diagram



6.1 STM32F429 Discovery board layout

Figure 3. Top layout

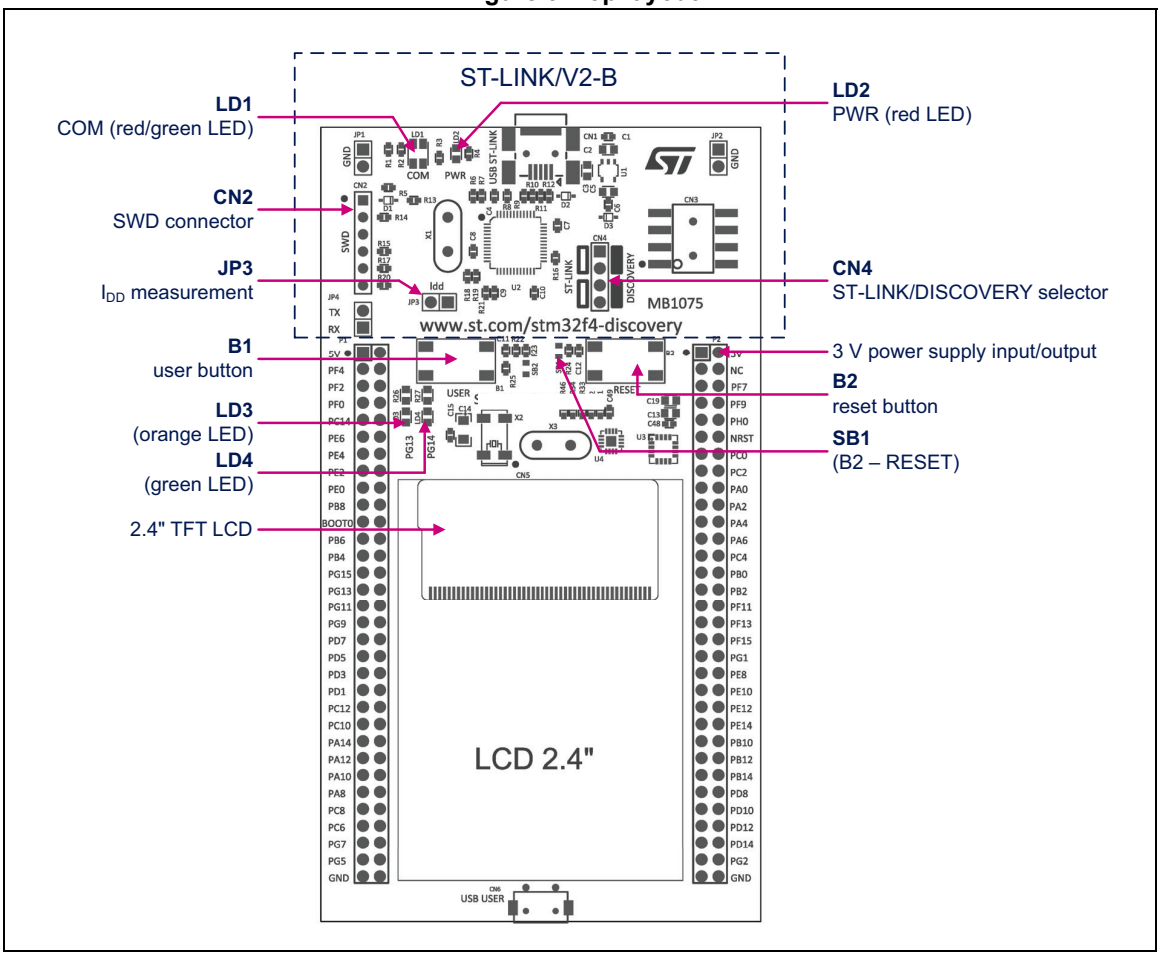
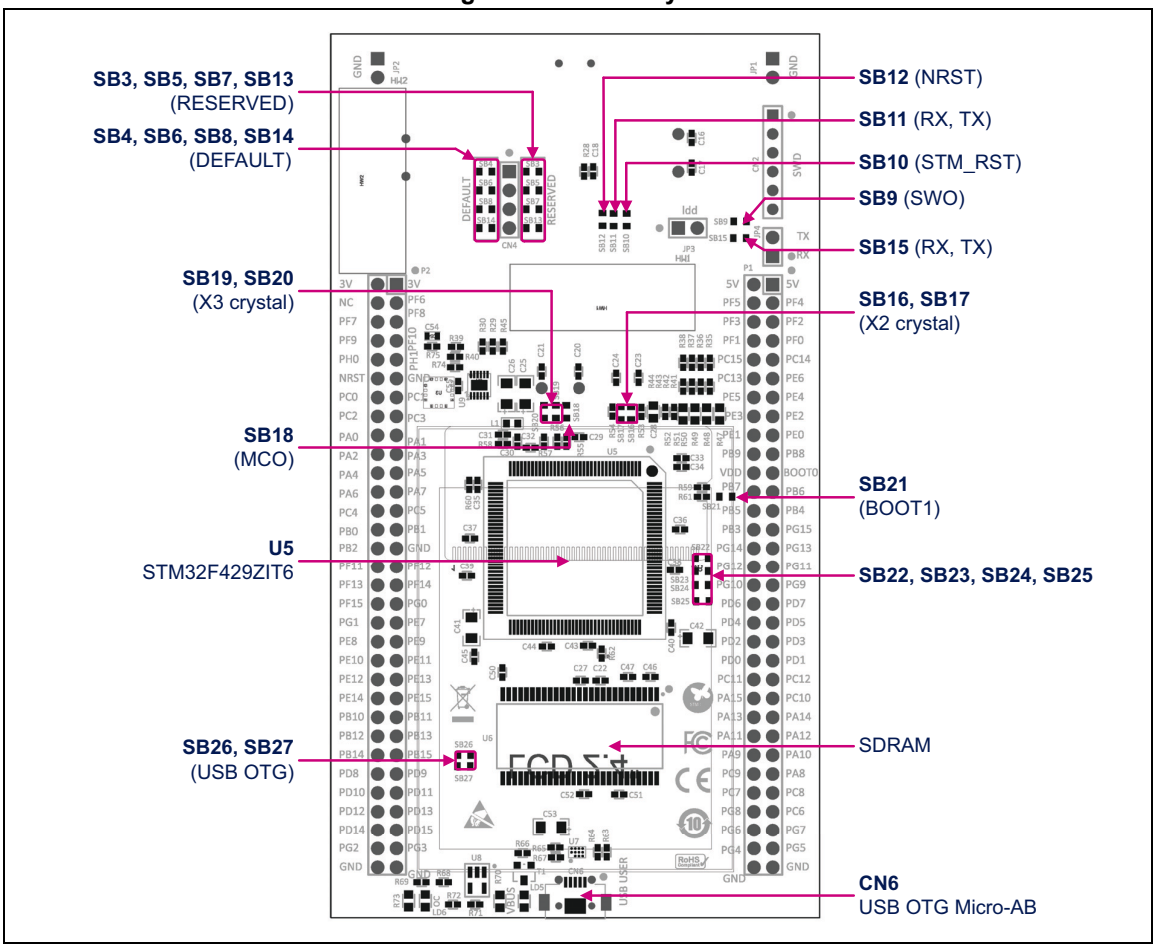
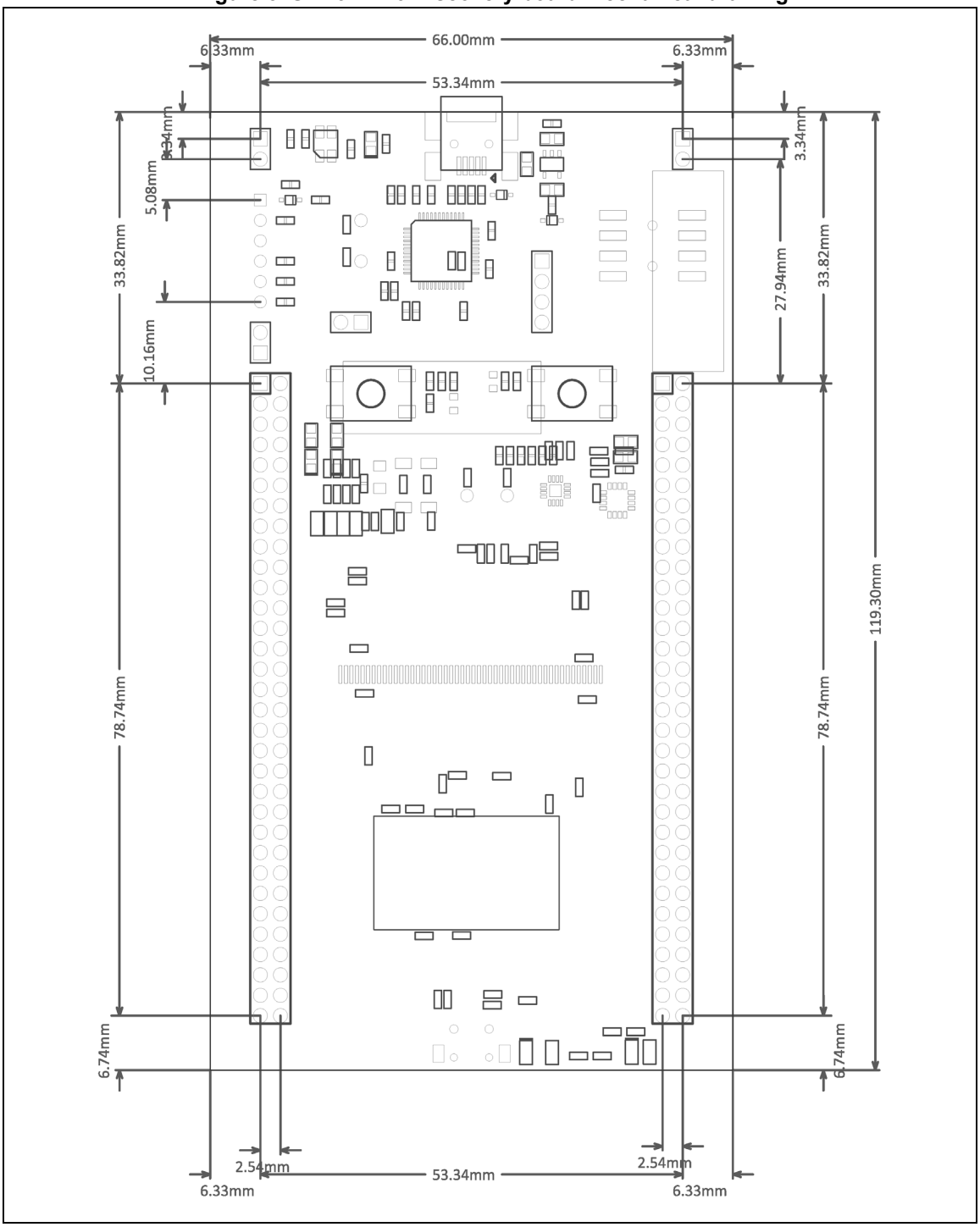


Figure 4. Bottom layout



6.2 Mechanical drawing

Figure 5. STM32F429 Discovery board mechanical drawing



6.4 Power supply and power selection

The power supply is provided either by the host PC through the USB cable or by an external 5 V power supply.

The D2 and D3 diodes protect the 5 V and 3 V pins from external power supplies:

- 5 V and 3 V can be used as output power supplies when another application board is connected to pins P1 and P2.
  In this case, the 5 V and 3 V pins deliver a 5 V or 3 V power supply and the power consumption must be lower than 100 mA.
- 5 V and 3 V can also be used as input power supply, e.g. when the USB connectors are not connected to the PC.
  In this case, the STM32F429 Discovery board must be powered by a power supply unit or by auxiliary equipment complying with the standard EN-60950-1: 2006+A11/2009, and must be Safety Extra Low Voltage (SELV) with limited power capability.

6.5 LEDs

- LD1 COM:
  The LD1 default status is red. LD1 turns to green to indicate that communications are in progress between the PC and the ST-LINK/V2-B.
- LD2 PWR:
  The red LED indicates that the board is powered.
- User LD3:
  The green LED is a user LED connected to the I/O PG13 of the STM32F429ZIT6.
- User LD4:
  The red LED is a user LED connected to the I/O PG14 of the STM32F429ZIT6.
- User LD5:
  The green LED indicates when VBUS is present on CN6 and is connected to PB13 of the STM32F429ZIT6.
- User LD6:
  The red LED indicates an overcurrent from VBUS of CN6 and is connected to the I/O PC5 of the STM32F429ZIT6.

6.6 Push-buttons

- B1 USER:
  User and Wake-Up button connected to the I/O PA0 of the STM32F429ZIT6.
- B2 RESET:
  The push-button connected to NRST is used to RESET the STM32F429ZIT6.

6.7 USB OTG supported

The STM32F429ZIT6 drives USB OTG High-Speed through its internal PHY, which limits it to USB OTG Full Speed on this board. The USB Micro-AB connector (CN6) allows the user to connect a host or device component, such as a USB key, mouse, or other.