UM1963
User manual

Getting started with the X-CUBE-SPN2 two-axis stepper motor driver software expansion for STM32Cube

Introduction

This document describes how to get started with the X-CUBE-SPN2 software expansion for STM32Cube.

X-CUBE-SPN2 provides complete middleware software support to develop an L6470-based application using the STM32 Nucleo platform and the two-axis stepper motor driver expansion board based on L6470 (X-NUCLEO-IHM02A1), within the STM32Cube software environment.

With this expansion board, any STM32 user can easily develop low voltage motor control applications for stepper motors. This software is highly portable across the range of MCU families, thanks to STM32Cube. The package contains a sample application enabling the motion control of two stepper motors. The software expands the range of STM32Cube-based packages.

Contents

1 What is STM32Cube?

1.1 STM32Cube overview

The STMCube™ initiative was developed by STMicroelectronics to help developers by reducing development effort, time and cost. STM32Cube covers the STM32 portfolio.

STM32Cube Version 1.x includes:

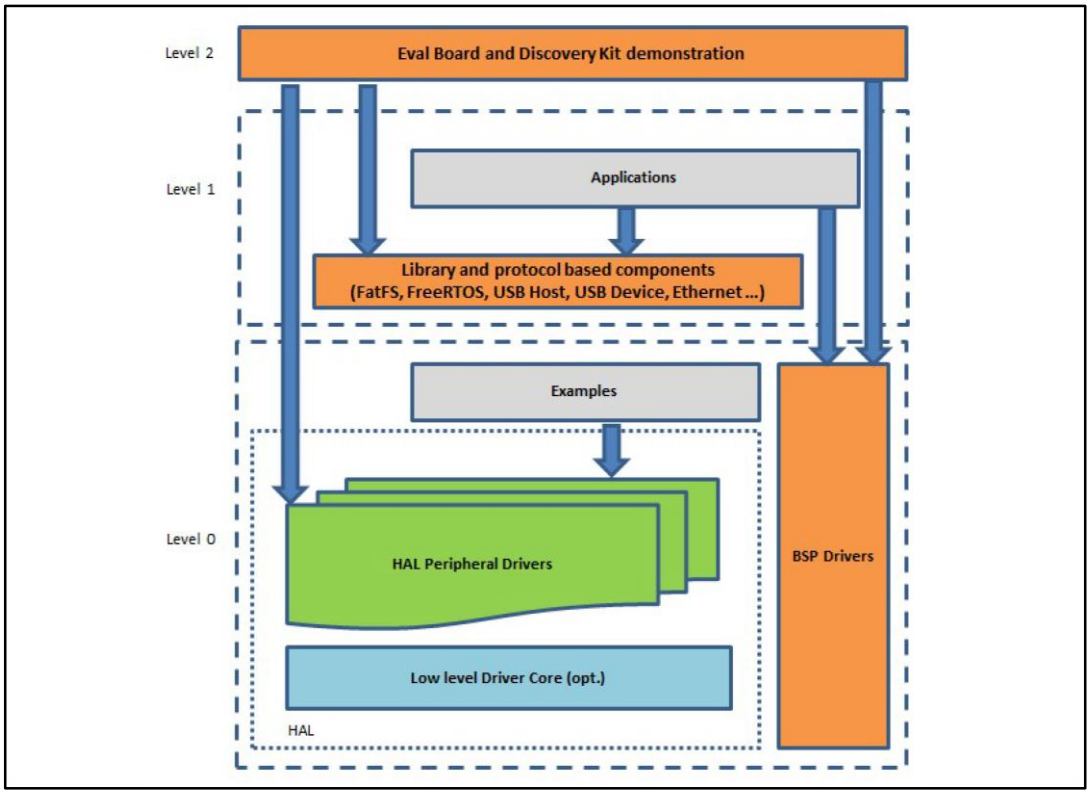
- The STM32CubeMX graphical software configuration tool that allows the generation of C initialization code using graphical wizards.
- A comprehensive embedded software platform specific to each series (such as STM32CubeF4 for the STM32F4 series)
  - The STM32Cube HAL embedded abstraction layer software, ensuring maximized portability across the STM32 portfolio
  - A consistent set of middleware components such as RTOS, USB, TCP/IP, Graphics
  - All embedded software utilities with a full set of examples

Information regarding STM32Cube is available on st.com at: http://www.st.com/stm32cube.

1.2 STM32Cube architecture

The STM32Cube firmware solution is built around three independent levels that can easily interact with one another, as described in the diagram below:

Figure 1: Firmware architecture



Level 0: This level is divided into three sub-layers:

- Board Support Package (BSP): this layer offers a set of APIs relative to the hardware components in the hardware boards (Audio codec, IO expander, Touchscreen, SRAM driver, LCD drivers. etc…) and composed of two parts:
  - Component: is the driver relative to the external device on the board and not related to the STM32, the component driver provides specific APIs to the external components of the BSP driver, and can be ported on any other board.
  - BSP driver: links the component driver to a specific board and provides a set of easy to use APIs. The API naming convention is BSP_FUNCT_Action(): e.g., BSP_LED_Init(), BSP_LED_On().

It is based on modular architecture allowing is to be easily ported on any hardware by just implementing the low level routines.

- Hardware Abstraction Layer (HAL): this layer provides the low level drivers and the hardware interfacing methods to interact with the upper layers (application, libraries and stacks). It provides generic, multi-instance and function-oriented APIs to help offload user application development time by providing ready to use processes. For example, for the communication peripherals (I2S, UART, etc.) it provides APIs for peripheral initialization and configuration, data transfer management based on polling, interrupt or DMA processes, and communication error management. The HAL Drivers APIs are split in two categories: generic APIs providing common, generic functions to all the STM32 series and extension APIs which provide special, customized functions for a specific family or a specific part number.
- Basic peripheral usage examples: this layer houses the examples built around the STM32 peripherals using the HAL and BSP resources only.

Level 1: This level is divided into two sub-layers:

- Middleware components: set of libraries covering USB Host and Device Libraries, STemWin, FreeRTOS, FatFS, LwIP, and PolarSSL. Horizontal interaction among the components in this layer is performed directly by calling the feature APIs, while vertical interaction with low-level drivers is managed by specific callbacks and static macros implemented in the library system call interface. For example, FatFs implements the disk I/O driver to access a microSD drive or USB Mass Storage Class.
- Examples based on the middleware components: each middleware component comes with one or more examples (or applications) showing how to use it. Integration examples that use several middleware components are provided as well.

Level 2: This level is a single layer with a global, real-time and graphical demonstration based on the middleware service layer, the low level abstraction layer and basic peripheral usage applications for board-based functions.

2 X-CUBE-SPN2 software, expansion for STM32Cube

2.1 Overview

The X-CUBE-SPN2 software package expands the STM32Cube functionality.

The key features of the X-CUBE-SPN2 are.

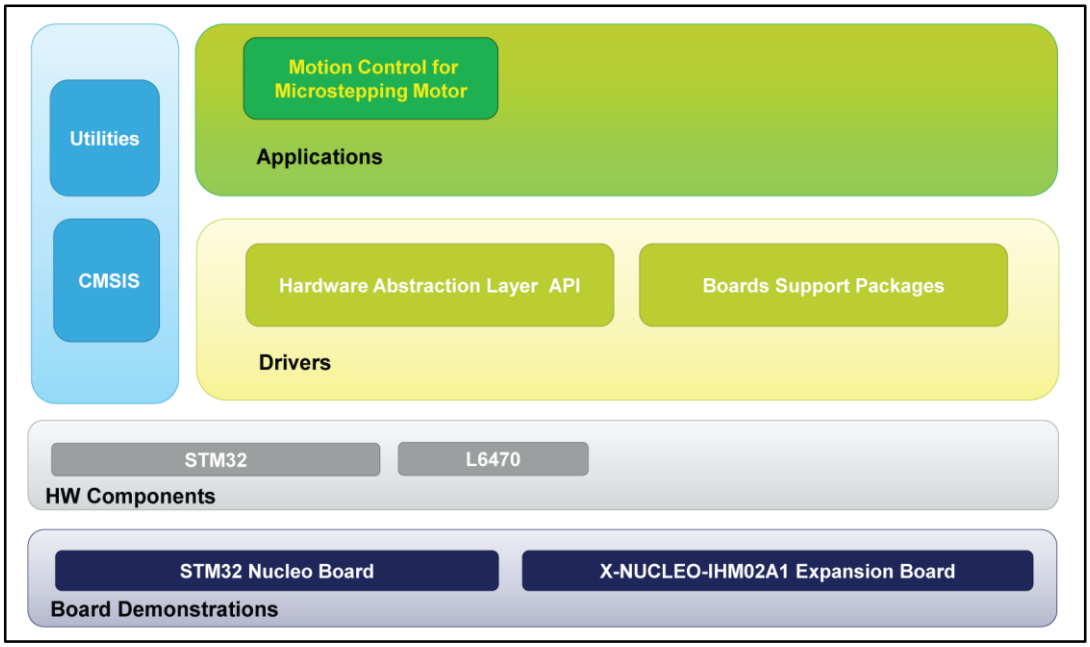
- Software library to build applications using a fully integrated micro stepping motor driver with motion engine and SPI (L6470)
- High portability across different MCU families, thanks to STM32Cube
- A sample application driving two stepper motors
- Free user-friendly license terms
- Example implementations available for the X-NUCLEO-IHM02A1 board plugged into a NUCLEO-F401RE, NUCLEO-F302R8 or NUCLEO-F072RB board. The example application can be developed for further experimentation with the code.

2.2 Architecture

This software is a fully compliant expansion for STM32Cube enabling development of applications using motor drivers. The software is based on the hardware abstraction layer for the STM32 microcontroller, STM32CubeHAL. The package extends STM32Cube by providing a Board Support Package (BSP) for the dual stepper motor driver expansion board.

The software layers used by the application software to access the two-axis stepper motor driver expansion board are:

Figure 2: X-CUBE-SPN2 software architecture



- The STM32Cube HAL driver layer, providing simple, generic and multi-instance APIs (application programming interfaces) to interact with the upper layers (application, libraries and stacks). It includes generic and extension APIs and is based on a generic architecture which allows the layers built on it (such as the middleware layer) to implement their functions without dependence on the specific hardware configuration

of a given Microcontroller Unit (MCU). This structure improves library code reusability and guarantees high portability across other devices.

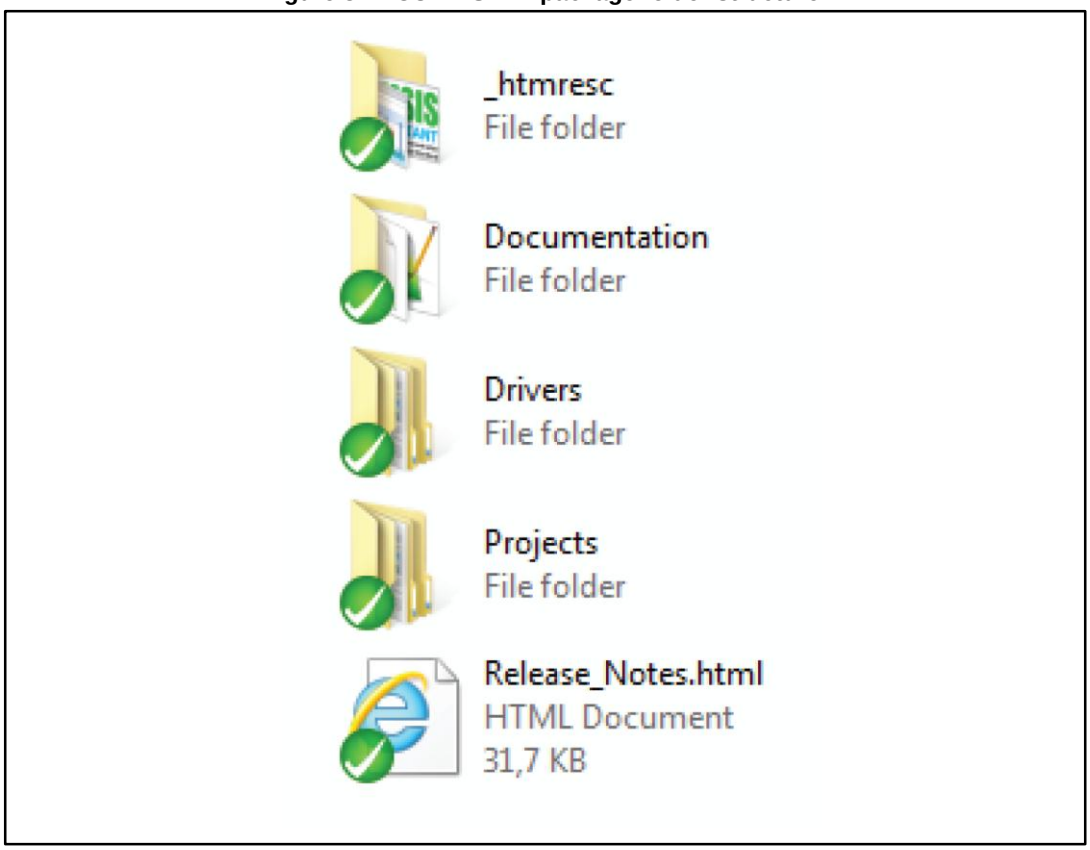
- The Board Support Package (BSP) layer, providing supporting software for the STM32 Nucleo board peripherals, except for the MCU. It has a set of APIs to provide a programming interface for certain board-specific peripherals (e.g., the LED, the user button etc.) and allows identification of the specific board version. For the dual stepper motor driver expansion board, it provides support for initializing and reading information from the dual stepper motor driver (L6470).

Figure 2: "X-CUBE-SPN2 software architecture" outlines the software architecture of the package.

2.3 Folders structure

This section provides an overview of the package folder structure. The figure below shows the architecture of the package.

Figure 3: X-CUBE-SPN2 package folder structure



The following folders are included in the software package:

- The Documentation folder contains a compiled HTML file generated from the source code and detailed documentation regarding the software components and APIs.
- The Drivers folder contains the HAL drivers, the board-specific drivers for each supported board or hardware platform, including those for the on-board components and the CMSIS layer, which is a vendor-independent hardware abstraction layer for the Cortex-M processor series.
- The Projects folder contains a sample application for the NUCLEO-F401RE, NUCLEO-F302R8 and NUCLEO-F072RB platforms to drive two stepper motors; it is

provided with three development environments, IAR EWARM, Keil MDK-ARM and System Workbench for STM32 SW4STM32 (free).

2.4 APIs

Detailed technical information about the APIs available to the user can be found in the compiled HTML file "X-CUBE-SPN2.chm" in the "Documentation" folder of the software package, where all the functions and parameters are fully described.

2.5 Sample application description

An example application using the X-NUCLEO-IHM02A1 expansion board with the NUCLEO-F401RE or NUCLEO-F302R8 or NUCLEO-F072RB board is provided in the "Projects" directory. Ready-to-use projects are available for multiple IDEs.

The target of this application is to perform some controlled movements of the two stepper motors connected to the X-NUCLEO-IHM02A1, stacked on one of the above mentioned STM32 NUCLEO boards.

3 System setup guide

3.1 Hardware description

This section describes the hardware components needed for developing a L6470-based application.

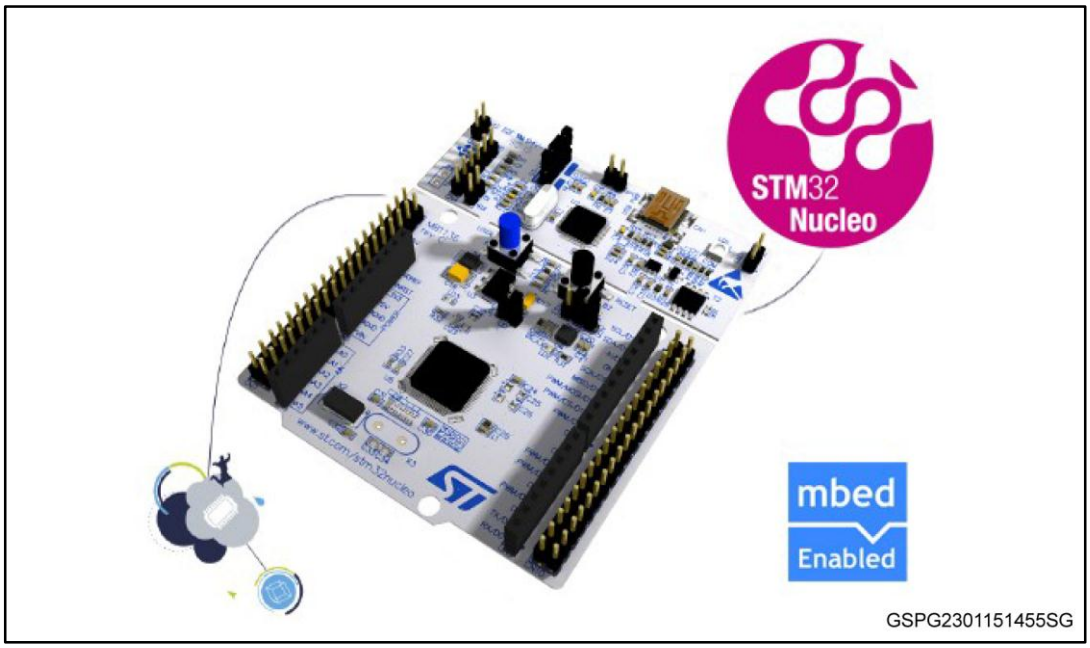
The following sub-sections describe the individual components.

3.1.1 STM32 Nucleo platform

The STM32 Nucleo boards provide an affordable and flexible way for users to try out new ideas and build prototypes with any of the STM32 microcontroller lines. The Arduino™ connectivity support and ST Morpho headers make it easy to expand the functionality of the STM32 Nucleo open development platform with a wide choice of specialized expansion boards. The STM32 Nucleo board does not require any separate probes as it integrates the ST-LINK/V2-1 debugger/programmer. The STM32 Nucleo board comes with the STM32 comprehensive software HAL library together with various packaged software examples.

Information about the STM32 Nucleo boards is available on www.st.com at http://www.st.com/stm32nucleo

Figure 4: STM32 Nucleo board



3.1.2 X-NUCLEO-IHM02A1 expansion board

The X-NUCLEO-IHM02A1 is a dual L6470 microstepping motor driver expansion board usable with the STM32 Nucleo board. It is also compatible with Arduino UNO R3 connector layout, and it is designed around two L6470 microstepping motor drivers connected in a daisy chain configuration. The X-NUCLEO-IHM02A1 interfaces with the STM32 MCU via SPI. The user can change the default device select pin by changing one resistor on the STM32 expansion board.

Figure 5: X-NUCLEO-IHM02A1 - two axes stepper motor driver expansion board

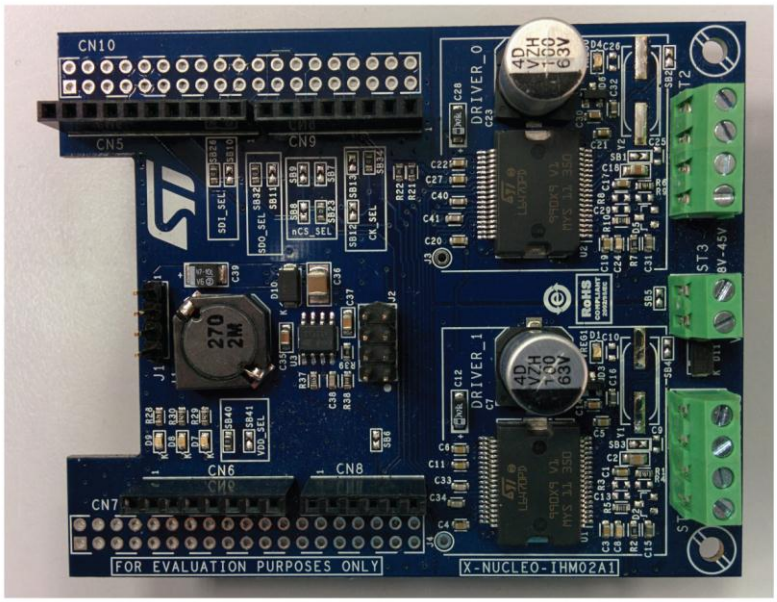
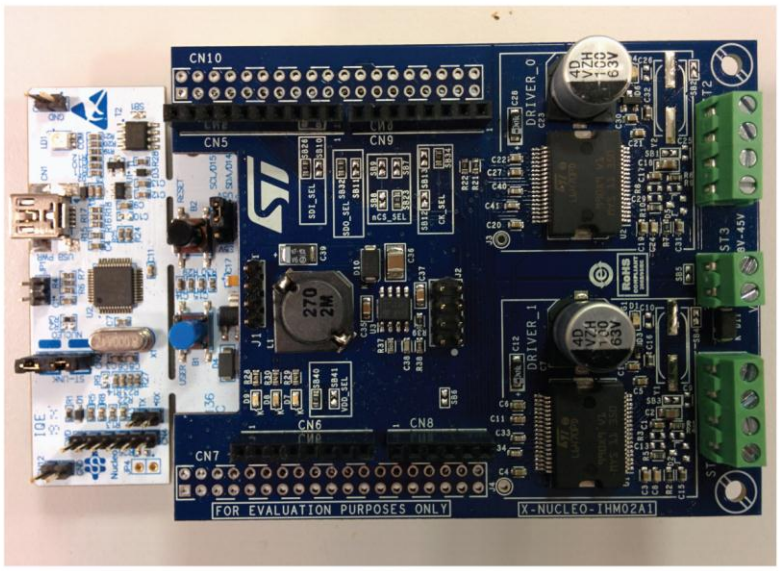


Figure 6: X-NUCLEO-IHM02A1 plugged on an STM32 Nucleo board



Information regarding the X-NUCLEO-IHM02A1 expansion board is available on www.st.com at http://www.st.com/x-nucleo.

3.2 Software description

The following software components are needed in order to set up a suitable development environment for creating an application based on the dual L6470 expansion board:

- X-CUBE-SPN2: an STM32Cube expansion for stepper motor application development. The X-CUBE-SPN2 firmware and associated documentation is available on ww.st.com.
- Development tool-chain and compiler. The STM32Cube expansion software supports the three following environments:

- IAR Embedded Workbench for ARM (EWARM),
- Keil Microcontroller Development Kit (MDK-ARM)
- System Workbench for STM32 (SW4STM32) [free]

3.3 Hardware software and system setup

This section describes the hardware and software setup procedures. It also describes the system setup needed for the above.

3.3.1 Hardware setup

The following hardware is needed to develop a L6470-based application:

1. One STM32 Nucleo board (suggested order code: NUCLEO-F401RE, NUCLEO-F302R8 or NUCLEO-F072RB);
2. One dual stepper motor driver expansion board based on L6470 (order code: X-NUCLEO-IHM02A1);
3. Two-axis stepper motors with L6470 compatible voltage and current;
4. One external power supply able to provide the right voltage and current for the two stepper motors;
5. One USB type A to Mini-B USB cable to connect the STM32 Nucleo board to the PC.

The example is set up to use motors like the Hybrid Stepping Motor 42BYGHM809 by Wantai Motor. If your motors have different parameters, modify the array named "MotorParameterInitData" in the "params.c" source file.

If the NUCLEO-F401RE is used, the SB15 has to be removed to properly use the SPI1.

3.3.2 Software setup

This section lists the minimum requirements for the developer to setup the SDK, run the sample application and customize applications PC.

3.3.3 Development tool chains and compilers

Select one of the Integrated Development Environments supported by the STM32Cube expansion software and follow the system requirements and setup information provided by the same IDE provider.

3.3.4 RS232 terminal

The user can chose from different software concerning the RS232 Terminal. This example uses PuTTY, a free, open-source terminal emulator, serial console and network file transfer application.

PuTTY is a very small program able to run on these operating systems:

- Microsoft Windows 95, 98, NT, ME, 2000, XP, 7 and 8;
- Unix.

3.3.5 System setup guide

This section describes how to setup different hardware components before writing and executing an application on the STM32 Nucleo board with the dual stepper motor driver expansion board.

3.3.6 STM32 Nucleo and dual L6470 expansion board setup

The STM32 Nucleo board integrates the ST-LINK/V2-1 debugger/programmer. The developer can download the relevant version of the ST-LINK/V2-1 USB driver (according to the Microsoft Windows OS) by searching STSW-LINK008 or STSW-LINK009 on www.st.com (according to the Microsoft Windows OS).

The X-NUCLEO-IHM02A1 two-axis stepper motor driver expansion board based on L6470 can be easily connected to the STM32 Nucleo board through the Arduino UNO R3 extension connector, see *Figure 6: "X-NUCLEO-IHM02A1 plugged on an STM32 Nucleo board"*. The two mounted L6470 stepper motor drivers are capable of interfacing with the external STM32 microcontroller on the Nucleo board via Serial Peripheral Interface (SPI). The two drivers are mounted in daisy chain configuration.

4 Send L6470 application command via UART

This section describes how to send any application command to both L6470s mounted on the X-NUCLEO-IHM02A1.

To try it, open the library (X-CUBE-SPN2) and uncomment the line regarding the tag MICROSTEPPING_MOTOR_USART_EXAMPLE inside the file main.c (line 64). Then download the new compiled code into the right STM32 Nucleo board.

4.1 X-NUCLEO-IHM02A1 and RS232 Terminal

As already mentioned, PuTTY is used to send and receive data via UART.

It is assumed that the software is already installed on the PC.

1. Make sure that the selected nCS pin matches the selected one in the firmware (refer to file xnucleoihm02a1.h).
2. Stack only one board on an STM32 Nucleo board.
3. Connect two stepper motors to the X-NUCLEO-IHM02A1 and supply the expansion board.
4. Attach the STM32 Nucleo board to the PC via USB cable. When the STM32 Nucleo board is attached to any PC, the latter sees a new COM port. The user right install the right drivers, which can be downloaded from www.st.com.
5. Launch PuTTY and set the right parameters regarding serial connection in the configuration form, see *Figure 7: "PuTTY configuration Connection\\Serial"*).

Figure 7: PuTTY configuration Connection\\Serial

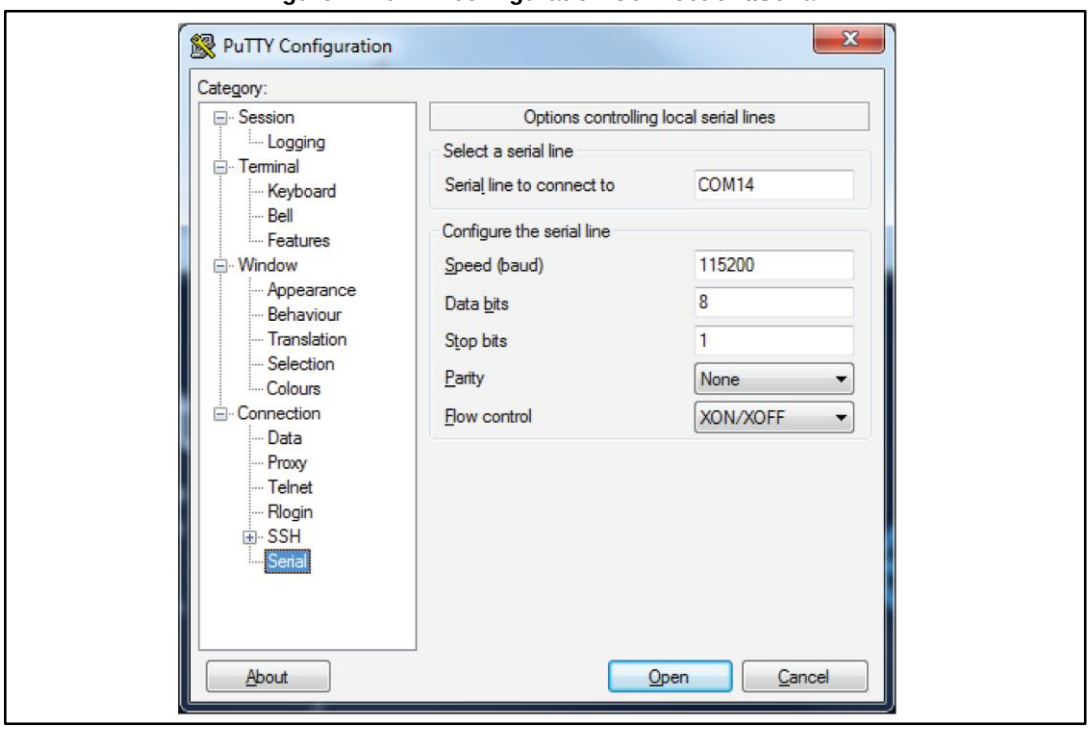
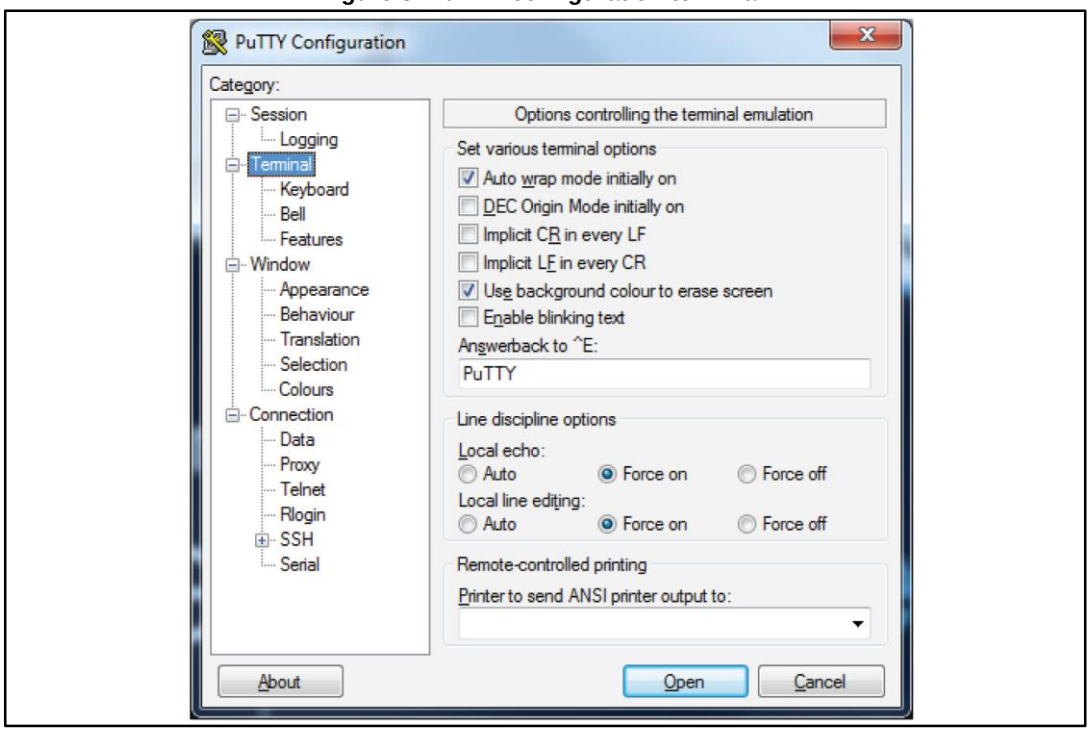


Figure 8: PuTTY configuration terminal

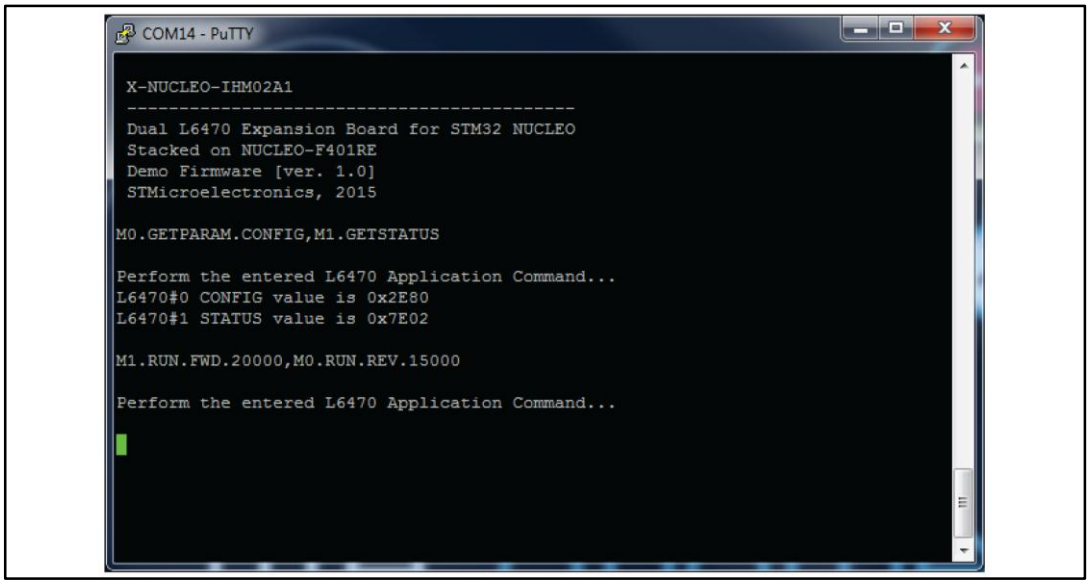


Enable the "local echo" in the terminal configuration, see *Figure 8: "PuTTY configuration terminal"*.

You can use the PC keyboard to send application commands to both L6470s mounted on the X-NUCLEO-IHM02A1. See *Figure 9: "PuTTY terminal"*.

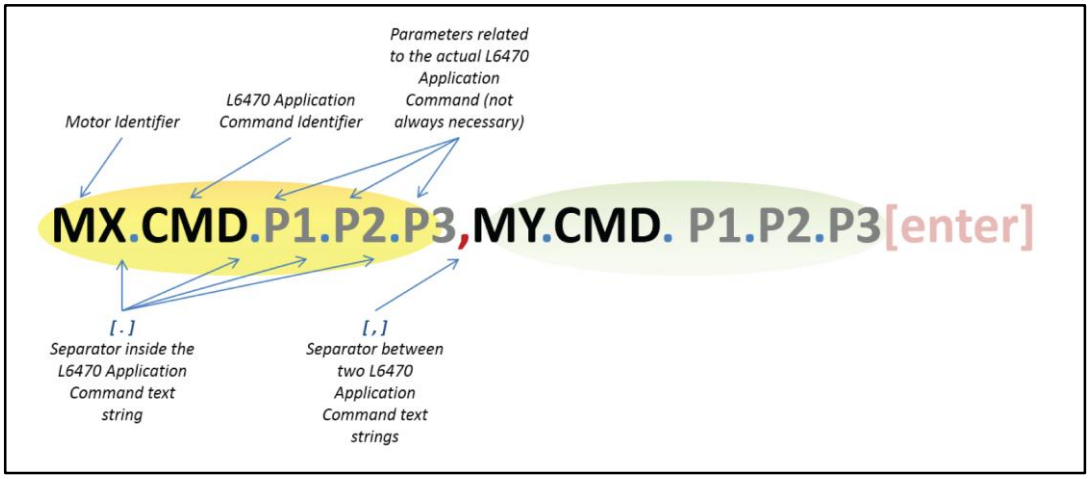
You can enter one or two L6470 Application Command text strings to address one or two motors at the same time. The second L6470 Application Command text string is not mandatory. The strings must be in upper case. *Figure 9: "PuTTY terminal"* shows a screenshot of a working PuTTY terminal.

Figure 9: PuTTY terminal



The string must adhere to the format in *Figure 10: "L6470 application command text string format"*.

Figure 10: L6470 application command text string format



The following tables explain the parts of the string.

Table 1: Motor identifier

Code	Meaning
M0	1st motor
M1	2nd motor

Table 2: L6470 application commands

CMD	P1	P2	P3	Action
NOP				Nothing
SETPARAM	PARAM	VALUE		Writes VALUE in PARAM register
GETPARAM	PARAM			Returns the stored value in PARAM register
RUN	DIR	SPD		Sets the target speed and the motor direction
STEPCLOCK	DIR			Puts the device in Step-clock mode and imposes DIR direction
MOVE	DIR	N_STEP		Makes N_STEP (micro)steps in DIR direction (not performable when motor is running)
GOTO	ABS_POS			Brings motor into ABS_POS position (minimum path)
GOTO_DIR	DIR	ABS_POS		Brings motor into ABS_POS position forcing DIR direction
GOUNTIL	ACT	DIR	SPD	Performs a motion in DIR direction with speed SPD until SW is closed, the ACT action is executed then a SoftStop takes place

CMD	P1	P2	P3	Action
RELEASESW	ACT	DIR		Performs a motion in DIR direction at minimum speed until the SW is released (open), the ACT action is executed then a HardStop takes place
GOHOME				Brings the motor into HOME position
GOMARK				Brings the motor into MARK position
RESETPOS				Resets the ABS_POS register (set HOME position)
RESETDEVICE				Device is reset to power-up conditions
SOFTSTOP				Stops motor with a deceleration phase
HARDSTOP				Stops motor immediately
SOFTHIZ				Puts the bridges in high impedance status after a deceleration phase
HARDHIZ				Puts the bridges in high impedance status immediately
GETSTATUS				Returns the STATUS register value

Table 3: L6470 registers

Register name	Register function
ABS_POS	Current position
EL_POS	Electrical position
MARK	Mark position
SPEED	Current speed
ACC	Acceleration
DEC	Deceleration
MAX_SPEED	Maximum speed
MIN_SPEED	Minimum speed
KVAL_HOLD	Holding KVAL
KVAL_RUN	Constant speed KVAL
KVAL_ACC	Acceleration starting KVAL
KVAL_DEC	Deceleration starting KVAL
INT_SPEED	Intersect speed
ST_SLP	Start slope
FN_SLP_ACC	Acceleration final slope
FN_SLP_DEC	Deceleration final slope
K_THERM	Thermal compensation factor
ADC_OUT	ADC output (the reset value is according to startup conditions)
OCD_TH	OCD threshold
STALL_TH	STALL threshold
FS_SPD	Full-step speed

Register name	Register function
STEP_MODE	Step mode
ALARM_EN	Alarm enable
CONFIG	IC configuration
STATUS	Status (the reset value is according to startup conditions)

Table 4: Motor directions

Code	Meaning
FWD	Forward
CPY	Reverse

Table 5: Action about ABS_POS [ACT]

Code	Meaning
RST	ABS_POS register is reset
CPY	ABS_POS register is copied into the MARK register

5 Revision history

Table 6: Document revision history

Date	Revision	Changes
23-Oct-2015	1	Initial release.

IMPORTANT NOTICE – PLEASE 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. 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 acknowledgement.

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 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.

ST and the ST logo are trademarks of ST. 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.