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

Laboratorio 5 - STM32/Arduino: GPIO, Giroscopio, comunicaciones, TinyML

Prof. MSc. Marco Villalta Fallas

Jose Daniel Blanco Solís - B71137
Mario Navarro Bejarano - B75398

II - 2024

Índice

1. Introducción

En este laboratorio se busca desarrollar un reconocedor de actividad humana básico utilizando la placa STM32F429 Discovery Kit, la biblioteca LibOpenCM3 y la biblioteca TensorFlow, aprovechando la capacidad del giroscopio para detectar movimientos en los ejes X, Y, Z. Este proyecto permitió adquirir habilidades en la integración de sensores, manejo de periféricos y creación de modelos de Aprendizaje de Maquina (ML).

El diseño del reconocedor de actividad humana se basa en el programa realizado para la implementación del laboratorio 4, el cual se enfoca en capturar las variaciones de los ejes del giroscopio y a través de un sistema de comunicación USART/USB transmitir los datos a una computadora, ademas, de incluir un LED indicador que parpadeará indicando que la transmisión USART esta habilitada.

En la implementación adaptada para este laboratorio posteriormente se almacenan los datos en un archivo .csv y este archivo se utiliza en conjunto a la biblioteca TensorFlow para crear un modelo, el cual se integra y ejecuta en el microcontrolador para detectar el movimiento realizado y registrarlo en la PC.

Nota: El código fuente se puede encontrar en el siguiente repositorio `https://github.com/marionabe/LabMicro/tree/main/Laboratorio5`

2. Nota Teórica

A continuación se muestra un diagrama del diseño general de la placa:

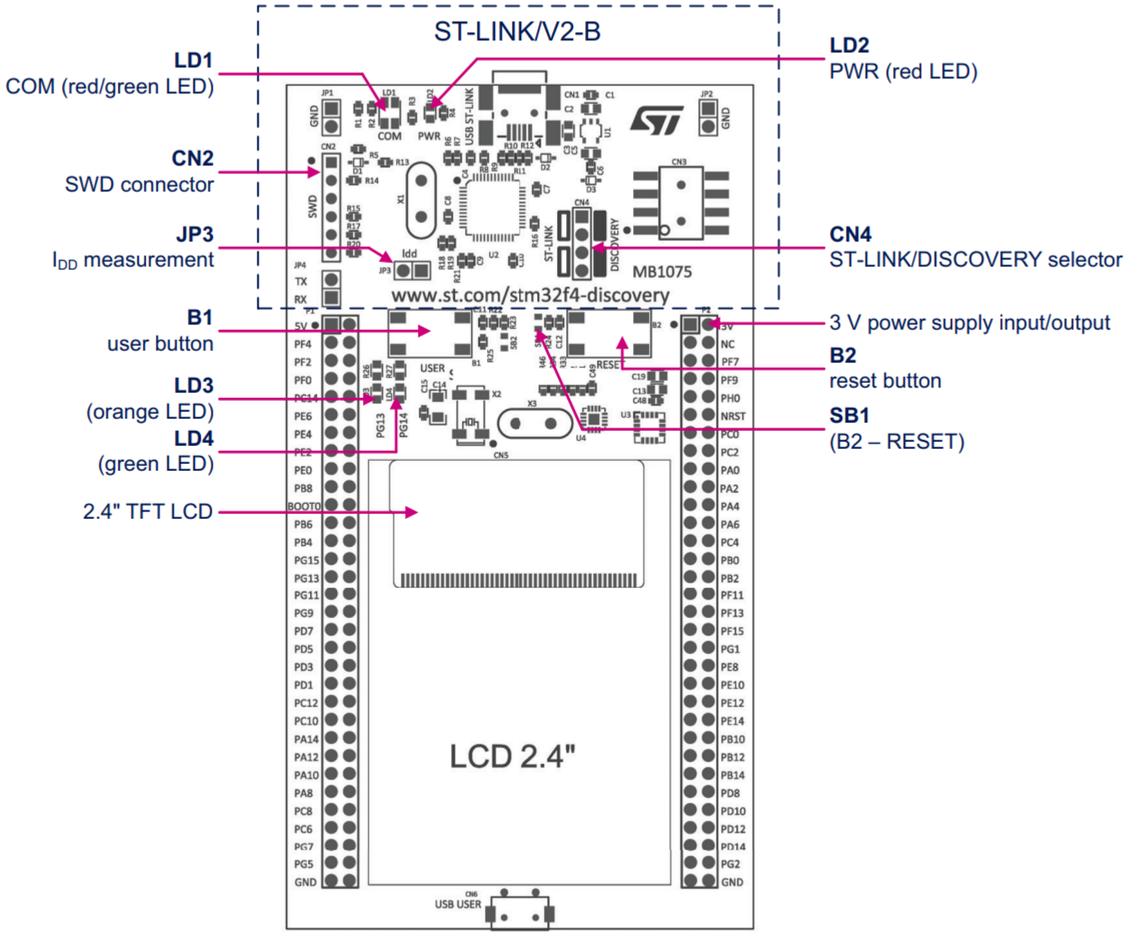


Figura 1: Diseño general del STM32F429i Discovery Kit [1]

2.1. Componentes empleados.

La siguiente tabla detalla los componentes y la cantidad utilizada.

Componente	Cantidad
STM32F429 Discovery Kit	1

2.2. STM32F429ZI

Los siguiente datos fueron tomados de la hoja de datos oficial del fabricante [1], la placa STM32F429 Discovery kit se basa en el microcontrolador STM32F429ZI, el cual es un microcontrolador de alto rendimiento de la serie STM32F4 desarrollado por STMicroelectronics, basado en el núcleo ARM Cortex-M4, este procesador opera a una frecuencia de 180 MHz e incorpora una Unidad de Punto Flotante (FPU) compatible con el estándar IEEE 754, lo que permite realizar operaciones de punto flotante de forma rápida y eficiente, además, incluye una caché de instrucciones y datos (I/D) que optimiza la velocidad de acceso a memoria y minimiza las latencias durante la ejecución del código. El Cortex-M4 también soporta procesamiento digital de señales (DSP), incorporando instrucciones SIMD (Single Instruction Multiple Data) que mejoran el rendimiento en algoritmos matemáticos, aumentando la eficiencia del sistema y ofrece un sistema de gestión de interrupciones capaz de manejar hasta 240 interrupciones con 16 niveles de prioridad, gestionandolas por el NVIC (Nested Vectored Interrupt Controller) para asegurar respuestas rápidas y eficientes a eventos de hardware software críticos.

El microcontrolador dispone de 2 MB de memoria Flash interna, utilizada para almacenar el firmware y las aplicaciones críticas, esta memoria permite múltiples ciclos de lectura y escritura para garantizar la estabilidad del sistema durante las actualizaciones, además, cuenta con 256 KB de SRAM interna para la ejecución de tareas en tiempo real y para aplicaciones que requieren un manejo intensivo de datos, como el procesamiento de señales, el sistema puede hacer uso de una memoria externa SDRAM de 64 Mbits para almacenamiento temporal y creación de buffers.

El STM32F429ZI ofrece varios puertos GPIO (General Purpose I/O) los cuales se configuran a través de registros específicos para definir su modo de operación, ya sea como entrada, salida, función alternativa o analógica.

El microcontrolador también dispone de hasta 14 temporizadores, algunos de ellos con funciones avanzadas como la generación de PWM (Pulse Width Modulation), cada temporizador se gestiona mediante los registros de control de temporizadores.

2.3. Giroscopio

La interfaz SPI permite la comunicación entre el microcontrolador y el giroscopio MEMS I3G4250D integrado en la placa, este es un sensor de movimiento de tres ejes que permite medir la velocidad angular en rangos ajustables de ± 245, ± 500 o ± 2000 grados por segundo (dps), lo que lo hace adecuado para aplicaciones de monitoreo de vibraciones y detección de movimientos, este giroscopio tambien cuenta con una interfaz de salida digital compatible con I2C, lo que facilita su integración con el microcontrolador y permite la captura de datos en tiempo real de manera eficiente, además, el I3G4250D incluye un filtro pasa bajos configurable que mejora la precisión en la medición de cambios de orientación y detecta variaciones sutiles en el movimiento en los ejes X, Y y Z.

2.4. LibOpenCM3

Es una biblioteca de código abierto diseñada específicamente para microcontroladores ARM Cortex-M, ofrece una interfaz ligera, modular y consistente que permite un mayor control sobre los periféricos, facilitando el desarrollo sobre registros, entre los módulos disponibles en libopencm3 se incluyen controladores para GPIO, temporizadores, buses de comunicación, convertidores analógico-digital, controladores DMA, watchdogs y periféricos más avanzados, esta biblioteca está escrita completamente en C y utiliza un sistema de compilación basado en Makefiles.

En este laboratorio se utilizó como la base para la programación de la placa STM32F429 Discovery Kit, buscando una configuración de los periféricos para adquirir datos del giroscopio, asimismo, facilitó la configuración de interrupciones, temporizadores y protocolos de comunicación serial como USART para el envío de datos a través del puerto USB.

2.5. TensorFlow

Es una biblioteca de código abierto desarrollada por Google para computación numérica y aprendizaje automático, permite construir, entrenar e implementar modelos de redes neuronales eficientes y escalables, su arquitectura permite su integración desde dispositivos embebidos hasta servidores de alto rendimiento, soportando ejecución tanto en CPU como GPU, la biblioteca está basada en grafos de flujo de datos, donde los nodos representan operaciones matemáticas y los bordes representan los tensores que fluyen entre operaciones, además, soporta diferentes niveles de abstracción que simplifican el diseño y el entrenamiento de modelos.

En este laboratorio se utilizó para construir un modelo de red neuronal con el fin de procesar secuencias de datos temporales provenientes del giroscopio de la placa STM32F429 Discovery Kit, los datos fueron recolectados, preprocesados y divididos en conjuntos de entrenamiento y prueba, buscando que el modelo aprendiera patrones en las variaciones angulares del sensor, una vez entrenado el modelo fue capaz de predecir comportamientos futuros del movimiento basándose en entradas anteriores.

3. Desarrollo

3.1. Componentes electrónicos

En este laboratorio solamente se utiliza la placa STM32F429 Discovery kit que tiene un valor aproximado al tipo de cambio actual del dólar de 25000 colones.

3.2. Giroscopio

3.2.1. Lectura

Esta sección se basa en el programa realizado para la implementación del laboratorio 4, para realizar la lectura de los valores en los ejes X, Y y Z del giroscopio, se utilizó parte del código que se proporciona en los ejemplos de LibOpenCM3, esto para funciones no relevantes para los objetivos de este laboratorio, como por ejemplo configurar el reloj del sistema.

En la figura 2 se muestra el código utilizado para configurar los pines y el SPI5, estos son los elementos necesarios para comunicarse con el giroscopio, primero se configuran los puertos GPIO, como salidas, así como las frecuencias, luego se configuran los registros SPI5 para habilitarlo a funcionar en modo maestro, dados los parámetros de funcionamiento de la interfaz SPI5 del giroscopio.

```c
void spi_setup(void)
{
    rcc_periph_clock_enable(RCC_SPI5);
    rcc_periph_clock_enable(RCC_GPIOC);
    rcc_periph_clock_enable(RCC_GPIOF);

    gpio_mode_setup(GPIOC, GPIO_MODE_OUTPUT, GPIO_PUPD_NONE, GPIO1);
    gpio_set(GPIOC, GPIO1);
    gpio_mode_setup(GPIOF, GPIO_MODE_AF, GPIO_PUPD_NONE,
        GPIO7 | GPIO8 | GPIO9);
    gpio_set_af(GPIOF, GPIO_AF5, GPIO7 | GPIO8 | GPIO9);

    spi_set_master_mode(SPI5);
    spi_set_baudrate_prescaler(SPI5, SPI_CR1_BR_FPCLK_DIV_64);
    spi_set_clock_polarity_0(SPI5);
    spi_set_clock_phase_0(SPI5);
    spi_set_full_duplex_mode(SPI5);
    spi_set_unidirectional_mode(SPI5);
    spi_enable_software_slave_management(SPI5);
    spi_send_msb_first(SPI5);
    spi_set_nss_high(SPI5);
    SPI_I2SCFGR(SPI5) &= ~SPI_I2SCFGR_I2SMOD;
    spi_enable(SPI5);

    rcc_periph_clock_enable(RCC_GPIOG);
    gpio_mode_setup(GPIOG, GPIO_MODE_OUTPUT, GPIO_PUPD_NONE, GPIO13 | GPIO14);

}
```

Figura 2: Código para configurar comunicación con SPI5

3.2.2. Almacenamiento

Para esta funcionalidad se desarrolló un script en Python encargado de la recepción y organización de los datos el cual acepta únicamente tres columnas de información correspondientes a los ejes X, Y y Z del sensor de movimiento y reorganiza los datos en estructuras adecuadas para su almacenamiento y análisis en formato de extensión csv.

Dado que la cantidad de datos por secuencia incide directamente en la estructura de entrada requerida por el modelo de TensorFlow, una vez integradas las implementaciones de las secciones anteriores se realizó la adquisición de datos registrando series de movimientos repetidos con una meta de tomar alrededor de 1000 muestras en un lapso de 15 segundos, una vez se confirmó la consistencia de los datos adquiridos se realizaron ajustes sobre los archivos de datos finales correspondientes a las distintas repeticiones de movimiento utilizadas, con el fin de garantizar que cada uno contuviera exactamente 1000 muestras para entrenar y evaluar el modelo. En la figura 3 se muestra el código utilizado para esto.

```python
import paho.mqtt.client as mqtt
import serial
import time
import json
import csv

# Configuraci�n del puerto serie
try:
    ser = serial.Serial(port='/dev/ttyACM0', baudrate=115200, timeout=1)
    print("Conexi�n con STM32 establecida.")
except serial.SerialException as e:
    print("Error al conectar con el puerto serie.")
    exit()

try:
    archivo = open('../Datos_obtenidos/Datos_Labo5.csv', 'w')  #Renombrar (Datos_Labo5.csv) para diferenciar toma de datos
    escritura_csv = csv.writer(archivo)
    datos_columnas = []
    encabezados = ['Eje X', 'Eje Y', 'Eje Z']
    print(encabezados)
    escritura_csv.writerow(encabezados)
except IOError as e:
    print("No se pudo crear el archivo CSV.")
    ser.close()
    exit()

# Bucle principal para la lectura de datos
try:
    while(1):
        data = ser.readline().decode('utf-8').replace('\r', "").replace('\n', "")
        data = data.split('\t')
        if len(data) >= 3:
            escritura_csv.writerow(data)
            print(data)
except KeyboardInterrupt:
    archivo.close()
    ser.close()
    print("Archivo CSV cerrado correctamente.")
```

Figura 3: Código para almacenamiento de datos en formato csv

3.3. Creación y entrenamiento del modelo

Una vez validada la consistencia de los datos el primer paso consistió en su preparación, esta etapa implicó la definición de una semilla aleatoria para asegurar reproducibilidad, la asignación de etiquetas a los datos según el tipo de movimiento que representa y la creación del vector de resultados esperados para cada muestra, posteriormente se mezclaron las muestras de forma aleatoria para evitar sesgos durante el entrenamiento y se dividieron en tres conjuntos: 60 % para entrenamiento, 20 % para prueba, y 20 % para validación. La red neuronal fue implementada utilizando Keras, específicamente empleando la función de activación ReLU, el optimizador RMSprop y la métrica de error absoluto medio (MAE) para evaluar las pérdidas. En las figura 4 se muestra el código utilizado para esto.

```
# Set a fixed random seed value, for reproducibility, this will allow us to get
# the same random numbers each time the notebook is run
SEED = 1337
np.random.seed(SEED)
tf.random.set_seed(SEED)

# the list of gestures that data is available for
GESTURES = [
    "Mov_Circulos",
    "Mov_Arriba_Abajo",
    "Mov_Extension_Brazo",
]

SAMPLES_PER_GESTURE = 146

NUM_GESTURES = len(GESTURES)

# create a one-hot encoded matrix that is used in the output
ONE_HOT_ENCODED_GESTURES = np.eye(NUM_GESTURES)

inputs = []
outputs = []

# read each csv file and push an input and output
for gesture_index in range(NUM_GESTURES):
  gesture = GESTURES[gesture_index]
  print(f"Processing index {gesture_index} for gesture '{gesture}'.")

  output = ONE_HOT_ENCODED_GESTURES[gesture_index]

  df = pd.read_csv("Datos_obtenidos/" + gesture + ".csv")

  # calculate the number of gesture recordings in the file
  num_recordings = int(df.shape[0] / SAMPLES_PER_GESTURE)

  print(f"\tThere are {num_recordings} recordings of the {gesture} gesture.")

  for i in range(num_recordings):
    tensor = []
    for j in range(SAMPLES_PER_GESTURE):
      index = i * SAMPLES_PER_GESTURE + j
      # normalize the input data, between 0 to 1:
      # - gyroscope is between: -2000 to +2000
      tensor += [
          (df['Eje X'][index] + 2000) / 4000,
          (df['Eje Y'][index] + 2000) / 4000,
          (df['Eje Z'][index] + 2000) / 4000
      ]

    inputs.append(tensor)
    outputs.append(output)

# convert the list to numpy array
inputs = np.array(inputs)
outputs = np.array(outputs)
print("inputs:",inputs)
print("Out", outputs)
print("Data set parsing and preparation complete.")

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

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

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

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

print("Data set randomization and splitting complete.")
print(inputs_validate);
print(outputs_validate)

# build the model and train it
model = tf.keras.Sequential()
model.add(tf.keras.layers.Dense(50, activation='relu')) # relu is used for performance
model.add(tf.keras.layers.Dense(NUM_GESTURES, activation='softmax')) # softmax is used, because we only expect one gesture to occur per input
model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])
history = model.fit(inputs_train, outputs_train, epochs=600, batch_size=2, validation_data=(inputs_validate, outputs_validate))
```

Figura 4: Código para la creación y entrenamiento del modelo

3.4. Detección de movimiento

Para la integración del modelo entrenado en el microcontrolador es necesario importar las bibliotecas de TensorFlow Lite, así como el modelo que encapsula los pesos y la arquitectura de la red neuronal como una biblioteca en C.

Para permitir que el modelo interprete correctamente las señales del sensor giroscópico y genere una predicción basada en los patrones previamente aprendidos, se definió la configuración indicando tanto los parámetros de entrada y salida del modelo como el tamaño de las tensores y los buffers requeridos.

El núcleo del programa consiste en un bucle que recopila datos en tiempo real desde el giroscopio hasta alcanzar el tamaño de entrada esperado por el modelo (1000 muestras organizadas en secuencias de tres variables: X, Y y Z), una vez acumuladas las muestras necesarias los datos son normalizados y procesados mediante el modelo obteniendo como salida la predicción correspondiente al movimiento registrado.

```
const char* GESTURES[] = {
  "Mov_Circulo",
  "Mov_Arriba_Abajo",
  "Mov_Extension_Brazo"
};

#define NUM_GESTURES (sizeof(GESTURES) / sizeof(GESTURES[0]))
```

```
    tflite::MicroErrorReporter micro_error_reporter;
    tflite::ErrorReporter * error_reporter = &micro_error_reporter;

    TfLiteTensor * input = nullptr;
    TfLiteTensor * output = nullptr;

    /* Loading the model */
    const tflite::Model * tf_model = tflite::GetModel(girosmodel);
    if(model->version() != TFLITE_SCHEMA_VERSION)
    {
        error_reporter->Report("Model provided is schema version %d not equal"
                               "to supported version %d.\n",
                               model->version(), TFLITE_SCHEMA_VERSION);
        return 1;
    }

    static tflite::MicroMutableOpResolver<4> micro_op_resolver;
    micro_op_resolver.AddConv2D();
    micro_op_resolver.AddMaxPool2D();
    micro_op_resolver.AddFullyConnected();
    micro_op_resolver.AddReshape();

    const int tensor_arena_size = 8*1024;
    static uint8_t tensor_arena[tensor_arena_size];

    static tflite::MicroInterpreter static_interpreter(tf_model, micro_op_resolver, tensor_arena, tensor_arena_size, error_reporter);

    TfLiteStatus allocate_status = static_interpreter.AllocateTensors();
    if( allocate_status != kTfLiteOk)
    {
        TF_LITE_REPORT_ERROR(error_reporter, "AllocateTensor() failed");
        return 1;
    }

    input = static_interpreter.input(0);
    output = static_interpreter.output(0);
```

```
        while (samplesRead < numSamples) {

            input->data.f[samplesRead * 6 + 3] = (gyr_x + 2000.0f) / 4000.0f;
            input->data.f[samplesRead * 6 + 4] = (gyr_y + 2000.0f) / 4000.0f;
            input->data.f[samplesRead * 6 + 5] = (gyr_z + 2000.0f) / 4000.0f;

            samplesRead++;

            if (samplesRead >= numSamples) {
                // Ejecutar inferencia
                TfLiteStatus invoke_status = static_interpreter.Invoke();
                if (invoke_status != kTfLiteOk) {
                    console_puts("Fallo en inferencia\n");
                    while (1);
                }

                // Obtener indice del gesto con mayor probabilidad
                int max_index = 0;
                float max_score = output->data.f[0];
                for (int i = 1; i < NUM_GESTURES; i++) {
                    if (output->data.f[i] > max_score) {
                        max_score = output->data.f[i];
                        max_index = i;
                    }
                }

                console_puts("Movimiento detectado: ");
                console_puts(GESTURES[max_index]);
                console_puts("\n");

                samplesRead = 0;
            }
        }
```

Figura 5: Código para la detección del movimiento efectuado

4. Análisis de los resultados

El proceso de entrenamiento fue monitoreado mediante la visualización de curvas de pérdida correspondientes a los conjuntos de entrenamiento y validación (figuras 6 y 7), lo que permitió evaluar el desempeño del modelo durante las iteraciones.

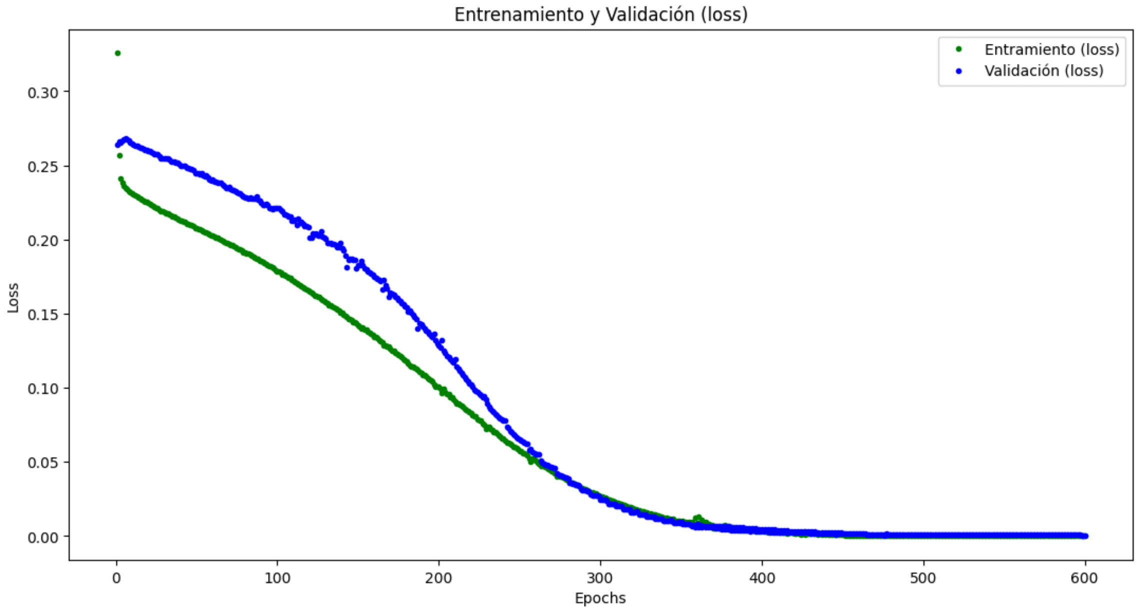


Figura 6: Entrenamiento y Validación (loss)

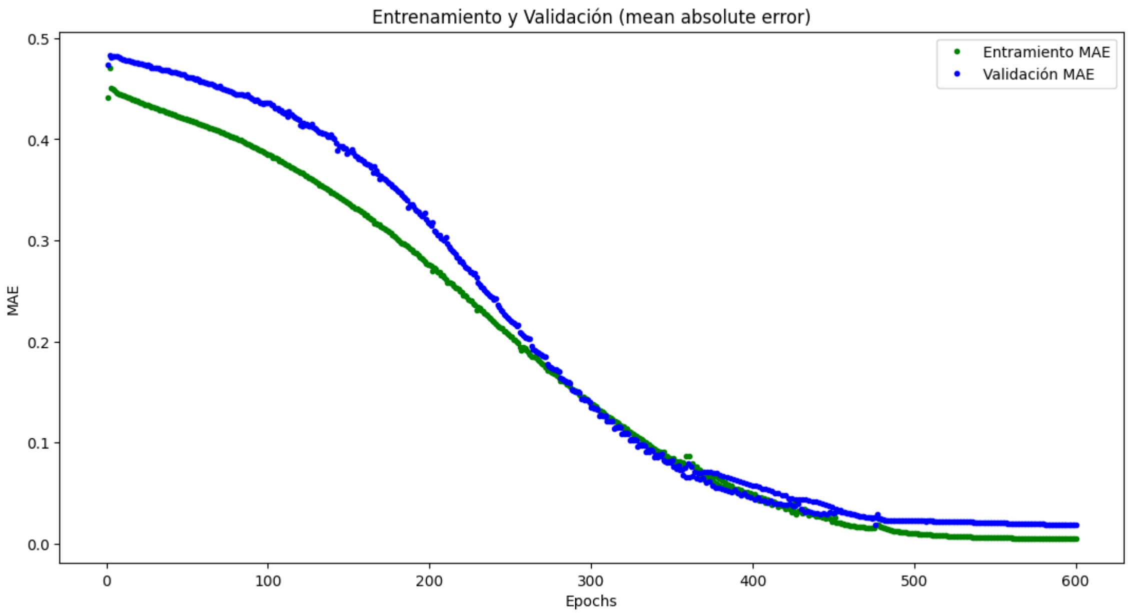


Figura 7: Entrenamiento y Validación (MAE)

Una vez finalizado el entrenamiento se utilizó el conjunto de validación para medir la precisión del modelo, en la comparación entre los resultados esperados y las predicciones generadas por la red neuronal, se observó una precisión aceptable (figura 8) la cual podría mejorar con ajustes finos en los hiperparámetros y/o aumentando la cantidad de datos disponibles.

```
predictions =
 [[0.831 0.06  0.108]
 [0.119 0.057 0.824]
 [0.242 0.294 0.464]]
actual =
 [[1. 0. 0.]
 [0. 0. 1.]
 [1. 0. 0.]]
```

Figura 8: Validación de precisión del modelo

Finalmente al ejecutar el modelo implementado en el microcontrolador se observa que funciona de manera estable en términos de ejecución, pero se presentan dificultades en la detección precisa de los movimientos, en varios casos el modelo no logró identificar adecuadamente el gesto, lo cual probablemente se debe a que en el inicio de la ventana los datos no coinciden con el inicio del movimiento, provocando que el patrón de entrada no contenga una repetición completa del gesto y resultando en una reducción de la precisión del modelo.

```
['-388', '56', '-23']
['-310', '-25', '-39']
['-254', '-39', '3']
['-263', '13', '35']
['-316', '52', '13']
['-313', '39', '-50']
['-292', '33', '-56']
['-270', '11', '-20']
['-304', '6', '27']
['-368', '29', '9']
['-424', '29', '-49']
['-436', '-30', '-47']
['-374', '-101', '0']
['-328', '-109', '-6']
['-337', '-46', '-19']
['-409', '39', '-28']
['-443', '-9', '-12']
Movimiento detectado: Mov_Arriba_Abajo
```

Figura 9: Salida en terminal de detección del movimiento (Arriba y Abajo)

```
['-47', '14', '-255']
['-53', '-14', '-258']
['-30', '-70', '-239']
['-30', '-42', '-245']
['11', '44', '-251']
['-29', '28', '-236']
['-52', '-25', '-180']
['-13', '-21', '-133']
['-26', '-2', '-127']
['16', '29', '-109']
['10', '-4', '-82']
Movimiento detectado: ---
```

Figura 10: Salida en terminal de detección del movimiento (Extensión de Brazo (Fallida))

```
['23', '-298', '-21']
['52', '-102', '-12']
['12', '-153', '-9']
['-7', '-304', '-14']
['5', '-413', '-27']
['16', '-382', '-26']
['3', '-219', '-16']
['-8', '-178', '-13']
['-23', '-253', '-10']
['-14', '-371', '-26']
['15', '-370', '-23']
['5', '-348', '-17']
['-31', '-365', '-7']
Movimiento detectado: Mov_Circulo
```

Figura 11: Salida en terminal de detección del movimiento (Circulo)

5. Conclusiones

En este proyecto se observo que es posible entrenar redes neuronales funcionales desde una computadora portátil utilizando herramientas como TensorFlow y ejecutarlas exitosamente en dispositivos como el STM32F429 Discovery Kit, sin embargo, también se identificaron desafíos significativos, específicamente en la captura y segmentación de los datos ya que la precisión del modelo se ve afectada cuando las muestras no están correctamente sincronizadas con los eventos reales que se desea detectar.

Sería recomendable implementar mecanismos de detección de eventos o segmentación automática para delimitar con mayor exactitud las ventanas de entrada y garantizar que las secuencias que alimentan el modelo representen de forma adecuada los gestos, sin embargo, estas mejoras requerirían un incremento en la complejidad del sistema y una mayor cantidad de datos de entrenamiento.

Referencias

[1] STMicroelectronics. Discovery kit with stm32f429zi mcu. https://www.st.com/resource/en/user_manual/um1670-discovery-kit-with-stm32f429zi-mcu-stmicroelectronics.pdf.

6. Apéndice

UM1670
User manual

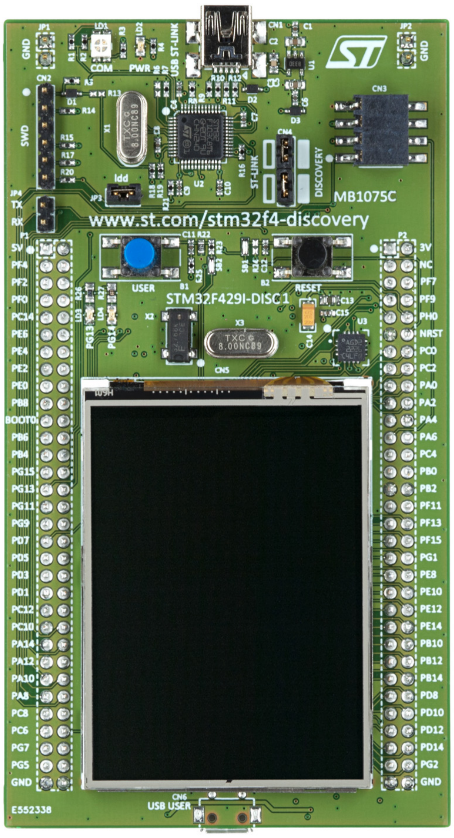
Discovery kit with STM32F429ZI MCU

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.

Contents

List of tables

List of figures

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

2 Ordering information

To order the Discovery kit with the STM32F429ZI microcontroller, refer to *Table 1*. Additional information is available from the datasheet and reference manual of the target microcontroller.

Table 1. Ordering information

Order code	Board reference	Target STM32
STM32F429I-DISC1(1)	MB1075	STM32F429ZIT6

1. Mbed Enabled™ STM32F429I-DISC1 with ST-LINK/V2-B replaces obsolete STM32F429I-DISCO with ST-LINK/V2.

2.1 Codification

The meaning of the codification is explained in *Table 2*.

Table 2. Codification explanation

32XXYYZDISCOVERY	Description	Example: 32F429IDISCOVERY
32XX	MCU series in STM32 32-bit Arm Cortex MCUs	STM32F4 Series
YY	MCU product line in the series	STM32F429
Z	STM32 Flash memory size: – I for 2 Mbytes	2 Mbytes
DISCOVERY	Discovery kit	Discovery kit

The order code is mentioned on a sticker placed on the top or bottom side of the board.

3 Development environment

3.1 System requirements

- Windows® OS (7, 8 and 10), Linux® 64-bit or macOS®[a]
- USB Type-A to Mini-B cable

3.2 Development toolchains

- IAR Systems - IAR Embedded Workbench®[b]
- Keil® - MDK-ARM[b]
- STMicroelectronics - STM32CubeIDE
- Arm® Mbed™[c] online (see http://mbed.org)

3.3 Demonstration software

The demonstration software, included in the STM32Cube MCU Package corresponding to the on-board microcontroller, is preloaded in the STM32 Flash memory for easy demonstration of the device peripherals in standalone mode. The latest versions of the demonstration source code and associated documentation can be downloaded from *www.st.com*.

4 Conventions

Table 3 provides the definition of some conventions used in the present document.

Table 3. ON/OFF conventions

Convention	Definition
Jumper JPx ON	Jumper fitted
Jumper JPx OFF	Jumper not fitted
Solder bridge SBx ON	SBx connections closed by solder
Solder bridge SBx OFF	SBx connections left open

a. macOS® is a trademark of Apple Inc. registered in the U.S. and other countries.

b. On Windows® only.

c. Arm and Mbed are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and or elsewhere.

5 Quick start

The STM32F429 Discovery is a low-cost and easy-to-use development kit to quickly evaluate and start development with an STM32F4 microcontroller.

Before installing and using the product, accept the Evaluation Product License Agreement from www.st.com/stm32f4-discovery.

For more information on the STM32F429 Discovery board and for demonstration software, visit www.st.com/stm32f4-discovery.

5.1 Getting started

Follow the sequence below to configure the STM32F429 Discovery board and launch the DISCOVER application:

1. Ensure that the jumpers JP3 and CN4 are set ON (Discovery mode).
2. Connect the STM32F429 Discovery board to a PC using a USB cable Type-A/Mini-B through the USB ST-LINK connector CN1, to power the board, the LEDs LD2 (PWR) and LD1 (COM).
3. The following applications are available on the screen:
   - Clock/Calendar and Game
   - Video Player and Image Browser (play videos and view images from the USB mass storage connected to CN6)
   - Performance monitor (watch the CPU load and run a graphical benchmark)
   - System Info
4. The demonstration software, as well as other software examples, are available at the www.st.com/stm32f4-discovery web page.
5. Develop applications starting from the examples.

6 Hardware layout

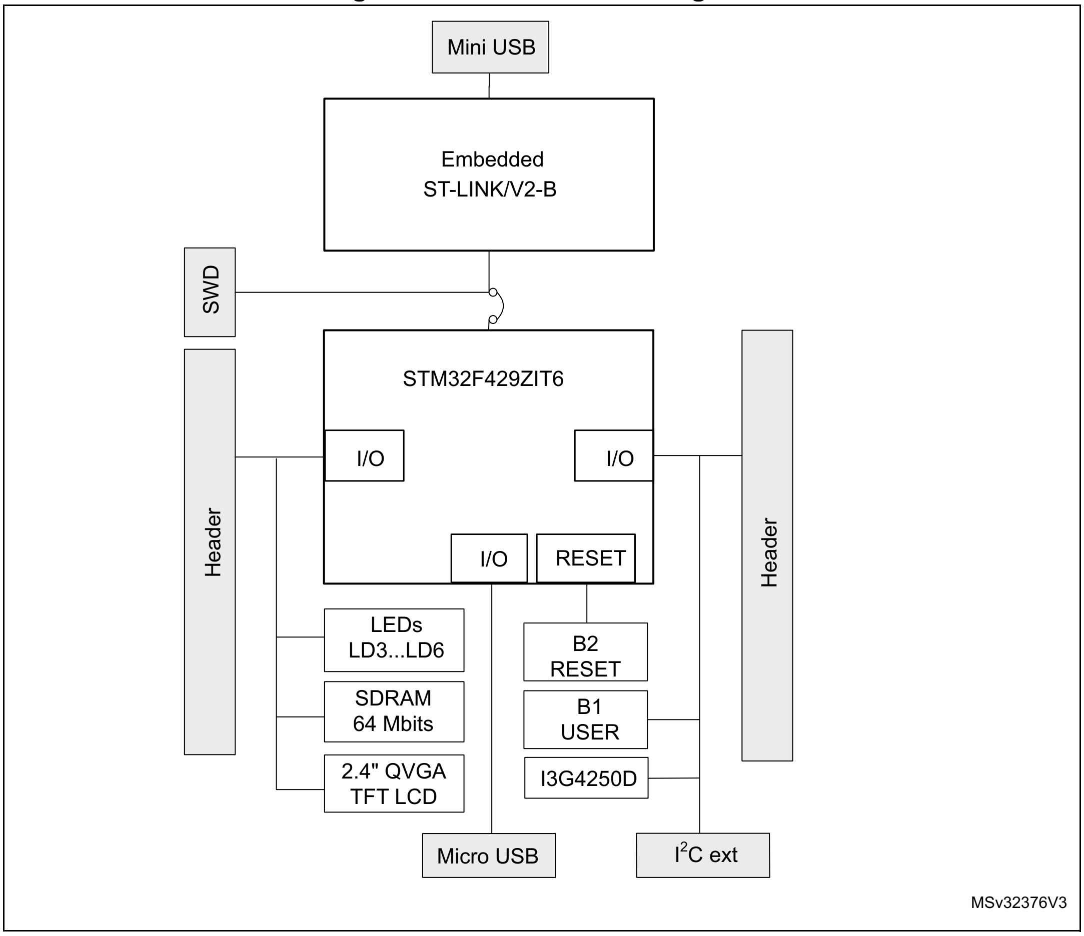
The STM32F429 Discovery board has been designed around the STM32F429ZIT6 microcontroller in a 144-pin LQFP package.

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

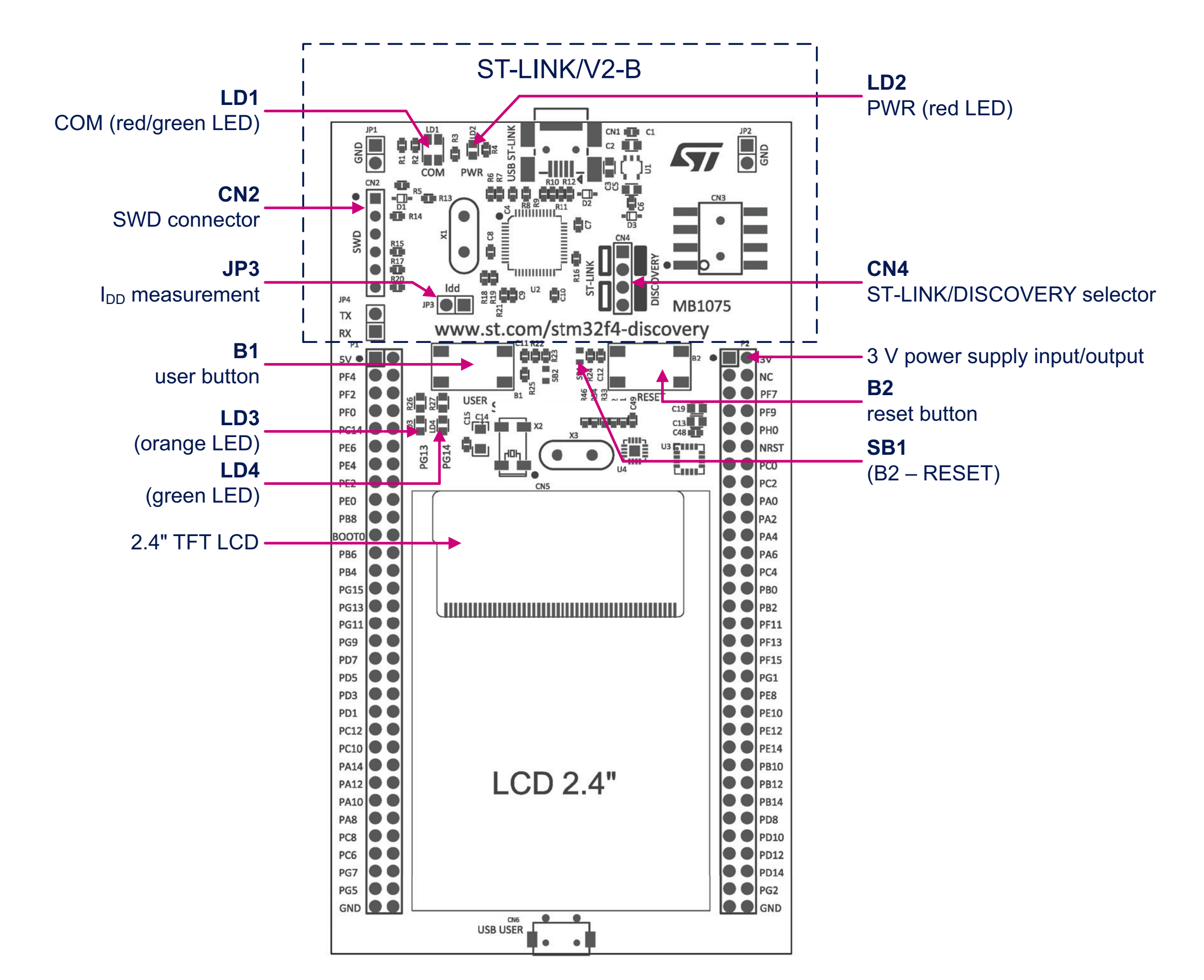
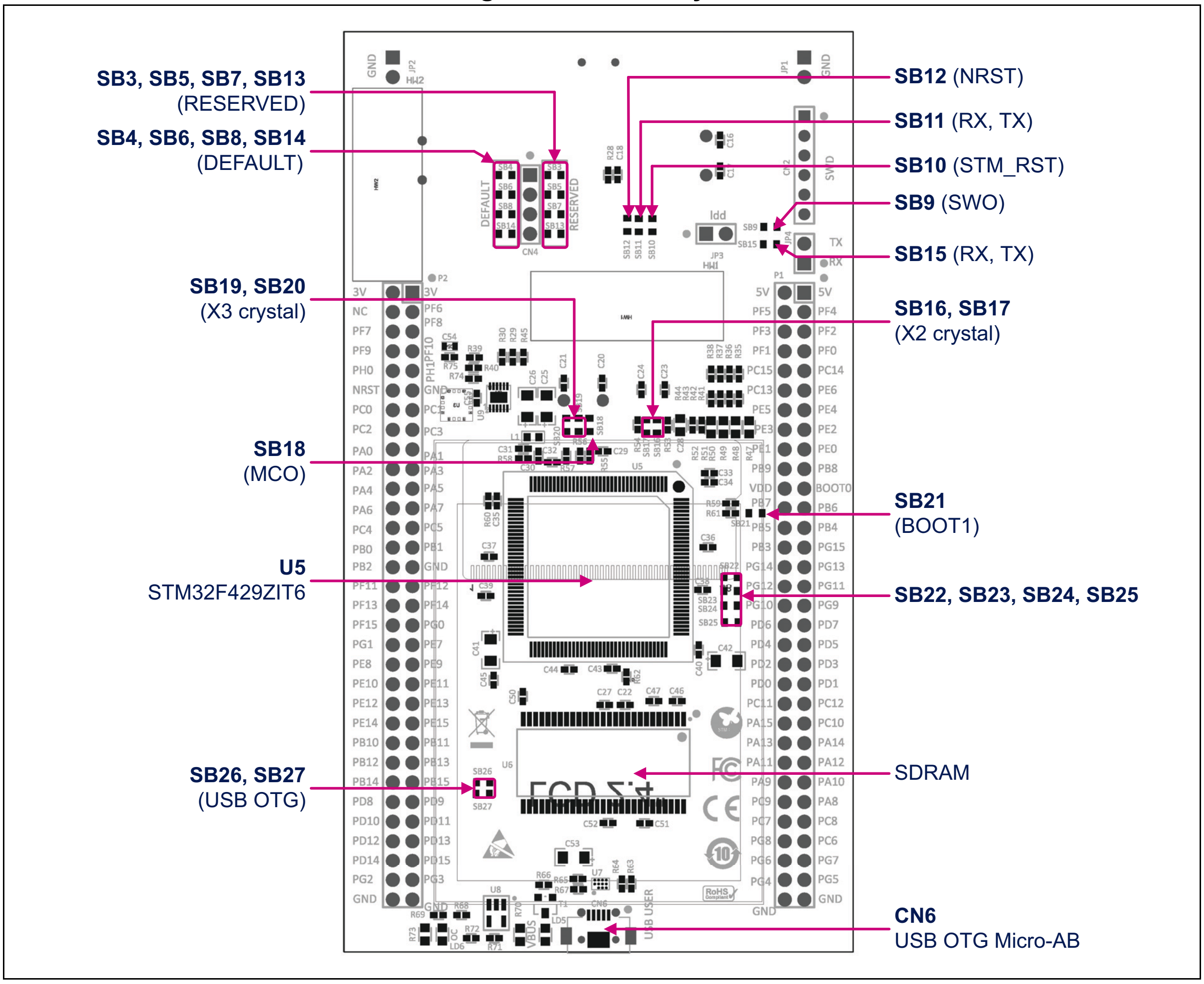
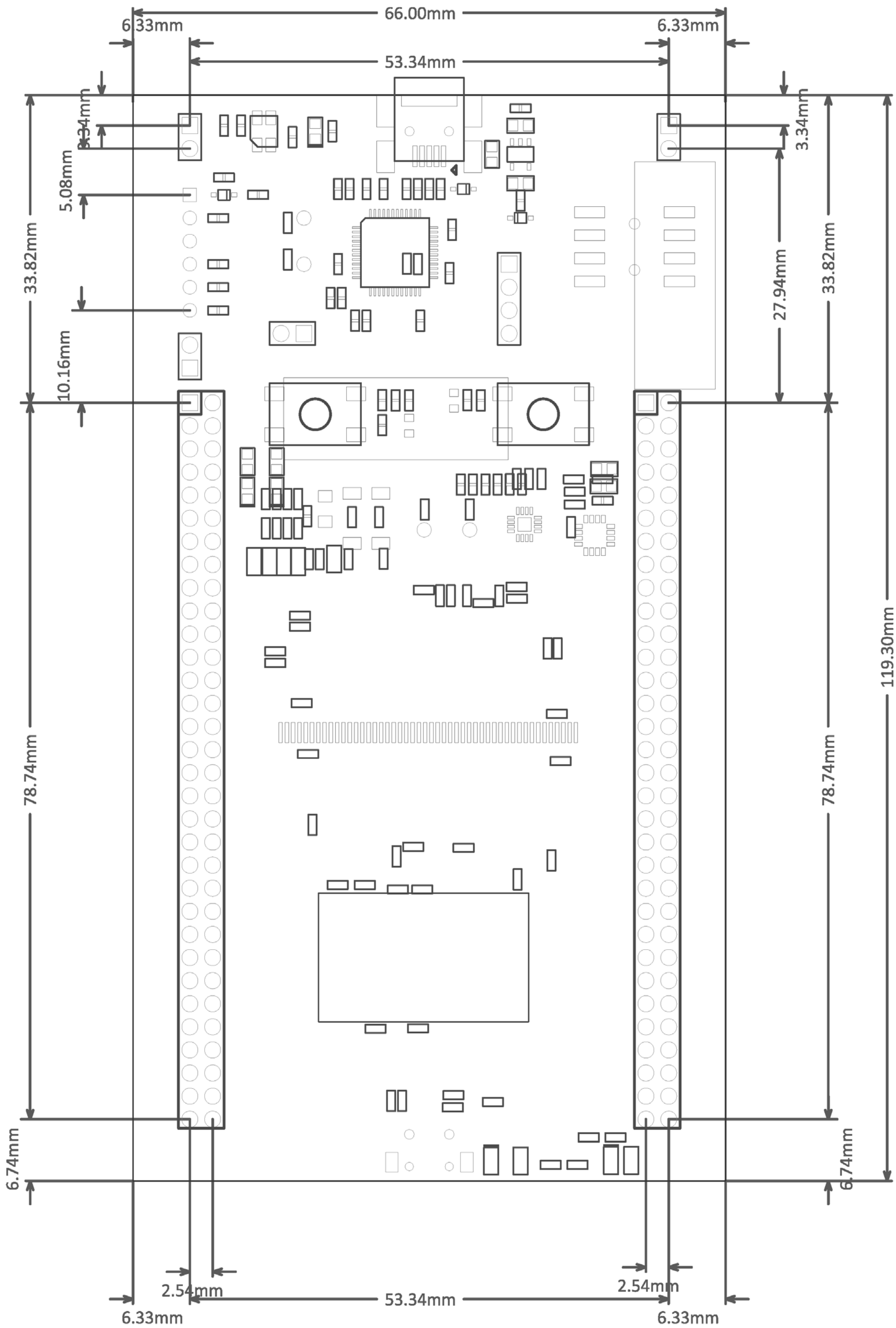


Figure 4. Bottom layout



6.2 Mechanical drawing

Figure 5. STM32F429 Discovery board mechanical drawing



6.3 Embedded ST-LINK/V2-B

The ST-LINK/V2-B on STM32F429I-DISC1 is embedded as a programming and debugging tool. Virtual COM port and USB mass storage features are supported by the ST-LINK/V2-B only for Mbed compatibility.

The ST-LINK/V2-B makes the STM32F4429I-DISC1 boards Mbed Enabled. The embedded ST-LINK/V2-B supports only SWD for STM32 devices. For information about debugging and programming features, refer to *ST-LINK/V2 in-circuit debugger/programmer for STM8 and STM32*, User manual (UM1075), which describes in detail all the ST-LINK/V2 features.

The changes in ST-LINK/V2-B versus ST-LINK/V2 version are listed below.

New features supported on ST-LINK/V2-B are:

- Virtual COM port interface on USB
- Mass storage interface on USB

Features not supported on ST-LINK/V2-B are:

- SWIM interface
- Minimum supported application voltage limited to 3 V
- USB power management request for more than 100 mA power on USB

Known limitation:

- Activating the readout protection on the ST-LINK/V2-B target, prevents the target application from running afterward. The target readout protection must be kept disabled on the ST-LINK/V2-B boards.

There are two different ways to use the embedded ST-LINK/V2-B, depending on the jumper states, as shown in *Table 4*:

- Programming or debugging the STM32 on board. Refer to *Section 6.3.4: Using ST-LINK/V2-B to program/debug the STM32F429ZIT6 on board* to program or debug the STM32F429ZIT6 on-board.
- Programming or debugging the STM32 in an external application board, using a cable connected to the SWD connector CN2. Refer to chapter *Section 6.3.5: Using ST-LINK/V2-B to program/debug an external STM32 application*.

Table 4. Jumper states

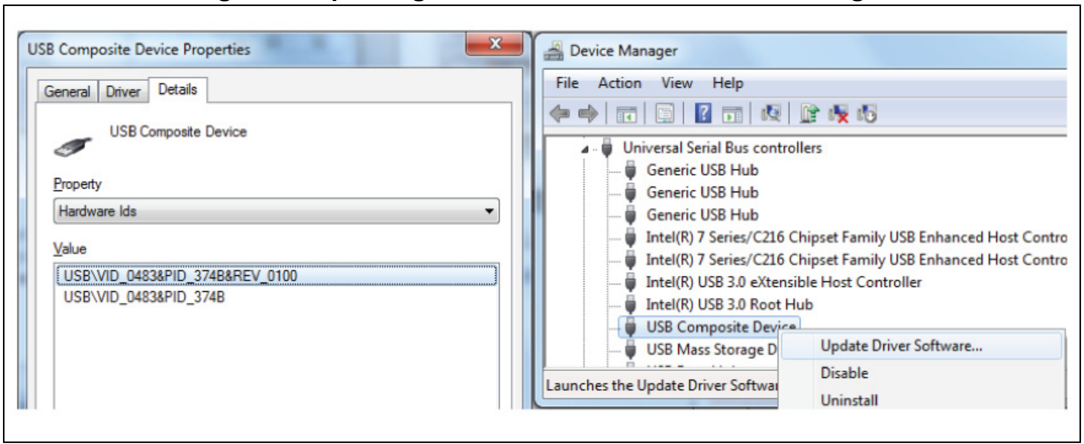
Jumper state	Description
Both CN4 jumpers ON	ST-LINK/V2-B functions enabled for on-board programming (default)
Both CN4 jumpers OFF	ST-LINK/V2-B functions enabled for application through external CN2 connector (SWD supported)

6.3.1 Drivers

The ST-LINK/V2-B requires a dedicated USB driver, which, for Windows® 7, 8, and 10 can be found at the *www.st.com* website. In case the STM32 Discovery is connected to the PC before the driver is installed, some Discovery interfaces may be declared as “Unknown” in the PC device manager. In this case, the user must install the driver files (see *Figure 6: Updating the list of drivers in Device Manager*) and update the driver of the connected device from the device manager.

Note: Prefer using the “USB Composite Device” handle for a full recovery.

Figure 6. Updating the list of drivers in Device Manager



6.3.2 ST-LINK/V2-B firmware upgrade

The ST-LINK/V2-B embeds a firmware upgrade mechanism for the in-situ upgrade through the USB port. As the firmware may evolve during the lifetime of the ST-LINK/V2-B product (for example new functionalities, bug fixes, support for new microcontroller families), it is recommended to visit the *www.st.com* website, before starting to use the Discovery board and periodically, to stay up-to-date with the latest firmware version.

6.3.3 VCP configuration

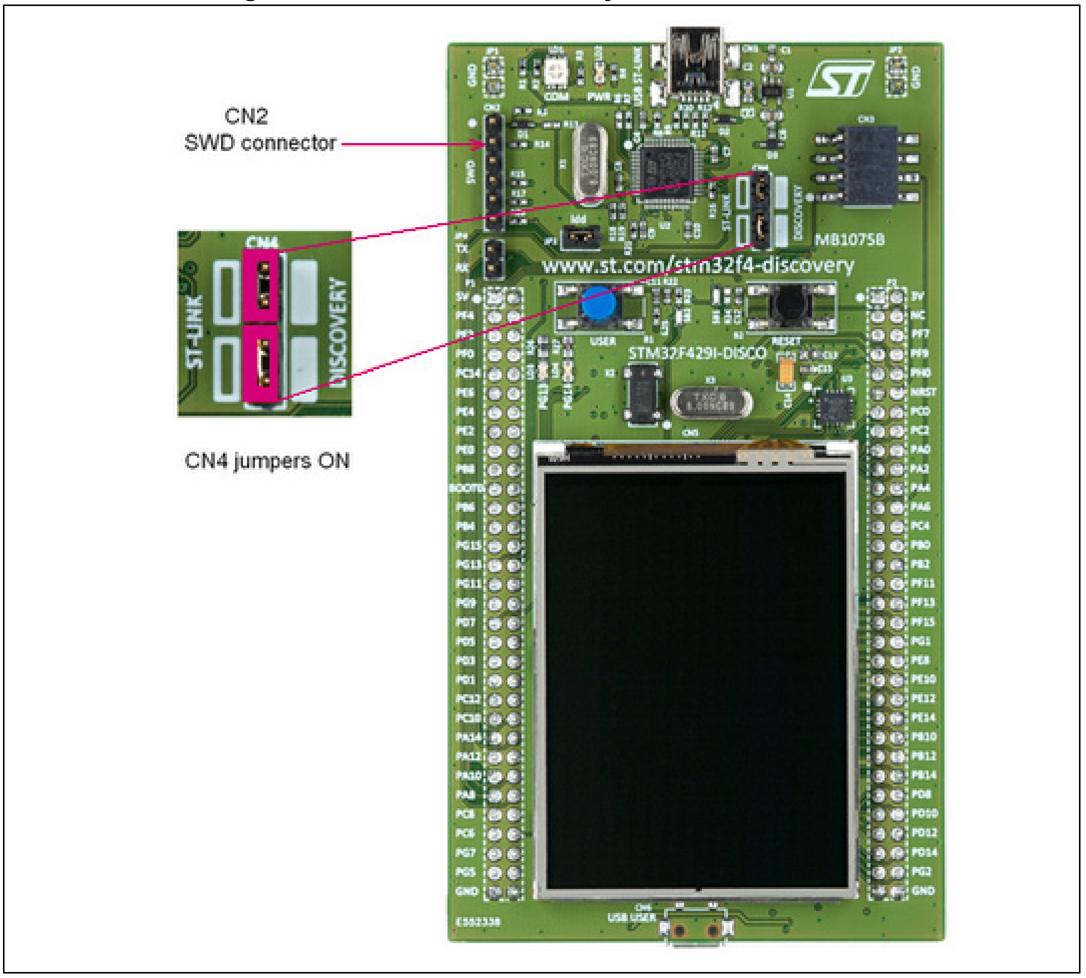
The ST-LINK/V2-B on STM32F429I-DISC1 supports Virtual COM port (VCP) on U2 pin 12 (ST-LINK_TX) and U2 pin 13 (ST-LINK_RX), which are connected to the STM32F429 target STM32 USART1 (PA9, PA10) for Mbed support, thanks to the SB11 and SB15 solder bridges.

The SB11 (PA9) and SB15 (PA10) default configurations for 32F429IDISCOVERY and STM32F429I-DISC1 are given in *Table 6: Solder bridges*.

6.3.4 Using ST-LINK/V2-B to program/debug the STM32F429ZIT6 on board

To program the STM32F429ZIT6 on board, simply plug in the two jumpers on CN4, as shown marked in red in *Figure 7*, but do not use the CN2 connector as it could disturb the communication with the STM32F429ZIT6 of the STM32F429 Discovery board.

Figure 7. STM32F429 Discovery board connections



6.3.5 Using ST-LINK/V2-B to program/debug an external STM32 application

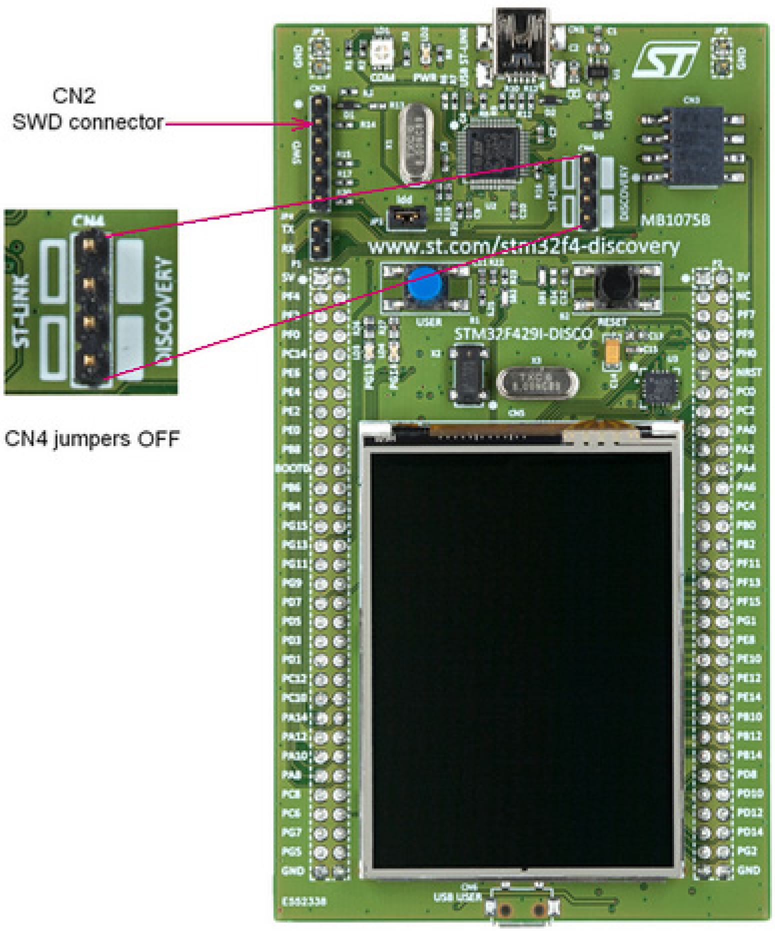
It is very easy to use the ST-LINK/V2-B to program the STM32 on an external application. Simply remove the two jumpers from CN4 as shown in *Figure 8* and connect the application to the CN2 debug connector according to *Table 5*.

Note: SB7 must be OFF if CN2 pin 5 is used in an external application.

Table 5. Debug connector CN2 (SWD)

Pin	CN2	Designation
1	VDD_TARGET	VDD from application
2	SWCLK	SWD clock
3	GND	Ground
4	SWDIO	SWD data input/output
5	NRST	RESET of target STM32
6	SWO	Reserved

Figure 8. ST-LINK/V2-B connections



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.

Two LEDs are dedicated to this module:

- LD5 (green LED) indicates when VBUS is active
- LD6 (red LED) indicates an overcurrent from a connected device

Note: The USB OTG host operation is realized once the Discovery is connected to the PC via the ST-LINK cable (mini USB). Once the USB OTG host device is detected and identified, then the USB ST-LINK cable can be removed.

6.8 Gyroscope MEMS (ST-MEMS I3G4250D)

The I3G4250D is a low-power, three-axis angular rate sensor. It includes a sensing element and an IC interface able to provide the measured angular rate to the external world through the I^2C/SPI serial interface.

The I3G4250D has a full-scale of ±245/±500/±2000 dps and is capable of measuring rates with a user-selectable bandwidth.

The STM32F429ZIT6 controls this motion sensor through the SPI interface.

6.9 TFT LCD (Thin-film-transistor liquid-crystal display)

The TFT LCD is a 2.4" display of 262 K colors. Its definition is QVGA (240 x 320 dots) and is directly driven by the STM32F429ZIT6 using the RGB protocol. It includes the ILI9341 LCD controller and can operate with a typical 2.8 V voltage.

6.10 64-Mbit SDRAM (1Mbit x 16-bit x 4-bank)

The 64-Mbit SDRAM is a high-speed CMOS, dynamic random-access memory designed to operate in 3.3 V memory systems containing 67,108,864 bits. It is internally configured as a quad-bank DRAM with a synchronous interface. Each 16,777,216-bit bank is organized as 4,096 rows by 256 columns by 16 bits. The 64-Mbit SDRAM includes auto-refresh, power-saving, and power-down modes. All signals are registered on the positive edge of the clock signal, CLK.

The STM32F429ZIT6 reads and writes data at 80 MHz.

6.11 JP3 (Idd)

Jumper JP3, labeled Idd, allows the consumption of STM32F429ZIT6 to be measured by removing the jumper and connecting an ammeter.

- Jumper ON: STM32F429ZIT6 is powered (default).
- Jumper OFF: an ammeter must be connected to measure the STM32F429ZIT6 current, (if there is no ammeter, the STM32F429ZIT6 is not powered).

6.12 OSC clock

6.12.1 OSC clock supply

The following information indicates all configurations for clock supply selection.

- **MCO from ST-LINK** (from MCO of the STM32F429ZIT6)
  This frequency cannot be changed, it is fixed at 8 MHz and connected to PH0-OSC_IN of the STM32F429ZIT6. The configuration needed is:
  - SB18 closed, SB19 open, R56 removed
  - SB20, R57, C20, C21, X3 = do not care
- **Oscillator on board** (from X3 crystal)
  For typical frequencies and its capacitors and resistors, refer to the STM32F429ZIT6 datasheet. The configuration needed is:
  - SB18, SB19, SB20 open
  - R56, R57, C20, C21, X3 soldered
- **Oscillator from external PH0** (from external oscillator through pin 10 of the P2 connector)
  The configuration needed is:
  - SB19 closed, SB18 open, R56 removed
  - SB20, R57, C20, C21, X3 = do not care
- **No external oscillato**r (from internal oscillator HSI only).
  PH0 and PH1 can be used as GPIO. The configuration needed is:
  - SB18 open, SB19 closed, SB20 closed, R56 removed, R57 removed
  - C20, C21, X3 = do not care

6.12.2 OSC 32 kHz clock supply

The following information indicates all configurations for the 32 kHz clock supply selection.

- **Oscillator on board** (from X2 Crystal, not provided).
  The configuration needed is:
  - SB16 open, SB17 open
  - R53, R54, C23, C24, X2 soldered
- **Oscillator from external PC14** (from external oscillator through pin 9 of P1 connector)
  The configuration needed is:
  - SB16 closed, R53 removed
  - SB17, R54, C23, C24, X2 = do not care
- **No external oscillator** (PC14 and PC15 can be used as GPIO.
  The configuration needed is:
  - SB16 closed, SB17 closed, R53 removed, R54 removed
  - C23, C24, X2 = do not care

6.13 Solder bridges

Table 6. Solder bridges

Bridge	State (1)	Description
SB19, 20 (X3 crystal)	OFF	X3, C20, C21, R56, and R57 provide a clock. PH0 and PH1 are disconnected from P2.
	ON	PH0 and PH1 are connected to P2. Remove only R56 and R57.
SB4, 6, 8,14 (default)	ON	Reserved, do not modify.
SB3, 5, 7,13 (reserved)	OFF	Reserved, do not modify.
SB22, 23, 24, 25	OFF	Reserved, do not modify.
SB16, 17 (X2 crystal)	OFF	X2, C23, C24, R53, and R54 deliver a 32 kHz clock. PC14 and PC15 are not connected to P2
	ON	PC14 and PC15 are only connected to P2. Remove only R53 and R54.
SB1 (B2-RESET)	**ON**	B2 push-button is connected to the NRST of STM32F429ZIT6.
	OFF	B2 push-button is not connected to the NRST of STM32F429ZIT6.
SB2 (B1-USER)	**ON**	B1 push-button is connected to PA0.
	OFF	B1 push-button is not connected to PA0.
SB11, 15 (RX,TX)	OFF	STM32F429 USART1 is not connected to ST-LINK, so VCP is disabled (Default configuration on 32F429IDISCOVERY).
	ON	STM32F429 USART1 is connected to ST-LINK, so VCP is enabled (default configuration on STM32F429I-DISC1).
SB12 (NRST)	**ON**	NRST signal of connector CN2 is connected to the NRST of STM32F429ZIT6.
	OFF	NRST signal is not connected.
SB9 (SWO)	**OFF**	SWO signal is not connected.
	ON	SWO signal of connector CN2 is connected to PB3.
SB10 (STM_RST)	**OFF**	No incidence on the NRST signal of STM32F429ZIT6.
	ON	NRST signal of STM32F429ZIT6 is connected to GND.
SB21 (BOOT0)	**ON**	BOOT0 signal of STM32F429ZIT6 is at level LOW through 510 Ω pull-down.
	OFF	BOOT0 signal of STM32F429ZIT6 is at level HIGH through 10 KΩ pull-up (not provided).
SB26, 27 (USB OTG)	**OFF**	PB14 and PB15 are only used for USB OTG and not connected to P2 to avoid noise.
	ON	PB14 and PB15 are connected to P2.
SB18 (MCO)	**OFF**	MCO signal of STM32F429ZIT6 is not used.
	ON	MCO clock signal from STM32F429ZIT6 is connected to OSC_IN of STM32F429ZIT6.

1. The default SBx state is shown in bold.

6.14 Extension connectors

The male headers P1 and P2 can connect the STM32F429 Discovery board to a standard prototyping/wrapping board. STM32F429ZIT6 GPIOs are available on these connectors. P1 and P2 can also be probed by an oscilloscope, a logic analyzer, or a voltmeter.

Table 7. STM32 pin description versus board functions

STM32 pin		Board functions																		
Main function	**LQFP144**	**System**	**VCP**	**SDRAM**	**LCD-TFT**	**LCD-RGB**	**LCD-SPI**	**I3G4250D**	**USB**	**LED**	**Push-button**	**I^{2}C Ext**	**Touch panel**	**Free I/O**	**Power supply**	**CN2**	**CN3**	**CN6**	**P1**	**P2**
BOOT0	138	BOOT0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	21	-
NRST	25	NRST	-	-	RESET	RESET	RESET	-	-	-	B2	-	-	-	-	5	-	-	-	12
PA0	34	-	-	-	-	-	-	-	-	-	B1	-	-	-	-	-	-	-	-	18
PA1	35	-	-	-	-	-	-	INT1	-	-	-	-	-	-	-	-	-	-	-	17
PA2	36	-	-	-	-	-	-	INT2	-	-	-	-	-	-	-	-	-	-	-	20
PA3	37	-	-	-	DB3	B5	-	-	-	-	-	-	-	-	-	-	-	-	-	19
PA4	40	-	-	-	VSYNC	VSYNC	-	-	-	-	-	-	-	-	-	-	-	-	-	22
PA5	41	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	21
PA6	42	-	-	-	DB6	G2	-	-	-	-	-	-	-	-	-	-	-	-	-	24
PA7	43	-	-	-	-	-	-	-	-	-	-	I2C_EXT_RST	-	-	-	-	4	-	-	23
PA8	100	-	-	-	-	-	-	-	-	-	-	SCL	SCL	-	-	-	3	-	53	-
PA9	101	-	USART1_TX	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	52	-
PA10	102	-	USART1_RX	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	51	-

Table 7. STM32 pin description versus board functions (continued)

STM32 pin		Board functions																		
Main function	**LQFP144**	**System**	**VCP**	**SDRAM**	**LCD-TFT**	**LCD-RGB**	**LCD-SPI**	**I3G4250D**	**USB**	**LED**	**Push-button**	**I^2C Ext**	**Touch panel**	**Free I/O**	**Power supply**	**CN2**	**CN3**	**CN6**	**P1**	**P2**
PA11	103	-	-	-	DB14	R4	-	-	-	-	-	-	-	-	-	-	-	-	50	-
PA12	104	-	-	-	DB15	R5	-	-	-	-	-	-	-	-	-	-	-	-	49	-
PA13	105	SWDIO	-		-	-	-	-	-	-	-	-	-	-	-	4	-	-	48	-
PA14	109	SWCLK	-	-	-	-	-	-	-	-	-	-	-	-	-	2	-	-	47	-
PA15	110	-	-	-	-	-	-	-	-	-	-	-	INT	-	-	-	-	-	46	-
PB0	46	-	-	-	DB13	R3	-	-	-	-	-	-	-	-	-	-	-	-	-	28
PB1	47	-	-	-	DB16	R6	-	-	-	-	-	-	-	-	-	-	-	-	-	27
PB2	48	BOOT1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	30
PB3	133	SWO	-	-	-	-	-	-	-	-	-	-	-	-	-	6	-	-	28	-
PB4	134	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	25	-
PB5	135	-	-	SDCKE1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	26	-
PB6	136	-	-	SDNE1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	23	-
PB7	137	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	24	-
PB8	139	-	-	-	DB4	B6	-	-	-	-	-	-	-	-	-	-	-	-	19	-
PB9	140	-	-	-	DB5	B7	-	-	-	-	-	-	-	-	-	-	-	-	20	-
PB10	69	-	-	-	DB8	G4	-	-	-	-	-	-	-	-	-	-	-	-	-	48
PB11	70	-	-	-	DB9	G5	-	-	-	-	-	-	-	-	-	-	-	-	-	47
PB12	73	-	-	-	-	-	-	-	ID	-	-	-	-	-	-	-	-	4	-	50

Table 7. STM32 pin description versus board functions (continued)

STM32 pin		Board functions																		
Main function	LQFP144	System	VCP	SDRAM	LCD-TFT	LCD-RGB	LCD-SPI	I3G4250D	USB	LED	Push-button	I^2C Ext	Touch panel	Free I/O	Power supply	CN2	CN3	CN6	P1	P2
PB13	74	-	-	-	-	-	-	-	VBUS	Green	-	-	-	-	-	-	-	1	-	49
PB14	75	-	-	-	-	-	-	-	DM	-	-	-	-	-	-	-	-	2	-	52 (1)
PB15	76	-	-	-	-	-	-	-	DP	-	-	-	-	-	-	-	-	3	-	51 (2)
PC0	26	-	-	SDNWE	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	14
PC1	27	-	-	-	-	-	-	CS	-	-	-	-	-	-	-	-	-	-	-	13
PC2	28	-	-	-	CSX	CSX	CSX	-	-	-	-	-	-	-	-	-	-	-	-	16
PC3	29	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	15
PC4	44	-	-	-	-	-	-	-	PSO	-	-	-	-	-	-	-	-	-	-	26
PC5	45	-	-	-	-	-	-	-	QC	Red	-	-	-	-	-	-	-	-	-	25
PC6	96	-	-	-	HSYNC	HSYNC	-	-	-	-	-	-	-	-	-	-	-	-	57	-
PC7	97	-	-	-	DB10	G6	-	-	-	-	-	-	-	-	-	-	-	-	56	-
PC8	98	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	55	-
PC9	99	-	-	-	-	-	-	-	-	-	-	SDA	SDA	-	-	-	1	-	54	-
PC10	111	-	-	-	DB12	R2	-	-	-	-	-	-	-	-	-	-	-	-	45	-
PC11	112	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	44	-
PC12	113	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	43	-
PC13	7		-		-	-	-	-	-	-	-	-	-	-	-	-	-	-	12	-
PC14	8	OSC32_IN	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	9	-

Table 7. STM32 pin description versus board functions (continued)

STM32 pin		Board functions																		
Main function	**LQFP144**	**System**	**VCP**	**SDRAM**	**LCD-TFT**	**LCD-RGB**	**LCD-SPI**	**I3G4250D**	**USB**	**LED**	**Push-button**	**I^2C Ext**	**Touch panel**	**Free I/O**	**Power supply**	**CN2**	**CN3**	**CN6**	**P1**	**P2**
PC15	9	OSC32_OUT	-		-	-	-	-	-	-	-	-	-	-	-	-	-	-	10	-
PD0	114	-	-	D2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	42	-
PD1	115	-	-	D3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	41	-
PD2	116	-	-		-	-	-	-	-	-	-	-	-	-	-	-	-	-	40	-
PD3	117	-	-		DB11	G7	-	-	-	-	-	-	-	-	-	-	-	-	39	-
PD4	118	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	38	-
PD5	119	-	-		-	-	-	-	-	-	-	-	-	-	-	-	-	-	37	-
PD6	122	-	-		DB0	B2	-	-	-	-	-	-	-	-	-	-	-	-	36	-
PD7	123	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	35	-
PD8	77	-	-	D13	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	54
PD9	78	-	-	D14	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	53
PD10	79	-	-	D15	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	56
PD11	80	-	-		TE	-	-	-	-	-	-	-	-	-	-	-	-	-	-	55
PD12	81	-	-		RDX	-	-	-	-	-	-	-	-	-	-	-	-	-	-	58
PD13	82	-	-		WRX	-	DCX	-	-	-	-	-	-	-	-	-	-	-	-	57
PD14	85	-	-	D0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	60
PD15	86	-	-	D1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	59
PE0	141	-	-	NBL0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	17	-
PE1	142	-	-	NBL1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	18	-
PE2	1	-	-		-	-	-	-	-	-	-	-	-	-	-	-	-	-	15	-
PE3	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	16	-
PE4	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	13	-
PE5	4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	14	-

Table 7. STM32 pin description versus board functions (continued)

STM32 pin		Board functions																		
Main function	LQFP144	System	VCP	SDRAM	LCD-TFT	LCD-RGB	LCD-SPI	I3G4250D	USB	LED	Push-button	I^{2}C Ext	Touch panel	Free I/O	Power supply	CN2	CN3	CN6	P1	P2
PE6	5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	11	-
PE7	58	-	-	D4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	37
PE8	59	-	-	D5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	40
PE9	60	-	-	D6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	39
PE10	63	-	-	D7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	42
PE11	64	-	-	D8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	41
PE12	65	-	-	D9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	44
PE13	66	-	-	D10	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	43
PE14	67	-	-	D11	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	46
PE15	68	-	-	D12	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	45
PF0	10	-	-	A0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	7	-
PF1	11	-	-	A1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	8	-
PF2	12	-	-	A2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	5	-
PF3	13	-	-	A3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	6	-
PF4	14	-	-	A4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3	-
PF5	15	-	-	A5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	4	-
PF6	18	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3
PF7	19	-	-	-	DCX	-	SCL	SCK	-	-	-	-	-	-	-	-	-	-	-	6
PF8	20	-	-	-	-	-	-	MISO	-	-	-	-	-	-	-	-	-	-	-	5
PF9	21	-	-	-	SDA	-	SDI/SDO	MOSI	-	-	-	-	-	-	-	-	-	-	-	8
PF10	22	-	-	-	ENABLE	DE	-	-	-	-	-	-	-	-	-	-	-	-	-	7
PF11	49	-	-	SDNRAS	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	32
PF12	50	-	-	A6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	31
PF13	53	-	-	A7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	34

Table 7. STM32 pin description versus board functions (continued)

STM32 pin		Board functions																		
Main function	LQFP144	System	VCP	SDRAM	LCD-TFT	LCD-RGB	LCD-SPI	I3G4250D	USB	LED	Push-button	I^2C Ext	Touch panel	Free I/O	Power supply	CN2	CN3	CN6	P1	P2
PF14	54	-	-	A8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	33
PF15	55	-	-	A9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	36
PG0	56	-	-	A10	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	35
PG1	57	-	-	A11	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	38
PG2	87	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	62
PG3	88	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	61
PG4	89	-	-	BA0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	62	-
PG5	90	-	-	BA1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	61	-
PG6	91	-	-	-	DB17	R7	-	-	-	-	-	-	-	-	-	-	-	-	60	-
PG7	92	-	-	-	DOTLCK	CLK	-	-	-	-	-	-	-	-	-	-	-	-	59	-
PG8	93	-	-	SDCLK	-	-	-	-	-	-	-	-	-	-	-	-	-	-	58	-
PG9	124	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	33	-
PG10	125	-	-	-	DB7	G3	-	-	-	-	-	-	-	-	-	-	-	-	34	-
PG11	126	-	-	-	DB1	B3	-	-	-	-	-	-	-	-	-	-	-	-	31	-
PG12	127	-	-	-	DB2	B4	-	-	-	-	-	-	-	-	-	-	-	-	32	-
PG13	128	-	-	-	-	-	-	-	-	Green	-	-	-	-	-	-	-	-	29	-
PG14	129	-	-	-	-	-	-	-	-	Red	-	-	-	-	-	-	-	-	30	-
PG15	132	-	-	SDNCAS	-	-	-	-	-	-	-	-	-	-	-	-	-	-	27	-
PH0	23	OSC_IN	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	10

Table 7. STM32 pin description versus board functions (continued)

STM32 pin		Board functions																		
Main function	LQFP144	System	VCP	SDRAM	LCD-TFT	LCD-RGB	LCD-SPI	I3G4250D	USB	LED	Push-button	I2C Ext	Touch panel	Free I/O	Power supply	CN2	CN3	CN6	P1	P2
PH1	24	OSC_OUT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	9
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	VDD	-	-	-	22	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3 V	-	5	-	-	1
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3 V	-	-	-	-	2
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	5 V	-	8	-	1	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	5 V	-	-	-	2	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	GND	3	7	5	63	11
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	GND	-	-	-	64	29
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	GND	-	-	-	-	63
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	GND	-	-	-	-	-

1. If SB27 is ON.
2. If SB26 is ON.

7 32F429IDISCOVERY information

7.1 Product marking

The sticker located on the top or bottom side of the PCB board shows the information about product identification such as board reference, revision, and serial number.

The first identification line has the following format: “MBxxxx-Variant-yzz”, where “MBxxxx” is the board reference, “Variant” (optional) identifies the mounting variant when several exist, "y" is the PCB revision and "zz" is the assembly revision: for example B01.

The second identification line is the board serial number used for traceability.

Evaluation tools marked as “ES” or “E” are not yet qualified and therefore not ready to be used as reference design or in production. Any consequences deriving from such usage will not be at ST charge. In no event, ST will be liable for any customer usage of these engineering sample tools as reference designs or in production.

“E” or “ES” marking examples of location:

- On the target STM32 that is soldered on the board (for illustration of STM32 marking, refer to the STM32 datasheet “Package information” paragraph at the *www.st.com* website).
- Next to the evaluation tool ordering part number that is stuck or silk-screen printed on the board.

The board reference for the 32F429IDISCOVERY base board is MB1075.

7.2 Board revision history

7.2.1 MB1075

Revision B01

The revision B-01 of the MB1075 board is the initial official release.

Revision C01

The revision C-01 of the MB1075 board removes the limitations of the revision B-01.

1. STM32F103C8T6 replaced by STM32F103CBT6 for ST-LINK/V2-B
2. Mbed Enabled™, SB11 and SB15 closed

Revision E01

The revision E-01 of the MB1075 board removes the limitations of the revision C-01.

1. U3: L3GD20 replaced by I3G4250D and R75, C54 added
2. Old STMPE811QTR kept and SX8651 added for two footprints, and U9, C55, and R74 added for SX8651, and original STMPE811QTR circuit kept
3. CN5 SF-TC240T-9370-T replaced by FRD240C48003-B: pin1 and pin3 for YU and YD swapped, pin8 connected with 3V
4. Updated solder bridges:
   – ON: SB3, SB5, SB7, SB13, SB19
   – OFF: SB4, SB6, SB8, SB14, SB18
5. For BOM for SX8651: as U9 SX8651IWLTRT is soldered, added R29, R30, and R45 4.7 KΩ resistors
6. Updated CN6 footprint
7. Line added on top silkscreen for CN5 when FPC is mounted

7.3 Known limitations

7.3.1 MB1075-F429

Revision B01

ST-LINK update from ST-LINK/V2 to ST-LINK/V2-B

Revision C01

EOL request for L3GD20 and STMPE811QTR

8 Revision history

Table 8. Document revision history

Date	Revision	Changes
10-Sep-2013	1	Initial release.
04-Mar-2016	2	New revision to introduce STM32F429I-DISC1 additional CPN that corresponds to Mbed-Enabled Discovery Kit. Updated: *Section : Introduction*, *Figure 1: STM32F429 Discovery board*, *Section 5: Quick start*, *Section 6: Hardware layout*, *Figure 2: Hardware block diagram*, *Figure 3: Top layout*, *Section 6.3: Embedded ST-LINK/V2 (or V2-B)*, *Table 5: Solder bridges*, *Table 6: STM32 pin description versus board functions*, *Section 7: Electrical schematics*.
20-Sep-2017	3	Updated *Section 6.9: TFT LCD (Thin-film-transistor liquid-crystal display)*.
02-Jul-2020	4	Reshuffle of document to align with latest standards: – *Introduction* to *Quick start* reordering – *Electrical schematics* removed – New *Table 2: Codification explanation* and *Section 7: 32F429IDISCOVERY information* Updated: – *Figure 2*, *Figure 3*, *Figure 4*, *Figure 5*, *Figure 7*, and *Figure 8*. – *Table 4: Jumper states* – *Section 6.3.4: Using ST-LINK/V2-B to program/debug the STM32F429ZIT6 on board* and *Section 6.3.5: Using ST-LINK/V2-B to program/debug an external STM32 application* for connector CN2, and *Table 6: Solder bridges* for connector CN3. – *Section 6.7: USB OTG supported* – Gyroscope ST-MEMS I3G4250D.
26-Aug-2020	5	Updated: – Clarification on protection diodes in *Section 6.4* Added: – Note on USB OTG host device management in *Section 6.7*