Getting started with X-CUBE-IPS industrial digital output software for STM32 Nucleo

Introduction

With the X-CUBE-IPS software package you can easily access the features of the ICs hosted in the below expansion boards for STM32 Nucleo:

- 0.5 A current rating with X-NUCLEO-OUT07A1 hosting IPS4260LM
- 0.6 A current rating with X-NUCLEO-DO40A1 hosting IPS4140HQ
- 0.7 A current rating with X-NUCLEO-OUT01A2, X-NUCLEO-OUT09A1, X-NUCLEO-OUT10A1, X-NUCLEO-OUT11A1, X-NUCLEO-OUT12A1, X-NUCLEO-OUT16A1, X-NUCLEO-OUT02A1, hosting respectively ISO8200BQ, IPS8160HQ, IPS161HF, ISO808, ISO808A, IPS8200HQ and ISO8200AQ
- 1.0 A current rating with X-NUCLEO-OUT13A1, X-NUCLEO-OUT14A1, X-NUCLEO-OUT19A1, X-NUCLEO-OUT17A1, X-NUCLEO-DO41A1, hosting respectively ISO808-1, ISO808A-1, IPS8160HQ-1, IPS8200HQ-1 and IPS4140HQ-1
- 2.5 A current rating with X-NUCLEO-OUT03A1 (hosting the IPS2050H), X-NUCLEO-OUT05A1 (hosting the IPS1025H), X-NUCLEO-OUT08A1 (hosting the IPS160HF), or X-NUCLEO-OUT15A1 (hosting the IPS1025HF)
- 5.0 A current rating with X-NUCLEO-DOL10A1 (hosting the IPS1050LQ)
- 5.7 A current rating with X-NUCLEO-OUT04A1 or X-NUCLEO-OUT06A1, hosting respectively the IPS2050H-32 and the IPS1025H-32

The expansion is built on STM32Cube software technology to ease portability across different STM32 microcontrollers.

The software comes with sample implementations for each expansion board supported in the package, for both NUCLEO-F401RE and NUCLEO-G431RB development boards.

Related links

Visit the STM32Cube ecosystem web page on www.st.com for further information

1 Acronyms and abbreviations

Table 1. List of acronyms

Acronym	Description
API	Application programming interface
BSP	Board support package
CMSIS	Cortex® microcontroller software interface standard
HAL	Hardware abstraction layer
IDE	Integrated development environment
LED	Light emitting diode
SPI	Serial peripheral interface

2 X-CUBE-IPS software expansion for STM32Cube

2.1 Overview

The X-CUBE-IPS software package expands the STM32Cube functionality.

The package key features are:

- Software package to build applications for high efficiency high-side and low-side switches:
  - octal (high-side): ISO8200BQ, ISO8200AQ,IPS8160HQ, IPS8160HQ-1, ISO808, ISO808-1, ISO808A, ISO808A-1, IPS8200HQ and IPS8200HQ-1
  - quad (high-side): IPS4140HQ and IPS4140HQ-1
  - quad (low-side): IPS4260LM
  - dual (high-side): IPS2050H and IPS2050H-32
  - single (high-side): IPS160HF, IPS161HF, IPS1025H, IPS1025H-32, and IPS1025HF
  - single (low-side): IPS1050LQ
- GPIOs, PWMs, and IRQs
- Fault/diagnostics interrupt handling
- Sample implementation available on the following expansion boards, when connected to a NUCLEO-F401RE or NUCLEO-G431RB development board:
  - X-NUCLEO-OUT01A2
  - X-NUCLEO-OUT02A1
  - X-NUCLEO-OUT03A1
  - X-NUCLEO-OUT04A1
  - X-NUCLEO-OUT05A1
  - X-NUCLEO-OUT06A1
  - X-NUCLEO-OUT07A1
  - X-NUCLEO-OUT08A1
  - X-NUCLEO-OUT09A1
  - X-NUCLEO-OUT10A1
  - X-NUCLEO-OUT11A1
  - X-NUCLEO-OUT12A1
  - X-NUCLEO-OUT13A1
  - X-NUCLEO-OUT14A1
  - X-NUCLEO-OUT15A1
  - X-NUCLEO-OUT16A1
  - X-NUCLEO-OUT17A1
  - X-NUCLEO-OUT19A1
  - X-NUCLEO-DO40A1
  - X-NUCLEO-DO41A1
  - X-NUCLEO-DOL10A1
- Easy portability across different MCU families, thanks to STM32Cube
- Compatible with STM32CubeMX, can be downloaded from st.com and installed directly into STM32CubeMX
- Free, user-friendly license terms

This software allows controlling the digital output of a single expansion board, or a properly configured stack of these expansion boards mounted upon a NUCLEO-F401RE or NUCLEO-G431RB development board.

It also allows you to program the expansion boards to be switched on and off using PWM with a specific frequency in the 0-100 Hz range (0.1 Hz resolution), and specific duty cycle in the 0-100% range (1% resolution).

The package includes an example to test the device functionality while driving the channels in the steady state and PWM.

2.2 Architecture

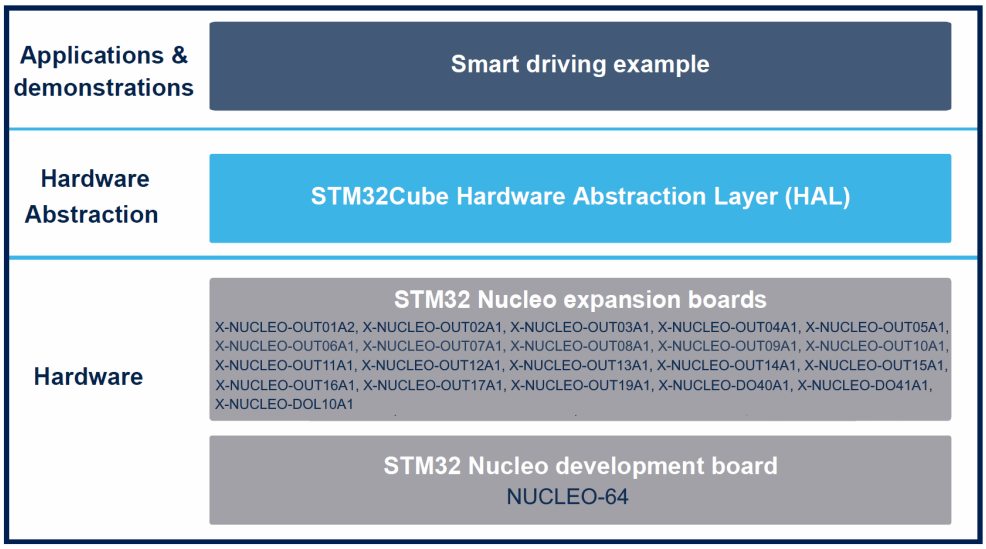
This software is a fully compliant expansion of STM32Cube architecture for the development of applications for high efficiency (octal, dual and single) high-side and (quad) low-side intelligent power switch (IPS) digital output modules.

The software is based on the STM32CubeHAL hardware abstraction layer for the STM32 microcontroller. The package extends STM32Cube by providing a board support package (BSP) for the digital output expansion boards based on the devices listed in Section 2.1: Overview.

The software layers used by the application software to access and use the industrial digital output expansion boards are:

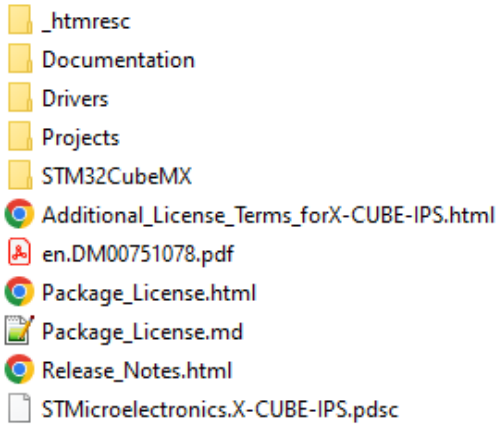
- **STM32Cube HAL layer**: consists of simple, generic and multi-instance APIs (application programming interfaces) which interact with the upper layer applications, libraries and stacks. These generic and extension APIs are based on a common framework so that overlying layers like middleware can function without requiring specific microcontroller unit (MCU) hardware information. This structure improves library code reusability and guarantees easy portability across other devices.
- **Board support package (BSP) layer**: provides software support for the STM32 Nucleo board peripherals, excluding the MCU. These specific APIs provide a programming interface for certain board specific peripherals like LEDs, user buttons, etc., and can also be used to fetch individual board version information. It also provides support for initializing, configuring and reading data.

Figure 1. X-CUBE-IPS expansion software architecture



2.3 Folder structure

Figure 2. **X-CUBE-IPS package folder structure**



The following folders are included in the software package:

- **htmresc** contains graphics for html pages
- **Documentation** contains a compiled HTML file generated from the source code, detailing the software components and APIs.
- **Drivers** contains:
  - STM32Cube HAL subfolders, specifically STM32G4xx_HAL_Driver and STM32F4xx_HAL_Driver. These files are not specific for the X-CUBE-IPS software but come directly from the STM32Cube framework and represent the hardware abstraction layer code for the STM32 MCUs.
  - a **CMSIS** folder, which contains the Cortex® microcontroller software interface standard files from Arm. These files are vendor-independent hardware abstraction layers for the Cortex-M processor series. This folder comes also unchanged from the STM32Cube framework.
  - a BSP folder containing the code required for the configuration of the expansion boards listed in Section 2.1: Overview, the drivers for the IC listed in Section 2.1: Overview, and the switch API functions.
- **Projects** contains sample applications for all supported IPS products, provided for NUCLEO-F401RE and NUCLEO-G431RB platforms.
- **STM32CubeMX** contains SW pack configuration files and all FTL template files needed for STM32CubeMX pack described by STMicroelectronics.X-CUBE-IPS.pdsc file.

2.3.1 BSPs

For the X-CUBE-IPS software, different BSPs are used:

- STM32F4xx-Nucleo, STM32G4xx_Nucleo
- iso8200bq
- iso8200aq
- ips2050h
- ips2050h_32
- ips1025h
- ips1025h_32
- ips4260l
- ips160hf
- ips8160hq
- ips161hf
- iso808
- iso808a
- iso808_1
- iso808a_1
- ips1025hf
- ips8200hq
- ips8200hq_1
- ips8160hq_1
- ips4140hq
- ips4140hq_1
- ips1050lq
- OUT01A2
- OUT02A1
- OUT03A1
- OUT04A1
- OUT05A1
- OUT06A1
- OUT07A1
- OUT08A1
- OUT09A1
- OUT10A1
- OUT11A1
- OUT12A1
- OUT13A1
- OUT14A1
- OUT15A1
- OUT16A1
- OUT17A1
- OUT19A1
- DO40A1
- DO41A1
- DOL10A1

2.3.1.1 STM32F4xx-Nucleo, STM32G4xx_Nucleo

Depending on the STM32 Nucleo development board used, these BSPs provide an interface to configure and use the development board peripherals with the expansion boards listed in Section 2.1: Overview.

Each folder (STM32F4xx-Nucleo, STM32G4xx_Nucleo) contains couples of .c/.h files (stm32[code]xx_nucleo.c/.h, where [code] is the MCU family code F4 or G4), which come from the STM32Cube framework without modification. They provide the functions to handle the user button and LEDs of the corresponding development board.

2.3.1.2 *ips1025h*

The ips1025h BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips1025h.

This folder contains:

- **ips1025h.c**: core functions of the IPS1025H driver
- **ips1025h.h**: declaration of the IPS1025H driver functions and their associated definitions

2.3.1.3 *ips1025h_32*

The ips1025h_32 BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips1025h_32.

This folder contains:

- **ips1025h_32.c**: core functions of the IPS1025H-32 driver
- **ips1025h_32.h**: declaration of the IPS1025H-32 driver functions and their associated definitions

2.3.1.4 *ips2050h*

The ips2050h BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips2050h.

This folder contains:

- **ips2050h.c**: core functions of the IPS2050H driver
- **ips2050h.h**: declaration of the IPS2050H driver functions and their associated definitions

2.3.1.5 *ips2050h_32*

The ips2050h_32 BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips2050h_32.

This folder contains:

- **ips2050h_32.c**: core functions of the IPS2050H-32 driver
- **ips2050h_32.h**: declaration of the IPS2050H-32 driver functions and their associated definitions

2.3.1.6 *ips1025hf*

The ips1025hf BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips1025hf.

This folder contains:

- **ips1025hf.c**: core functions of the IPS1025HF driver
- **ips1025hf.h**: declaration of the IPS1025HF driver functions and their associated definitions

2.3.1.7 *ips8160hq*

The ips8160hq BSP component provides the driver functions for the STMicroelectronics intelligent power switch device in the folder Drivers\BSP\Components\ips8160hq.

This folder contains:

- **ips8160hq.c**: core functions of the IPS8160HQ driver
- **ips8160hq.h**: declaration of the IPS8160HQ driver functions and their associated definitions

2.3.1.8 *ips8160hq_1*

The ips8160hq_1 BSP component provides the driver functions for the STMicroelectronics intelligent power switch device in the folder Drivers\BSP\Components\ips8160hq_1.

This folder contains:

- **ips8160hq_1.c**: core functions of the IPS8160HQ-1 driver

- **ips8160hq_1.h**: declaration of the IPS8160HQ-1 driver functions and their associated definitions

2.3.1.9 *ips160hf*

The ips160hf BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips160hf.

This folder contains:

- **ips160hf**: core functions of the IPS160HF driver
- **ips160hf**: declaration of the IPS160HF driver functions and their associated definitions

2.3.1.10 *ips161hf*

The ips161hf BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips161hf.

This folder contains:

- **ips161hf.c**: core functions of the IPS161HF driver
- **ips161hf.h**: declaration of the IPS161HF driver functions and their associated definitions

2.3.1.11 *iso808*

The iso808 BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\iso808.

This folder contains:

- **iso808.c**: core functions of the ISO808 driver
- **iso808.h**: declaration of the ISO808 driver functions and their associated definitions

2.3.1.12 *iso808_1*

The iso808_1 BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\iso808_1.

This folder contains:

- **iso808_1.c**: core functions of the ISO808-1 driver
- **iso808_1.h**: declaration of the ISO808-1 driver functions and their associated definitions

2.3.1.13 *iso808a*

The iso808a BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\iso808a.

This folder contains:

- **iso808a.c**: core functions of the ISO808A driver
- **iso808a.h**: declaration of the ISO808A driver functions and their associated definitions

2.3.1.14 *iso808a_1*

The iso808a_1 BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\iso808a_1.

This folder contains:

- **iso808a_1.c**: core functions of the ISO808A-1 drivers
- **iso808a_1.h**: declaration of the ISO808A-1 driver functions and their associated definitions

2.3.1.15 *iso8200bq*

The iso8200bq BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\iso8200bq.

This folder contains:

- **iso8200bq.c**: core functions of the ISO8200BQ driver
- **iso8200bq.h**: declaration of the ISO8200BQ driver functions and their associated definitions

2.3.1.16 *ips8200hq*

The ips8200hq BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips8200hq.

This folder contains:

- **ips8200hq.c**: core functions of the IPS8200HQ driver
- **ips8200hq.h**: declaration of the IPS8200HQ driver functions and their associated definitions

2.3.1.17 *ips8200hq_1*

The ips8200hq_1 BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips8200hq_1.

This folder contains:

- **ips8200hq_1.c**: core functions of the IPS8200HQ-1 driver
- **ips8200hq_1.h**: declaration of the IPS8200HQ-1 driver functions and their associated definitions

2.3.1.18 *ips4260l*

The ips4260l BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips4260l.

This folder contains:

- **ips4260l.c**: core functions of the IPS4260LM driver
- **ips4260l.h**: declaration of the IPS4260LM driver functions and their associated definitions

2.3.1.19 *iso8200aq*

The iso8200aq BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\iso8200aq.

This folder contains:

- **iso8200aq.c**: core functions of the ISO8200AQ driver
- **iso8200aq.h**: declaration of the ISO8200AQ driver functions and their associated definitions

2.3.1.20 *ips4140hq*

The ips4140hq BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips4140hq.

This folder contains:

- **ips4140hq.c**: core functions of the IPS4140HQ driver
- **ips4140hq.h**: declaration of the IPS4140HQ driver functions and their associated definitions

2.3.1.21 *ips4140hq_1*

The ips4140hq_1 BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips4140hq_1.

This folder contains:

- **ips4140hq_1.c**: core functions of the IPS4140HQ-1 driver
- **ips4140hq_1.h**: declaration of the IPS4140HQ-1 driver functions and their associated definitions

2.3.1.22 *ips1050lq*

The ips1050lq BSP component provides the driver functions for the STMicroelectronics intelligent power switch devices in the folder Drivers\BSP\Components\ips1050lq.

This folder contains:

- **ips1050lq.c**: core functions of the IPS1050LQ driver
- **ips1050lq.h**: declaration of the IPS1050LQ driver functions and their associated definitions

2.3.1.23 *OUT08A1 and OUT10A1*

The OUT08A1 and OUT10A1 BSP components contain board support package files for the X-NUCLEO-OUT08A1 and X-NUCLEO-OUT10A1 expansion boards respectively. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins.

Through these functions, the channel can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.24 *OUT03A1, OUT04A1, OUT05A1 and OUT06A1*

These BSP components contain board support package files for the X-NUCLEO-OUT03A1, X-NUCLEO-OUT04A1, X-NUCLEO-OUT05A1, X-NUCLEO-OUT06A1 expansion boards respectively and they are dedicated to the functions necessary to drive the power switches in the steady-state and in PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.25 *OUT09A1 and OUT19A1*

The OUT09A1 and OUT19A1 BSP components contain board support package files for the X-NUCLEO-OUT09A1 and X-NUCLEO-OUT19A1 expansion boards respectively. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.26 *OUT11A1 and OUT13A1*

The OUT11A1 and OUT13A1 BSP components contain board support package files for the X-NUCLEO-OUT11A1 and X-NUCLEO-OUT13A1 expansion boards respectively. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, Direct Control Mode or Synchronous Control Mode can be managed, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.27 *OUT12A1 and OUT14A1*

The OUT12A1 and OUT14A1 BSP components contain board support package files for the X-NUCLEO-OUT12A1 and X-NUCLEO-OUT14A1 expansion boards respectively. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs and SPI peripheral.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, using the SPI interface, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.28 *OUT15A1*

The OUT15A1 BSP component contains board support package files for the X-NUCLEO-OUT15A1 expansion board. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, the channel can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.29 *OUT01A2*

The OUT01A2 BSP component contains board support package files for the X-NUCLEO-OUT01A2 expansion board. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.30 *OUT16A1 and OUT17A1*

The OUT16A1 and OUT17A1 BSP components contain board support package files for the X-NUCLEO-OUT16A1 and X-NUCLEO-OUT17A1 expansion boards respectively. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs or SPI peripheral.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, using a parallel or serial (SPI) interface, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.31 *OUT07A1*

The OUT07A1 BSP components contain board support package files for the X-NUCLEO-OUT07A1 expansion board. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, using a parallel interface, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle

2.3.1.32 *OUT02A1*

The OUT02A1 BSP component contains board support package files for the X-NUCLEO-OUT02A1 expansion board. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs and the SPI peripheral.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.33 *DO40A1 and DO41A1*

The DO40A1 and DO41A1 BSP components contain board support package files for the X-NUCLEO-DO40A1 and X-NUCLEO-DO41A1 expansion boards respectively. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.1.34 *DOL10A1*

The DOL10A1 BSP component contains board support package files for the X-NUCLEO-DOL10A1 expansion board. These files are dedicated to the functions necessary to drive the power switches in the steady-state and in the PWM mode using the GPIOs.

The files are also used to obtain the status of the diagnostics and output feedback pins. Through these functions, one or more channels can be set, reset, or configured in the PWM mode with a specific frequency and duty cycle.

2.3.2 Projects

For each STM32 Nucleo platform, one example project is available in the folders:

- Projects\NUCLEO-F401RE\Examples\OUT01A2\EightChannels
- Projects\NUCLEO-F401RE\Examples\OUT02A1\DaisyChain
- Projects\NUCLEO-F401RE\Examples\OUT02A1\Regular_8_Channels
- Projects\NUCLEO-F401RE\Examples\OUT02A1\Regular_16_Channels
- Projects\NUCLEO-F401RE\Examples\OUT03A1\DefaultBoard
- Projects\NUCLEO-F401RE\Examples\OUT03A1\FourBoards
- Projects\NUCLEO-F401RE\Examples\OUT04A1\DefaultBoard
- Projects\NUCLEO-F401RE\Examples\OUT04A1\FourBoards
- Projects\NUCLEO-F401RE\Examples\OUT05A1\DefaultBoard
- Projects\NUCLEO-F401RE\Examples\OUT05A1\FourBoards
- Projects\NUCLEO-F401RE\Examples\OUT06A1\DefaultBoard
- Projects\NUCLEO-F401RE\Examples\OUT06A1\FourBoards
- Projects\NUCLEO-F401RE\Examples\OUT07A1\FourChannels
- Projects\NUCLEO-F401RE\Examples\OUT08A1\DefaultBoard
- Projects\NUCLEO-F401RE\Examples\OUT08A1\FourBoards
- Projects\NUCLEO-F401RE\Examples\OUT09A1\EightChannels
- Projects\NUCLEO-F401RE\Examples\OUT10A1\DefaultBoard
- Projects\NUCLEO-F401RE\Examples\OUT10A1\FourBoards
- Projects\NUCLEO-F401RE\Examples\OUT11A1\DirectMode
- Projects\NUCLEO-F401RE\Examples\OUT11A1\SynchronousMode
- Projects\NUCLEO-F401RE\Examples\OUT12A1\DaisyChain
- Projects\NUCLEO-F401RE\Examples\OUT12A1\Regular_8_Channels
- Projects\NUCLEO-F401RE\Examples\OUT12A1\Regular_16_Channels
- Projects\NUCLEO-F401RE\Examples\OUT13A1\DirectMode
- Projects\NUCLEO-F401RE\Examples\OUT13A1\SynchronousMode
- Projects\NUCLEO-F401RE\Examples\OUT14A1\DaisyChain
- Projects\NUCLEO-F401RE\Examples\OUT14A1\Regular_8_Channels
- Projects\NUCLEO-F401RE\Examples\OUT14A1\Regular_16_Channels
- Projects\NUCLEO-F401RE\Examples\OUT15A1\DefaultBoard
- Projects\NUCLEO-F401RE\Examples\OUT15A1\TwoBoards
- Projects\NUCLEO-F401RE\Examples\OUT16A1\DaisyChain
- Projects\NUCLEO-F401RE\Examples\OUT16A1\SPI_8_Channels
- Projects\NUCLEO-F401RE\Examples\OUT16A1\Parallel_8_Channels
- Projects\NUCLEO-F401RE\Examples\OUT17A1\DaisyChain
- Projects\NUCLEO-F401RE\Examples\OUT17A1\SPI_8_Channels
- Projects\NUCLEO-F401RE\Examples\OUT17A1\Parallel_8_Channels
- Projects\NUCLEO-F401RE\Examples\OUT19A1\EightChannels
- Projects\NUCLEO-F401RE\Examples\DO40A1\FourChannels
- Projects\NUCLEO-F401RE\Examples\DO41A1\FourChannels
- Projects\NUCLEO-F401RE\Examples\DOL10A1\OneChannel_LS
- Projects\NUCLEO-G431RB\Examples\OUT01A2\EightChannels
- Projects\NUCLEO-G431RB\Examples\OUT02A1\DaisyChain
- Projects\NUCLEO-G431RB\Examples\OUT02A1\Regular_8_Channels
- Projects\NUCLEO-G431RB\Examples\OUT02A1\Regular_16_Channels
- Projects\NUCLEO-G431RB\Examples\OUT03A1\DefaultBoard
- Projects\NUCLEO-G431RB\Examples\OUT03A1\FourBoards
- Projects\NUCLEO-G431RB\Examples\OUT04A1\DefaultBoard
- Projects\NUCLEO-G431RB\Examples\OUT04A1\FourBoards

- Projects\NUCLEO-G431RB\Examples\OUT05A1\DefaultBoard
- Projects\NUCLEO-G431RB\Examples\OUT05A1\FourBoards
- Projects\NUCLEO-G431RB\Examples\OUT06A1\DefaultBoard
- Projects\NUCLEO-G431RB\Examples\OUT06A1\FourBoards
- Projects\NUCLEO-G431RB\Examples\OUT07A1\FourChannels
- Projects\NUCLEO-G431RB\Examples\OUT08A1\DefaultBoard
- Projects\NUCLEO-G431RB\Examples\OUT08A1\FourBoards
- Projects\NUCLEO-G431RB\Examples\OUT09A1\EightChannels
- Projects\NUCLEO-G431RB\Examples\OUT10A1\DefaultBoard
- Projects\NUCLEO-G431RB\Examples\OUT10A1\FourBoards
- Projects\NUCLEO-G431RB\Examples\OUT11A1\DirectMode
- Projects\NUCLEO-G431RB\Examples\OUT11A1\SynchronousMode
- Projects\NUCLEO-G431RB\Examples\OUT12A1\DaisyChain
- Projects\NUCLEO-G431RB\Examples\OUT12A1\Regular_8_Channels
- Projects\NUCLEO-G431RB\Examples\OUT12A1\Regular_16_Channels
- Projects\NUCLEO-G431RB\Examples\OUT13A1\DirectMode
- Projects\NUCLEO-G431RB\Examples\OUT13A1\SynchronousMode
- Projects\NUCLEO-G431RB\Examples\OUT14A1\DaisyChain
- Projects\NUCLEO-G431RB\Examples\OUT14A1\Regular_8_Channels
- Projects\NUCLEO-G431RB\Examples\OUT14A1\Regular_16_Channels
- Projects\NUCLEO-G431RB\Examples\OUT15A1\DefaultBoard
- Projects\NUCLEO-G431RB\Examples\OUT15A1\TwoBoards
- Projects\NUCLEO-G431RB\Examples\OUT16A1\DaisyChain
- Projects\NUCLEO-G431RB\Examples\OUT16A1\SPI_8_Channels
- Projects\NUCLEO-G431RB\Examples\OUT16A1\Parallel_8_Channels
- Projects\NUCLEO-G431RB\Examples\OUT17A1\DaisyChain
- Projects\NUCLEO-G431RB\Examples\OUT17A1\SPI_8_Channels
- Projects\NUCLEO-G431RB\Examples\OUT17A1\Parallel_8_Channels
- Projects\NUCLEO-G431RB\Examples\OUT19A1\EightChannels
- Projects\NUCLEO-G431RB\Examples\DO40A1\FourChannels
- Projects\NUCLEO-G431RB\Examples\DO41A1\FourChannels
- Projects\NUCLEO-G431RB\Examples\DOL10A1\OneChannel_LS

Each example has a folder dedicated to the targeted IDE:

- **EWARM** contains the project files for IAR
- **MDK-ARM** contains the project files for Keil
- **STM32CubeIDE** contains the project files for STM32CubeIDE

Each folder in the previous list contains a file readme.html which contains several details about the related project examples and specifically the list of source files provided.

2.4 Software required resources

2.4.1 X-NUCLEO-OUT01A2

The MCU controls IPS8200BQ via GPIOs.

Thus, when using one X-NUCLEO-OUT01A2 expansion board, and using the proper example project called EightChannels, eight GPIO signals (IN1 to IN8) plus one GPIO dedicated to the interrupt management (STATUS) are required. The software also uses a PWM timer to generate the periodic patterns on the output channels for the expansion boards.

It is also possible to evaluate a combination of expansion boards stacked through the Arduino connectors. In this case, the expansion boards must be properly configured to avoid any conflict between signals. The X-NUCLEO-OUT01A2 offer some flexibility to remap default signals into alternate positions. See their related schematic diagrams. No example projects are provided for this alternative use, customers have to adapt the EightChannels project code to their specific needs.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.2 X-NUCLEO-OUT02A1

The MCU controls ISO8200AQ via SPI interface and GPIOs.

Thus, when using one X-NUCLEO-OUT02A1 expansion board, and using the proper example project called **Regular_8_Channels**, one SPI peripheral (ISO_CLK, uC_MISO, uC_MOSI signals), one GPIO (ISO_SS) used as device select, one GPIO (ISO_OUT_EN) used to enable output lines and two GPIOs dedicated to the interrupt management (ISO_FAULT and ISO_PGOOD pins) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channel for the expansion board.

It is also possible to evaluate a 16-channels digital output module by stacking two X-NUCLEO-OUT02A1 with shared or independent supply rail and independent loads.

This can be achieved in two different ways:

1. Configuring two independent stacked boards to get an 8+8 channels system.
   In this case, the two boards must be properly configured: the first one (board 0) can be left in default configuration, for the second one (board 1) it is necessary to unsolder some resistors from the default positions and solder them in different positions according to the scheme described below.

Table 2. X-NUCLEO-OUT02A1 - Configuration of a stack of two independent expansion boards

Board no.	ISO_CLK	uC_MISO	uC_MOSI	ISO_SS	ISO_OUT_EN	ISO_FAULT	ISO_PGOOD
Board 0	R1	R2	R3	R4	R6	R7	R5
Board 1	R1	R2	R3	R113	R111	R107	R112

*Important: To enable this configuration the example project to be used is the one available in Examples\OUT02A1**Regular_16_Channels**.*

2. Configuring two stacked boards using Daisy Chain feature to get a 16 channels system.
   In this case, the two boards must be properly configured: for the first one (board 0) and the second one (board 1) it is necessary to unsolder some resistors from the default positions and solder them in different positions according to the scheme described below.

Table 3. X-NUCLEO-OUT02A1 - Configuration of a stack of two expansion boards (Daisy Chain)

Board no.	ISO_CLK	ISO_SDIO1	uC_MISO	uC_MOSI	ISO_SS	ISO_OUT_EN	ISO_FAULT	ISO_PGOOD
Board 0	R1	R105	--	R3	R4	R6	R7	R5
Board 1	R1	R105	R2	--	R4	R6	R107	R112

Important: *To enable this configuration the example project to be used is the one available in Examples\OUT02A1**DaisyChain**.*

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.3 X-NUCLEO-OUT03A1, X-NUCLEO-OUT04A1

The MCU controls IPS2050H and IPS2050H-32 via GPIOs.

Thus, when using one X-NUCLEO-OUT03A1 expansion board or one X-NUCLEO-OUT04A1 expansion board, and using the proper example project called DefaultBoard, two GPIO signals (IN1 and IN2 pins) plus two GPIOs dedicated to the interrupt management (FLT1, FLT2 pins) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channels for the expansion boards.

It is also possible to evaluate an eight-channel digital output module by stacking up to four X-NUCLEO-OUT03A1 and\or X-NUCLEO-OUT04A1 with shared or independent supply rail and independent loads.

In this case, the example project called FourBoards must be used and the additional expansion boards must be properly configured. For the second, third, or fourth board, it is necessary to unsolder four resistors for each board from the default position and solder them in different positions related to the board number, following the scheme described below.

Table 4. X-NUCLEO-OUT03A1, X-NUCLEO-OUT04A1 - Configuration of a stack of four expansion boards

Board no.	IN1	IN2	FLT1	FLT2
Board 0	R101	R102	R103	R104
Board 1	R131	R132	R133	R134
Board 2	R111	R112	R113	R114
Board 3	R121	R122	R123	R124

Important: *When using board 2 and board 3, two jumpers must close the morpho connectors pins in the STM32 Nucleo development board:*

- *CN7.35-36 closed*
- *CN10.25-26 closed*

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.4 X-NUCLEO-OUT05A1, X-NUCLEO-OUT06A1

The MCU controls IPS1025H and IPS1025H-32 via GPIOs.

Thus, when using one X-NUCLEO-OUT05A1 expansion board or one X-NUCLEO-OUT06A1 expansion board, and using the proper example project called DefaultBoard, one GPIO signal (IN1) plus two GPIOs dedicated to the interrupt management (FLT1, FLT2 pins) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channels for the expansion boards.

It is also possible to evaluate a four-channel digital output module by stacking up to four X-NUCLEO-OUT05A1 and\or X-NUCLEO-OUT06A1 with shared or independent supply rail and independent loads.

In this case, the example project called FourBoards must be used and the additional expansion boards must be properly configured. For the second, third, or fourth board, it is necessary to unsolder three resistors for each board from the default position and solder them in different positions related to the board number, following the scheme described below.

Table 5. X-NUCLEO-OUT05A1, X-NUCLEO-OUT06A1 - Configuration of a stack of four expansion boards

Board no.	IN1	FLT1	FLT2
Board 0	R101	R103	R114
Board 1	R102	R104	R117
Board 2	R115	R116	R107
Board 3	R120	R119	R118

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.5 X-NUCLEO-OUT07A1

The MCU controls IPS4260LM via GPIOs.

Thus, when using one X-NUCLEO-OUT07A1 expansion board, and using the example project called **FourChannels**, four GPIO signals (IN1 to IN4) plus two GPIO dedicated to the interrupt management (FLT_L and OL_L) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channels for the expansion boards.

It is also possible to evaluate a combination of expansion boards stacked through the Arduino connectors. In this case, the expansion boards must be properly configured to avoid any conflict between signals. The X-NUCLEO-OUT07A1 offers some flexibility to remap default signals into alternate positions. See its related schematic diagrams. No example projects are provided for this alternative use, customers have to adapt the FourChannels project code to their specific needs.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.6 X-NUCLEO-OUT08A1, X-NUCLEO-OUT10A1

The MCU controls IPS160HF and IPS161HF via GPIOs.

Thus, when using one X-NUCLEO-OUT08A1 or X-NUCLEO-OUT10A1 expansion board, and using the proper example project called DefaultBoard, three GPIO signals (IN1, Nch-Drv, OUT_FB pins) plus a GPIO dedicated to the interrupt management (DIAG pin) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channel for the expansion board.

It is also possible to evaluate a quad-channel digital output module by stacking four X-NUCLEO-OUT08A1 or four X-NUCLEO-OUT10A1, or a mix of them, with a shared or independent supply rail and independent loads.

In this case, the example project called FourBoards must be used and the additional expansion boards must be properly configured. For the second, third, and fourth board, it is necessary to unsolder four resistors from the default position and solder them in different positions, following the scheme described below.

Table 6. X-NUCLEO-OUT08A1, X-NUCLEO-OUT10A1 - Configuration of a stack of four expansion boards

Board no.	IN1	DIAG	Nch-DRV	OUT_FB
Board 0	R101	R103	R102	R104
Board 1	R111	R112	R124	R131
Board 2	R121	R125	R130	R123
Board 3	R132	R133	R134	R122

Important: *When using board 1 and board 3, two jumpers must close the morpho connectors pins in the STM32 Nucleo development board:*

- *CN7.35-36 closed*
- *CN10.25-26 closed*

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.7 X-NUCLEO-OUT09A1, X-NUCLEO-OUT19A1

The MCU controls IPS8160HQ and IPS8160HQ-1 via GPIOs.

Thus, when using one X-NUCLEO-OUT09A1 expansion board or one X-NUCLEO-OUT19A1 expansion board, and using the example project called EightChannels, eight GPIO signals (IN1 to IN8) plus one GPIO dedicated to the interrupt management (STATUS) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channels for the expansion boards.

It is also possible to evaluate a combination of expansion boards stacked through the Arduino connectors. In this case, the expansion boards must be properly configured to avoid any conflict between signals. The X-NUCLEO-OUT09A1 and X-NUCLEO-OUT19A1 offer some flexibility to remap default signals into alternate positions. See their related schematic diagrams. No example projects are provided for this alternative use, customers have to adapt the EightChannels project code to their specific needs.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.8 X-NUCLEO-OUT11A1, X-NUCLEO-OUT13A1

The MCU controls ISO808 and ISO808-1 via GPIOs.

Thus, when using one X-NUCLEO-OUT11A1 expansion board or one X-NUCLEO-OUT13A1 expansion board, eight GPIO signals (IN1 to IN8), two GPIOs (LOAD and SYNCH) used to control the device operating mode (*Synchronous Control Mode* or *Direct Control Mode*), one GPIO (OUT_EN) used to enable output lines and one GPIO dedicated to the interrupt management (STATUS pin) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channel for the expansion boards.

To enable *Synchronous Control Mode* the example project to be used is the one available in Examples\OUT11A1\SynchronousMode or Examples\OUT13A1\SynchronousMode respectively.

To enable *Direct Control Mode* the example project to be used is the one available in Examples\OUT11A1\DirectMode or Examples\OUT13A1\DirectMode respectively.

It is also possible to evaluate a combination of expansion boards stacked through the Arduino connectors. In this case, the expansion boards must be properly configured to avoid any conflict between signals. The X-NUCLEO-OUT11A1 and X-NUCLEO-OUT13A1 offer some flexibility to remap default signals into alternate positions. See their related schematic diagrams. No example projects are provided for this alternative use, customers have to adapt the code of the provided projects to their specific needs.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.9 X-NUCLEO-OUT12A1, X-NUCLEO-OUT14A1

The MCU controls ISO808A and ISO808A-1 via SPI interface and GPIOs.

Thus, when using one X-NUCLEO-OUT12A1 expansion board or one X-NUCLEO-OUT14A1 expansion board, and using the proper example project called Regular_8_Channels, one SPI peripheral (SPI_CLK, SPI_MISO, SPI_MOSI signals), one GPIO (SPI_SS) used as device select, one GPIO (OUT_EN) used to enable output lines and two GPIOs dedicated to the interrupt management (STATUS and PGOOD pins) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channel for the expansion board.

It is also possible to evaluate a 16-channels digital output module by stacking two X-NUCLEO-OUT12A1 and\or X-NUCLEO-OUT14A1 with shared or independent supply rail and independent loads.

This can be achieved in two different ways:

1. Configuring two independent stacked boards to get an 8+8 channels system.
   In this case, the two boards must be properly configured: the first one (board 0) can be left in default configuration, for the second one (board 1) it is necessary to unsolder some resistors from the default positions and solder them in different positions according to the scheme described below.

Table 7. X-NUCLEO-OUT12A1, X-NUCLEO-OUT14A1 - Configuration of a stack of two independent expansion boards

Board no.	SPI_CLK	SPI_MISO	SPI_MOSI	SPI_SS	OUT_EN	STATUS	PGOOD
Board 0	R106	R105	R104	R103	R119	R108	R107
Board 1	R106	R105	R104	R114	R109	R113	R111

Important: To enable this configuration the example project to be used is the one available in Examples\OUT12A1\Regular_16_Channels or Examples\OUT14A1\Regular_16_Channels respectively.

2. Configuring two stacked boards using Daisy Chain feature to get a 16 channels system.
   In this case, the two boards must be properly configured: for the first one (board 0) and the second one (board 1) it is necessary to unsolder some resistors from the default positions and solder them in different positions according to the scheme described below.

Table 8. X-NUCLEO-OUT12A1, X-NUCLEO-OUT14A1 - Configuration of a stack of two expansion boards (Daisy Chain)

Board no.	SPI_CLK	DaisyChain	SPI_MISO	SPI_MOSI	SPI_SS	OUT_EN	STATUS	PGOOD
Board 0	R106	R102	--	R104	R103	R119	R108	R107
Board 1	R106	R102	R105	--	R103	R109	R113	R111

Important: To enable this configuration the example project to be used is the one available in Examples\OUT12A1\DaisyChain or Examples\OUT14A1\DaisyChain respectively.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.10 X-NUCLEO-OUT15A1

The MCU controls IPS1025HF via GPIOs.

Thus, when using one X-NUCLEO-OUT15A1 expansion board, three GPIO signals (IN1, Nch-Drv, OUT_FB pins) plus two GPIOs dedicated to the interrupt management (FLT1, FLT2 pins) are required.

The software also uses a PWM timer to generate the periodic patterns on the output channel for the expansion board, and using the proper example project called DefaultBoard.

It is also possible to evaluate a dual-channel digital output module by stacking two X-NUCLEO-OUT15A1 with shared or independent supply rail and independent loads.

In this case, the example project called TwoBoards must be used and the additional expansion board must be properly configured. For the second board, it is necessary to unsolder five resistors from the default position and solder them in different positions, following the scheme described below.

Table 9. X-NUCLEO-OUT15A1 - Configuration of a stack of two expansion boards

Board no.	IN1	FLT1	FLT2	Nch-DRV	OUT_FB
Board 0	R101	R103	R114	R110	R108
Board 1	R102	R104	R107	R115	R116

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.11 X-NUCLEO-OUT16A1, X-NUCLEO-OUT17A1

The MCU controls IPS8200HQ and IPS8200HQ-1 via the SPI interface and GPIOs.

Thus, when using one X-NUCLEO-OUT16A1 expansion board or one X-NUCLEO-OUT17A1 expansion board, the user can choose the proper example project between the two available: SPI_8_Channels and Parallel_8_Channels.

Selecting **SPI_8_Channels**, one SPI peripheral (SPI_CLK, SPI_MISO, SPI_MOSI signals), one GPIO (SPI_SS) used as device select, one GPIO (OUT_EN) used to enable output lines, three GPIO input lines (SEL2, SEL1, WDEN), used to read the hardware setup about the SPI/parallel interface, SPI data width flag (8 or 16 bits) and watchdog feature enable status, one GPIO (WD) used to manage the watchdog timer, and three GPIOs dedicated to the interrupt management (FAULT, PGOOD, and TWARN pins) are required.

Selecting **Parallel_8_Channels**, eight GPIO (IN1-8 signals), one GPIO input line (SEL2) used to read the hardware setup about the SPI/parallel interface, and three GPIOs dedicated to the interrupt management (FAULT, PGOOD and TWARN pins) are required.

It is also possible to evaluate a 16-channels digital output module by stacking two X-NUCLEO-OUT16A1 and\or X-NUCLEO-OUT17A1 with shared or independent supply rail and independent loads, and selecting the available project **DaisyChain**. In this case, the two boards must be configured as Daisy chain mode with jumpers and switch available on board.

In this scenario one SPI peripheral (SPI_CLK, SPI_MISO, SPI_MOSI signals), one GPIO (SPI_SS) used as device select, one GPIO (OUT_EN) used to enable output lines, three GPIO input lines (SEL2, SEL1, WDEN), used to read the hardware setup about the SPI/parallel interface, SPI data width flag (8 or 16 bits) and watchdog feature enable status, one GPIO (WD) used to manage the watchdog timer, and three GPIOs dedicated to the interrupt management (FAULT, PGOOD, and TWARN pins) are required.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder of the software package).

The software also uses a PWM timer to generate the periodic patterns on the output channel for the expansion board.

2.4.12 X-NUCLEO-DO40A1 and X-NUCLEO-DO41A1

The MCU controls IPS4140HQ or IPS4140HQ-1 via GPIOs.

Thus, when using one X-NUCLEO-DO40A1 or one X-NUCLEO-DO41A1 expansion board, and using the example project called **FourChannels**, four GPIO signals (IN1 to IN4) plus four GPIO signals (STATUS1 to STATUS4) dedicated to the interrupt management are required.

The software also uses a PWM timer to generate the periodic patterns on the output channels for the expansion boards.

It is also possible to evaluate a combination of expansion boards stacked through the Arduino connectors. In this case, the expansion boards must be properly configured to avoid any conflict between signals. The X-NUCLEO-DO40A1 and X-NUCLEO-DO41A1 offer some flexibility to remap default signals into alternate positions. See their related schematic diagrams. No example projects are provided for this alternative use, customers have to adapt the FourChannels project code to their specific needs.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.4.13 X-NUCLEO-DOL10A1

The MCU controls IPS1050LQ via GPIOs.

Thus, when using one X-NUCLEO-DOL10A1 expansion board, and using the example project called **OneChannel_LS**, four GPIO signals (IN, IPD1, IPD2, IPD3) plus one GPIO signal (OVT) dedicated to the interrupt management are required.

The software also uses a PWM timer to generate the periodic patterns on the output channels for the expansion boards.

It is also possible to evaluate a combination of expansion boards stacked through the Arduino connectors. In this case, the expansion boards must be properly configured to avoid any conflict between signals. The X-NUCLEO-DOL10A1 offers some flexibility to remap default signals into alternate positions. See its related schematic diagram. No example projects are provided for this alternative use, customers have to adapt the **OneChannel_LS** project code to their specific needs.

For further details, see the jumper configuration described in Section 3.4: Board setup and the documentation file (readme.html in the proper example project folder).

2.5 APIs

The X-CUBE-IPS software APIs are defined in:

- Drivers\BSP\OUT01A2\out01a2.h, out01a2_bus.h
- Drivers\BSP\OUT02A1\out02a1.h, out02a1_bus.h
- Drivers\BSP\OUT03A1\out03a1.h, out03a1_bus.h
- Drivers\BSP\OUT04A1\out04a1.h, out04a1_bus.h
- Drivers\BSP\OUT05A1\out05a1.h, out05a1_bus.h
- Drivers\BSP\OUT06A1\out06a1.h, out06a1_bus.h
- Drivers\BSP\OUT07A1\out07a1.h, out07a1_bus.h
- Drivers\BSP\OUT08A1\out08a1.h, out08a1_bus.h
- Drivers\BSP\OUT09A1\out09a1.h, out09a1_bus.h
- Drivers\BSP\OUT10A1\out10a1.h, out10a1_bus.h
- Drivers\BSP\OUT11A1\out11a1.h, out11a1_bus.h
- Drivers\BSP\OUT12A1\out12a1.h, out12a1_bus.h
- Drivers\BSP\OUT13A1\out13a1.h, out13a1_bus.h
- Drivers\BSP\OUT14A1\out14a1.h, out14a1_bus.h
- Drivers\BSP\OUT15A1\out15a1.h, out15a1_bus.h
- Drivers\BSP\OUT16A1\out16a1.h out16a1_bus.h
- Drivers\BSP\OUT17A1\out17a1.h out17a1_bus.h
- Drivers\BSP\OUT19A1\out19a1.h, out19a1_bus.h
- Drivers\BSP\DO40A1\do40a1.h, do40a1_bus.h
- Drivers\BSP\DO41A1\do41a1.h, do41a1_bus.h
- Drivers\BSP\DOL10A1\dol10a1.h, dol10a1_bus.h

Detailed technical information about the APIs available to the user can be found in a compiled HTML file located inside the “Documentation” folder of the software package where all the functions and parameters are described.

2.6 Sample application description

A sample application is provided in each example project folder.

This example code helps to understand how to use the APIs provided for each project and it shows how to manage device output states, driving single or multiple, if available, input channels, and how to implement a pulse width modulation with the output lines of each device.

This example code implements a simple cyclic state machine that sends, at each state, different commands to the device: to evolve from a state to the next one, a pressure of the Blue button available in the Nucleo boards is requested. After the last state the code restarts from the first one.

Detailed description of this code is available for each provided example project.

2.6.1 OUT01A2

A sample application using the X-NUCLEO-OUT01A2 expansion board with either NUCLEO-F401RE or NUCLEO-G431RB boards is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT01A2\DirectMode\readme.html
- Examples\OUT01A2\SynchronousMode\readme.html

in the proper STM32 Nucleo board folder.

2.6.2 OUT02A1

A sample application using the X-NUCLEO-OUT02A1 expansion board with either NUCLEO-F401RE or NUCLEO-G431RB boards is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT02A1\DaisyChain\readme.html
- Examples\OUT02A1\Regular_8_Channels\readme.html
- Examples\OUT02A1\Regular_16_Channels\readme.html

in the proper STM32 Nucleo board folder.

2.6.3 OUT03A1 and OUT04A1

A sample application using the X-NUCLEO-OUT03A1 or X-NUCLEO-OUT04A1 expansion boards with either a NUCLEO-F401RE or NUCLEO-G431RB development board is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT03A1\DefaultBoard\readme.html
- Examples\OUT03A1\FourBoards\readme.html
- Examples\OUT04A1\DefaultBoard\readme.html
- Examples\OUT04A1\FourBoards\readme.html

in the proper STM32 Nucleo board folder.

2.6.4 OUT05A1 and OUT06A1

A sample application using the X-NUCLEO-OUT05A1 or X-NUCLEO-OUT06A1 expansion boards with either a NUCLEO-F401RE or NUCLEO-G431RB development board is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT05A1\DefaultBoard\readme.html
- Examples\OUT05A1\FourBoards\readme.html
- Examples\OUT06A1\DefaultBoard\readme.html
- Examples\OUT06A1\FourBoards\readme.html

in the proper STM32 Nucleo board folder.

2.6.5 OUT07A1

A sample application using the X-NUCLEO-OUT07A1 expansion board with either a NUCLEO-F401RE or NUCLEO-G431RB development board is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT07A1\FourChannels\readme.html

in the proper STM32 Nucleo board folder.

2.6.6 OUT08A1 and OUT10A1

A sample application using the X-NUCLEO-OUT08A1 or X-NUCLEO-OUT10A1 expansion board with either NUCLEO-F401RE or NUCLEO-G431RB boards is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT08A1\DefaultBoard\readme.html
- Examples\OUT08A1\FourBoards\readme.html
- Examples\OUT10A1\DefaultBoard\readme.html
- Examples\OUT10A1\FourBoards\readme.html

in the proper STM32 Nucleo board folder.

2.6.7 OUT09A1 and OUT19A1

A sample application using the X-NUCLEO-OUT09A1 or X-NUCLEO-OUT19A1 expansion boards with either a NUCLEO-F401RE or NUCLEO-G431RB development board is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT09A1\EightChannels\readme.html
- Examples\OUT19A1\EightChannels\readme.html

in the proper STM32 Nucleo board folder.

2.6.8 OUT11A1 and OUT13A1

A sample application using the X-NUCLEO-OUT11A1 or X-NUCLEO-OUT13A1 expansion board with either NUCLEO-F401RE or NUCLEO-G431RB boards is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT11A1\DirectMode\readme.html
- Examples\OUT11A1\SynchronousMode\readme.html
- Examples\OUT13A1\DirectMode\readme.html
- Examples\OUT13A1\SynchronousMode\readme.html

in the proper STM32 Nucleo board folder.

2.6.9 OUT12A1 and OUT14A1

A sample application using the X-NUCLEO-OUT12A1 or X-NUCLEO-OUT14A1 expansion board with either NUCLEO-F401RE or NUCLEO-G431RB boards is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT12A1\DaisyChain\readme.html
- Examples\OUT12A1\Regular_8_Channels\readme.html
- Examples\OUT12A1\Regular_16_Channels\readme.html
- Examples\OUT14A1\DaisyChain\readme.html
- Examples\OUT14A1\Regular_8_Channels\readme.html
- Examples\OUT14A1\Regular_16_Channels\readme.html

in the proper STM32 Nucleo board folder.

2.6.10 OUT15A1

A sample application using one or two X-NUCLEO-OUT15A1 expansion boards with either a NUCLEO-F401RE or NUCLEO-G431RB development board is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT15A1\DefaultBoard\readme.html
- Examples\OUT15A1\TwoBoards\readme.html

in the proper STM32 Nucleo board folder.

2.6.11 OUT16A1 and OUT17A1

A sample application using the X-NUCLEO-OUT16A1 or X-NUCLEO-OUT17A1 expansion board with either NUCLEO-F401RE or NUCLEO-G431RB boards is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\OUT16A1\DaisyChain\readme.html
- Examples\OUT16A1\SPI_8_Channels\readme.html
- Examples\OUT16A1\Parallel_8_Channels\readme.html
- Examples\OUT17A1\DaisyChain\readme.html
- Examples\OUT17A1\SPI_8_Channels\readme.html
- Examples\OUT17A1\Parallel_8_Channels\readme.html

in the proper STM32 Nucleo board folder.

2.6.12 DO40A1 and DO41A1

A sample application using the X-NUCLEO-DO40A1 or X-NUCLEO-DO41A1 expansion boards with either a NUCLEO-F401RE or NUCLEO-G431RB development board is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\DO40A1\FourChannels\readme.html
- Examples\DO41A1\FourChannels\readme.html

2.6.13 DOL10A1

A sample application using the X-NUCLEO-DOL10A1 expansion board with either a NUCLEO-F401RE or NUCLEO-G431RB development board is provided in the “Projects” directory. Ready to be built projects are available for multiple IDEs.

A detailed description is available in:

- Examples\DOL10A1\OneChannel_LS\readme.html

in the proper STM32 Nucleo board folder.

3 System setup guide

3.1 Hardware description

3.1.1 STM32 Nucleo

STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line.

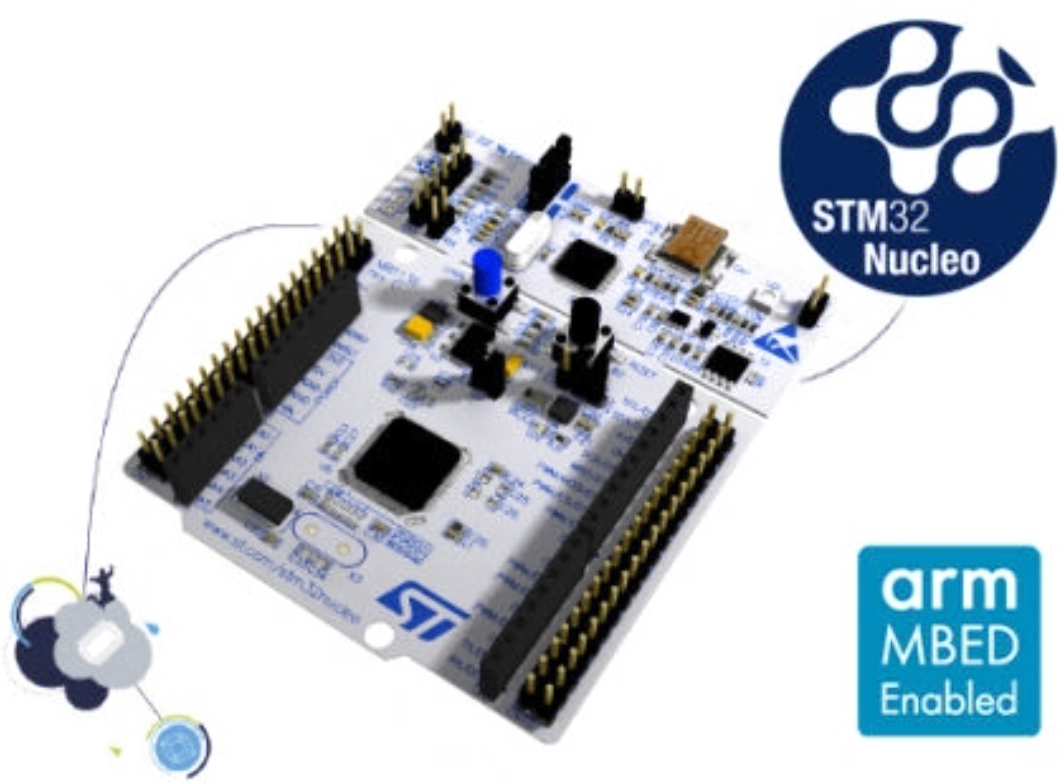
The Arduino™ connectivity support and ST morpho connectors make it easy to expand the functionality of the STM32 Nucleo open development platform with a wide range of specialized expansion boards to choose from.

The NUCLEO-F401RE development board does not require separate probes as it integrates the ST-LINK/V2-1 debugger/programmer.

The NUCLEO-G431RB development board does not require separate probes as it integrates the STLINK-V3 debugger/programmer.

The STM32 Nucleo board comes with the comprehensive STM32 software HAL library together with various packaged software examples.

Figure 3. STM32 Nucleo board



3.1.2 X-NUCLEO-OUT01A2 expansion board

The X-NUCLEO-OUT01A2 is an industrial digital output expansion board for STM32 Nucleo. It provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the ISO8200BQ octal high-side smart power solid state relay, with embedded galvanic isolation, in a digital output module connected to 0.7 A industrial loads.

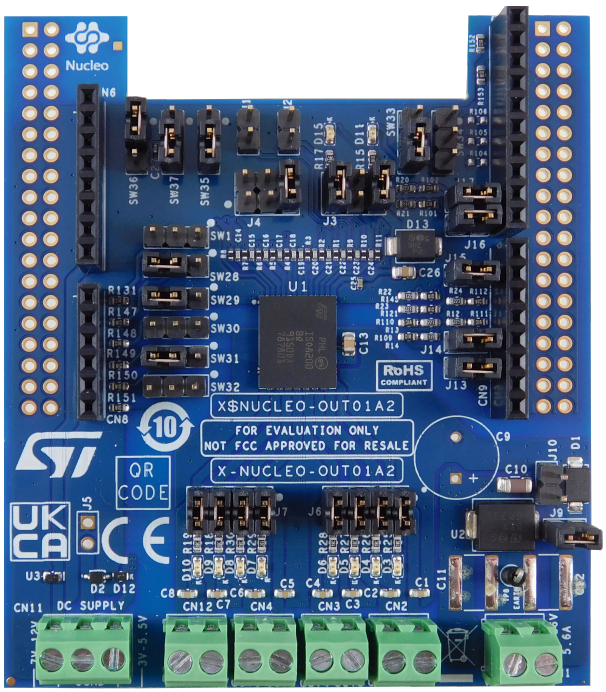
The X-NUCLEO-OUT01A2 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors.

The galvanic isolation between the microcontroller and the process stage is guaranteed by the ISO8200BQ.

The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of a X-NUCLEO-OUT01A2 stacked on other expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 4. X-NUCLEO-OUT01A2 expansion board



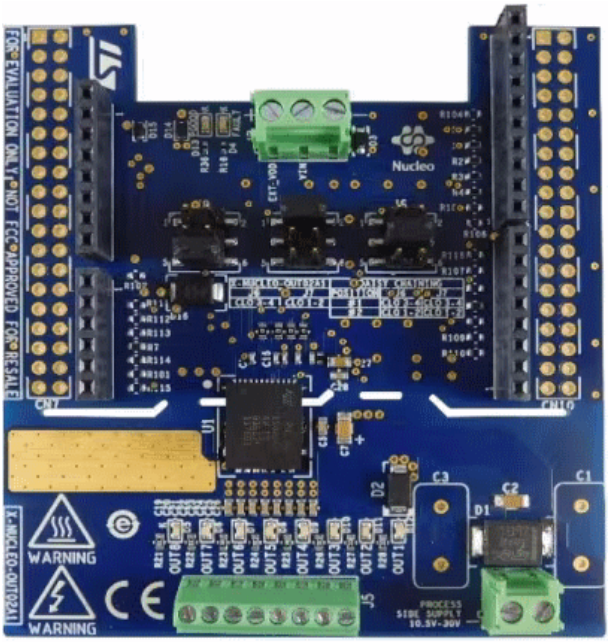
3.1.3 X-NUCLEO-OUT02A1 expansion board

The X-NUCLEO-OUT02A1 industrial digital output expansion boards for STM32-Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the ISO8200AQ octal high-side smart power solid state relay, with embedded galvanic isolation and 20MHz SPI control interface, in a digital output module connected to 0.7 A industrial loads.

The X-NUCLEO-OUT02A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors. The galvanic isolation between the microcontroller and the process stage is guaranteed by the ISO8200AQ device. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a 16-channel digital output system enabling the daisy chaining feature on two X-NUCLEO-OUT02A1 stacked expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 5. X-NUCLEO-OUT02A1 expansion board



3.1.4 X-NUCLEO-OUT03A1 expansion board

The X-NUCLEO-OUT03A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS2050H (dual high-side smart power solid state relay) in a digital output module connected to 2.5 A (max.) industrial loads.

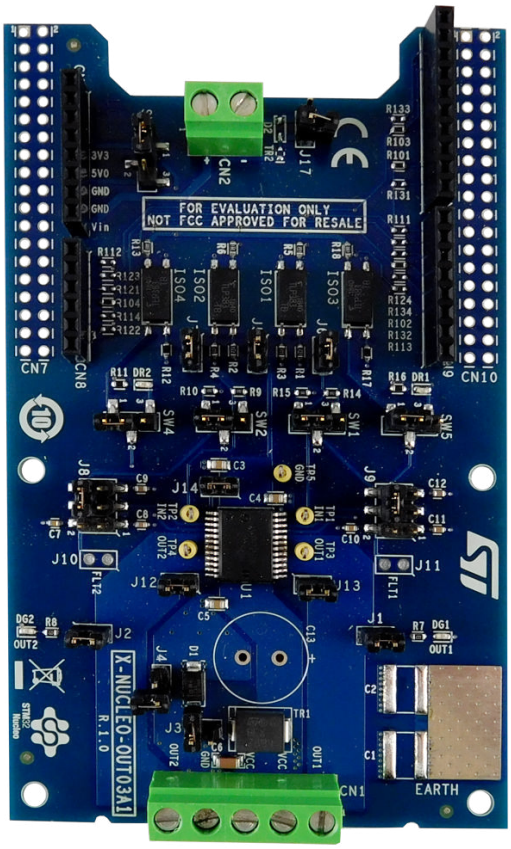
The X-NUCLEO-OUT03A1 interfaces with the microcontroller on the STM32 Nucleo via 5 kV optocouplers driven by GPIO pins, Arduino UNO R3 (default configuration), and ST morpho (optional, not mounted) connectors.

The expansion board can be connected to either a NUCLEO-F401RE or NUCLEO-G431RB development board.

It is also possible to evaluate a system composed by up to four stacked X-NUCLEO-OUT03A1 expansion boards: hardware settings described in Section 2.4 must be followed.

As an example, a system with four X-NUCLEO-OUT03A1 expansion boards allows you to evaluate an eight-channel digital output module with 2.5 A (max.) capability each.

Figure 6. X-NUCLEO-OUT03A1 expansion board



3.1.5 X-NUCLEO-OUT04A1 expansion board

The X-NUCLEO-OUT04A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS2050H-32 (dual high-side smart power solid state relay) in a digital output module connected to 5.7 A (max.) industrial loads.

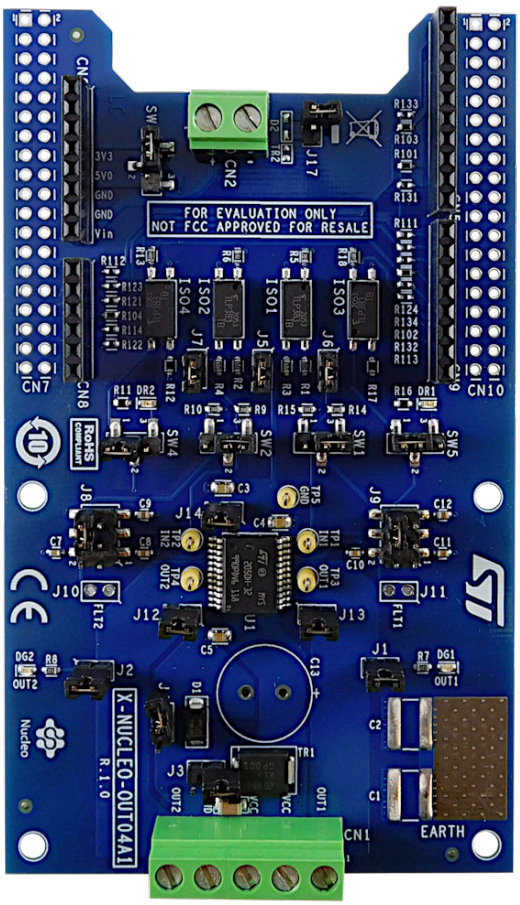
The X-NUCLEO-OUT04A1 interfaces with the microcontroller on the STM32 Nucleo via 5 kV optocouplers driven by GPIO pins, Arduino UNO R3 (default configuration) and ST morpho (optional, not mounted) connectors.

The expansion board can be connected to either a NUCLEO-F401RE or NUCLEO-G431RB development board.

It is also possible to evaluate a system composed by up to four stacked X-NUCLEO-OUT04A1 expansion boards: hardware settings described in Section 2.4 must be followed.

As an example, a system with four X-NUCLEO-OUT04A1 expansion boards allows you to evaluate an eight-channel digital output module with 5.7 A (max.) capability each.

Figure 7. X-NUCLEO-OUT04A1 expansion board



3.1.6 X-NUCLEO-OUT05A1 expansion board

The X-NUCLEO-OUT05A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS1025H single high-side smart power solid state relay, in a digital output module connected to 2.5 A industrial loads.

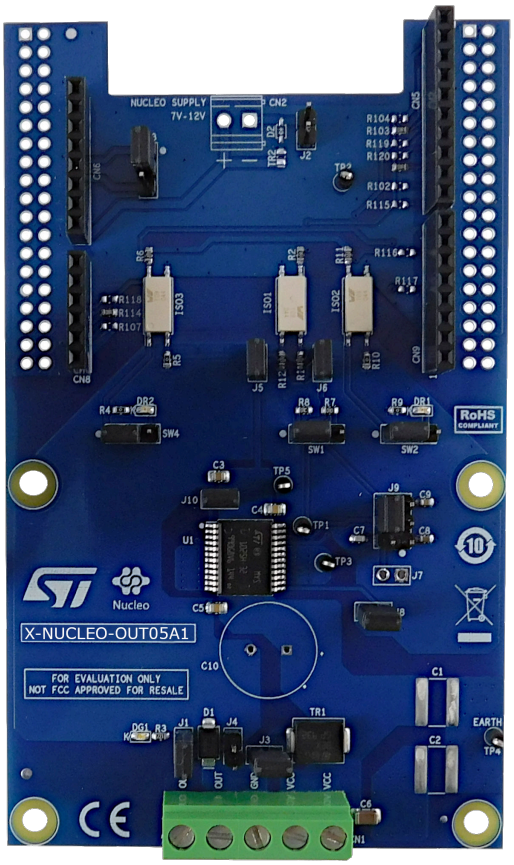
The X-NUCLEO-OUT05A1 interfaces with the microcontroller on the STM32 Nucleo via 5 kV optocouplers driven by GPIO pins and Arduino R3 connectors.

The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of up to four stacked X-NUCLEO-OUT05A1 expansion boards: hardware settings described in Section 2.4 must be followed.

As an example, a system with four X-NUCLEO-OUT05A1 expansion boards allows you to evaluate a quad channel digital output module.

Figure 8. X-NUCLEO-OUT05A1 expansion board



3.1.7 X-NUCLEO-OUT06A1 expansion board

The X-NUCLEO-OUT06A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS1025H-32 single high-side smart power solid state relay, in a digital output module connected to 5.7 A industrial loads.

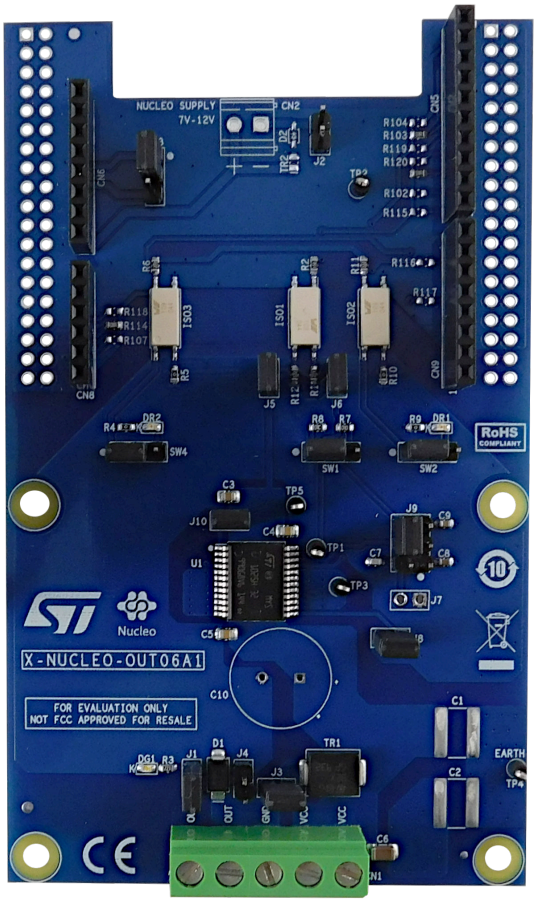
The X-NUCLEO-OUT06A1 interfaces with the microcontroller on the STM32 Nucleo via 5 kV optocouplers driven by GPIO pins and Arduino UNO R3 connectors.

The expansion board can be connected to either a NUCLEO-F401RE or NUCLEO-G431RB development board.

It is also possible to evaluate a system composed by up to four stacked X-NUCLEO-OUT06A1 expansion boards: hardware settings described in Section 2.4 must be followed.

As an example, a system with four X-NUCLEO-OUT06A1 expansion boards allows you to evaluate a quad channel digital output module.

Figure 9. X-NUCLEO-OUT06A1 expansion board



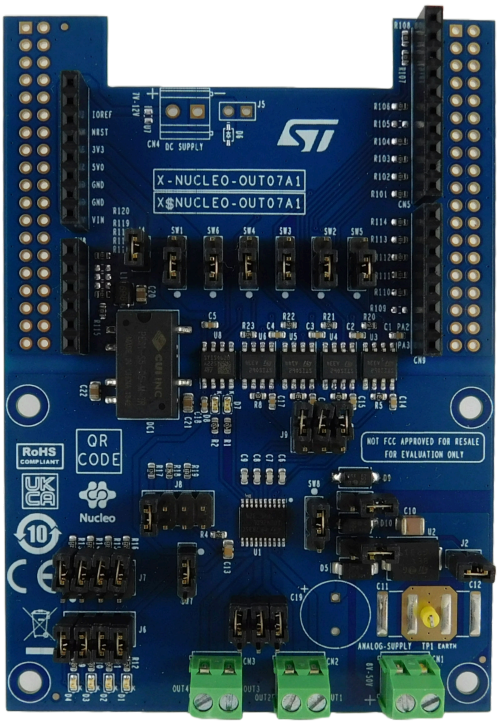
3.1.8 X-NUCLEO-OUT07A1 expansion board

The X-NUCLEO-OUT07A1 industrial digital output expansion boards for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS4260LM quad low-side intelligent power switch, in a digital output module connected to 0.5 A industrial loads.

The X-NUCLEO-OUT07A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of an X-NUCLEO-OUT07A1 stacked on other expansion boards: hardware settings described in Section 2.4: Software required resources must be followed.

Figure 10. X-NUCLEO-OUT07A1 expansion board



3.1.9 X-NUCLEO-OUT08A1 expansion board

The X-NUCLEO-OUT08A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible evaluation and development environment for 2 A (typ.) digital output modules, featuring the safe driving and smart diagnostic capabilities of the IPS160HF single high-side switch.

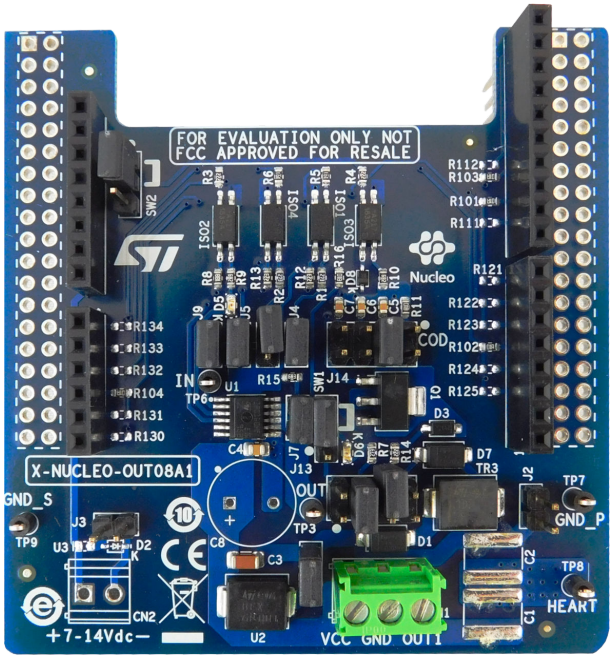
The X-NUCLEO-OUT08A1 interfaces with the microcontroller on the STM32 Nucleo via 3 kV optocouplers driven by GPIO pins and Arduino™ UNO R3 (default configuration) and ST morpho (optional, not mounted) connectors.

The expansion board should be connected to either a NUCLEO-F401RE or NUCLEO-G431RB development board, and can also be stacked with another X-NUCLEO-OUT08A1 or X-NUCLEO-OUT10A1.

Up to four X-NUCLEO-OUT08A1 expansion boards can be stacked to evaluate up to a quad channel digital output module with 2 A (typ.) capability each: hardware settings described in Section 2.4 must be followed.

It is also possible to evaluate the typical cascade architecture of a single channel digital output module for safety applications: in this scenario, the first shield output is connected to the supply of the second one. Dedicated on-board hardware can be enabled or disabled to activate fast discharge of high capacitive loads, output voltage sensing, and an additional surge pulse output line protection.

Figure 11. X-NUCLEO-OUT08A1 expansion board



3.1.10 X-NUCLEO-OUT09A1 expansion board

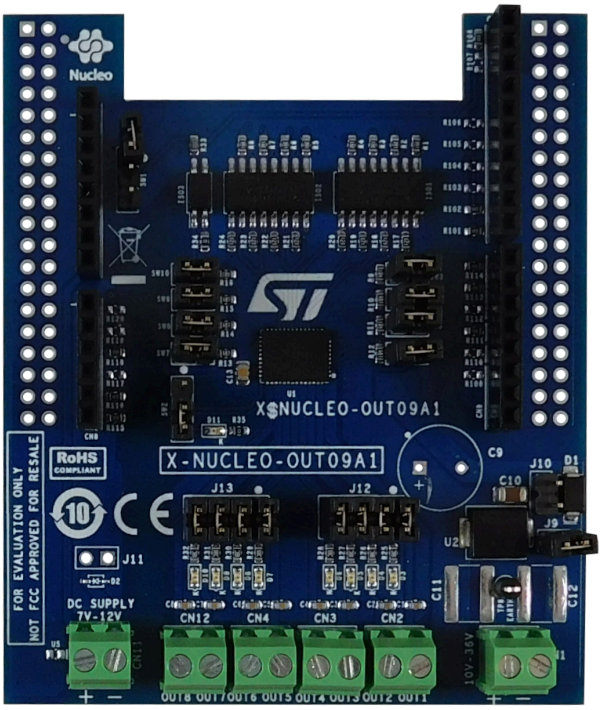
The X-NUCLEO-OUT09A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS8160HQ octal high-side smart power solid state relay, in a digital output module connected to 0.7 A industrial loads.

The X-NUCLEO-OUT09A1 interfaces with the microcontroller on the STM32 Nucleo via 3 kV and 3.7 kV optocouplers driven by GPIO pins and Arduino® R3 connectors.

The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of a X-NUCLEO-OUT09A1 stacked on other expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 12. X-NUCLEO-OUT09A1 expansion board



3.1.11 X-NUCLEO-OUT10A1 expansion board

The X-NUCLEO-OUT10A1 industrial digital output expansion board for STM32 Nucleo provides an affordable and easy-to-use solution for the development of 0.5 A (typ.) digital output modules, letting you easily evaluate the IPS161HF driving and diagnostic capabilities with industrial loads.

The X-NUCLEO-OUT10A1 interfaces with the microcontroller on the STM32 Nucleo via 3 kV optocouplers driven by GPIO pins and Arduino™ UNO R3 (default configuration) and ST morpho (optional, not mounted) connectors.

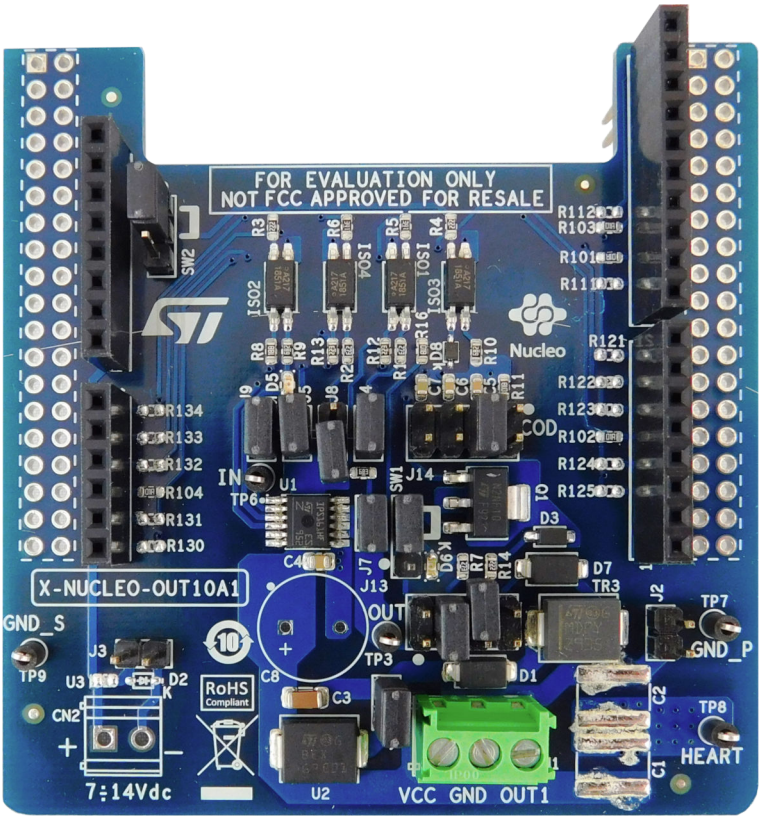
The expansion board should be connected to either a NUCLEO-F401RE or NUCLEO-G431RB development board, and can be stacked with another X-NUCLEO-OUT10A1 or an X-NUCLEO-OUT08A1.

Up to four X-NUCLEO-OUT10A1 expansion boards can be stacked to evaluate up to a quad channel digital output module with 0.5 A (typ.) capability each: hardware settings described in Section 2.4 must be followed.

It is also possible to evaluate the typical cascade architecture of a single channel digital output module for safety applications: in this scenario, the first shield output is connected to the supply of the second one.

Dedicated on-board hardware can be enabled or disabled to activate fast discharge of high capacitive loads, output voltage sensing, and an additional surge pulse output line protection.

Figure 13. X-NUCLEO-OUT10A1 expansion board



3.1.12 X-NUCLEO-OUT11A1 expansion board

The X-NUCLEO-OUT11A1 is an industrial digital output expansion board for STM32 Nucleo. It provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the ISO808 octal high-side smart power solid state relay, with embedded galvanic isolation, in a digital output module connected to 0.7 A industrial loads.

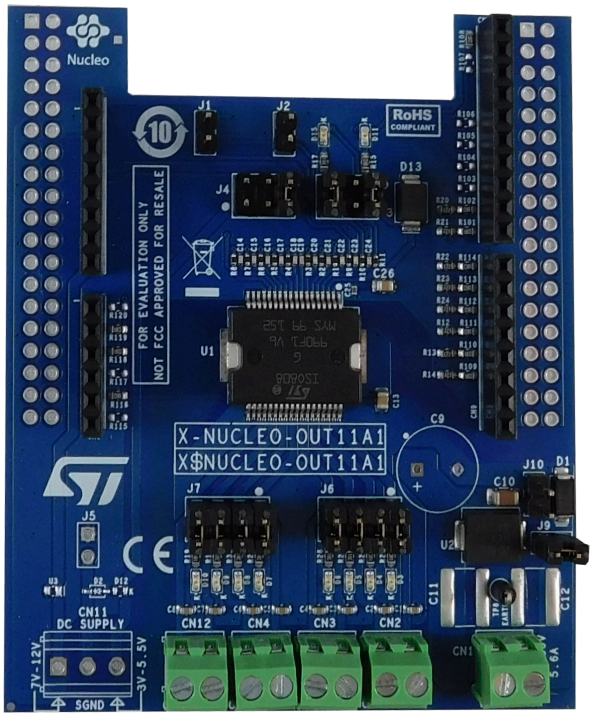
The X-NUCLEO-OUT11A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors.

The galvanic isolation between the microcontroller and the process stage is guaranteed by the ISO808.

The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of a X-NUCLEO-OUT11A1 stacked on other expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 14. X-NUCLEO-OUT11A1 expansion board



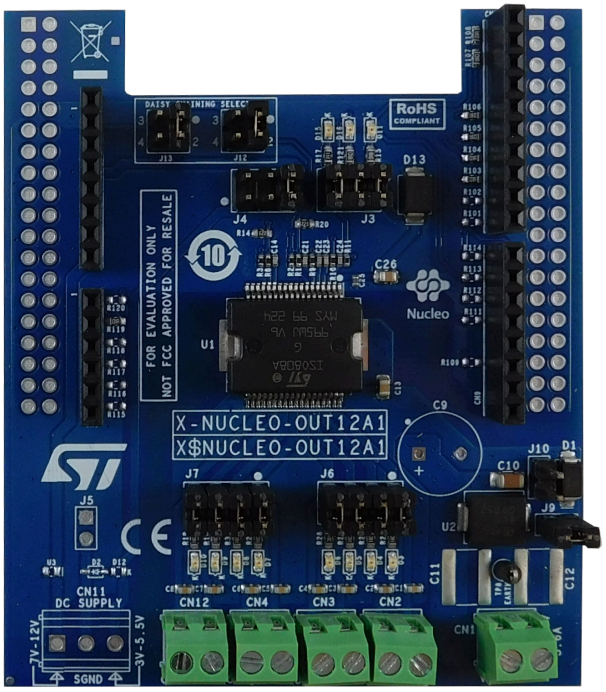
3.1.13 X-NUCLEO-OUT12A1 expansion board

The X-NUCLEO-OUT12A1 industrial digital output expansion boards for STM32-Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the ISO808A octal high-side smart power solid state relay, with embedded galvanic isolation and 20MHz SPI control interface, in a digital output module connected to 0.7 A industrial loads.

The X-NUCLEO-OUT12A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors. The galvanic isolation between the microcontroller and the process stage is guaranteed by the ISO808A device. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a 16-channel digital output system enabling the daisy chaining feature on two X-NUCLEO-OUT12A1 stacked expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 15. X-NUCLEO-OUT12A1 expansion board



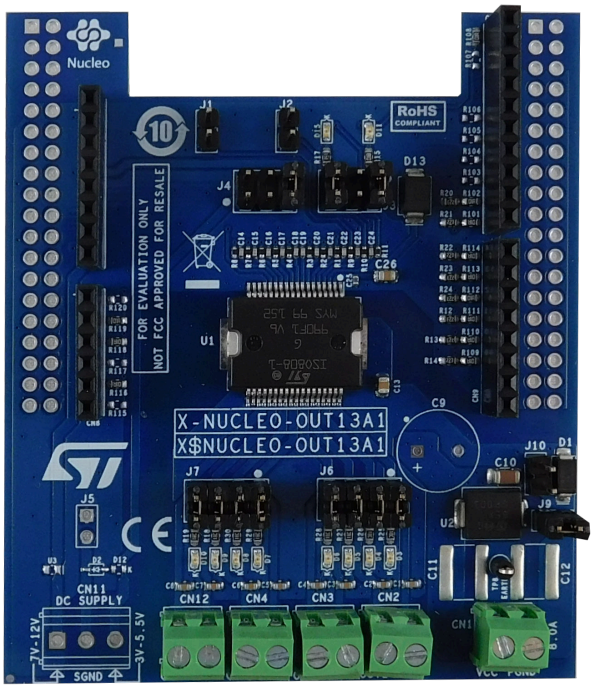
3.1.14 X-NUCLEO-OUT13A1 expansion board

The X-NUCLEO-OUT13A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the ISO808-1 octal high-side smart power solid state relay with embedded galvanic isolation, in a digital output module connected to 1.0 A industrial loads.

The X-NUCLEO-OUT13A1 interfaces with the microcontroller on the STM32 Nucleo via Arduino® R3 connectors. The ISO808-1 integrated technology guarantees a 2 kV_{RMS} galvanic isolation.

The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board. It is also possible to evaluate a system composed of an X-NUCLEO-OUT13A1 stacked on other expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 16. X-NUCLEO-OUT13A1 expansion board



3.1.15 X-NUCLEO-OUT14A1 expansion board

The X-NUCLEO-OUT14A1 is an industrial digital output expansion board for STM32 Nucleo. It provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the ISO808A-1 octal high-side smart power solid state relay, with embedded galvanic isolation and 20MHz SPI control interface, in a digital output module connected to 1.0 A industrial loads.

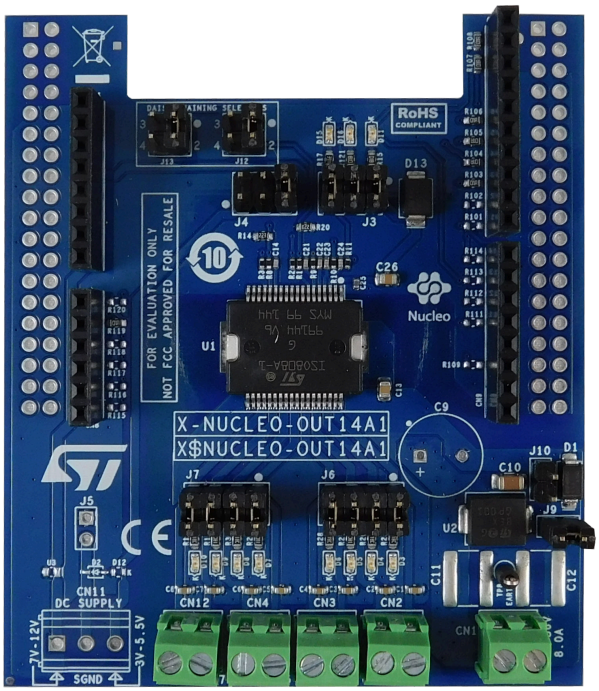
The X-NUCLEO-OUT14A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors.

The galvanic isolation between the microcontroller and the process stage is guaranteed by the ISO808A-1.

The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a 16 channel digital output system enabling the daisy chaining feature on two X-NUCLEO-OUT14A1 stacked expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 17. X-NUCLEO-OUT14A1 expansion board



3.1.16 X-NUCLEO-OUT15A1 expansion board

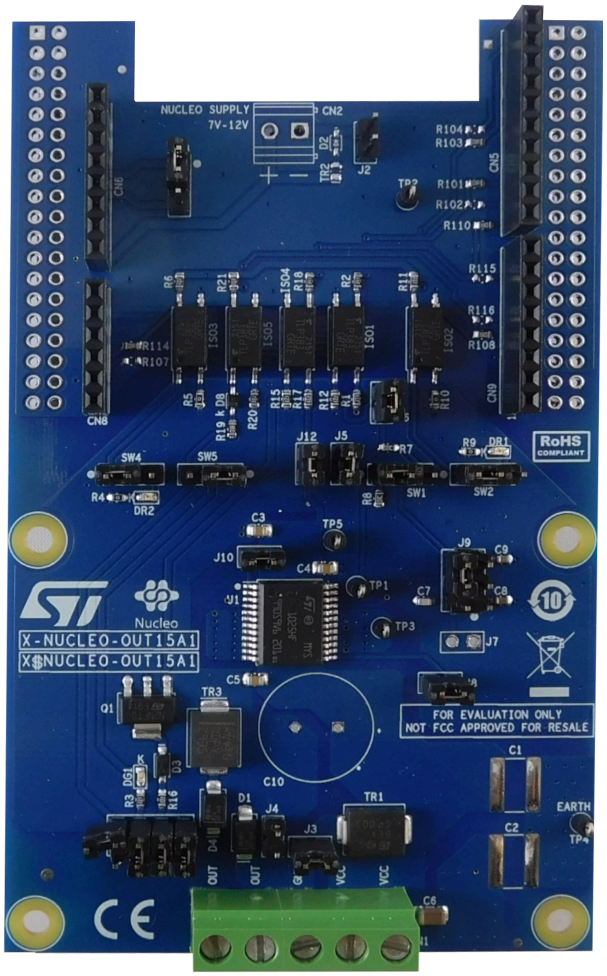
The X-NUCLEO-OUT15A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible evaluation and development environment for 2.5 A (typical) digital output modules. It features the safe driving and smart diagnostic capabilities of the IPS1025HF high efficiency single high-side switch.

The X-NUCLEO-OUT15A1 interfaces with the microcontroller on the STM32 Nucleo via 3 kV optocouplers driven by GPIO pins, with the Arduino® UNO R3 (default configuration), and the ST morpho (optional, not mounted) connectors.

The expansion board can be connected to either a NUCLEO-F401RE or NUCLEO-G431RB development board. It can also be stacked with another X-NUCLEO-OUT15A1.

Two X-NUCLEO-OUT15A1 expansion boards allow you to evaluate a dual-channel digital output module with 2.5A (typical) capability each: hardware settings described in Section 2.4 must be followed.

Figure 18. X-NUCLEO-OUT15A1 expansion board



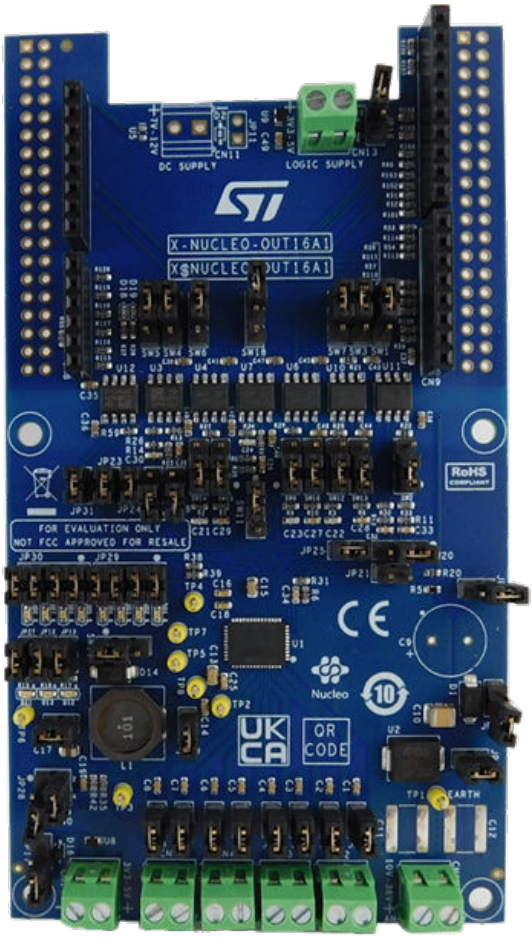
3.1.17 X-NUCLEO-OUT16A1 expansion board

The X-NUCLEO-OUT16A1 industrial digital output expansion boards for STM32-Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS8200HQ octal high-side smart power solid state relay with serial/parallel selectable interface on-chip, in a digital output module connected to 0.7 A industrial loads.

The X-NUCLEO-OUT16A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a 16-channel digital output system enabling the daisy chaining feature on two X-NUCLEO-OUT16A1 stacked expansion boards properly configured in Daisy Chain Mode and driving them with the dedicated example project *DaisyChain* available in the software package.

Figure 19. X-NUCLEO-OUT16A1 expansion board



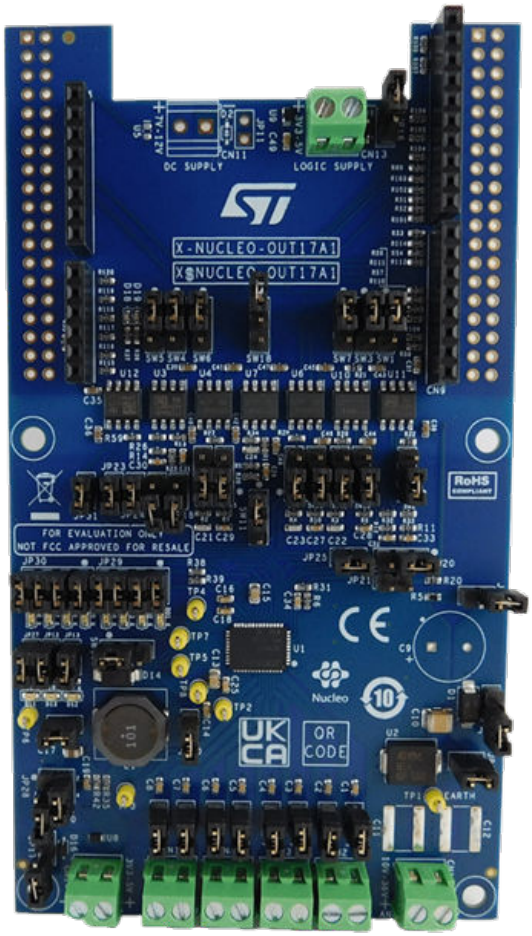
3.1.18 X-NUCLEO-OUT17A1 expansion board

The X-NUCLEO-OUT17A1 industrial digital output expansion boards for STM32-Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS8200HQ-1 octal high-side smart power solid state relay with serial/parallel selectable interface on-chip, in a digital output module connected to 1.0 A industrial loads.

The X-NUCLEO-OUT17A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a 16-channel digital output system enabling the daisy chaining feature on two X-NUCLEO-OUT17A1 stacked expansion boards properly configured in Daisy Chain Mode and driving them with the dedicated example project *DaisyChain* available in the software package.

Figure 20. X-NUCLEO-OUT17A1 expansion board



3.1.19 X-NUCLEO-OUT19A1 expansion board

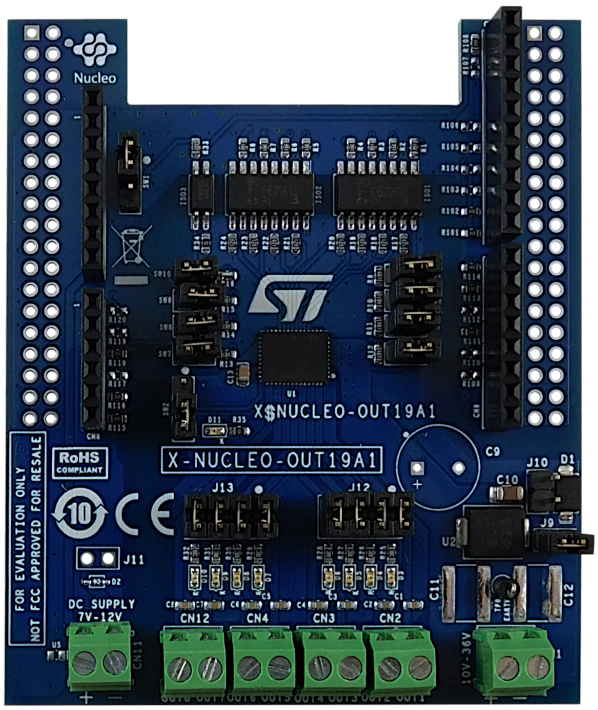
The X-NUCLEO-OUT19A1 industrial digital output expansion board for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS8160HQ-1 octal high-side smart power solid state relay, in a digital output module connected to 1 A industrial loads.

The X-NUCLEO-OUT19A1 interfaces with the microcontroller on the STM32 Nucleo via 3 kV and 3.7 kV optocouplers driven by GPIO pins and Arduino® R3 connectors.

The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of a X-NUCLEO-OUT19A1 stacked on other expansion boards: hardware settings described in Section 2.4 must be followed.

Figure 21. X-NUCLEO-OUT19A1 expansion board



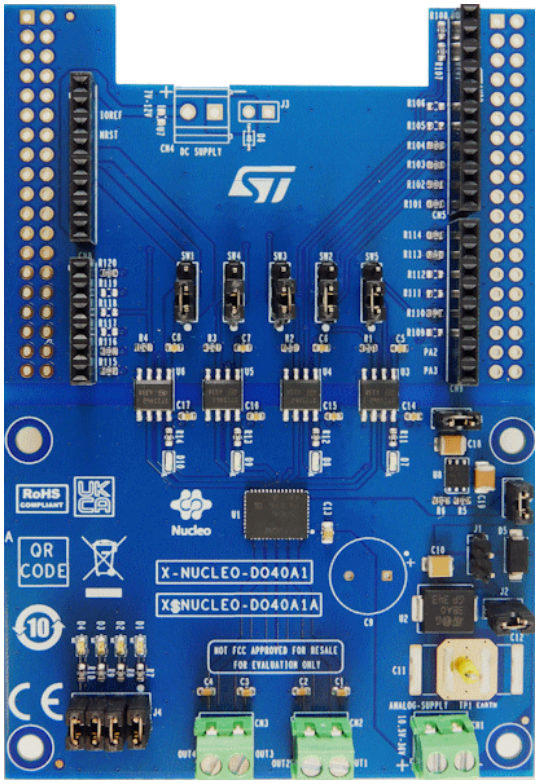
3.1.20 X-NUCLEO-DO40A1 expansion board

The X-NUCLEO-DO40A1 industrial digital output expansion boards for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS4140HQ quad high-side intelligent power switch, in a digital output module connected to 0.6 A industrial loads.

The X-NUCLEO-DO40A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of an X-NUCLEO-DO40A1 stacked on other expansion boards: hardware settings described in Section 2.4: Software required resources must be followed.

Figure 22. X-NUCLEO-DO40A1 expansion board



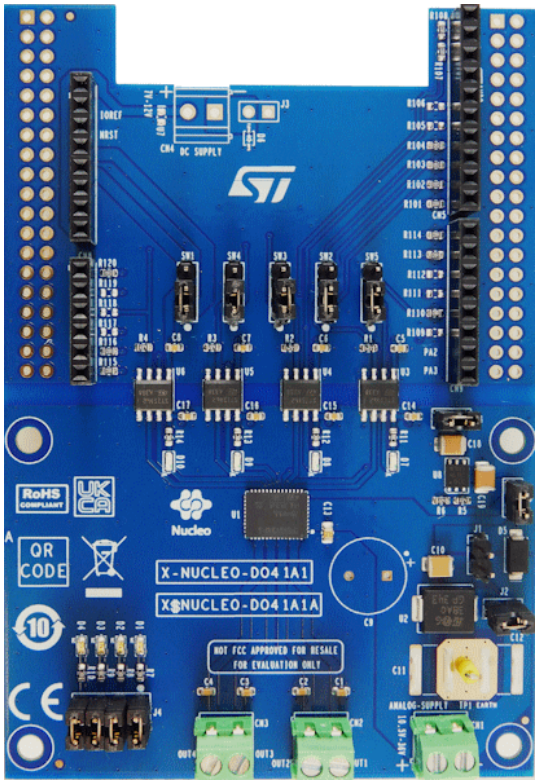
3.1.21 X-NUCLEO-DO41A1 expansion board

The X-NUCLEO-DO41A1 industrial digital output expansion boards for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS4140HQ-1 quad high-side intelligent power switch, in a digital output module connected to 1.0 A industrial loads.

The X-NUCLEO-DO41A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® R3 connectors. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of an X-NUCLEO-DO41A1 stacked on other expansion boards: hardware settings described in Section 2.4: Software required resources must be followed.

Figure 23. X-NUCLEO-DO41A1 expansion board



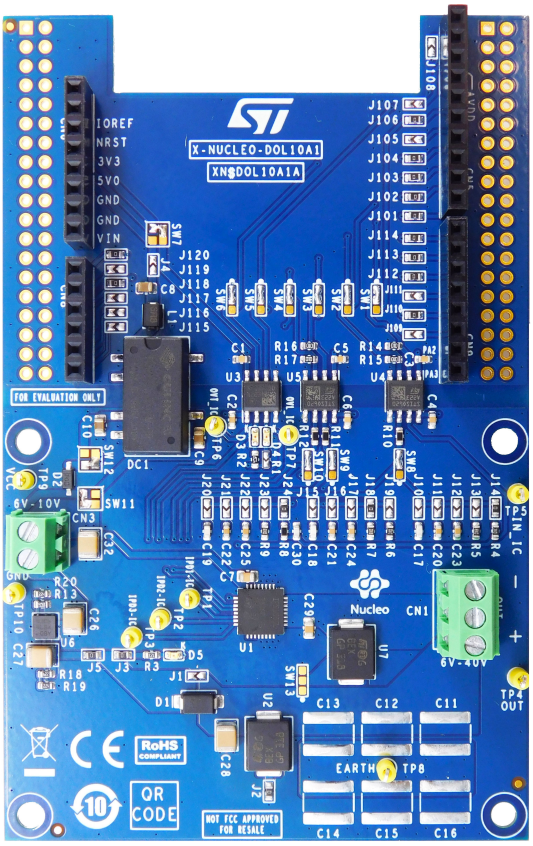
3.1.22 X-NUCLEO-DOL10A1 expansion board

The X-NUCLEO-DOL10A1 industrial digital output expansion boards for STM32 Nucleo provides a powerful and flexible environment for the evaluation of the driving and diagnostic capabilities of the IPS1050HQ single low-side intelligent power switch, in a digital output module connected to 5.0 A industrial loads.

The X-NUCLEO-DOL10A1 directly interfaces with the microcontroller on the STM32 Nucleo driven by GPIO pins and Arduino® UNO R3 connectors. The expansion board can be connected to either a NUCLEO-F401RE or a NUCLEO-G431RB development board.

It is also possible to evaluate a system composed of an X-NUCLEO-DOL10A1 stacked on other expansion boards: hardware settings described in Section 2.4: Software required resources must be followed.

Figure 24. X-NUCLEO-DOL10A1 expansion board



3.2 Hardware setup

The following hardware components are needed:

1. One USB type A to Mini-B USB cable to connect the STM32 Nucleo to the PC when using a NUCLEO-F401RE
2. One USB type A to Micro-B USB cable when using a NUCLEO-G431RB
3. An external power supply (8 - 33 V) and the associated wires to supply the system expansion boards

3.3 Software setup

The following software components are needed to set up a suitable development environment for creating applications for the STM32 Nucleo equipped with one or more industrial digital output expansion boards:

- X-CUBE-IPS: an expansion for STM32Cube dedicated to applications development which require the use of:
  - ISO8200BQ
  - ISO8200AQ
  - IPS2050H
  - IPS2050H-32
  - IPS1025H
  - IPS1025H-32
  - IPS4260LM
  - IPS160HF
  - IPS8160HQ
  - IPS161HF
  - ISO808
  - ISO808A
  - ISO808-1
  - ISO808A-1
  - IPS1025HF
  - IPS8200HQ
  - IPS8200HQ-1
  - IPS8160HQ-1
  - IPS4140HQ
  - IPS4140HQ-1
  - IPS1050LQ

  The X-CUBE-IPS firmware and related documentation is available on www.st.com.
- Development tool-chain and compiler: the STM32Cube expansion software supports the three following environments:
  - IAR Embedded Workbench for ARM® (EWARM) toolchain + ST-LINK
  - RealView Microcontroller Development Kit (MDK-ARM-STR) toolchain + ST-LINK
  - STM32CubeIDE + ST-LINK

3.4 Board setup

3.4.1 STM32 Nucleo development board

Configure the STM32 Nucleo development board with the following jumper positions:

- NUCLEO-F401RE
  - JP5 on U5V for firmware flashing
  - JP1 open
  - JP6 closed
  - CN2 closed 1-2, 3-4
  - CN3 open
  - CN4 open
  - CN11 closed
  - CN12 closed
- NUCLEO-G431RB
  - JP5 closed 1-2 (5V_STLK for firmware flashing)
  - JP1, JP7 open
  - JP3, JP6 closed
  - JP8 closed 1-2
  - CN4 open
  - CN11 closed
  - CN12 closed

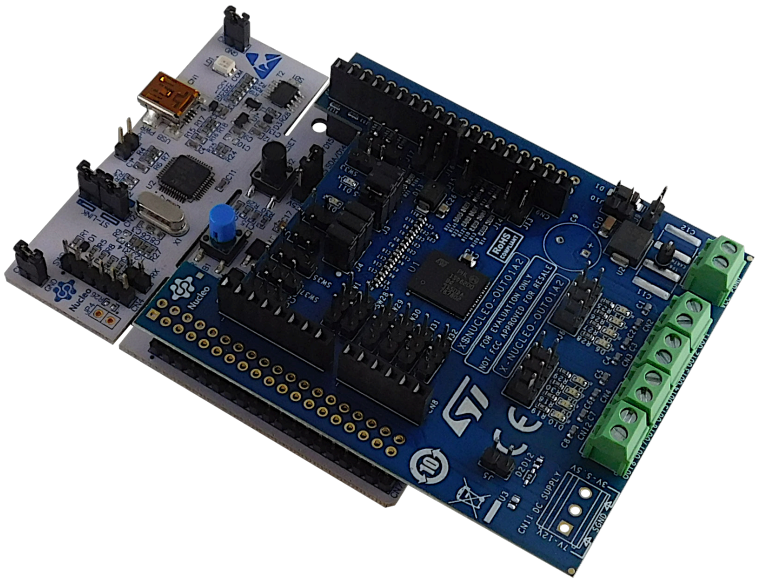
3.4.2 X-NUCLEO-OUT01A2 expansion board

The X-NUCLEO-OUT01A2 must be configured as follows:

- J5 Open
- J3 Closed 1-2, 5-6
- J4 Closed 5-6
- J6 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT1-4
- J7 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT5-8
- J9, J10, J13, J14, J15, J16, J17 Closed
- SW1 All open
- SW28 Closed 2-3
- SW29 Closed 2-3
- SW30 All Open
- SW31 Closed 2-3
- SW32 All Open
- SW33 Closed 2-3
- SW34 All Open
- SW35 Closed 2-3
- Direct Mode:
  - J1, J2 Closed
  - SW36 Closed 1-2
  - SW37 Closed 2-3
- Synchronous Mode:
  - J1, J2 Open
  - SW36 Closed 2-3
  - SW37 Closed 2-3

Step 1. Plug the X-NUCLEO-OUT01A2 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 25. X-NUCLEO-OUT01A2 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT01A2 expansion board on by connecting CN1 connector pin 1 (VCC) and pin 2 (GND) to the DC power supply (which must be set between 15 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE).

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT01A2\DirectMode for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT01A2\SynchronousMode for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT01A2\DirectMode for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT01A2\SynchronousMode for NUCLEO-G431RB.

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example. Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

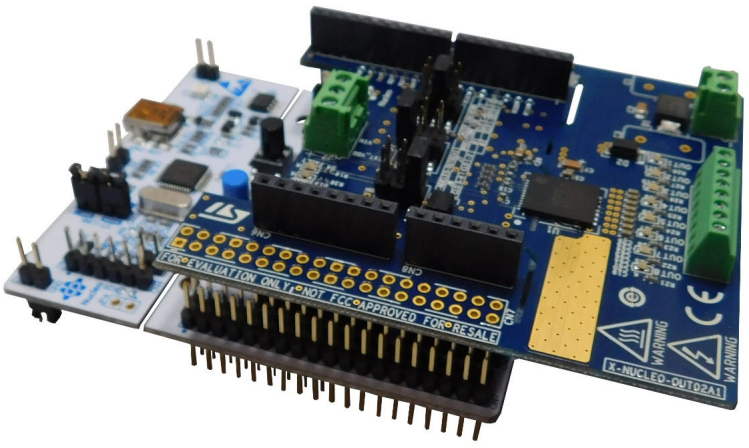
3.4.3 X-NUCLEO-OUT02A1 expansion board

The X-NUCLEO-OUT02A1 must be configured as follows:

- J8 Closed 5-6
- J6, J7 for **Daisy Chain** setup:
  - Board 0:
    - J6: Closed 3-4
    - J7: Closed 3-4
  - Board 1:
    - J6: Closed 1-2
    - J7: Closed 1-2
- J6, J7 for **Regular 16 Channels** setup:
  - Board 0:
    - J6: Closed 3-4
    - J7: Closed 1-2
  - Board 1:
    - J6: Closed 3-4
    - J7: Closed 1-2
- J6, J7 for **Regular 8 Channels** setup:
  - Board 0:
    - J6: Closed 3-4
    - J7: Closed 1-2

Step 1. Plug the X-NUCLEO-OUT02A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 26. X-NUCLEO-OUT02A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector J1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT02A1 expansion board on by connecting J1 connector pin 1 (VCC) and pin 2 (GND) to the DC power supply (which must be set between 15 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT02A1\DaisyChain for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT02A1\Regular_8_Channels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT02A1\Regular_16_Channels for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT02A1\DaisyChain for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT02A1\Regular_8_Channels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT02A1\Regular_16_Channels for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

3.4.4 X-NUCLEO-OUT03A1 and X-NUCLEO-OUT04A1 expansion boards

The X-NUCLEO-OUT03A1 or X-NUCLEO-OUT04A1 must be configured as follows:

- SW1, SW2, SW3 Closed 1-2
- SW4
  - Closed 1-2 to route FLT2 signal from device to microcontroller only
  - Closed 2-3 to drive the DR2 red LED only
- SW5
  - Closed 1-2 to route FLT1 signal from device to microcontroller only
  - Closed 2-3 to drive the DR1 red LED only
- J1, J2, J5, J6, J7, J12, J13, J14 Closed
- J3, J4, J10, J11, J17 Open
- J8, J9 Closed 4-6

Step 1. Plug the X-NUCLEO-OUT03A1 or X-NUCLEO-OUT04A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 27. X-NUCLEO-OUT03A1 expansion board connected to an STM32 Nucleo development board

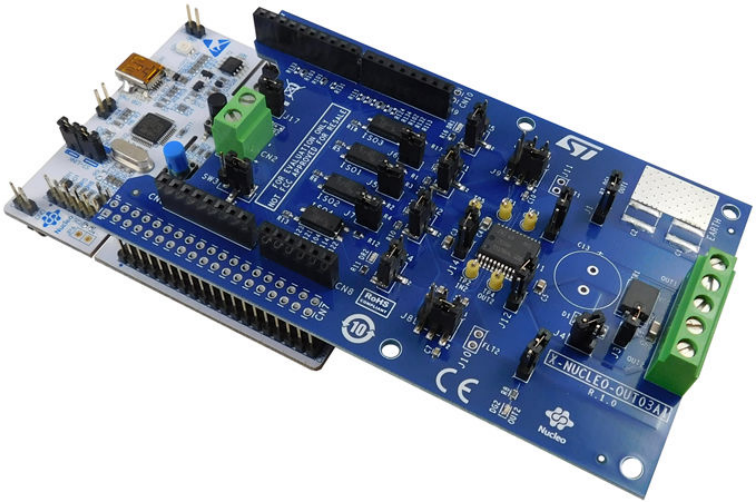
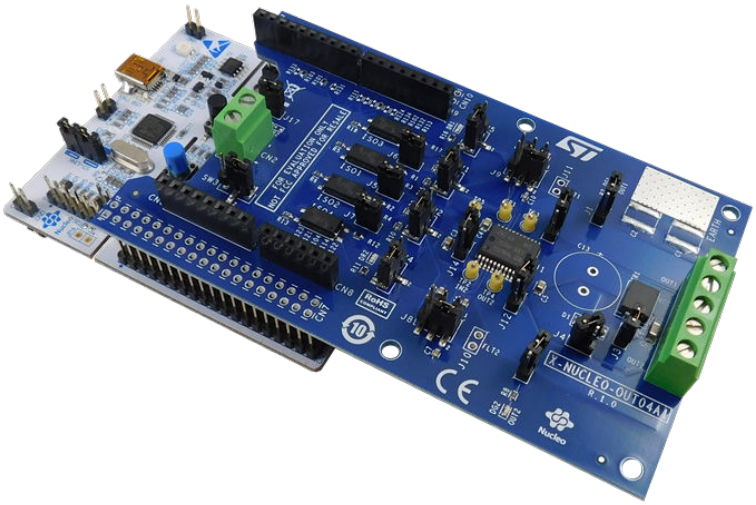


Figure 28. X-NUCLEO-OUT04A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT03A1 or X-NUCLEO-OUT04A1 expansion board by connecting CN1 connector pin 2 or 3 (VCC) and 4 (GND) to the DC power supply (which must be set between 8 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT03A1\DefaultBoard for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT03A1\FourBoards for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT04A1\DefaultBoard for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT04A1\FourBoards for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT03A1\DefaultBoard for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT03A1\FourBoards for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT04A1\DefaultBoard for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT04A1\FourBoards for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

3.4.5 X-NUCLEO-OUT05A1 and X-NUCLEO-OUT06A1 expansion boards

The X-NUCLEO-OUT05A1 or X-NUCLEO-OUT06A1 must be configured as follows:

- SW1, SW3 Closed 1-2
- SW2
  - Closed 1-2 to route FLT1 signal from device to microcontroller only
  - Closed 2-3 to drive the DR1 red LED only
- SW4
  - Closed 1-2 to route FLT2 signal from device to microcontroller only
  - Closed 2-3 to drive the DR2 red LED only
- J1, J3, J5, J6, J8, J10 Closed
- J2, J4, J7 Open
- J9 Closed 4-6

Step 1. Plug the X-NUCLEO-OUT05A1 or X-NUCLEO-OUT06A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 29. X-NUCLEO-OUT05A1 expansion board connected to an STM32 Nucleo development board

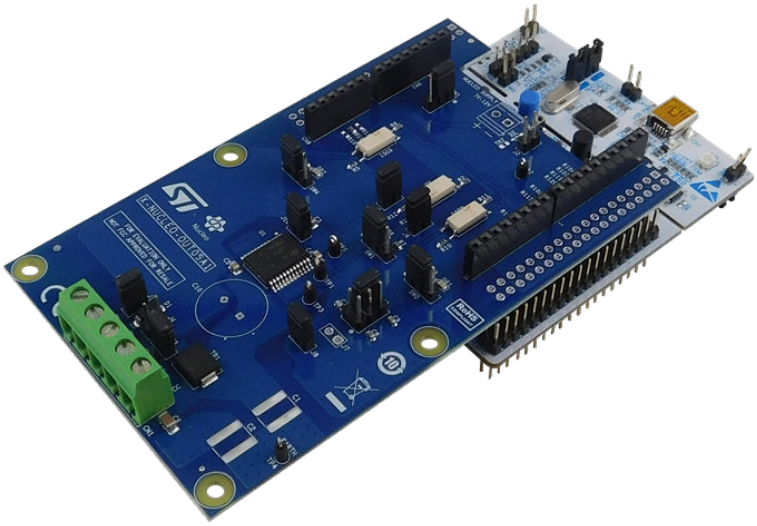
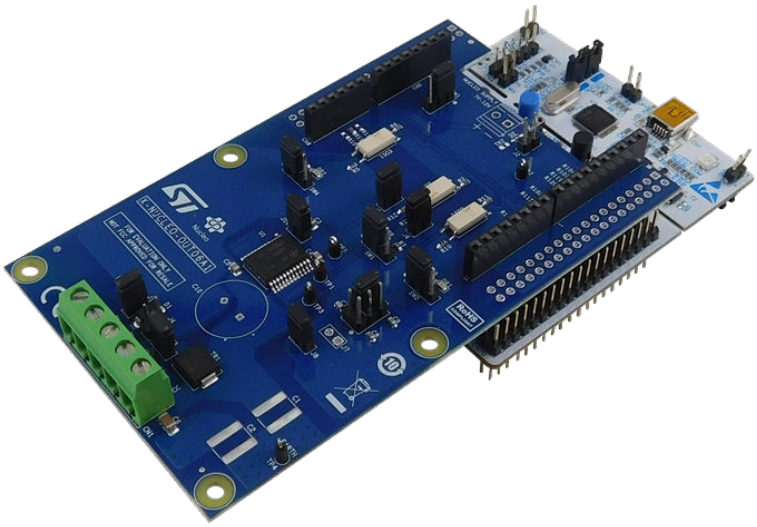


Figure 30. X-NUCLEO-OUT06A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT05A1 or X-NUCLEO-OUT06A1 expansion board by connecting CN1 connector pin 4 or 5 (VCC) and 3 (GND) to the DC power supply (which must be set between 8 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil®, EWARM from IAR, or STM32CubeIDE).

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT05A1\DefaultBoard for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT05A1\FourBoards for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT06A1\DefaultBoard for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT06A1\FourBoards for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT05A1\DefaultBoard for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT05A1\FourBoards for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT06A1\DefaultBoard for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT06A1\FourBoards for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

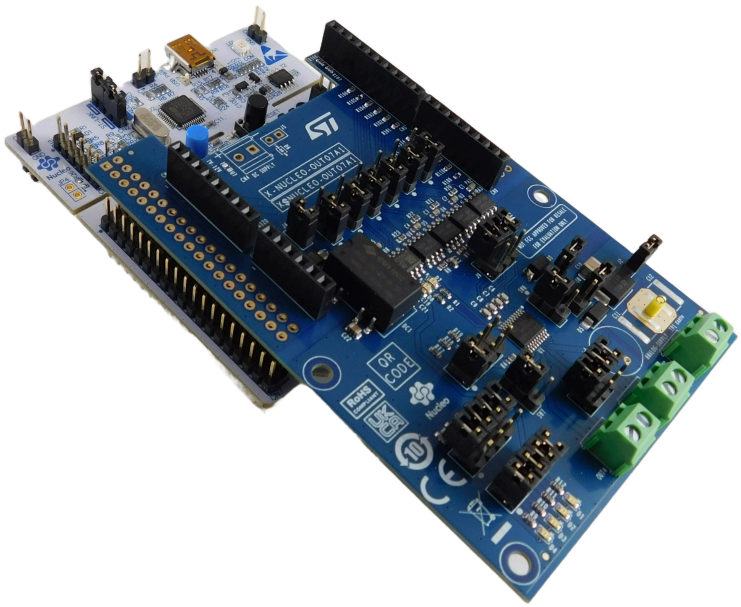
3.4.6 X-NUCLEO-OUT07A1 expansion board

The X-NUCLEO-OUT07A1 expansion board must be configured in the following way:

- J1 Open
- J2 Closed
- J3 Open
- J4 Closed
- J5 Open
- J6 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT1-4
- J7 all Open
- J8 Closed 1-2
- J9 all Open
- J10 all Open
- SW1 Closed 1-2
- SW2 Closed 1-2
- SW3 Closed 1-2
- SW4 Closed 1-2
- SW5 Closed 1-2
- SW6 Closed 1-2
- SW7 Closed 1-2
- SW8 Closed 1-2

Step 1. Plug the X-NUCLEO-OUT07A1 expansion board on top of the STM32 Nucleo via the Arduino connectors.

Figure 31. X-NUCLEO-OUT07A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT07A1 expansion board by connecting CN1 connector pin 1 (VCC) and 2 (GND) to the DC power supply (which must be set between 8 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil®, EWARM from IAR, or STM32CubeIDE).

Step 5. Depending on the STM32 Nucleo board used, open the software project from:

– Projects\NUCLEO-F401RE\Examples\OUT07A1\FourChannels for NUCLEO-F401RE

or

– Projects\NUCLEO-G431RB\Examples\OUT07A1\FourChannels for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description

3.4.7 X-NUCLEO-OUT08A1 and X-NUCLEO-OUT10A1 expansion boards

The X-NUCLEO-OUT08A1 or X-NUCLEO-OUT10A1 expansion board must be configured in the following way:

- J1, J4, J5, J7, J8, J9 Closed
- J13 Closed: 1-2, 3-4, 5-6
- J14 Closed: 1-2, 3-4
- SW1: Closed 2-3
- SW2: Closed 1-2
- All other jumpers Open

Step 1. Plug the X-NUCLEO-OUT08A1 or X-NUCLEO-OUT10A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 32. X-NUCLEO-OUT08A1 expansion board connected to an STM32 Nucleo development board

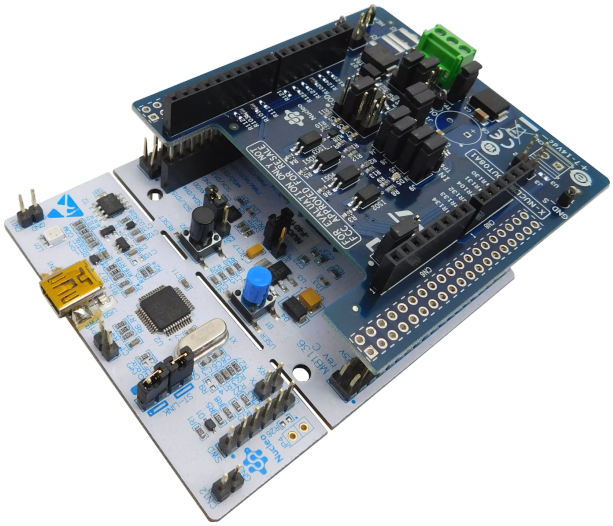
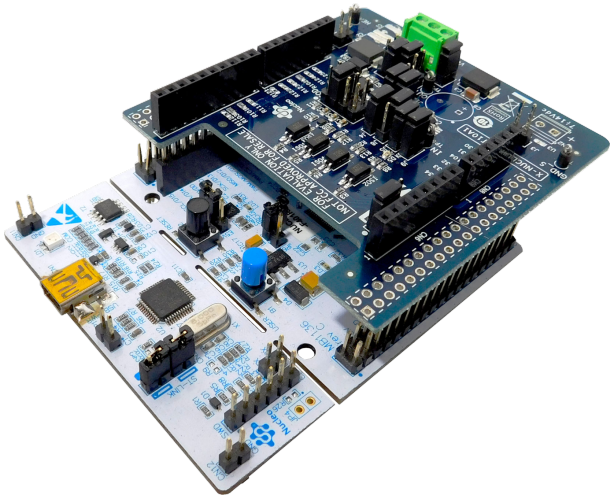


Figure 33. X-NUCLEO-OUT10A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT08A1 or X-NUCLEO-OUT10A1 expansion board on by connecting its connectors CN1 1(V_{CC}), 2(GND) to the DC power supply (which must be set between 8 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT05A1\DefaultBoard for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT05A1\FourBoards for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT06A1\DefaultBoard for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT06A1\FourBoards for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT05A1\DefaultBoard for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT05A1\FourBoards for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT06A1\DefaultBoard for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT06A1\FourBoards for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

3.4.8 X-NUCLEO-OUT09A1 and X-NUCLEO-OUT19A1 expansion boards

The X-NUCLEO-OUT09A1 and X-NUCLEO-OUT19A1 must be configured as follows:

- SW1 Closed 2-3
- SW2:
  - Closed 1-2 to route STATUS signal from device to microcontroller only
  - Closed 2-3 to drive the D11 red LED only
- SW3 to SW10 Closed
- J9 Closed
- J10 Open
- J12 Closed 1-2, 3-4, 5-6, 7-8 to enable green LEDs D3, D4, D5, D6 (OUT1, OUT2, OUT3, OUT4)
- J13 Closed 1-2, 3-4, 5-6, 7-8 to enable green LEDs D7, D8, D9, D10 (OUT5, OUT6, OUT7, OUT8)

Step 1. Plug the X-NUCLEO-OUT09A1 or X-NUCLEO-OUT19A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 34. X-NUCLEO-OUT09A1 expansion board connected to an STM32 Nucleo development board

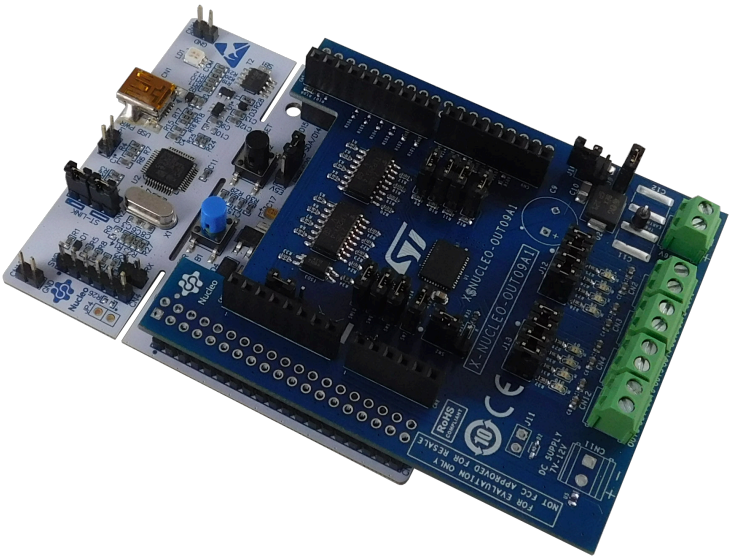
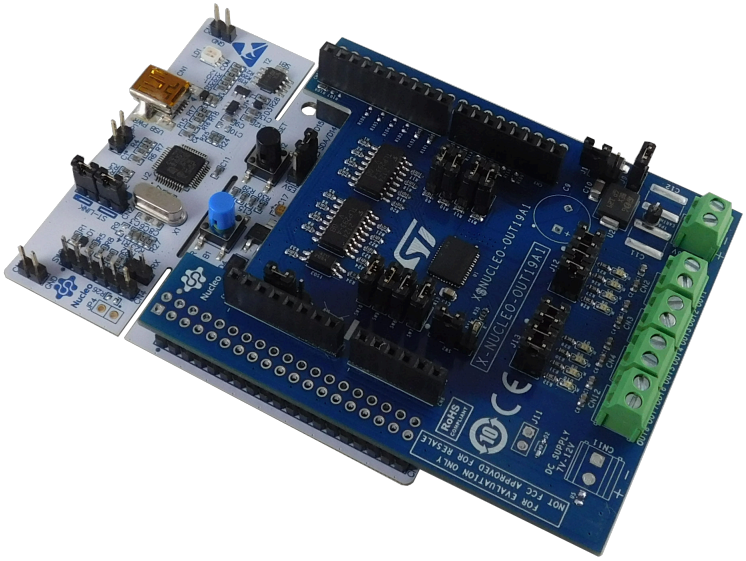


Figure 35. X-NUCLEO-OUT19A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT09A1 or X-NUCLEO-OUT19A1 expansion board on by connecting CN1 connector pin 1 (VCC) and pin 2 (GND) to the DC power supply (which must be set between 15 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT09A1\EightChannels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT19A1\EightChannels for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT09A1\EightChannels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT19A1\EightChannels for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example. Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

3.4.9 X-NUCLEO-OUT11A1 and X-NUCLEO-OUT13A1 expansion boards

The X-NUCLEO-OUT11A1 and X-NUCLEO-OUT13A1 must be configured as follows:

- J1, J2, J5 Open
- J3 Closed 1-2, 5-6
- J4 Closed 5-6
- J6 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT1-4
- J7 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT5-8
- J9, J10 Closed

Step 1. Plug the X-NUCLEO-OUT11A1 or X-NUCLEO-OUT13A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 36. X-NUCLEO-OUT11A1 expansion board connected to an STM32 Nucleo development board

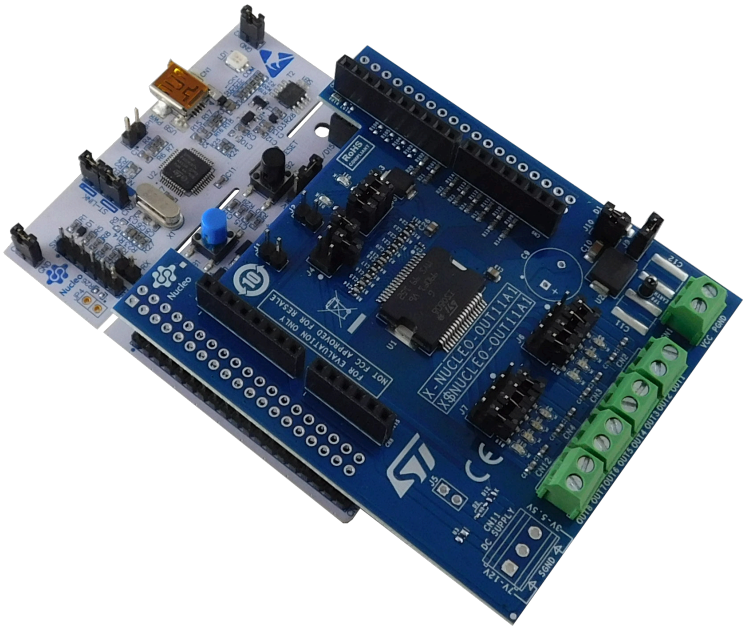
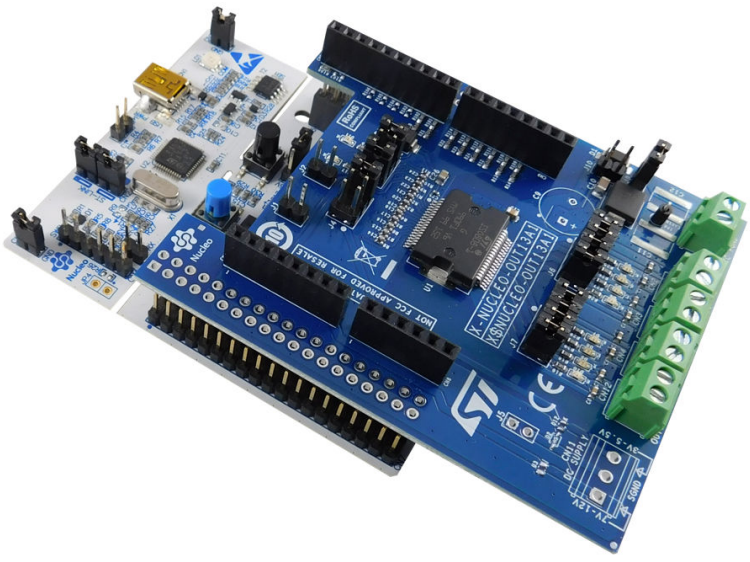


Figure 37. X-NUCLEO-OUT13A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT11A1 or X-NUCLEO-OUT13A1 expansion board on by connecting CN1 connector pin 1 (VCC) and pin 2 (GND) to the DC power supply (which must be set between 15 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT11A1\DirectMode for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT11A1\SynchronousMode for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT13A1\DirectMode for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT13A1\SynchronousMode for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT11A1\DirectMode for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT11A1\SynchronousMode for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT13A1\DirectMode for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT13A1\SynchronousMode for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

3.4.10 X-NUCLEO-OUT12A1 and X-NUCLEO-OUT14A1 expansion boards

The X-NUCLEO-OUT12A1 and X-NUCLEO-OUT14A1 must be configured as follows:

- J5 Open
- J3 Closed 1-2, 3-4, 5-6
- J4 Closed 5-6
- J6 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT1-4
- J7 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT5-8
- J9, J10 Closed
- J12, J13 for **Daisy Chain** setup:
  - Board 0:
    - J12: Closed 1-2
    - J13: Closed 3-4
  - Board 1:
    - J12: Closed 3-4
    - J13: Closed 1-2
- J12, J13 for **Regular 16 Channels** setup:
  - Board 0:
    - J12: Closed 1-2
    - J13: Closed 1-2
  - Board 1:
    - J12: Closed 1-2
    - J13: Closed 1-2
- J12, J13 for **Regular 8 Channels** setup:
  - Board 0:
    - J12: Closed 1-2
    - J13: Closed 1-2

Step 1. Plug the X-NUCLEO-OUT12A1 or X-NUCLEO-OUT14A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 38. X-NUCLEO-OUT12A1 expansion board connected to an STM32 Nucleo development board

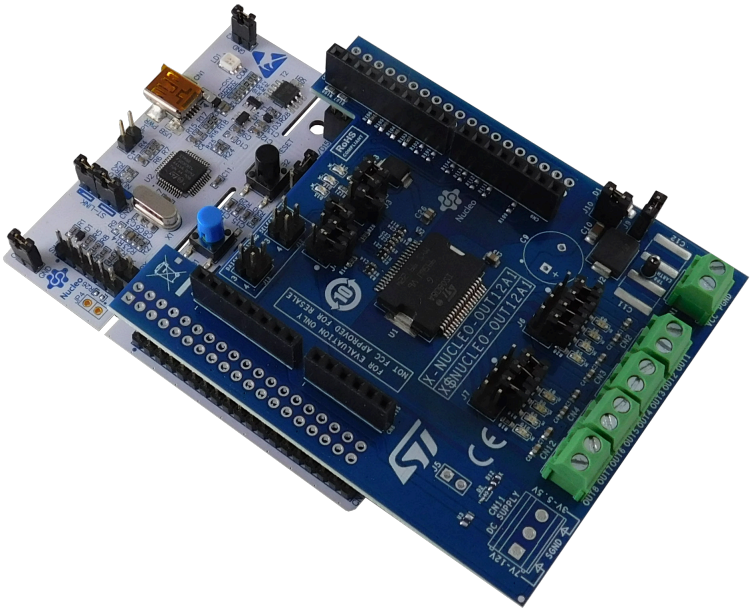
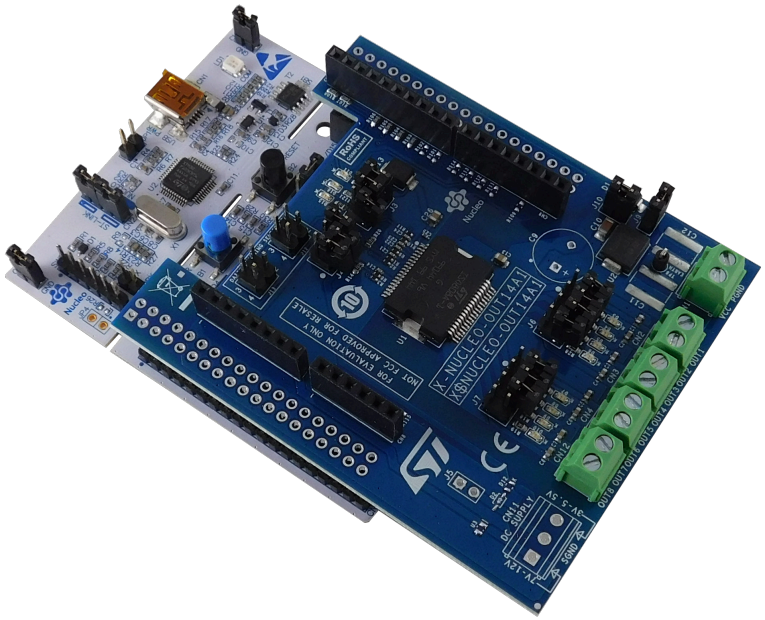


Figure 39. X-NUCLEO-OUT14A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT12A1 or X-NUCLEO-OUT14A1 expansion board on by connecting CN1 connector pin 1 (VCC) and pin 2 (GND) to the DC power supply (which must be set between 15 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT12A1\DaisyChain for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT12A1\Regular_8_Channels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT12A1\Regular_16_Channels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT14A1\DaisyChain for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT14A1\Regular_8_Channels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT14A1\Regular_16_Channels for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT12A1\DaisyChain for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT12A1\Regular_8_Channels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT12A1\Regular_16_Channels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT14A1\DaisyChain for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT14A1\Regular_8_Channels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT14A1\Regular_16_Channels for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

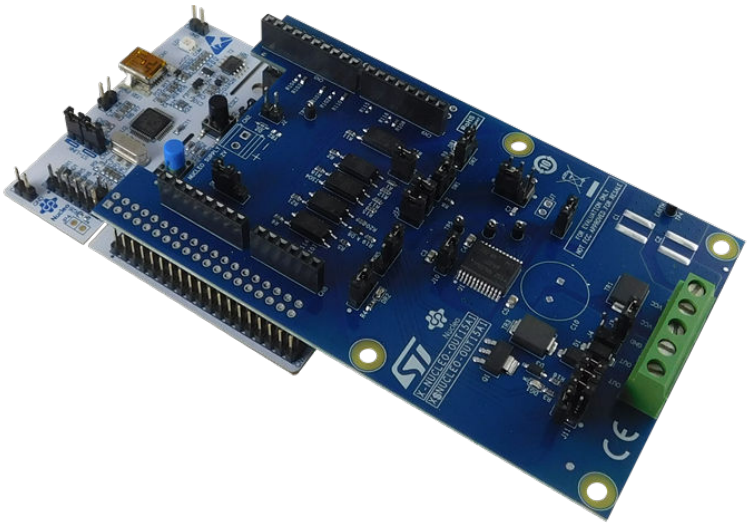
3.4.11 X-NUCLEO-OUT15A1 expansion board

The X-NUCLEO-OUT15A1 must be configured as follows:

- SW1 Cosed 2-3
- SW2
  - Closed 1-2 to route FLT1 signal from device to microcontroller only
  - Closed 2-3 to drive the DR1 red LED only
- SW3, SW5 Closed 1-2
- SW4
  - Closed 1-2 to route FLT2 signal from device to microcontroller only
  - Closed 2-3 to drive the DR2 red LED only
- J2 Open
- J3, J4, J5, J6, J7, J8, J10, J12 Closed
- J9 Closed 4-6
- J11 Closed 1-2, 3-4, 5-6

Step 1. Plug the X-NUCLEO-OUT15A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 40. X-NUCLEO-OUT15A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-OUT15A1 expansion board by connecting CN1 connector pin 4 or 5 (VCC) and 3 (GND) to the DC power supply (which must be set between 8 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil®, EWARM from IAR, or STM32CubeIDE).

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT15A1\DefaultBoard for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT15A1\TwoBoards for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT15A1\DefaultBoard for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT15A1\TwoBoards for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example.

Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

3.4.12 X-NUCLEO-OUT16A1 and X-NUCLEO-OUT17A1 expansion boards

The X-NUCLEO-OUT16A1 and X-NUCLEO-OUT17A1 must be configured as follows:

(Common settings)

- SW1 Closed 1-2
- SW17 Closed 1-2
- JP1, JP2, JP3, JP4, JP5, JP6, JP7, JP8 Closed to enable OUT1-8 output lines
- JP9 Closed
- JP10 Open
- JP11 Not mounted
- JP12 Closed
- JP13 Closed
- JP14 Open
- JP15 Closed
- JP16 Open
- JP17 Open
- JP18 Open
- JP19 Open
- JP20 Closed
- JP23 Closed
- JP24 Closed
- JP25 Closed
- JP27 Closed
- JP28 Closed 2-4
- JP29 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT1-4
- JP30 Closed 1-2, 3-4, 5-6, 7-8 to enable active state led for OUT5-8
- JP31 Closed

(Parallel 8 conf specific settings)

- SW3 Closed 1-2
- SW4 Closed 1-2
- SW5 Closed 1-2
- SW6 Closed 1-2
- SW7 Closed 1-2
- SW9 Closed 1-2
- SW10 Closed 1-2
- SW11 Closed 1-2
- SW12 Closed 1-2
- SW13 Closed 1-2
- SW14 Closed 1-2
- SW15 Closed 1-2
- SW18 Open
- SW20 Closed 1-2
- JP21 Open
- JP22 Open

(SPI 8 conf specific settings)

- SW3 (**Watchdog Timeout Enable/Disable**)
  - Closed 1-2 (WDEN L: MCU freeze detection Off)
  - Closed 2-3 (WDEN H: MCU freeze detection On)
- SW4 Closed 2-3
- SW5 Closed 2-3

- SW6 Closed 2-3
- SW7 Closed 2-3
- SW9 Closed 2-3
- SW10 Closed 2-3
- SW11 Closed 2-3
- SW12 Closed 2-3
- SW13 Closed 2-3
- SW14 Closed 2-3
- SW15 Closed 2-3
- SW18 Closed 1-2
- SW20 Closed 2-3
- JP21 Closed
- JP22
  - Open (SEL1 L: SPI 8 bits)
  - Closed (SEL1 H: SPI 16 bits)

(Daisy Chain conf specific settings)

- SW3 (**Watchdog Timeout Enable/Disable**)
  - Closed 1-2 (WDEN L: MCU freeze detection Off)
  - Closed 2-3 (WDEN H: MCU freeze detection On)
- SW4 Closed 2-3
- SW5 Closed 2-3
- SW6 (**DAISY_CHAIN/MOSI**)
  - Board 0:
    - Closed 2-3
  - Board 1:
    - Closed 1-2
- SW7 Closed 2-3
- SW9 Closed 2-3
- SW10 Closed 2-3
- SW11 Closed 2-3
- SW12 Closed 2-3
- SW13 Closed 2-3
- SW14 Closed 2-3
- SW15 Closed 2-3
- SW18 (**SPI_MISO/DAISY_CHAIN**)
  - Board 0:
    - Closed 2-3
  - Board 1:
    - Closed 1-2
- SW20 Closed 2-3
- JP21 Closed
- JP22
  - Open (SEL1 L: SPI 8 bits)
  - Closed (SEL1 H: SPI 16 bits)

Step 1. Plug the X-NUCLEO-OUT16A1 or X-NUCLEO-OUT17A1 expansion board on top of the STM32 Nucleo via the Arduino connectors.

Figure 41. X-NUCLEO-OUT16A1 expansion board connected to an STM32 Nucleo development board

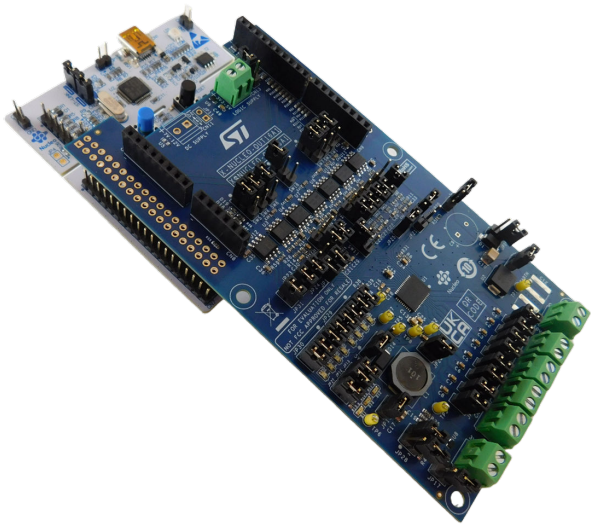
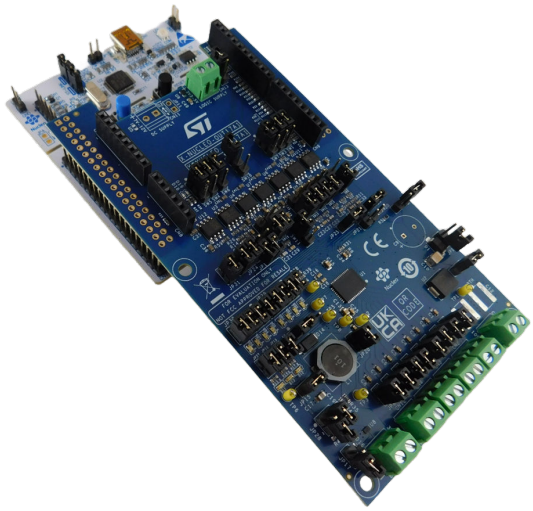


Figure 42. X-NUCLEO-OUT17A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo development board by connecting a USB cable between the CN1 connector and a PC USB port.

Step 3. Power the X-NUCLEO-OUT16A1 or X-NUCLEO-OUT17A1 expansion board by properly connecting CN1 connector pin 1 (VCC) and 2 (GND) to the DC power supply (which must be set between 8 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo development board used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\OUT16A1\DaisyChain for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT16A1\SPI_8_Channels6 for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT16A1\Parallel_8_Channels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT17A1\DaisyChain for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT17A1\SPI_8_Channels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\OUT17A1\Parallel_8_Channels for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\OUT16A1\DaisyChain for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT16A1\SPI_8_Channels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT16A1\Parallel_8_Channels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT17A1\DaisyChain for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT17A1\SPI_8_Channels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\OUT17A1\Parallel_8_Channels for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example. Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

3.4.13 X-NUCLEO-DO40A1 and X-NUCLEO-DO41A1 expansion boards

The X-NUCLEO-DO40A1 and X-NUCLEO-DO41A1 must be configured as follows:

- J1, J3 Open
- J2, J5, J6 Closed
- J4 Closed 1-2, 3-4, 5-6, 7-8 to enable green LEDs D1, D2, D3, D4 (OUT1, OUT2, OUT3, OUT4)
- SW1, SW2, SW3, SW4, SW5 Closed 1-2

Step 1. Plug the X-NUCLEO-DO40A1 or X-NUCLEO-DO41A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO connectors.

Figure 43. X-NUCLEO-DO40A1 expansion board connected to an STM32 Nucleo development board

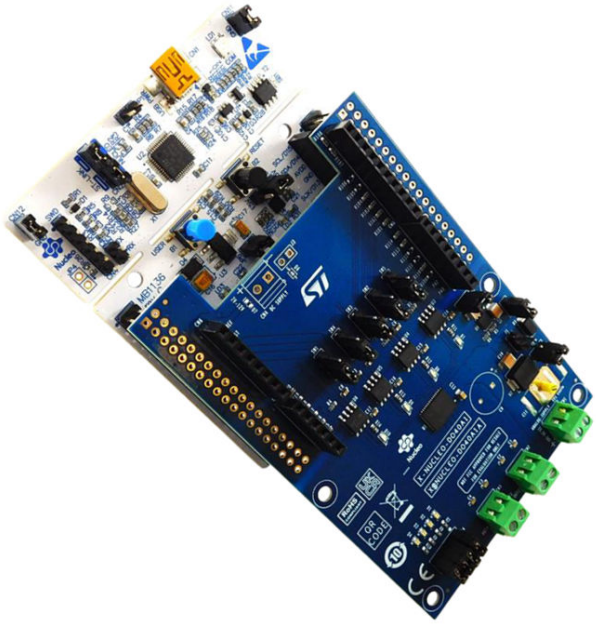
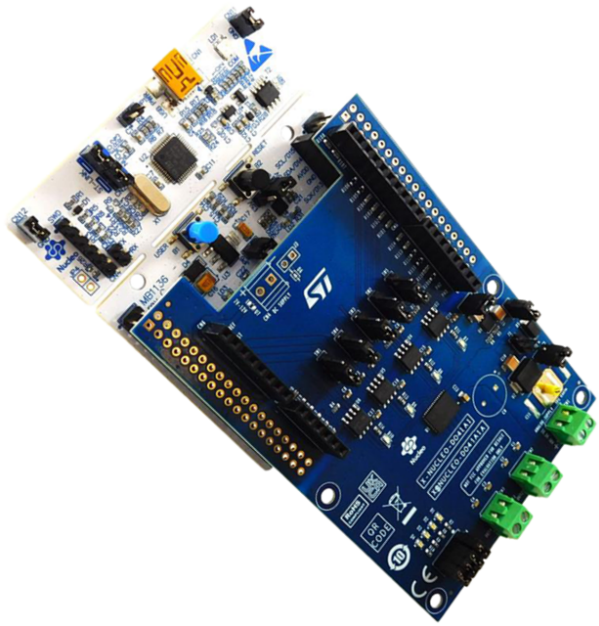


Figure 44. X-NUCLEO-DO41A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-DO40A1 or X-NUCLEO-DO41A1 expansion board on by connecting CN1 connector pin 1 (VCC) and pin 2 (GND) to the DC power supply (which must be set between 15 and 33 V).

Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

- Projects\NUCLEO-F401RE\Examples\DO40A1\FourChannels for NUCLEO-F401RE
- Projects\NUCLEO-F401RE\Examples\DO41A1\FourChannels for NUCLEO-F401RE

or

- Projects\NUCLEO-G431RB\Examples\DO40A1\FourChannels for NUCLEO-G431RB
- Projects\NUCLEO-G431RB\Examples\DO41A1\FourChannels for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example. Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

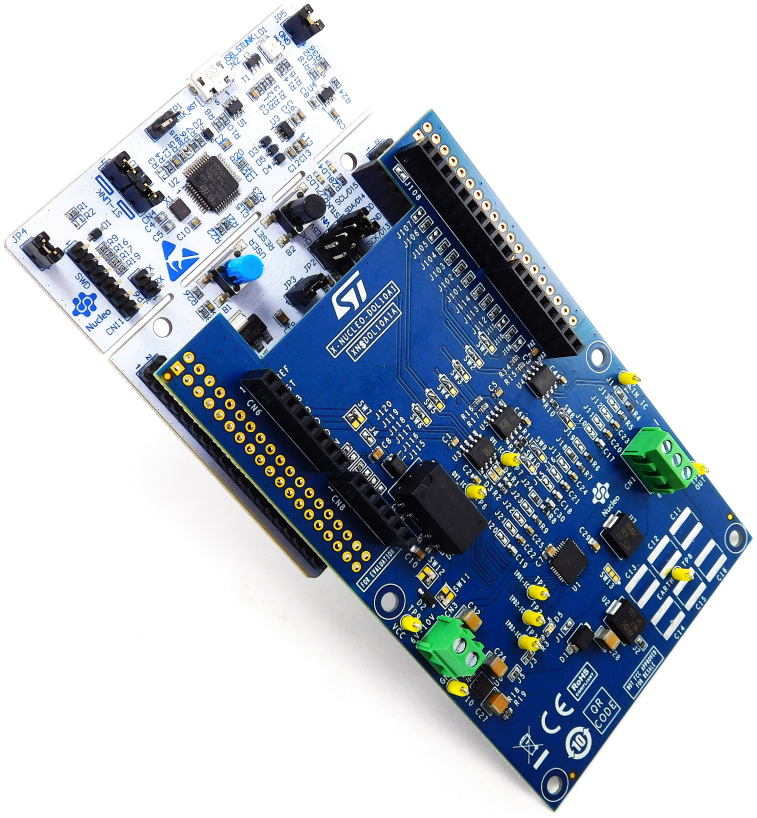
3.4.14 X-NUCLEO-DOL10A1 expansion board

The X-NUCLEO-DOL10A1 must be configured as follows:

- J1, J4 Open
- J10, J11, J12, J13, J15, J16, J17, J19, J20, J21, J22, J23 Open
- J108, J107, J105, J111, J109, J119, J117, J116, J115 Open
- J2, J3, J5 Closed
- J14, J18, J24 Closed
- J106, J104, J103, J102, J101, J114, J113, J112, J110, J120, J118 Closed
- SW1, SW2, SW3, SW4, SW5, SW6, SW7 Closed 1-2
- SW8, SW9, SW10, SW11, SW12 Closed 2-3
- SW13 all open

Step 1. Plug the X-NUCLEO-DOL10A1 expansion board on top of the STM32 Nucleo via the Arduino® UNO R3 connectors.

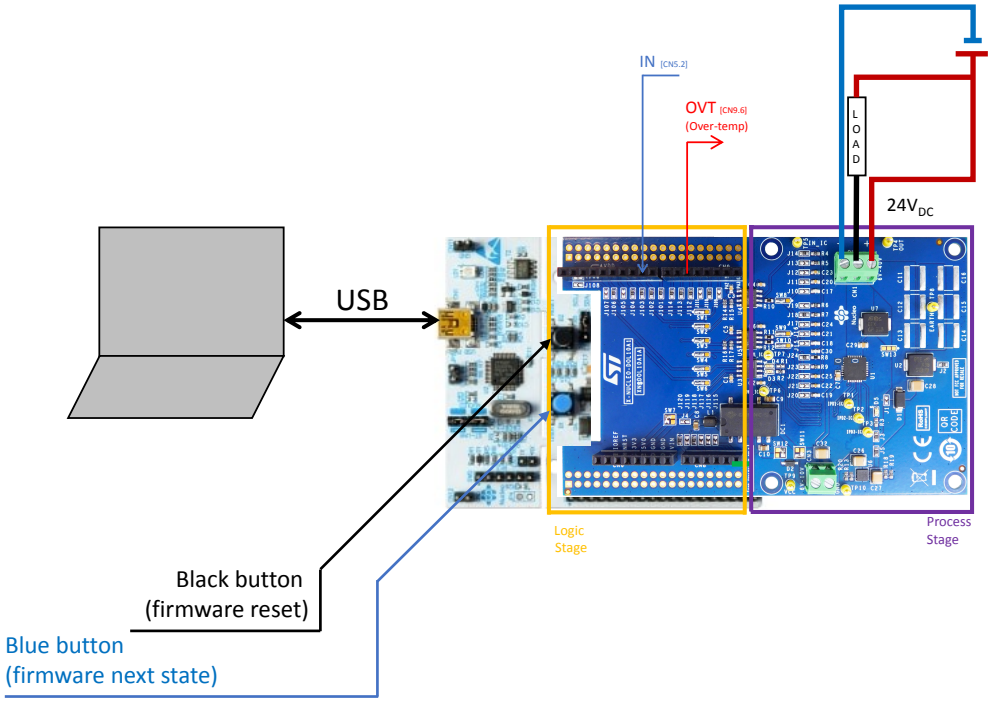
Figure 45. X-NUCLEO-DOL10A1 expansion board connected to an STM32 Nucleo development board



Step 2. Power the STM32 Nucleo board via USB cable between connector CN1 and a PC USB port.

Step 3. Power the X-NUCLEO-DOL10A1 expansion board on by connecting CN1 connector pin 1 (VCC) and pin 3 (GND) to the DC power supply (which must be set between 15 and 33 V), and connect the load between CN1 connector pin 2 (OUT) and the DC power supply positive pole. An example of connection setup is reported below:

Figure 46. X-NUCLEO-DOL10A1 expansion board connection setup



Step 4. Open your preferred toolchain (MDK-ARM from Keil, EWARM from IAR, or STM32CubeIDE)

Step 5. Depending on the STM32 Nucleo board and IDE used, open the software project from:

– Projects\NUCLEO-F401RE\Examples\DOL10A1\OneChannel_LS for NUCLEO-F401RE

or

– Projects\NUCLEO-G431RB\Examples\DOL10A1\OneChannel_LS for NUCLEO-G431RB

Step 6. Rebuild all files and load your image into target memory.

Step 7. Run the example. Each time the user button is pressed, a new command is applied at the digital output as described in Sample application description.

Revision history

Table 10. **Document revision history**

Date	Revision	Changes
09-Jun-2022	1	Initial release.
14-Dec-2022	2	Updated introduction, Section 2.1 Overview, Section 2.2 Architecture, Section 2.3 Folder structure, Section 2.3.1 BSPs, Section 2.3.1.1 STM32F4xx-Nucleo, STM32G4xx_Nucleo, Section 2.3.2 Projects, Section 3.2 Hardware setup, and Section 3.3 Software setup. Added Section 2.3.1.9 ips160hf, Section 2.3.1.16 OUT08A1 and OUT10A1, Section 2.4.3 X-NUCLEO-OUT08A1, X-NUCLEO-OUT10A1, Section 2.4.6 X-NUCLEO-OUT11A1, X-NUCLEO-OUT13A1, Section 2.4.7 X-NUCLEO-OUT12A1, X-NUCLEO-OUT14A1, Section 2.6.6 OUT11A1 and OUT13A1, Section 2.6.7 OUT12A1 and OUT14A1, Section 2.6.3 OUT08A1 and OUT10A1, X-NUCLEO-OUT08A1 expansion board, X-NUCLEO-OUT10A1 expansion board, X-NUCLEO-OUT11A1 expansion board, X-NUCLEO-OUT12A1 expansion board, X-NUCLEO-OUT13A1 expansion board, X-NUCLEO-OUT14A1 expansion board, Section 3.4.5 X-NUCLEO-OUT08A1 and X-NUCLEO-OUT10A1 expansion boards, Section 3.4.5 X-NUCLEO-OUT08A1 and X-NUCLEO-OUT10A1 expansion boards, Section 3.4.8 X-NUCLEO-OUT11A1 and X-NUCLEO-OUT13A1 expansion boards, and Section 3.4.9 X-NUCLEO-OUT12A1 and X-NUCLEO-OUT14A1 expansion boards.
27-Sep-2023	3	Updated introduction, *Section 2.1 Overview, Section 2.2 Architecture, Section 2.3.1 BSPs.* Removed section IPS1025H_2050H. Added *Section 2.3.1.3 ips1025h_32, Section 2.3.1.4 ips2050h, Section 2.3.1.5 ips2050h_32.* Updated *Section 2.3.1.2 ips1025h, Section 2.3.1.6 ips1025hf, Section 2.3.1.7 ips8160hq,* *Section 2.3.1.8 ips8160hq_1, Section 2.3.1.9 ips160hf, Section 2.3.1.11 iso808,* *Section 2.3.1.12 iso808_1 , Section 2.3.1.13 iso808a, Section 2.3.1.14 iso808a_1,* *Section 2.3.1.15 iso8200bq, Section 2.3.1.16 OUT08A1 and OUT10A1, Section 2.3.1.17 OUT03A1,* *OUT04A1, OUT05A1 and OUT06A1, Section 2.3.1.19 OUT11A1 and OUT13A1,* *Section 2.3.1.19 OUT11A1 and OUT13A1, Section 2.3.1.20 OUT12A1 and OUT14A1,* *Section 2.3.2 Projects.* Added *Section 2.3.1.2 ips1025h* and *Section 2.3.1.22 OUT01A2.* Updated *Section 2.4.1 X-NUCLEO-OUT03A1, X-NUCLEO-OUT04A1, Section 2.4.2 X-NUCLEO-OUT05A1,*

Date	Revision	Changes
		X-NUCLEO-OUT06A1, Section 2.4.3 X-NUCLEO-OUT08A1, X-NUCLEO-OUT10A1, *Section 2.4.4 X-NUCLEO-OUT09A1, X-NUCLEO-OUT19A1, Section 2.4.5 X-NUCLEO-OUT15A1,* *Section 2.4.6 X-NUCLEO-OUT11A1, X-NUCLEO-OUT13A1, Section 2.4.7 X-NUCLEO-OUT12A1, X-NUCLEO-OUT14A1,* *Section 2.4.8 X-NUCLEO-OUT01A2, Section 2.5 APIs, Section 2.6 Sample application description,* *Section 3.1 Hardware description, Section 3.2 Hardware setup,* *Section 3.3 Software setup, Section 3.4 Board setup.* Added *Section 3.1.2 X-NUCLEO-OUT01A2 expansion board*
02-Nov-2023	4	Updated *Section Introduction, Section 2.1: Overview,* *Section 2.2: Architecture, Section 2.3: Folder structure,* *Section 2.3.1: BSPs, Section 2.3.2: Projects,* *Section 2.5: APIs* and *Section 3.3: Software setup*. Added *Section 2.3.1: BSPs, Section 2.3.1.16: ips8200hq,* *Section 2.3.1.17: ips8200hq_1, Section 2.3.1.26: OUT16A1 and OUT17A1,* *Section 2.4.9: X-NUCLEO-OUT16A1, X-NUCLEO-OUT17A1,* *Section 2.6.9: OUT16A1 and OUT17A1,* *Section 3.4.11: X-NUCLEO-OUT16A1 and X-NUCLEO-OUT17A1* *expansion boards, Section 3.1.16: X-NUCLEO-OUT16A1* expansion board and *Section 3.1.17: X-NUCLEO-OUT17A1 expansion board*.
22-Jan-2024	5	Updated *Section Introduction, Section 2.1: Overview,* *Section 2.2: Architecture, Section 2.3.1: BSPs,* *Section 2.3.2: Projects, Section 2.5: APIs* and *Section 3.3: Software setup.* Added *Section 2.3.1.18: ips4260lm,* *Section 2.3.1.27: OUT07A1, Section 2.4.10: X-NUCLEO-OUT07A1,* *Section 2.6.10: OUT07A1, Section 3.1.7: X-NUCLEO-OUT07A1 expansion board* and *Section 3.4.5: X-NUCLEO-OUT07A1 expansion board.*
24-Jul-2024	6	Added reference to STM32CubeMX. Updated Sections *Section 2.1: Overview, Section 2.3: Folder structure,* *Section 2.3.1: BSPs, Section 2.3.2: Projects,* *Section 2.4: Software required resources, Section 2.5: APIs,* Sample application description, *Section 3.4.2: X-NUCLEO-OUT01A2* *expansion board, Section 3.4.4: X-NUCLEO-OUT03A1* *and X-NUCLEO-OUT04A1 expansion boards,* *Section 3.4.5: X-NUCLEO-OUT05A1 and X-NUCLEO-OUT06A1* *expansion boards, Section 3.4.6: X-NUCLEO-OUT07A1* *expansion board, Section 3.4.7: X-NUCLEO-OUT08A1* *and X-NUCLEO-OUT10A1 expansion boards,*

Date	Revision	Changes
		Section 3.4.8: X-NUCLEO-OUT09A1 and X-NUCLEO-OUT19A1 *expansion boards , Section 3.4.11: X-NUCLEO-OUT15A1* *expansion board, Section 3.4.9: X-NUCLEO-OUT11A1* *and X-NUCLEO-OUT13A1 expansion boards,* *Section 3.4.10: X-NUCLEO-OUT12A1 and X-NUCLEO-OUT14A1* *expansion boards, Section 3.4.12: X-NUCLEO-OUT16A1* *and X-NUCLEO-OUT17A1 expansion boards*
06-Dec-2024	7	Updated *Section Introduction, Section 2.1: Overview,* *Section 2.2: Architecture, Section 2.3.1: BSPs,* *Section 2.3.2: Projects, Section 2.5: APIs,* *Section 2.6: Sample application description,* *Section 3.3: Software setup* Added *Section 2.3.1.19: iso8200aq,* *Section 2.3.1.20: ips4140hq, Section 2.3.1.21: ips4140hq_1,* *Section 2.3.1.32: OUT02A1, Section 2.3.1.33: DO40A1 and* *DO41A1, Section 2.4.2: X-NUCLEO-OUT02A1,* *Section 2.4.12: X-NUCLEO-DO40A1 and X-NUCLEO-DO41A1,* *Section 2.6.2: OUT02A1, Section 2.6.12: DO40A1* *and DO41A1, Section 3.1.3: X-NUCLEO-OUT02A1 expansion* *board, Section 3.1.20: X-NUCLEO-DO40A1 expansion* *board , Section 3.1.21: X-NUCLEO-DO41A1 expansion* *board , Section 3.4.3: X-NUCLEO-OUT02A1 expansion* *board, Section 3.4.13: X-NUCLEO-DO40A1 and X-NUCLEO-DO41A1 expansion boards*
22-Jul-2025	8	Updated Section Introduction, Section 2.1: Overview, Section 2.2: Architecture, Section 2.3.1: BSPs, Section 2.3.2: Projects, Section 2.5: APIs, Section 3.3: Software setup Added Section 2.3.1.22: ips1050lq, Section 2.3.1.34: DOL10A1, Section 2.4.13: X-NUCLEO-DOL10A1, Section 2.6.13: DOL10A1, Section 3.1.22: X-NUCLEO-DOL10A1 expansion board , Section 3.4.14: X-NUCLEO-DOL10A1 expansion board

Contents

List of tables

List of figures

IMPORTANT NOTICE – READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice.

In the event of any conflict between the provisions of this document and the provisions of any contractual arrangement in force between the purchasers and ST, the provisions of such contractual arrangement shall prevail.

The purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgment.

The purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of the purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

If the purchasers identify an ST product that meets their functional and performance requirements but that is not designated for the purchasers' market segment, the purchasers shall contact ST for more information.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.