Digital Radio Projects

IP400 Experimenter Nucleo CC2 Radio Node

Document Number: IP400-CC2
Revision: 1.1a
Status: Released
Written By: Martin C. Alcock, M. Sc, MIEEE, VE6VH

Table of Contents

List of Tables

List of Figures

References

[1] gnu.org, "General Public Licence," [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html. [Accessed 25th February 2018].

[2] ST Microelectronics, "STM32 Nucleo-64 development board with STM32WL33CC MCU," [Online]. Available: https://www.st.com/en/evaluation-tools/nucleo-wl33cc2.html. [Accessed 26 01 2025].

[3] ST Microelectronics, "STM32WL33CC: Sub-GHz Wireless Microcontrollers.," ST Microelectronics, [Online]. Available: https://www.st.com/en/microcontrollers-microprocessors/stm32wl33cc.html. [Accessed 28 1 2025].

[4] ST Microelectronics, "Integrated Development Environment for STM32," [Online]. Available: https://www.st.com/en/development-tools/stm32cubeide.html. [Accessed 26 1 2025].

[5] AdaFruit Industries, "Adafruit Ultimate GPS Logger Shield," [Online]. Available: https://www.adafruit.com/product/1272#technical-details. [Accessed 26 1 2025].

Revision Status

Revision	Date	Description
1.0	March 24th, 2025	Separated from previous documentation, released
1.1	May 3rd, 2025	Added LPUART connection description
1.1a	May 6th, 2025	Corrected pinout for LPUART

Table 1 Revision status

Reference Documents

Author	Issue Date	Document Number	Description
M. Alcock	Jan 2025	IP400-PHY	IP400 Physical Layer Specification
M. Alcock	Mar 2025	IP400--SPI	IP400 Radio Node SPI Protocol specification

Table 2 Reference Documents

Intellectual Property Notice

Disclaimer

This document is a preliminary release for a product still in development and may be subject to change in future revisions. The software described herein may be subject to unpredictable behaviour without notice. You are advised to keep a can of RAID™ Ant, Roach and Program Bug killer handy. Spray liberally on the affected area when needed.

If any page in this document is blank, it is completely unintentional.

Introduction

The IP400 project was launched to experiment with digital mesh networking on the 400 MHz band, using commercial devices off the shelf. The first hardware to be utilized is the Nucleo-CC2 [2] evaluation module, which features an STM32WL33 microcontroller [3], and is readily available.

This module is designed for the experimenter, as it not only has the capability to develop code for the processor, but also has Arduino-style connectors that can be used as a platform for different types of experimentation, including adding a GPS receiver or other modules. A built-in debug capability enables downloading and software breakpoints to be inserted for debugging capabilities.

Software is developed using the STMCube IDE [4] development environment. This software is freely available and can be downloaded from the manufacturer's site, and runs on several operating system platforms, including Windows®, and Linux. Familiarity with that environment is suggested to use this module.

It can also be connected to a raspberry Pi using the SPI connections, to take advantage of code developed for the host.

ST Micro NUCLEO WL33CC2

Figure 1 Phase 1 Nucleo board

The module consists of two PCB's, one contains the STM32WL33 microcontroller, the second an STLink debugger with a type 'C' USB connection. To install the firmware, the STM32CubeIDE must be used. Download the zip file and source code from the Github site, unzip the project, import it into the IDE, and add the source files from the IP400 directory. Compile the code and run it using the IDE downloader.

Start a PuTTy session in serial mode to the first COM port on the debugger, set it to 115200 bits/sec and DEC VT100 emulation mode. This can remain in place, even when the code is restarted or the reset key on the board is hit.

The basic functionality includes a menu selection to set station or radio parameters, a mesh table builder, packet repeater, and a simple chat application.

Using the Second UAR/T

The second UAR/T (also know as LPUAR/T, or Low Power UAR/T), can be utilized for a variety of applications, such as a GPS receiver, or to implement a simple modem connection. The Nucleo board has several connectors, including the Arduino style which can support GPS receivers or any type of prototyping board.

Signal	WL33 Pin	GPIO	Morpho CN4
LPUART_TX	PB6	54	35
LPUART_RX	PB7	55	37
GND	-	-	32

Table 3 LPUART connections

Implementing a USB Connection

An external USB connection to the second LPUART can be implemented with an external module such as those illustrated in Figure 1Figure 2. Any evaluation board for a device from such manufacturers as Future Technology Devices International (FTDI), or Cypress can be utilized. The modules can be connected to the Nucleo board using the pins referenced in Table 3. Ensure that the Tx and Rx are crossed over correctly.

Figure 2 USB to UART Module examples

Connecting a GPS receiver

A GPS receiver may be connected to the Nucleo using the Arduino connectors. The recommended module is the 'Ultimate GPS logging shield' from Adafruit industries [5]. The connection requires a small board modification as shown in Figure 3.

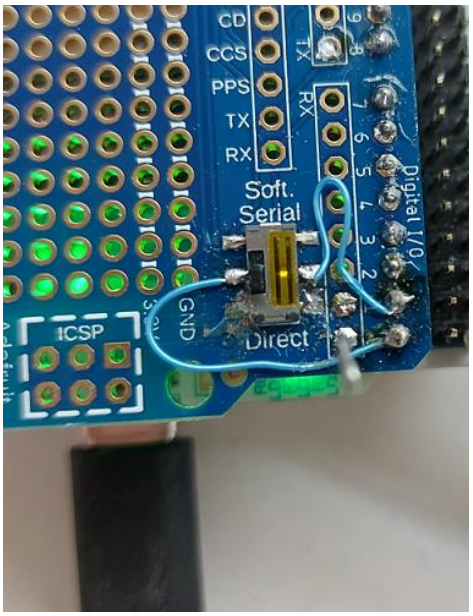


Figure 3 Adafruit GPS setup

The modification is as follows:

1. Using some soldering braid and an Exacto knife, remove the solder from both pins at the lower left hand and right hand side of the slide switch, then lift the pin or remove it completely.
2. With an ohmmeter, check there is no continuity between the left pin on the switch and the I/O pin labelled '1', and the right pin and the I/O pin labelled '0'.
3. Solder a piece of #30 Kynar wire to the centre pin on the left side, then to the pin labelled '0' on the connector, and a second wire from the right hand side to the pin labelled '1' on the connector.
4. Check continuity with an ohmmeter between the slide switch and the new pin locations.
5. Ensure that the switch is in the 'Direct' (towards the removed pins) position.

Connecting the Nucleo to a Raspberry Pi

As the same code runs both on the Pi HAT as on the Nucleo board, a Nucleo board can be connected to a RPI to offer the same functionality as the HAT board, however the code has to be compiled and installed on the Nucleo using the IDE. Unzip and import the WL33_NUCLEO_UART project, add the latest code to the workspace, compile and download to the Pi.

The connection is made using cable jumpers from CN4 on the Nucleo board's morpho connector, to the HAT connector on the raspberry Pi. The table below lists the signal name, it origin on the WL33 chip, the GPIO designation, and the pin where it can be found on the morpho, plus the corresponding pin on the Raspberry PI HAT connector.

Signal	WL33 Pin	GPIO	Morpho CN4	RPi
SPI_MOSI	PB8	33	15	19
SPI_MISO	PB9	34	13	21
SPI_SCK	PB11	31	11	23
GND	-	-	9	25
Test Point	PA6	-	3	-

Table 4 SPI connections from Nucleo to RPi

Figure 4 illustrates the Pi to Nucleo connections using jumpers.

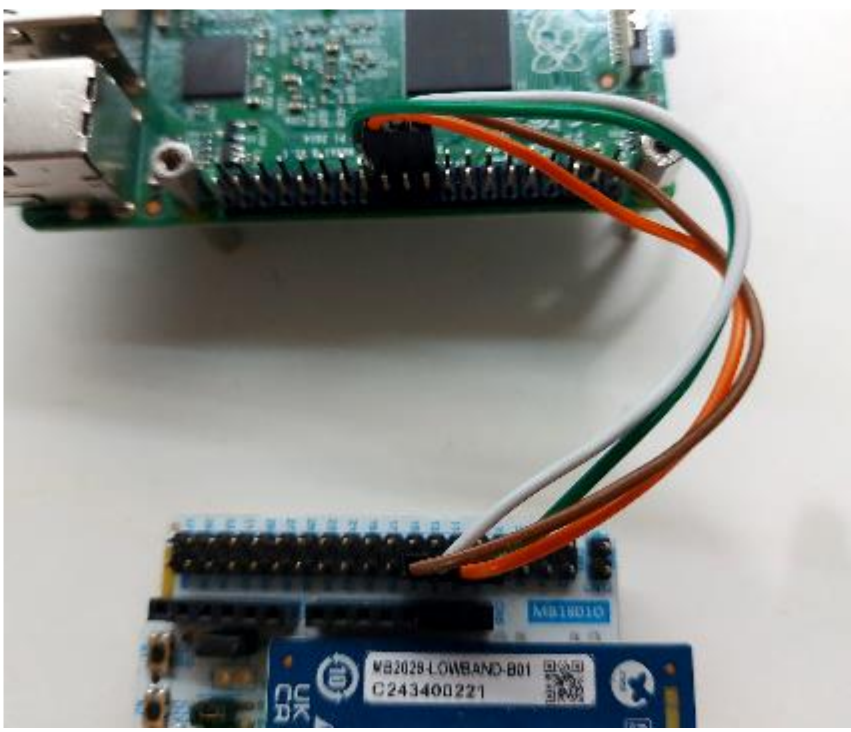


Figure 4 Pi and Nucleo connections

Test Point

A test point is provided at Pin 3 of the Morpho connector. This is toggled each time a data exchange occurs between the Pi and the Nucleo board. It can be connected to an oscilloscope and should appear as a square wave toggling every 20ms, or a 40 ms period.

The frequency of the toggling can be changed in the code, see the referenced document on the SPI specification.

Accessing the Menu

If you are not using the supplied image, there are three steps that need to be carried out to be able to connect to the application menu:

1. Run the configuration program 'raspi-config', and choose Interface Options, then Serial Port (I6). Turn off the login shell by answering 'no' to the first question. Enable the serial port by answering 'Yes' to the second question. Then re-boot the Pi.
2. Download and install minicom if you do not have it, with 'sudo apt-get install minicom'.
3. Launch minicom with the command 'minicom -b 115200 -D /dev/ttyAM0.

You may also want to enable SSH access if you have an ethernet connection to the Pi and PuTTY on a remote machine.

Arduino Connectors

Figure 5 illustrates the Arduino connector on the Nucleo main board.

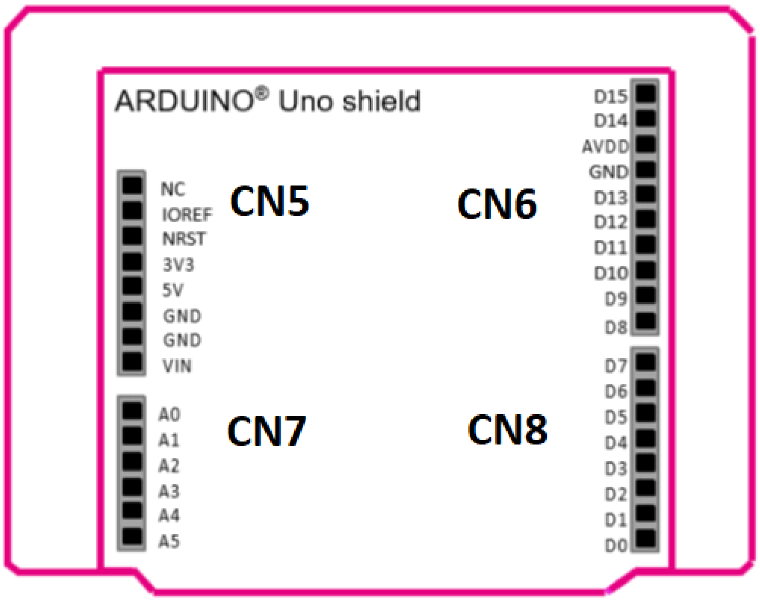


Figure 5 Arduino Connector

Left hand side connector pinouts (CN5 and CN7)

Connector	Pin Number	Pin Name	MCU Pin	Function
CN5	1	NC	-	
	2	3V3 (IOREF)	-	IOREF
	3	NRST	NRST	Low Active Reset
	4	3V3	-	3.3V output
	5	5V	-	5V
	6	GND	-	GND
	7	GND	-	GND
	8	VIN	-	Ext Supply 7-12V
CN7	1	A0	PB0	ADC_VINM1
	2	A1	PB1	ADC1_VINP1/LCA
	3	A2	PB2	ADC1_VINM0/LCB
	4	A3	PB3	ADC_VINP0
	5	A4	PB4	PVD_VIN
	6	A5	PB5	ADC_VINP3/LD3

Table 5 Left hand Arduino Connector Pinouts

Right hand side connectors (CN6 & CN8)

Connector	Pin Number	Pin Name	MCU Pin	Function
CN8	1	D0	PB7	LPUART1_RX
	2	D1	PB6	LPUART1_TX
	3	D2	PA9	USART1_TX
	4	D3	PA0	GPIO/B1
	5	D4	PA8	USART1_RX
	6	D5	PA13	GPIO
	6	D6	PA12	GPIO
	7	D7	PA4	GPIO
CN6	1	D8	PA5	GPIO
	2	D9	PA14	GPIO
	3	D10	PB10	SPI1_NSS
	4	D11	PB9	SPI1_MOSI
	5	D12	PB9	SPI1_MISO
	6	D13	PB11	SPI1_SCK
	7	GND		
	8	AVDD		
	9	D14	PA7	I2C2_SDA
	10	D15	PA6	I2C2_SCL

Table 6 Right hand Arduino Connector Pinouts

Node MCU Pin Usage

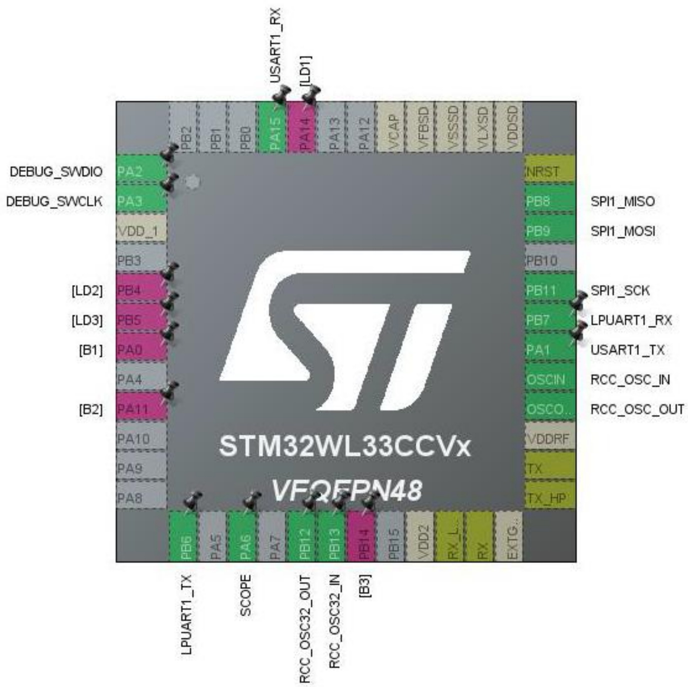


Figure 6 Node MCU Pin Usage

Table 7 lists the pins used by the node MCU application. The power and clock pins are not listed.

Pin	Name	Usage
PA2	SWDIO	Serial data used by debugger
PA3	SWCLK	Serial clock used by debugger
PA14	LD1	Blue LED
PB4	LD2	Green LED
PB5	LD3	Red LED
PA0	B1	Pushbutton 1
PA11	B2	Pushbutton 2
PB6	LPUART_TX	LPUART Tx data. Connected to Arduino CN8/D1
PA6	SCOPE	Scope trigger used by SPI module
PB14	B3	Pushbutton 3
PA1	USART1_TX	USART1 Tx. Connected to STLink VCOM port
PB7	LPUART_RX	LPUART Rx data. Connected to Arduino CN8/C0
PB11	SPI1_SCK	SPI Serial clock for Rpi connection
PB9	SPI1_MOSI	SPI Master out, slave in
PB8	SPI_MISO	SPI Master in, slave out
PA15	USART1_RX	USART1 Rx. Connected to STLink VCOM port

Table 7 Node MCU pin utilization

Using the STM32Cube IDE

If you are unfamiliar with the STM32Cube Integrated Development Environment [4], use the instructions in this chapter to compile and download the code.

Conditional Compilation

Globally used variables for conditional compilation for the are all contained in the include file config.h, Table 8 lists them and their effect on compilation.

Conditional	Value	Effect
_BOARD_TYPE	PI_BOARD	Compiles code for the raspberry pi board
	NUCLEO_BOARD	Compiles code for the Nucleo board
__ENABLE_GPS	0	Omits GPS code
	1	Includes GPS code for LPUART

Table 8 Conditional Compilation

Other modules may contain conditionals for debug purposes, which are local to that module only.

Compiling using Cube IDE

Import the correct project for your hardware platform and then add the most recent code to it. The steps are shown below.

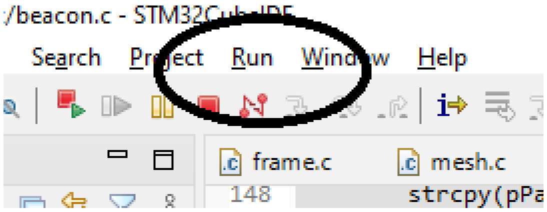
1. Step 1: Go to the File->Import menu, then choose General->File System, then click next.
2. Step2: Using the browse button, navigate to the IP400 directory that you downloaded from GitHub. Click on the 'Src' directory.
3. Step 3: Click the box in the left pane, and make sure the "Into Folder" is pointing to your project folder, then IP400/Src.
4. Step 4: Click Finish.
5. Step5: Repeat steps 2-4 for the 'Inc' directory.

Then use the 'build project' to build it.

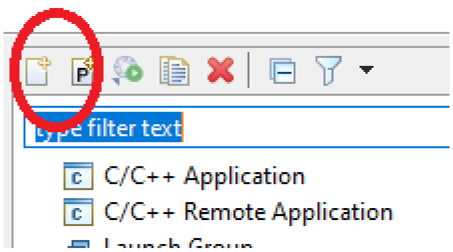
Using the debugger

There are seven steps involved in running from the IDE:

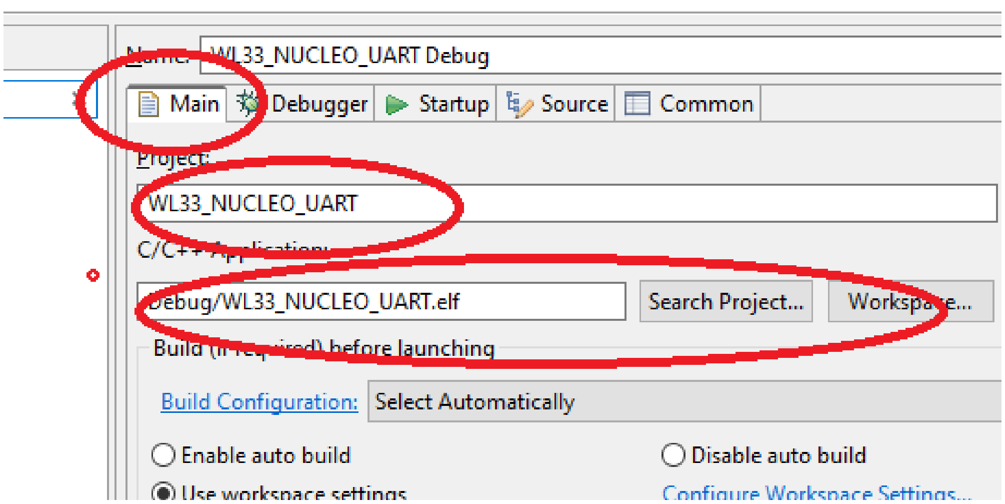
Step 1: Choose the RUN menu item, the Debug Configurations:



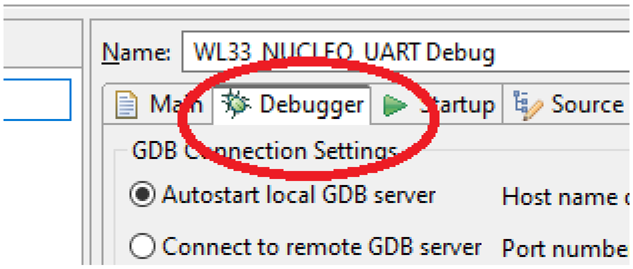
Step 2: A dialog box will launch. Click the 'new configuration' icon:



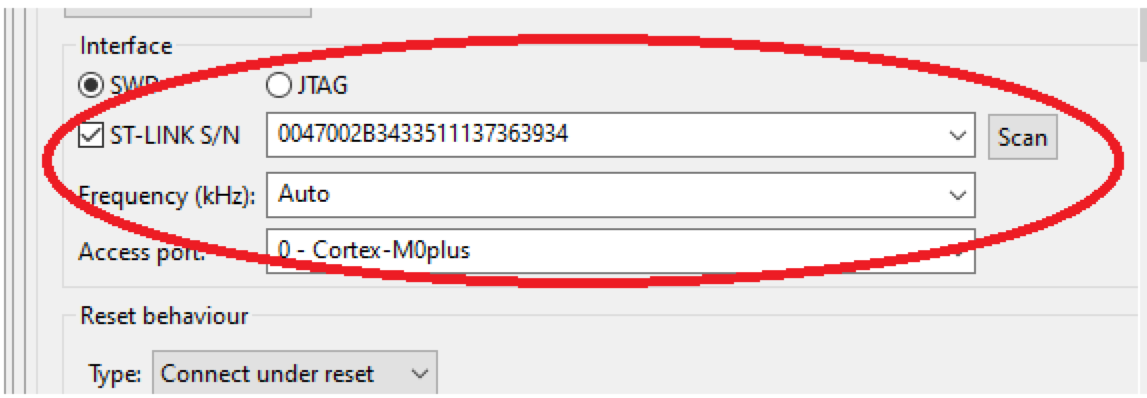
Step 3: on the Main tab, select your project and the .elf object file:



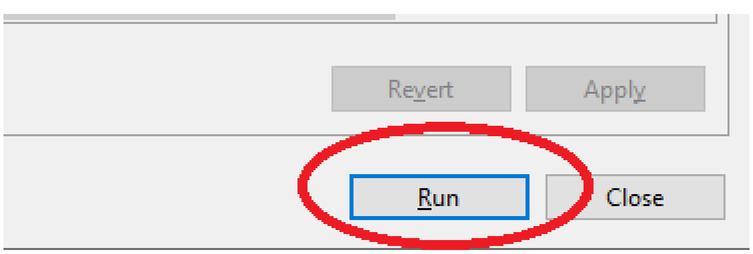
Step 4: Ensure your Nucleo is connected to your PC, the click the 'Debugger' tab.



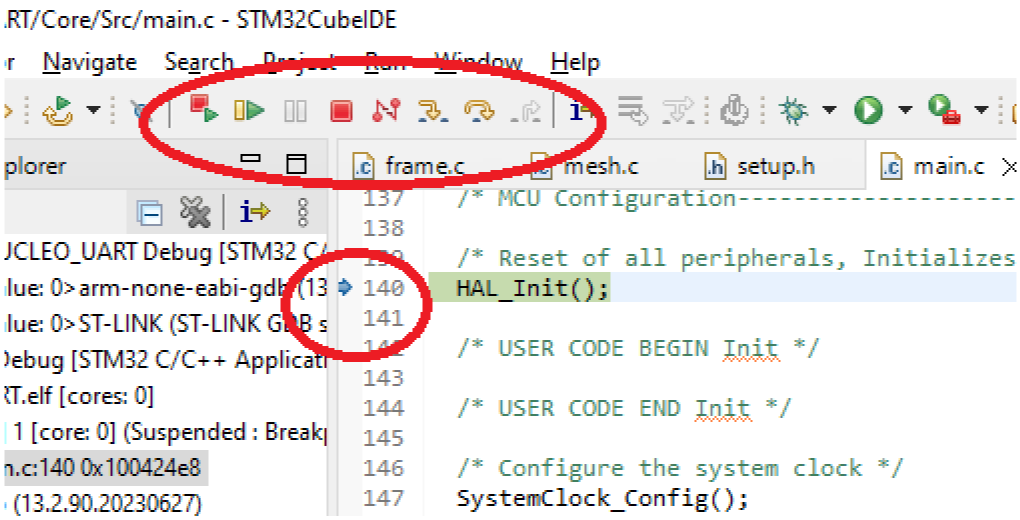
Step 5: Select the debugger tab, click the ST-LINK check box, and scan. Then the serial number of your Debugger.



Step 6: Click the 'Run' button.



Step 7: An editor will open and there will be an arrow next to the first line of executable code.



Click the green arrow to run it. The Red square will terminate it, and the combination. Will reload the code. Under the run menu you will find a restart that issues a reset to the device.