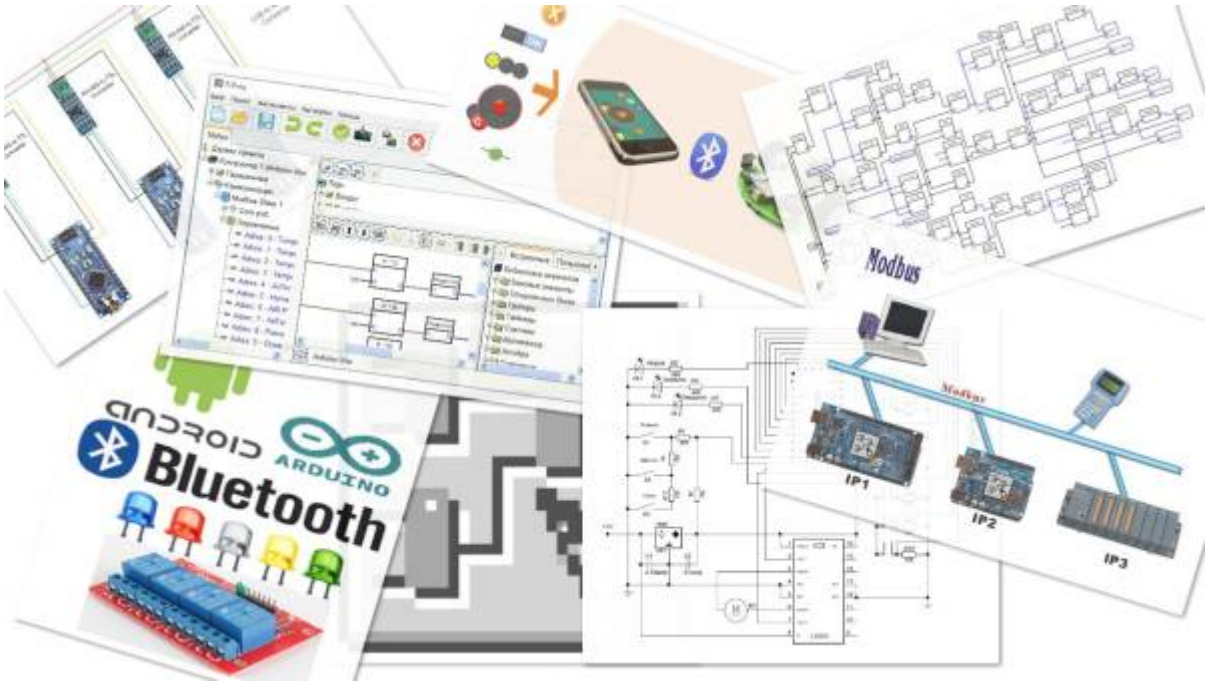
FLprog - Arduino programming for the reluctant, Introduction

Original blog post: august 27, 2022



There are many different programs that allow you to do graphical arduino programming ie: ardublock, XODide, MyOpenLab. However, this program compared to the others is external is not an add-on to the Arduino IDE like ardublock. It is not a completely new programming language with a very high input threshold, nor does it need to be connected to a computer like MyOpenLab. FLprog is a completely external program that generates code that we can upload to the arduino. After interpreting our graphical output, the programme simply opens an Arduino IDE window with our code.

On the author's website https://flprog.ru/en/ under projects, we can see the many things the programme community has done. There you can find projects ranging from controllers for a heating cooker to programs for small toys and robots.

The programme is based on a very simple principle, namely that we have blocks/blocks from which we assemble our programme, then we connect them by wires/switches, which allows data or logical state to flow from one block to another. This allows any idea to be realised instantly with virtually zero input threshold.

I have so far managed to create something like a parking sensor in this programme. This is a device that, using an ultrasonic sensor, emits periodic beeps whose periodicity depends on the distance in front of the ultrasonic sensor. In simple terms, the sensor beeps more frequently if something is approaching it.

Below is the process of creating such a design.

First, we need to find or purchase parts, the following parts were used in this project:

- HCSR04 proximity sensor
- Arduino UNO microcontroller

FLprog supports Arduino Uno, Arduino Leonardo, Arduino MEGA and many others.

- TDA2030 18W Audio Amplifier Module.

Arduino buzzers have a very irritating sound this will help to offset this effect slightly.

- Some old speakers

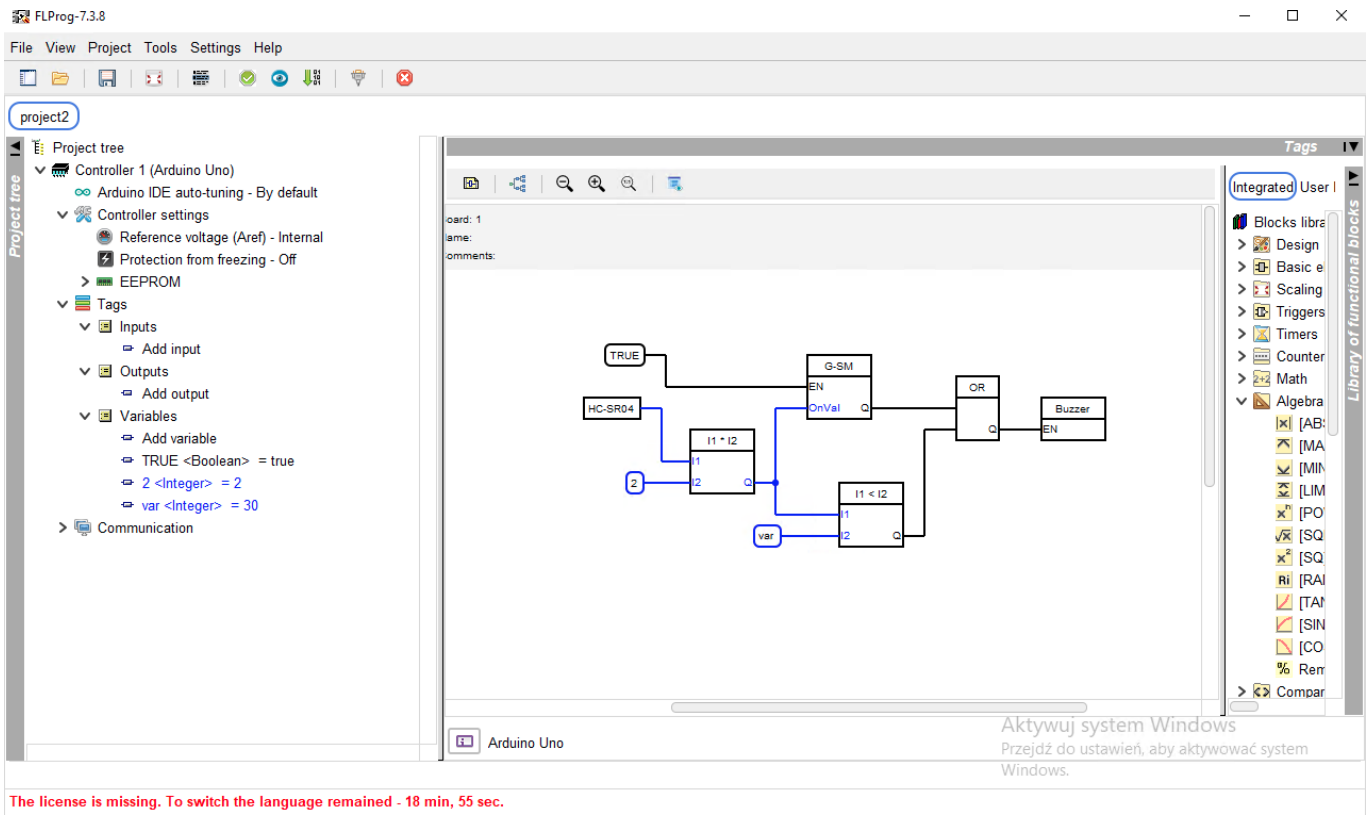
3 ohm to 16 ohm impedance.

- Power source

In my project, the arduino is powered from a workshop power supply. The audio amplifier is powered by the 5V line from the arduino, as nothing here will draw dizzying amounts of current

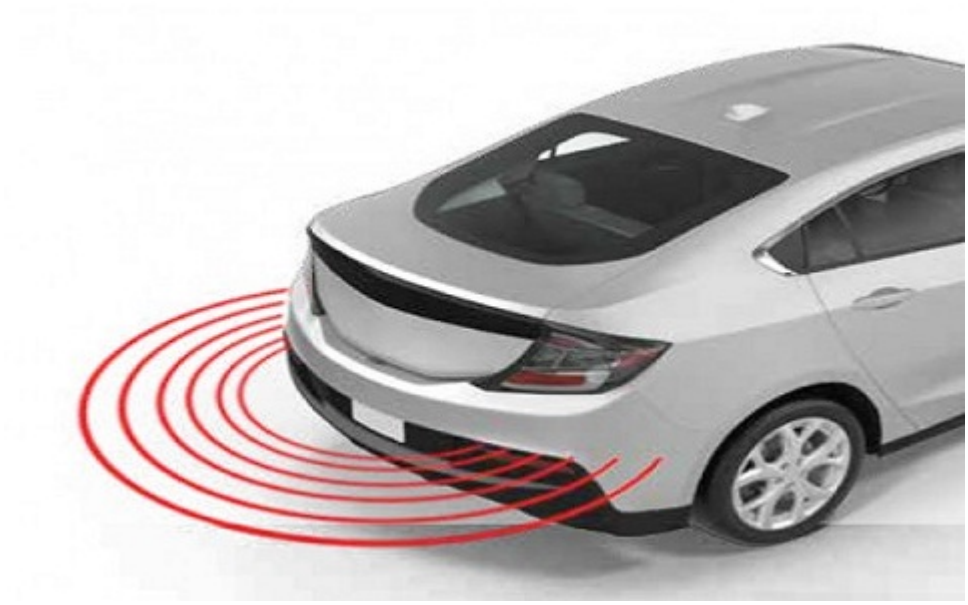
Then we need to connect everything. Connect the sensor to the pins on the arduino, it's not important which ones specifically, it's important that they have a ~ sign, because it means that they support PWM and this is important for us, in the program you can later correct the number of the pin used. Next, connect the amplifier, connect its power supply, connect the speaker and connect one of the pins to the input of the amplifier.

Then open the program and start playing. Below we have the whole project already made. When creating the project, press File>New>New Project for the controller and then in the additional window choose the FBD (function block diagram) option. Select the microcontroller you are interested in from the list.



In this way, we have created a new empty project.

Now it is worth explaining what we have on the screen because there is not little of it. On the right side of the interface, we have available blocks that allow us to create the programme. On the left side, we can see our: variables, arduino inputs and outputs, communication ports, internal memory and microcontroller jamming protection. Next, to start work normally, we would have to add inputs and outputs, but in this case we will use ready-made blocks adapted to our elements. We start by dragging our HC-SR04 sensor block from the list on the right, sensors>distance sensor tab. By double-clicking on this block, we can set on which pins we have connected the Echo and Trig signals of our sensor. In the catalogue note, we can read that this sensor is able to detect objects at a maximum distance of 4. We also set this distance in this window. Now we have a function block which outputs the value of the distance between the object and the sensor in integer format. FLprog highlights such connections as well as integer variables in blue. Now, as our number from the sensor will be rather small because we are interested in objects close by, with the idea that we are building a praking sensor. We therefore create an integer variable with the value 2 and take the math>MUL(*) multiplication block. We connect everything as in the picture, the multiplication is alternating so it doesn't matter to which input of the block we connect what. Next, we want to generate pulses from length proportional to distance. We take the Timers>Generator block from the tray. Then, in the parameters of this block, we set our generator as a symmetrical multivibrator and then select that the pulse length is dependent on an external parameter. We also need to enable the generator so we create a boolean variable with the value true and connect it to the enable input. Next, we need to create a condition that below a certain distance the speaker will start beeping continuously. We use the comparison>comparator block for this. And we choose the appropriate comparison, create a variable for the minimum distance and link everything in the right way and now our comparator will have the output in HIGH state when the distance drops below about 30cm. We then have to somehow combine the two signals in such a way that each is equivalent, i.e. the generator turns on the loudspeaker as needed. People who know logic gates will say ... OR fits here and yes, that's right, that's the gate we are talking about, we connect it. And now the last block is left>Piezo Speaker, add it to the project, connect it to the output of the OR gate and set the parameters, in the settings we set the pin which we connected to the input of the amplifier, we also choose the option continuosly the option is explained in the window, and then the frequency we want it to generate, I set it to 1kHz I think it is relatively little annoying such a sound.



In this way we have a parking sensor, which, although very simple, works. I invite you to check my other projects and I recommend you to start the adventure with arduino and FLprog yourself.