Land Tiger Project

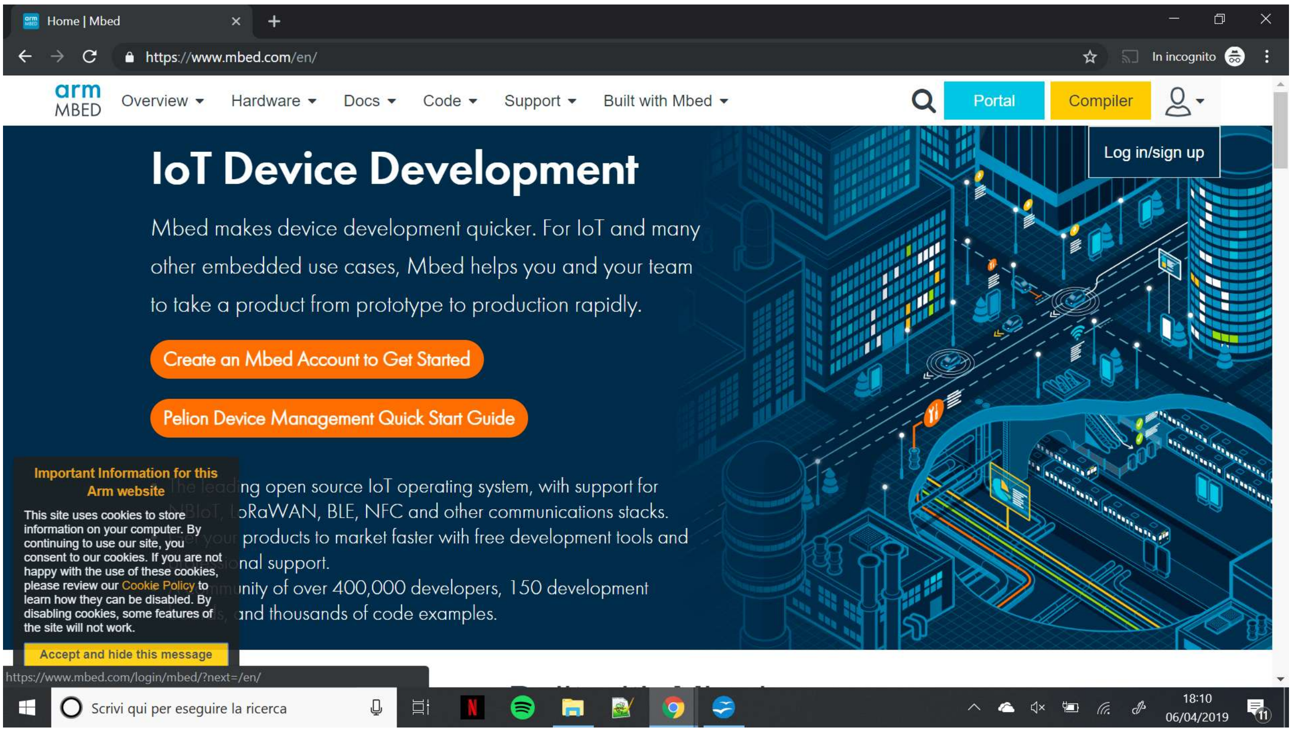
# Intro and Setup

For my project I choosed Mbed, an open source operating system made by Arm. It's a good operating system for IoT and embedded systems in general. Apart from a very active community, it has a wide variety of supported boards that act like a plug'n play devices and are very convenient to program.
Ofcourse the LandTiger is not one of the supported boards.
Fortunately Mbed support a board called “mbed LPC1768” that, as you can probably guess, has the same processor as that of the LandTiger.
I will now guide you through the various steps needed to create a program, compile it and flash it in to the board.
As we will see to flash the program is a bit more complex than “usual”, but if you get used to it, it will only take a couple of minutes for the entire procedure.

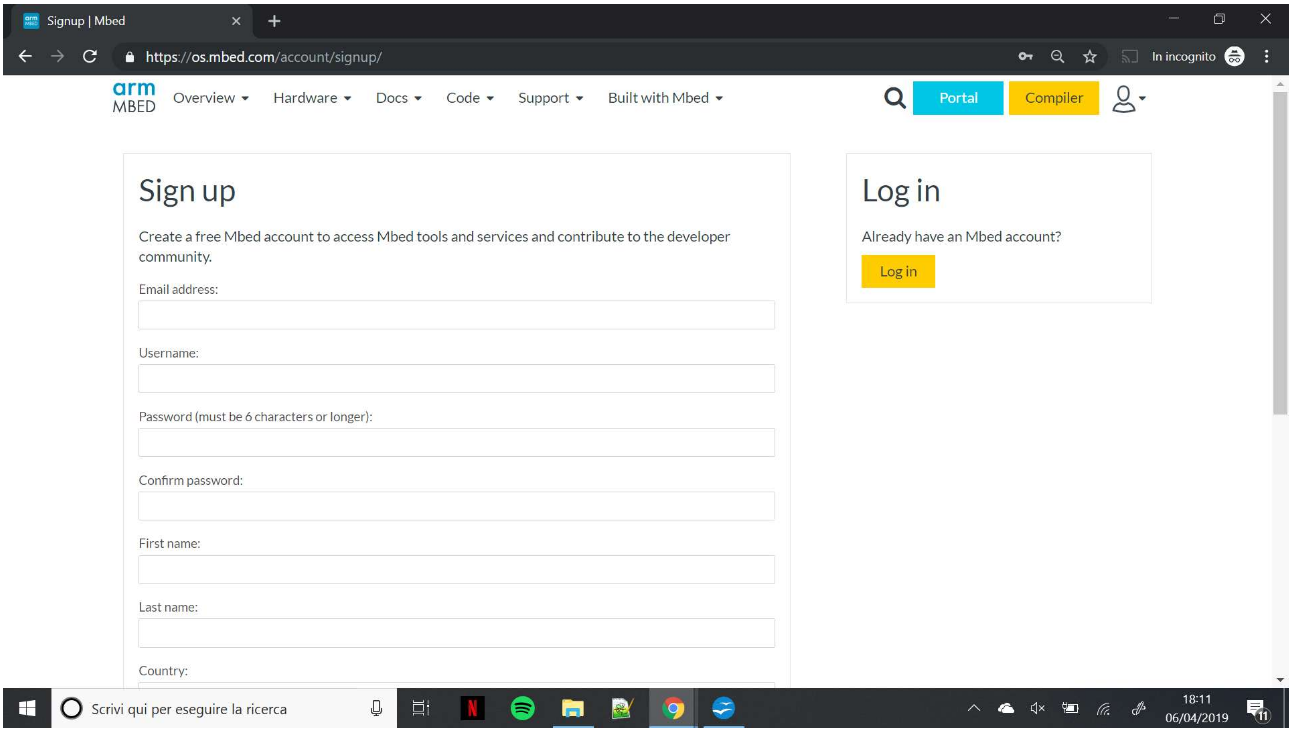
What do you need:

- A LandTiger board.
- One or two serial cable (if you want to use the board as a USB function like a mouse or a Keyboard).
- An Ulink 2 (or a cheaper USB to JTAG programmer) to flash your code into the board.
- Keil uVision 4 (the free ediction is perfectly fine).
- Hexplorer (https://sourceforge.net/p/hexplorer/wiki/Home/)
- An account on the Mbed official website (https://www.mbed.com/en/)
- A copy of the landtiger user manual uploaded by the user wim. This is a very good document since it shows you how the correspondence between the official supported “mbed LPC1768” and our LandTiger (https://os.mbed.com/media/uploads/wim/landtiger_v2.0_-_manual__v1.1.pdf).
- A good amount of patience.

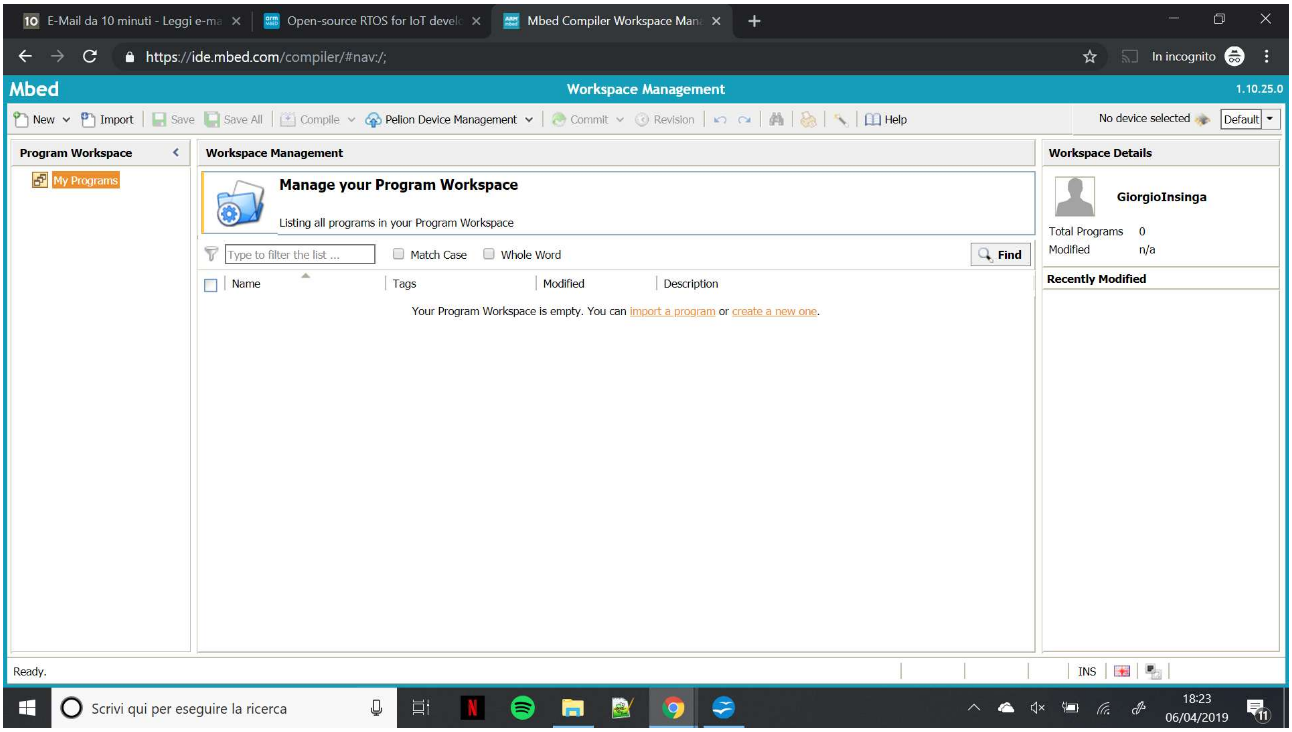
Let's see how to set everyhing:



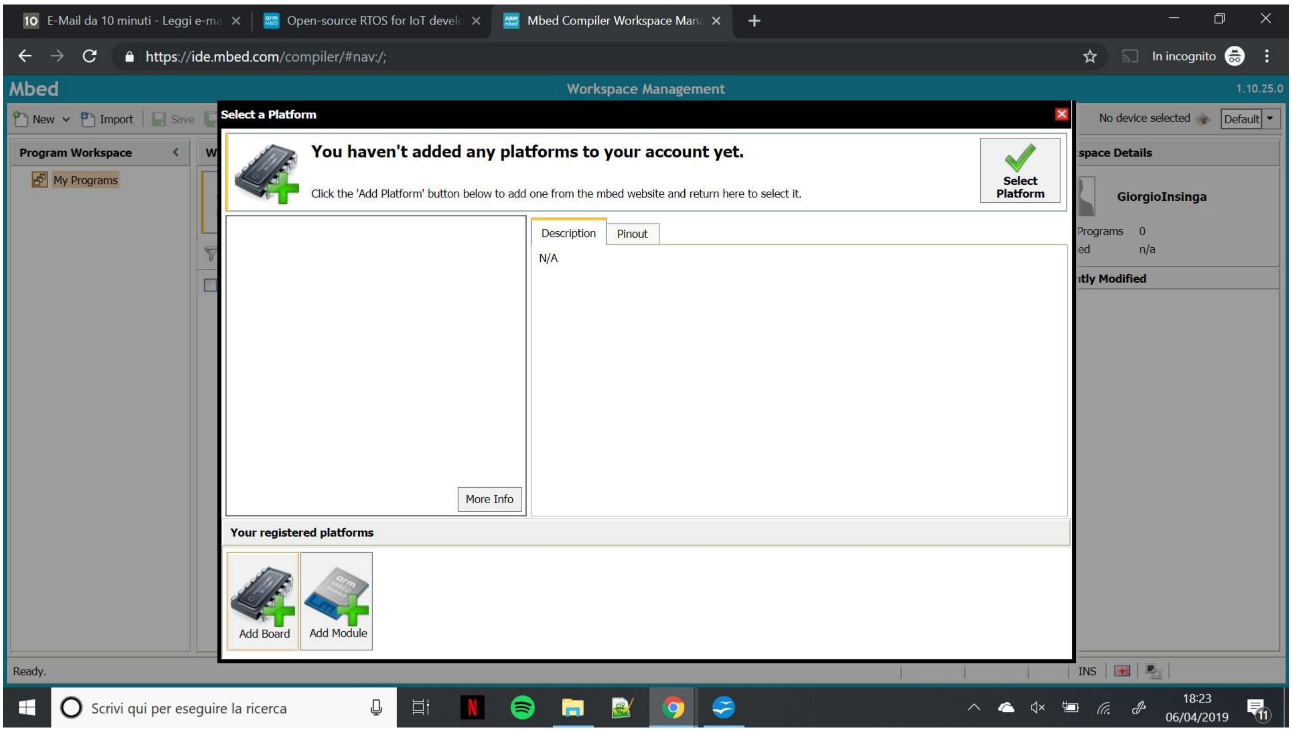

This is the homepage of Mbed. Just click on the sign up link to set up your account.



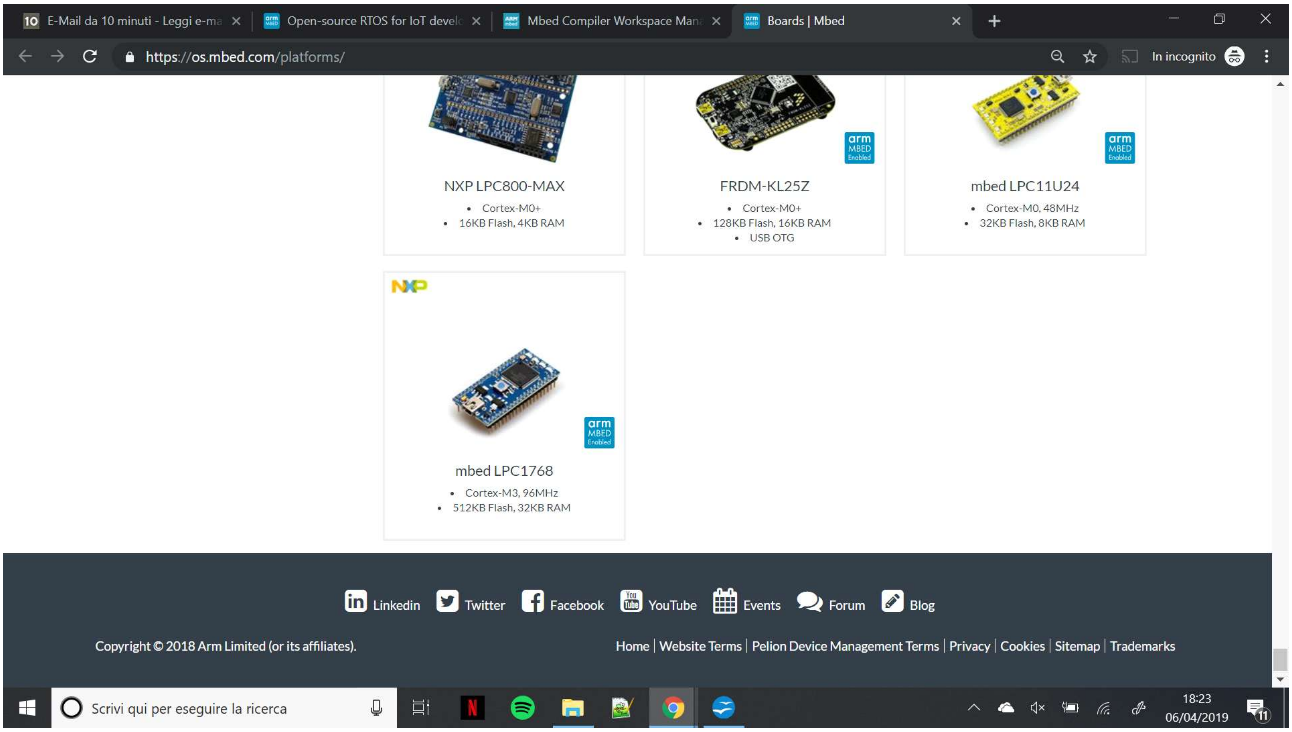

Fill the form and ensure that you're not a robot (in case you're a robot, just lie).



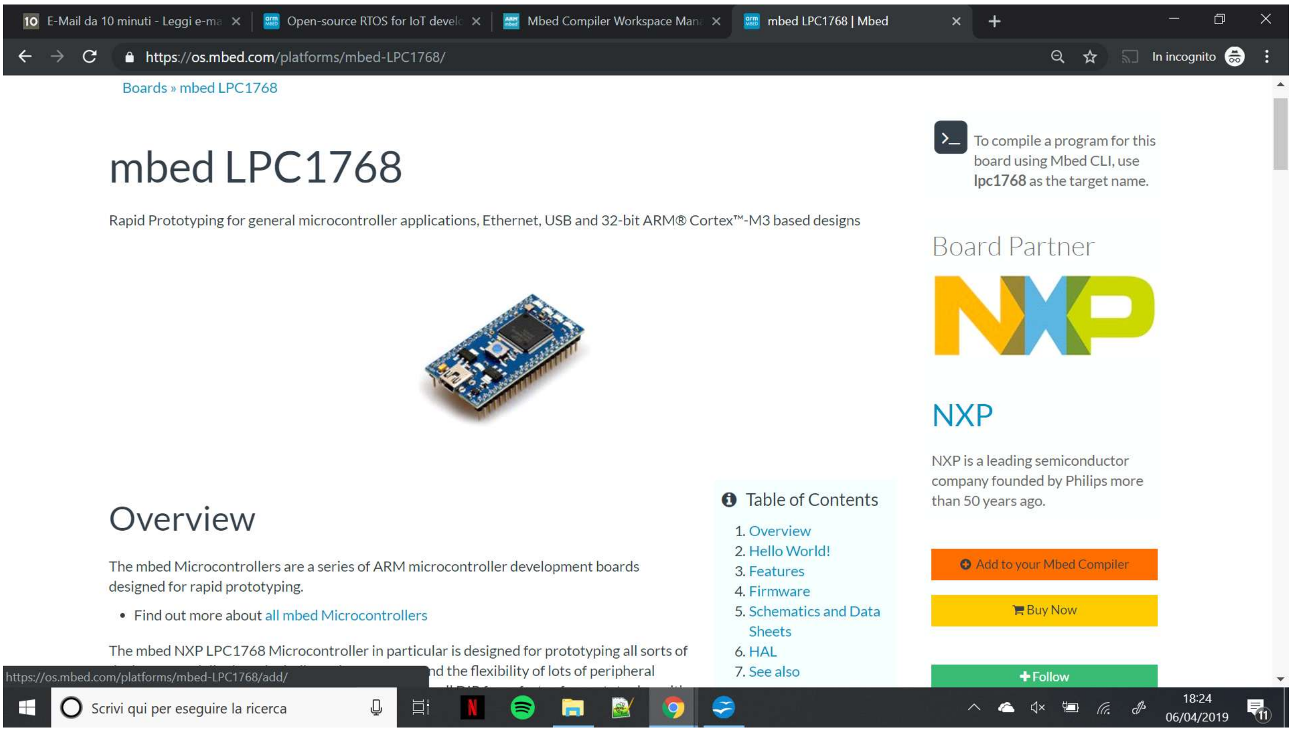
After your registration (you can use a 10 minute mail provider to sign up), you should be redirected to the homepage. Click compiler and this windows will show up. From this page you'll be able to manage all your project. As you can notice on the top-right there is no device selected yet.



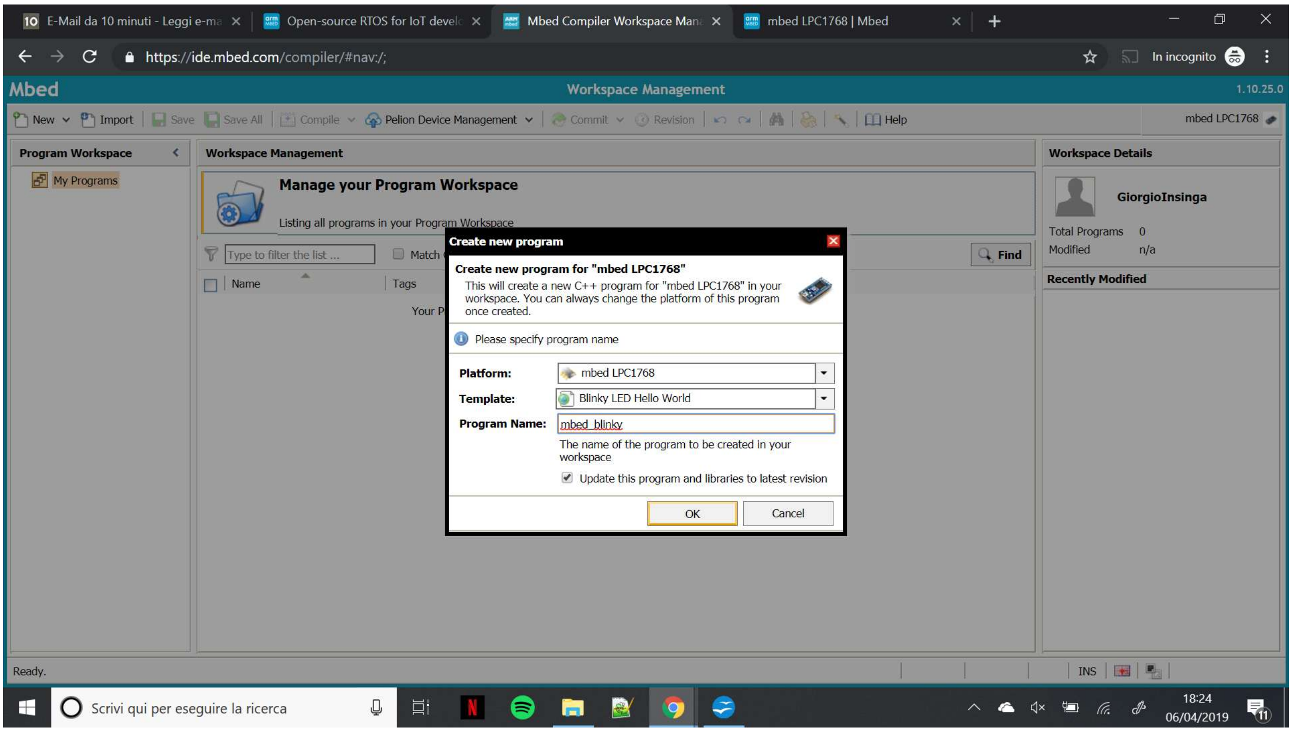
Click on “No device selected”. Here you will manage all your devices. If you change the active board,  your program will still work as expected (as long as you remap the pins), you just need to recompile it.
Click “Add Board” and you will be redirected to the list of all the supported Mbed board.



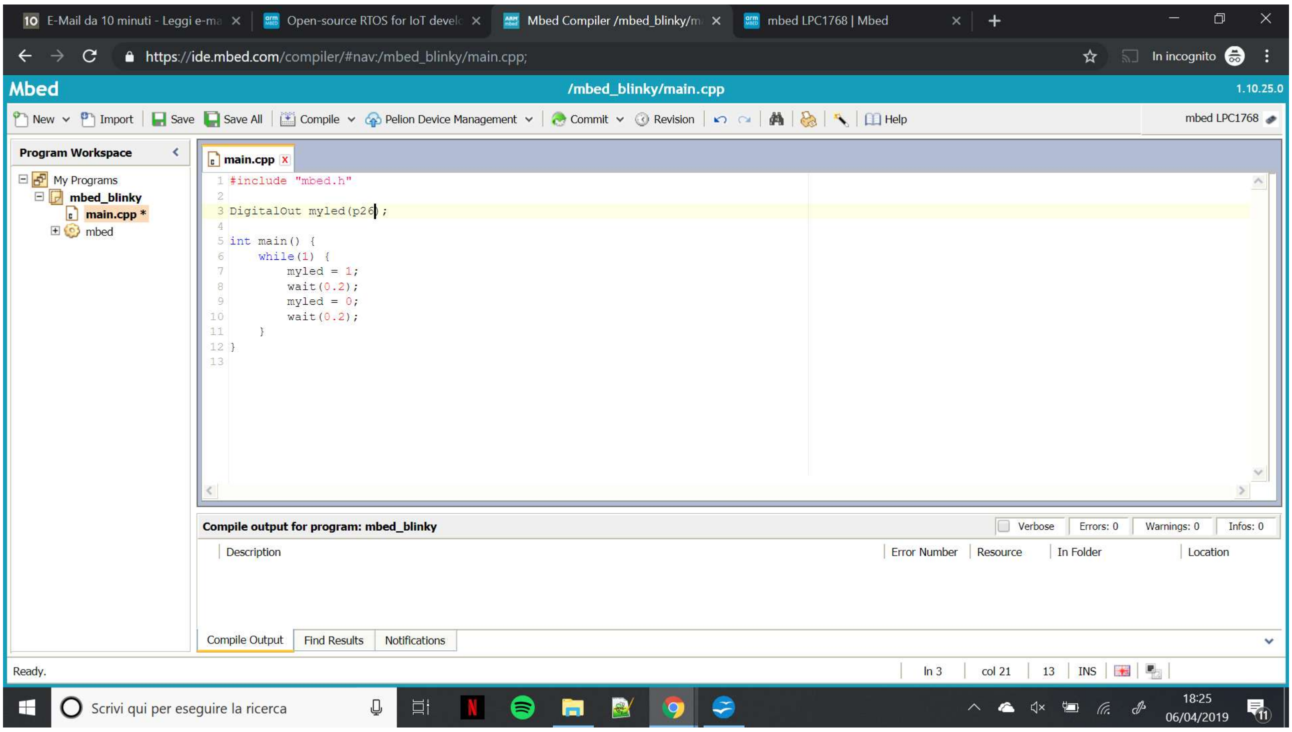

On the bottom of the page, you will find the “mbed LPC1768”. As I said before, we will use this board as a reference because fortunately it uses the same chip as the LandTiger: the NXP LPC1768.



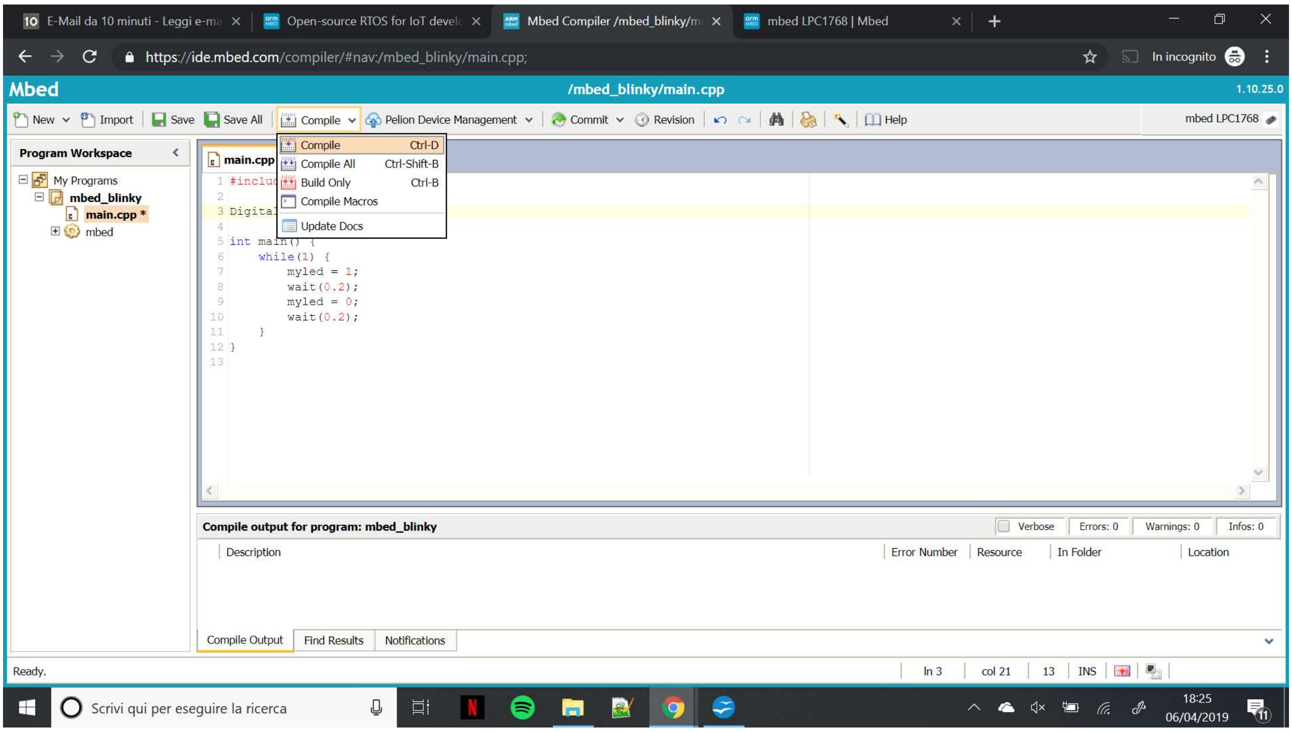

From this page, you just need to click “Add to your Mbed Compiler” and the board will be automatically applied to your compiler.



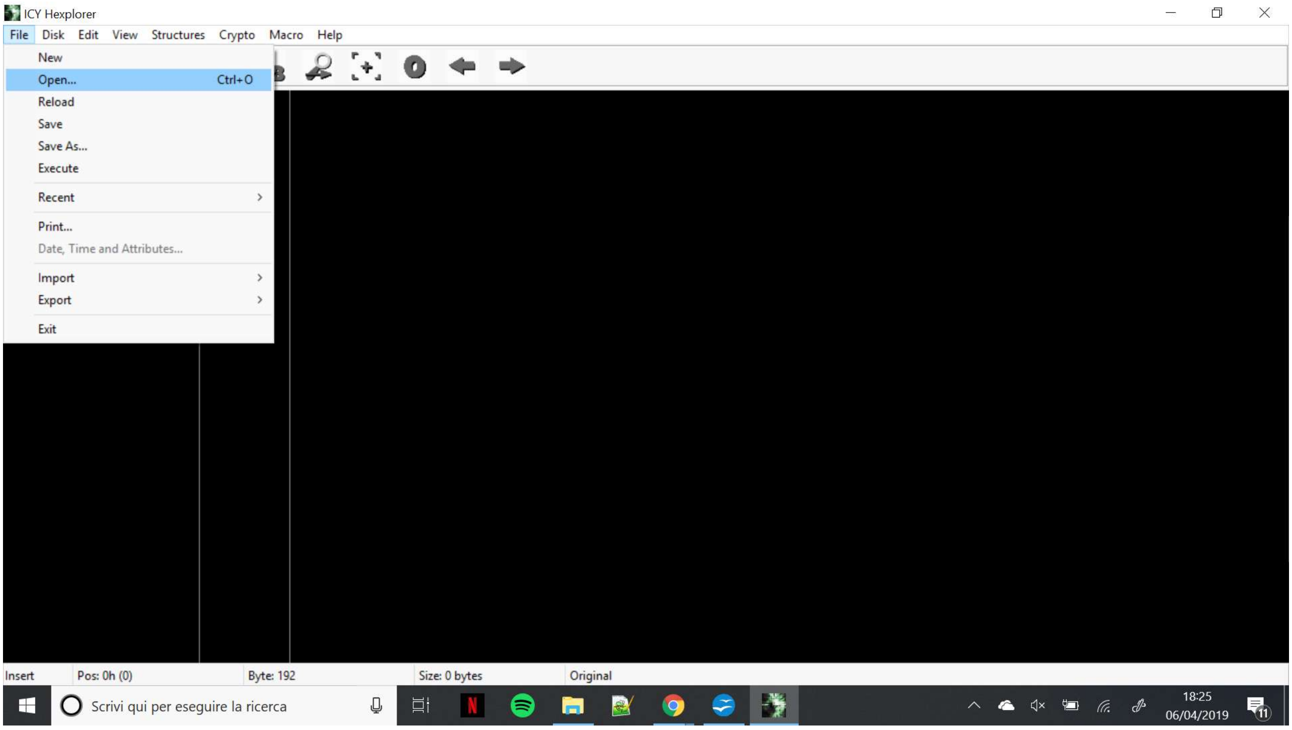
Returning to the compiler you can click the “New” button on the top menu. From here you can specify the desidered platform, the template (for example empty or the blinky ones) and of course the Program Name. Let's select the standard one for this tutorial, just click “OK”.



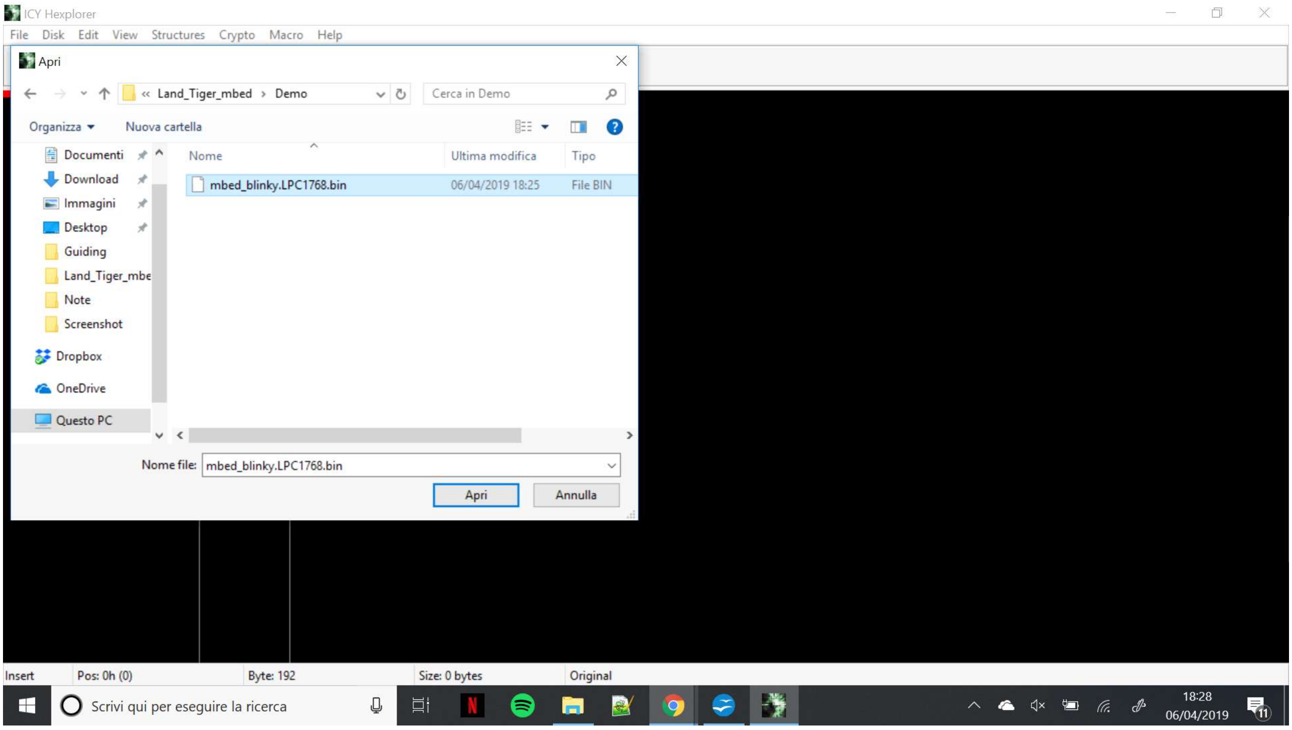
This is the blink example, I just replaced the pin “myled” to p26 (since our board have a different organization of the pins).



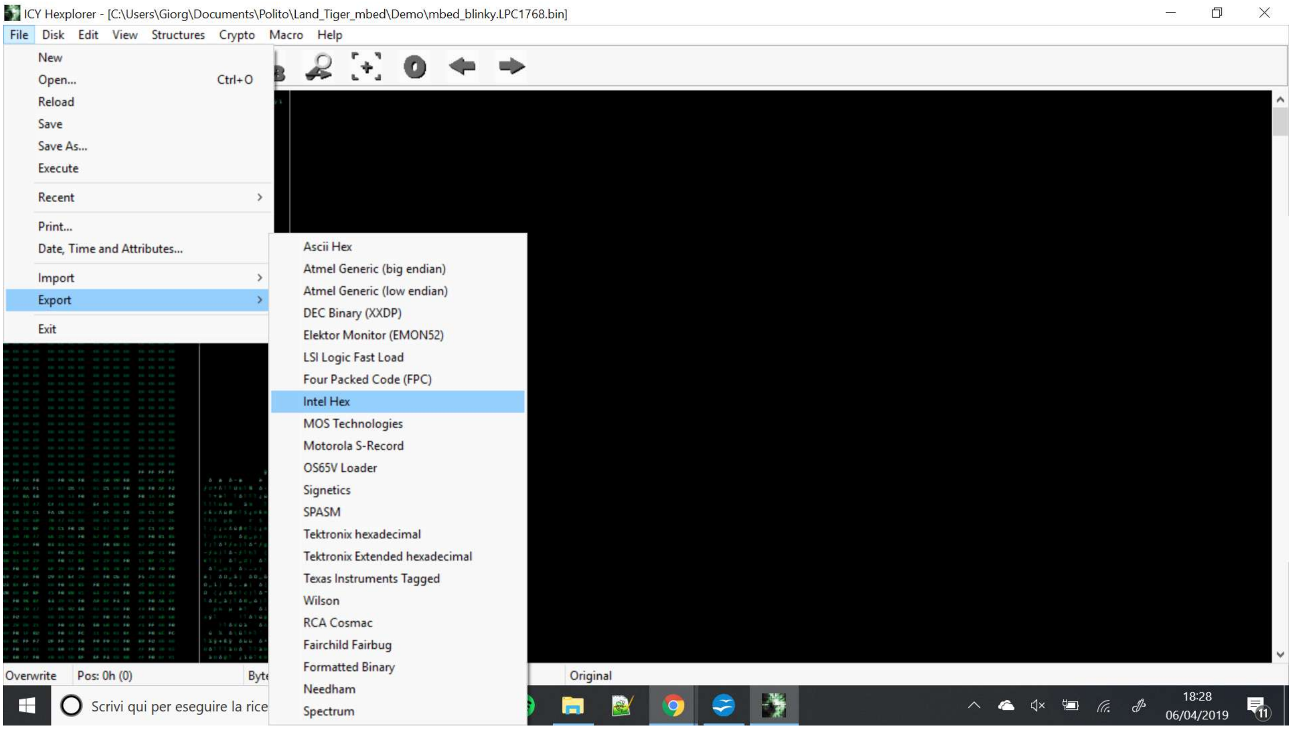

Click on compile from the top menu and then again click compile. This will give you the .bin file that we'll use in the next steps.



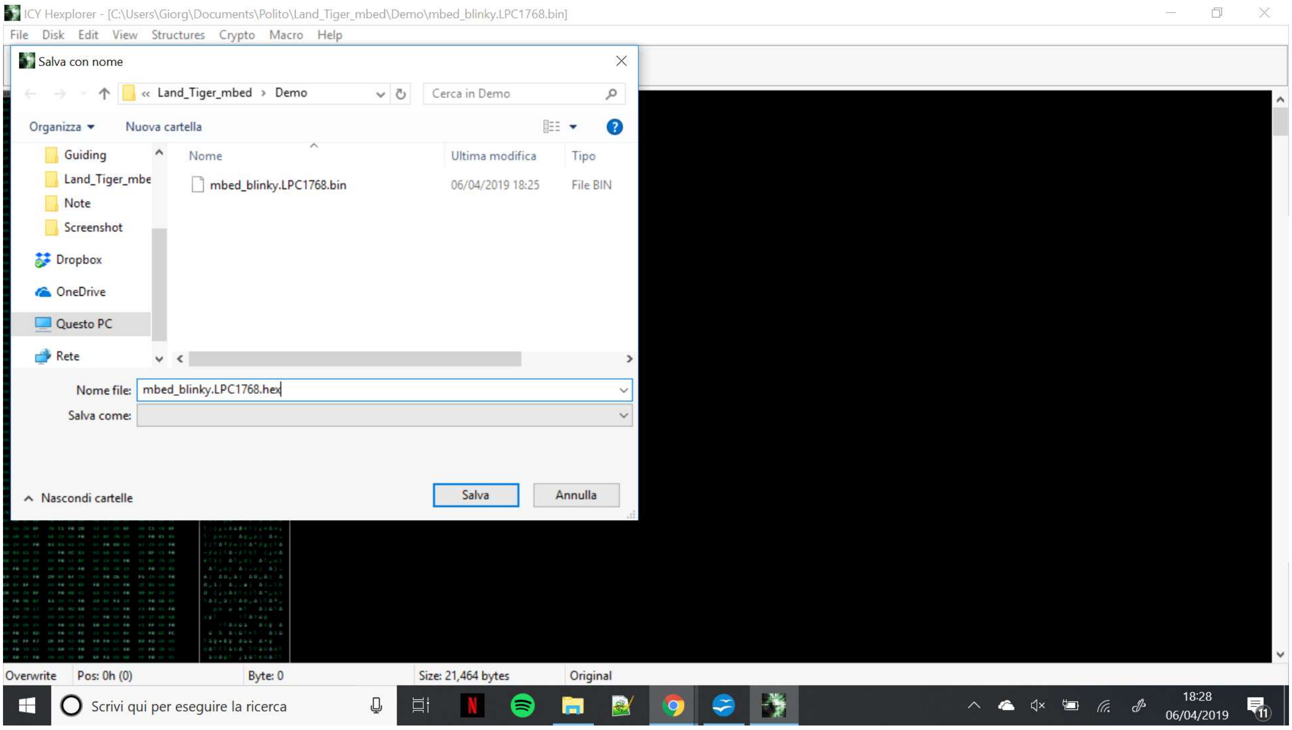

Open Hexplorer and then File → Open...



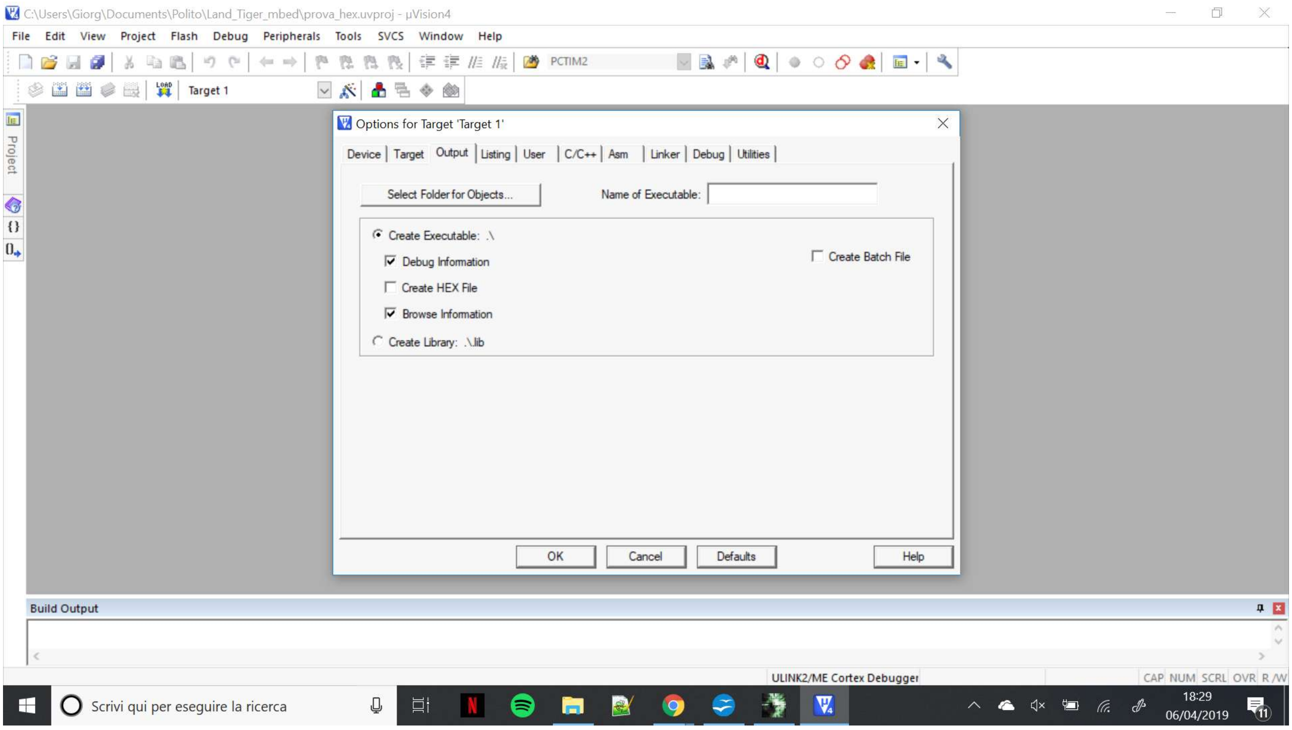

Select the .bin file you obtained from the mbed website.



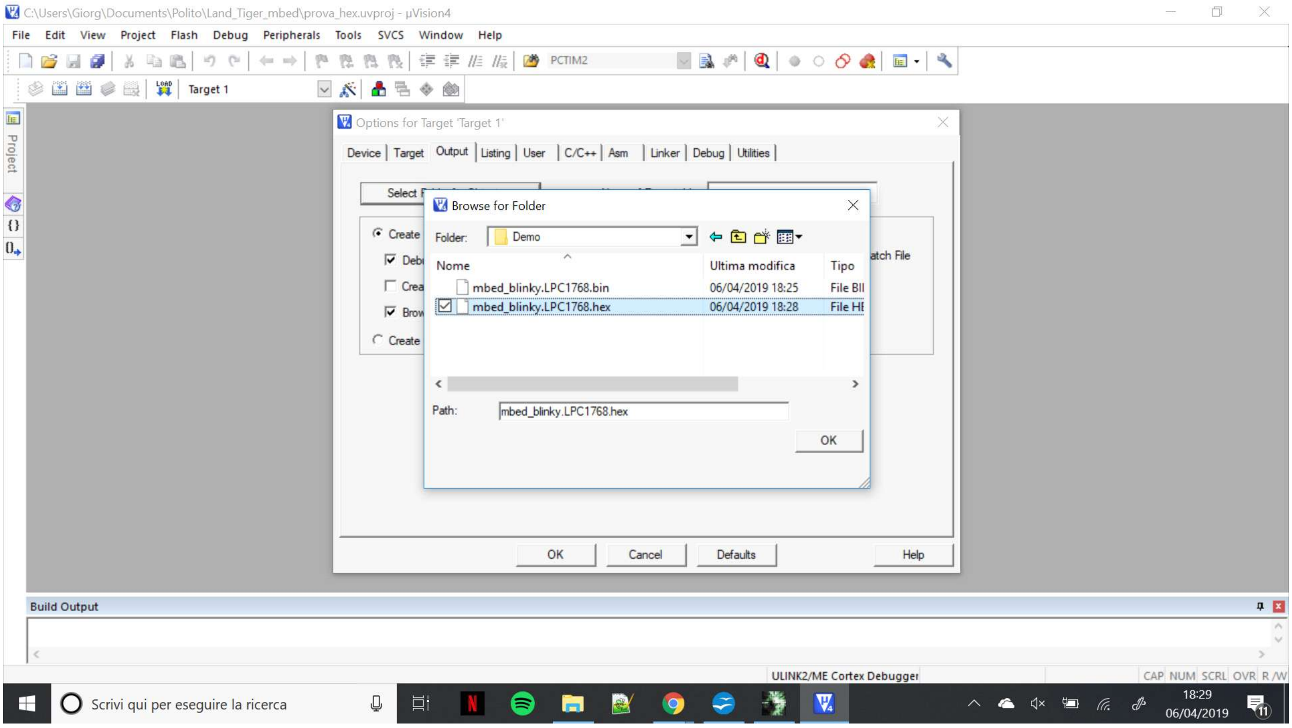

Click on File → Export → Intel Hex



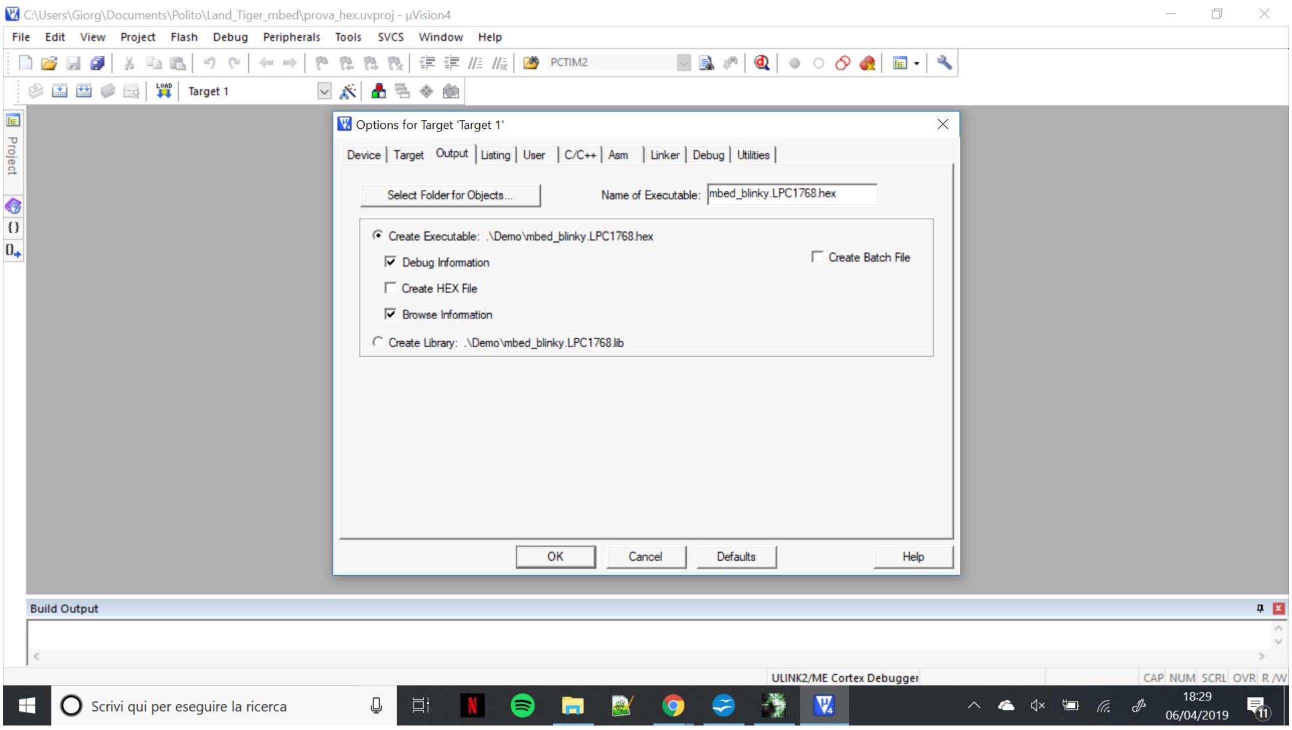
Save your file adding the .hex extension, this is the final file that you'll upload to the board.



Open Keil and then click the “magic wand” simbol. In the menu that will show up, select the option tab.

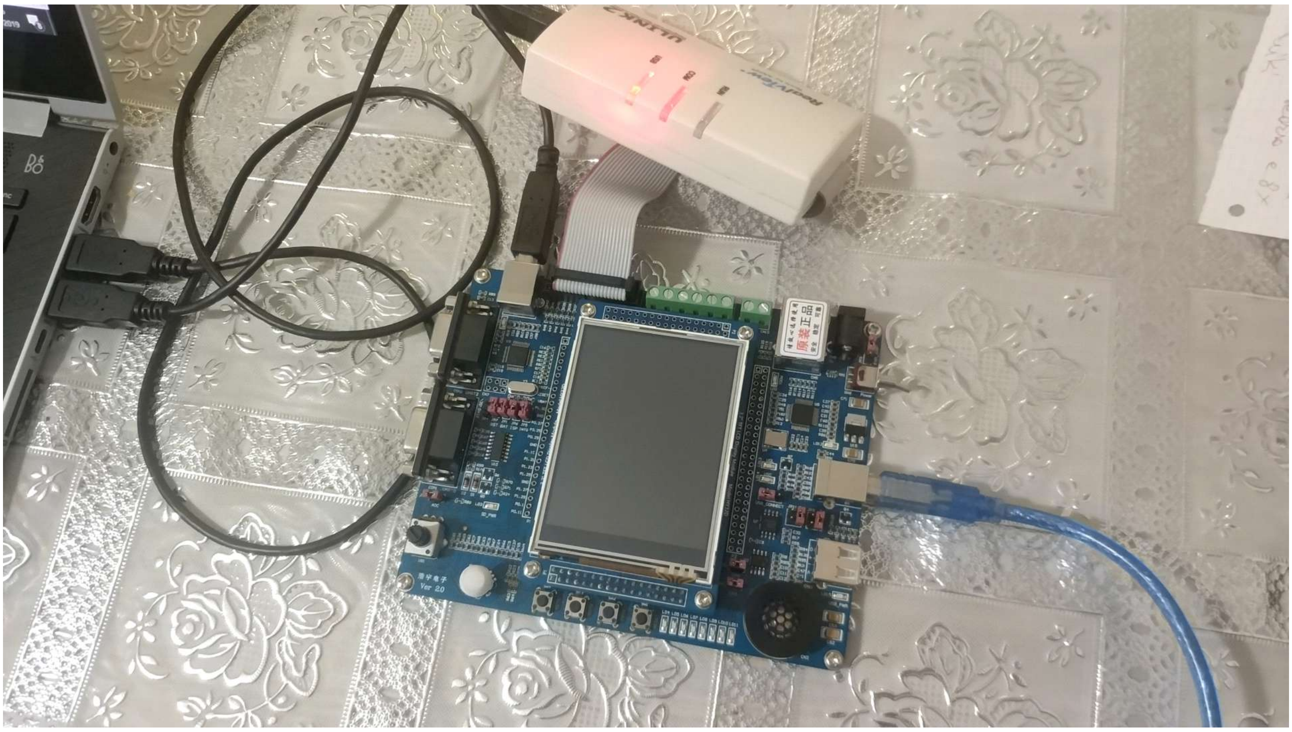


Select the folder that contains your file, not the file itself. Selecting the file will anyway set the right folder  but you still need to follow the next step.



Finally insert in “Name of Executable” the name of your .hex file and then click “OK” on the bottom left.

Clicking on the “Load” button should now correctly load your program in the board. It's important that after the flashing of the board. <u>Remember to push the reset button on the board after the procedure, otherwise your program will not be executed.</u>



This is how to connect the board to the PC. Notice the JTAG cable and the Serial cable near to it; to program the board you need these two cable (the serial one is used just as a power source for the board). Notice also the blue serial cable. This one is used for the “USB slave interface” and it will be necessary in all of the project that I will show. Be aware: your program will not be executed if it included some USB component and this cable is not connected to the pc.
Notice the position of the various jumper. Copy exactly this configuration to have the board powered by the serial cable and set as a USB slave.

Now that we have seen everything about the setup I will shortly describe what I have done for my project. The source code for everything will be included in this page.

## Simple Mouse

This was my first try with the mouse interface. I simply mapped the cursor's movement to the joystick and the left and the right click to two buttons of the Landtiger.

## Take me home country roads

This is one of the two demos that I made to show the capabilities of the landtiger board. In this program I am using it to emulate a mouse and a keyboard. You just need to open Google Chrome, put it full screen, upload the code to the board and run it.
What this will do it's actually very simple (but actually funny); At first the cursor will go to the top left corner (just a simple loop that ensure that you'll reach the corner starting from whatever place in the screen). Then it will go to the address bar of the browser and write "Take me home country roads". At this point the board will send a return to start the search. After that it will simply go down with the pointer, click the video to open it and click it again to make it play.
I inserted many delays to give time to observe the various steps and to give enough time to load the results from Google and to load the youtube video.
PS: please note that this will only works if you use Google as searching engine.

## Gotta catch'em all

After the youtube videos I had one idea: "why don't I use the LandTiger to play games?". But I didn't want to use it as a simple mouse, I just wanted an all-in-one solution. But here I had a problem. The landtiger have only some buttons and a joystick so I needed something that can be played with few inputs. Pokemon was the answer.
To use this demo you need to download one GBA emulator and one Pokemon rom (others game should work too).
I mapped the joystick of the landtiger to the arrows of the keyboard and I mapped the 2 buttons of the board to the key A, B and the joystick button to the return key (Start key in the GBA emulator).

## WiFi Keyboard

I had a lot of troubles with the display of the Landtiger. There were different mbed libraries for the Landtiger's display, but there was a problem. From what I discovered these libraries were made for an older version of MBED and are not working anymore. I spent literally days trying to figure out how to fix them, without success. But the previous week I had promised to have a working keyboard and I wanted to stay true to my word so I came up with this "weird" solution.
This demo uses a very good board based on the ESP-8266. You can get one of these board for around 2.5€ from China. They are very good and they are very good for IoT project with the integrated WiFi capabilities. Since my original goal was to make a touch display I used this board to make a WiFi keyboard. Here's a brief description on how I did it.
I programmed the board to act as an access point and to show a Keyboard-shaped HTML page when accessing the root address. From here you can press a key to have it printed on the computer through the LandTiger board.
When you press a button you are actually pressing a link to a page named as the key that you pressed. Here is the trick: none of these pages existed. So what should you get? A 404 page. I programmed the board to print the keyboard everytime you get this error and to use the not existing link to understand which key you pressed.
But now the problem was: how to bring this information to the LandTiger? I disconnected the display to reveal the pins that it uses. Since in any case the LandTiger wasn't able to print the entire ASCII code and I didn't need all the characters, I just needed to transfer 5 bits at a time.
At first I tried to connect 5 cables between the ESP-8266's board to 5 of the databus pins used by the display in common with the leds of the LANDTIGER. I should say that it was a bit late in the night, because it actually took me 30 minutes to understand why I couldn't read the input from this pins. At some point I though "is there some sort of diode... oh.". Genius idea to use the pins in series with the light emitting DIODES.
But were there other pins to use? Well yes, but actually no. I needed 5 pins for the parallel transmission but there were only 4 usable as an input. That pins can also be used for an SPI connection. So why not to use them for SPI? Because both of the boards are made to be an SPI master, not a slave. Of course there are libraries to make them act as a slave, but I also read bad things about the stability of this solutions and I had no time to write the code by myself.
So I programmed a 3 bits parallel comunication splitting the communication in two parts: at first the board send the first 3 LSBs and then it send the 2 remaing MSBs. One pin is used to identify which bits are sent (it's high for the first 3 bits and becomes low for the reaming ones). The communication is made slow enough to overcome potential timing problems between the boards.
After correctly decoding the message, the LandTiger send it to the PC.
PS: sometimes the response time of the ESP-8266 board is very slow. Some say that this is a problem of the overhead introduced by the Arduino environment.

## Touch keyboard and trackpad

This was the last program for my project. After fixing one of the libraries for the display of the LandTiger Board i simply designed a keyboard for it and a big area reserved for the trackpad (that acts like a joystick).

When you press a key it's square will light up and a sound will be emitted from the board.
The touch in the display is not so precise; to compensate the board acquires 400 samples and takes the average.
The most difficult part after fixing the library was to design the keyboard and to align everything correctly.