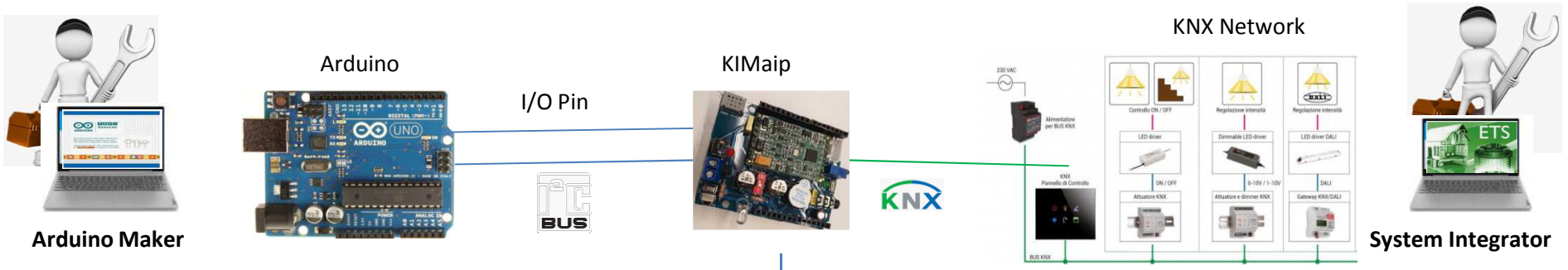
KIMaip I2C / KNX Gateway use case



By Arduino IDE:

Programming in c ++ of your device to read the sensors, execute the logics and use the KIMalib library to send and receive the values on the communication objects mapped in the KIMaip device.
Through the library you can also retrieve the data saved in the KNX user parameters to use them in your logic.

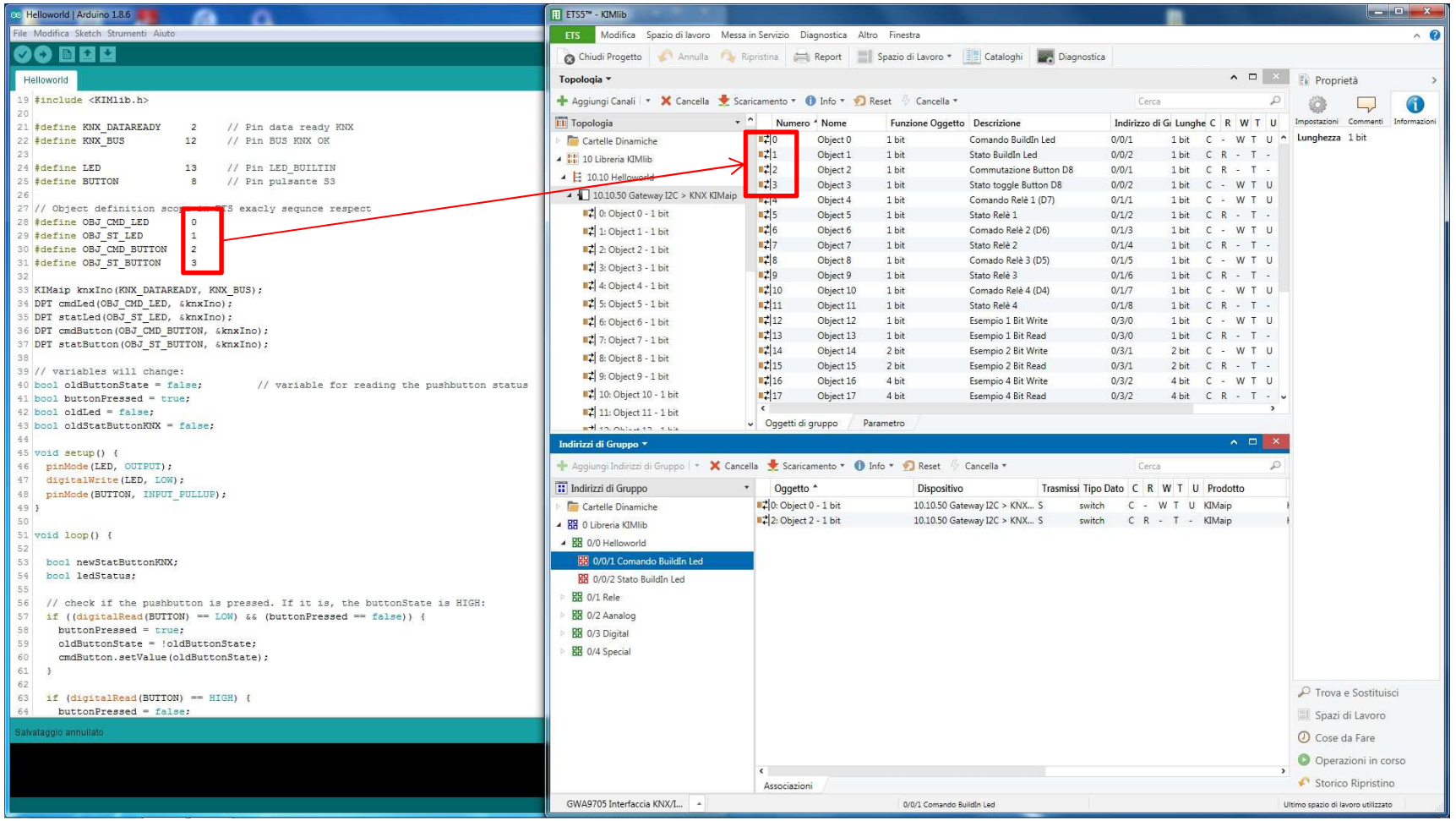
By ETS:

•Configure KNX physical address.
• Enable & Set objects data length (1Bit, 1Bite, ecc..)
• Set communication flag (C,R,W,T,U,)
• Configure group addresses
• Link communication objects in group addresses
• Download the programming and the KNX application.

In the library folder there is an ETS (V5) file that contains the KIMaip device to copy and use in your project.

The ETS file is the same one used to load the DEMO application used to make the examples contained in the library work.
The device present in it must be configured according to the needs of your project:
- General configurations
- Type of communication object (1Bit, 2Bit, 1Byte ... etc ..)
- User Parameter preset value

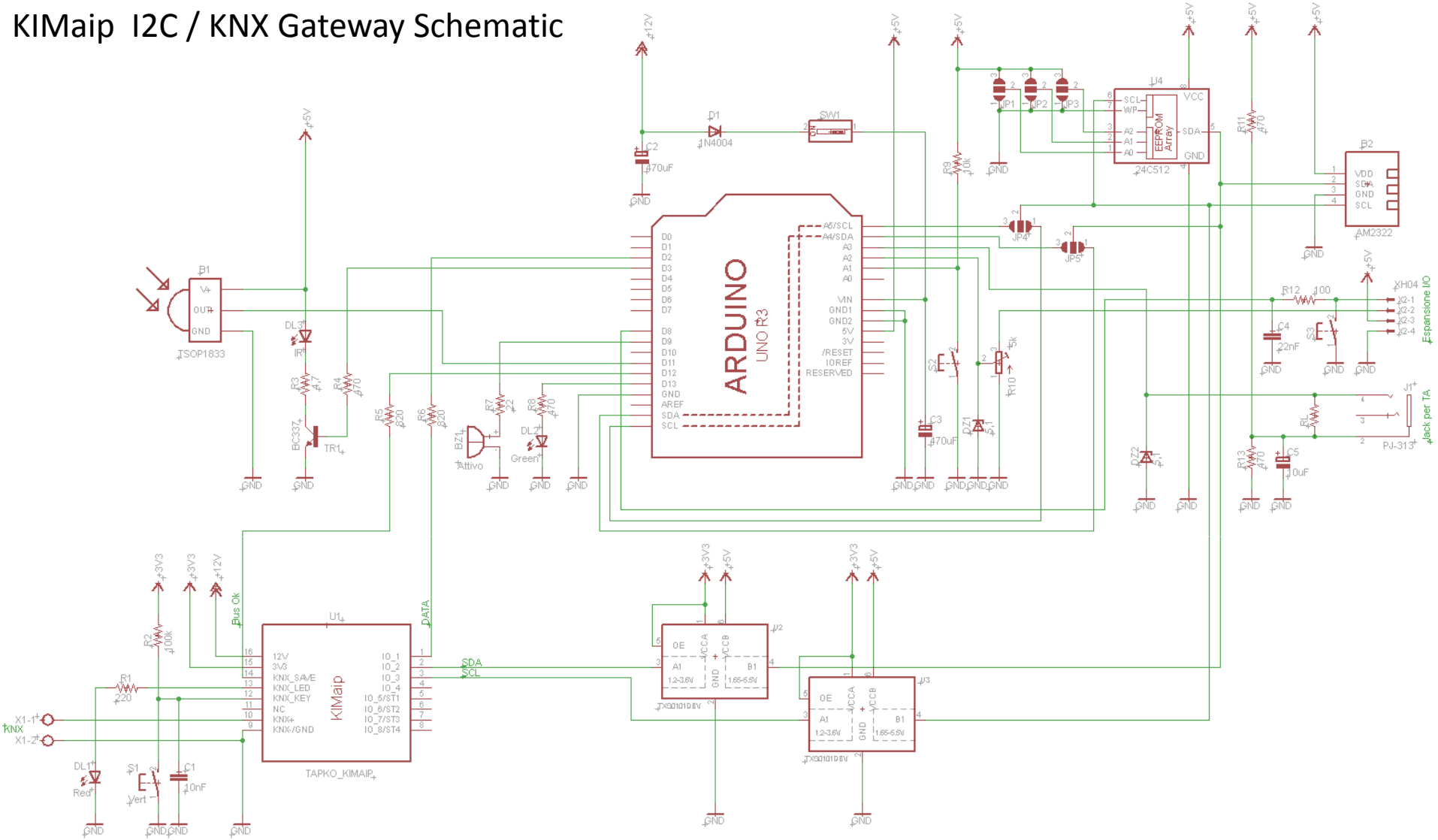
KIMlib, library for Tapko KIMaip module; I2C / KNX Gateway



Maps the addresses of the KNX communication objects in the same way that they are numbered in the ETS project, uses the same type of data (1Bit, 1Byte, etc.) and uses the KIMlib library to read and write data on them.

You can also read the user parameters (1byte) that you set via ETS, in this case they are mapped sequentially starting from 0

KIMaip I2C / KNX Gateway Schematic



Gadget on board (full version)

Tapko KIMaip.
Red LED (KNX).
Button (KNX).
3,3V to 5V level adapter.
Switch for Arduino Pwr selection (KNX / Vin).

Power Drive IR TX.
Ir RX.
512kb I2C Eeprom.
Buzzer.
Green Led.
Button (S2).

I2C Temperature + Humidity Sensor.
Input for TA (predisposition for RL).
Button with debounce (S3).
Connector for DI (S3) + AI expansionExt. (A2).
AI expansionExt. adjustment trimmer.