**Re-creating Sabancı University’s Radio by Using Internet Technologies**

***Given by:***

*Ahmet Çavuşoğlu*

*Efe Aydın*

***To:***

*İbrahim Tekin*

**ABSTRACT**

The aim of the research was building an internet radio for university which can be used by students of the university and they can be accessed from anywhere there is internet.During the researchs and amplications we applied lots of diffrent methods and solutions in order to create a product. However, there was a general method we followed during the project. This method is in order: first find and assimilate the theoretical knowledge. Then understand the results in the experimental environment and finally customize it according to the needs of the project. As a result of the study, the user interface and receiver and transmitter circuits were successfully completed and tested. In order to make the project truly user-friendly, the user interface must continue to be developed, and software such as a firewall must be added to protect the wifi network and data. In the circuit part, amplifier circuits should be developed and extra circuit parts and elements that can store large data sets should be added.

**Definition of The Project:**

Internet radio is a kind of a radio which is developed by using internet and wifi technologies. Internet radio is superior than normal radio thanks to the features which are economical, easy to access and use. The reason why it is economic, wifi signals are open to public use. Therefore, people who make broadcasting don’t pay for it.

**Short History of The Internet Radio:**

Severe Tire Damage was the first band to perform live on the Internet. (June 24, 1993)Moreover, Carl Malamud (Born July 2, 1959) ,American technologist, author, and public domain advocate, pioneered to the band during these years. Internet Multicasting Service was founded by him. He was responsible for developing the first Internet radio station and he created the website called Internet 1996 World exposition which recieved 5 million visitors. Furthermore, novadays there are several internet web sites broadcasting live radio such as (<https://www.accuradio.com/> , <https://www.internet-radio.com/>).

**Introduction:**

In the past, University’s Radio was being used by the Sabancı community. However, the interest on the radio deacreased and some technical issues emerged. That’s the reason why, university doesn’t have an active radio at the moment. On the other hand, RadioSU tries to broadcasting by using popular online platforms such as discord, twitch. But these broadcastes don’t be followed by the Sabancı community and also, community don’t know these broadcastes. Unless RadioSU doesn’t inculed novality, they can’t attract the community. Our project aims that, conserving radio culture by implementing recent technologies which is internet. Additionally, giving significant information about the news of the Sabancı University, playing songs to entertain users, to attract users by intriguing interface are our goals either. With an internet radio which is modern and easy to accessible, the attraction of community to the radio will increase.

**Methods and Materials:**

Our Project has 2 different parts. First part is Software and the second one is hardware. Software part contains 3 different coding language. These are python, html and arduino. The website of the Project was written by html and python while electronic microcontrollers were written by arduino language. The website is based on flask module and also includes xampp and phpmyadmin database structures. The materials which used to create the website were Udemy, AI, useful articles. The code was written on Visual Studio Code. The software part does not include any human participation. The electronic part of the project does not work independently from the software part, but rather works in harmony to complement each other. For this reason, theoretical and practical applications were used together. When starting the work, the needs of the project were determined in the first stage, and then the theoretical knowledge that would meet these needs was researched. After theoretical information was researched, an original design was created in accordance with the project. Most of the theoretical research was provided by forums on the internet and datasheets of the components. The design was initially tested in the Falstad simulation program. In this way, the states of the passive and active electronic parts when connected to the power supply and whether the incoming signals were processed correctly were tested. After the circuit was established and the codes were loaded, a multimeter was used for dc examination and an oscilloscope for ac examination to understand that the circuits were working correctly. However, the materials and theoretical research listed here, as well as all the circuits and software, were not made in one go. Each circuit and software created in the first stage was installed and tested independently of each other. Because otherwise, it would be very difficult to find the cause of an error that would occur throughout the project.

**Materials and Useful Resources:**

Equipments to be taken from Ibrahim Tekin and CoSpace:

* NodeMcu ESP8266
* bc237 (BJT Transistor)
* Breadboard
* Jumper wires
* Wifi ip
* Resistors
* Capasitors

Other equipments:

* Visual Studio Code
* Computer
* Arduino
* Python

Useful Resources:

* Internet
* Articles
* Videos
* Experiences
* RadyoSU

**Results:**

For the website, we planned to upload and download voice files to the website. In addition, only registered users can use the website was planned either. However, the connection between xampp, phpmyadmin applications and the flask module on python did not work. Therefore, registration part of the website not working. In addition, Html language includes several types of function. Hence, some Titles and photos did not located well. On the other hand, uploading voice files, showing on the right place of the page and downloading by the user ,the main purposes of the project, are working.

You can find the flask and database codes below of the page.

First part of the code several modules imported such as flask, MySQL, wtforms, os, uuid and passlib.hash

RegistrationForm(Form) ,class, includes the registration codes. Name, username, email and password required for this part and the password is converted to secret code by sha256\_crypt.encrypt then uploaded into MySQL database.

The function route starts the localhost:5000 and then shows the index.html which is the main page of the website by returning the template. There are several functions for different pages. For instance, register, login ,main page, about.. etc. And all functions return different “html” code. The function fort he download and upload is shown below the codes. First of all, the function connects the folder thanks to the “os” module which stores the voicefiles. Then, asks to the user which folder you want to upload from the files in computer. At the end, uploads the voice file to the folder that stores the files. Finally, for always open website, “debug==True” is used. This code tries to integrate new codes without crash the website.

A screen shot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

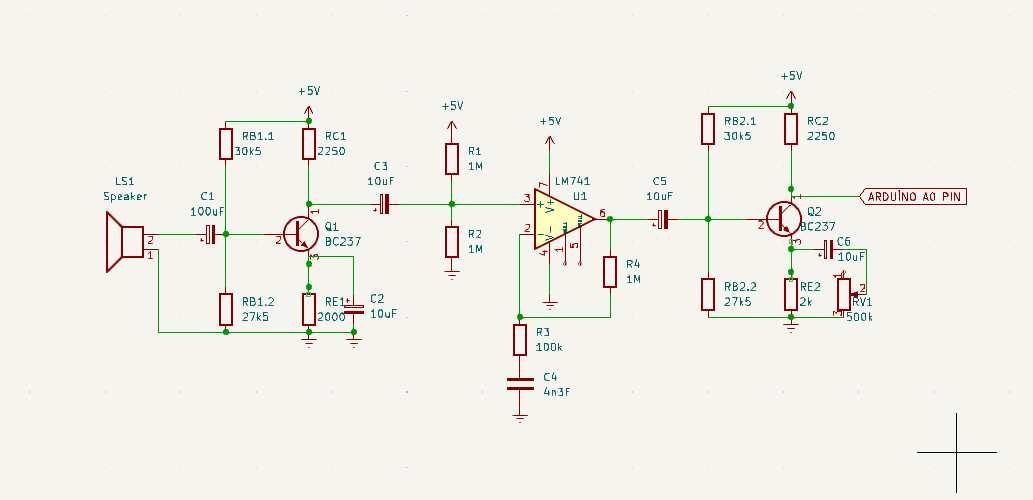
A screenshot of a computer

Description automatically generated

The aim of our project was to create the server and receiver of an internet radio that university students could use. For this purpose, we prepared a receiver and a transmitter circuit. In the first stage of the tranmitter, there is a diode protection circuit. Then, signal pass from emitter fallower circuit. After, analog signal is converting to digital by Arduino Nano board. Moreover, the digital signal transmit via Udp communication with help of the esp8266NodeMCU. That’s the end of the transmitter. Then, esp8266NodeMCU which is located in the receiver receive the digital data with Udp communication and send to Arduino Nano board. Arduino Nano and the DAC circuit transform the digital data to analog form. In the end of the DAC a buffer circuit which is an emiter fallower circuit drive the speaker. Let’s start to analyze every stage more detailly in a way of circuits and codes.

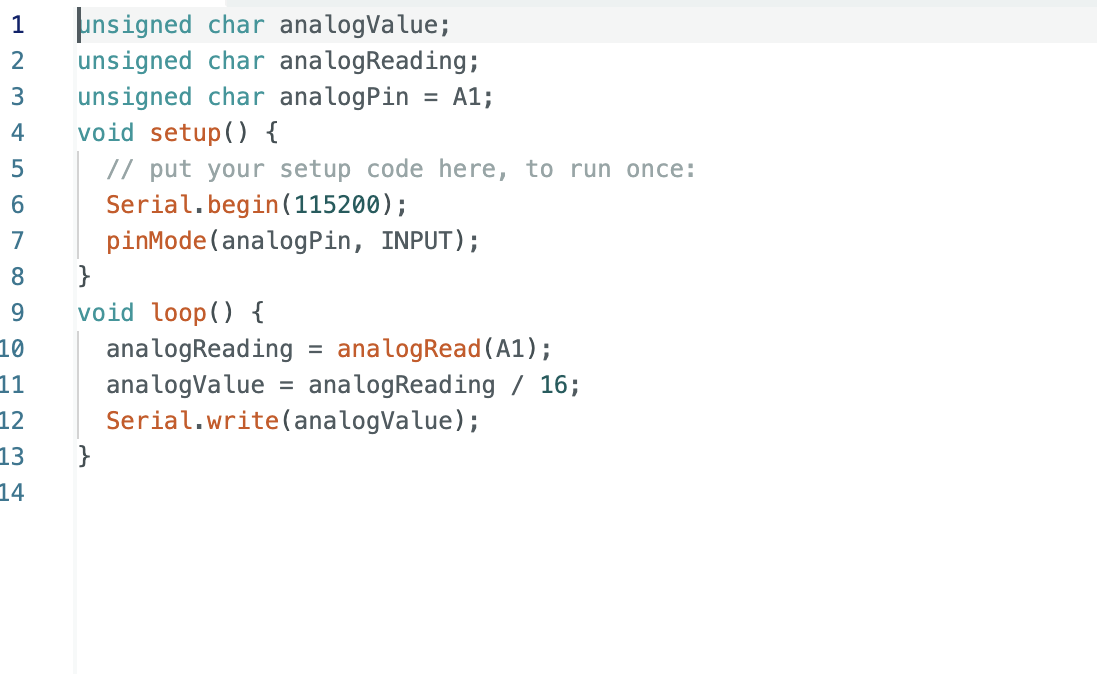
**TRANSMITTER**

**Circuit Part**



The transmitter starts with a stereo intro. The stereo input then passes through a diode protection circuit. In this way, it protects the Arduino microcontroller when the external audio signal exceeds 5 volt peak-to-peak voltage. Due to the diodes used in this circuit, the input signal can pass a maximum peak-to-peak voltage of 1.4 volts without cutting. In addition, before the signal reaches the diodes, it passes through a 1K ohm resistor, limiting the current that will pass from diodes. In this way, diodes will not pass max power they can handle. After the diode protection circuit, the signal comes to the (emitter fallower) circuit. In the first stage, it comes to the base end of the transistor through the coupling capacitor, which only allows the passage of the ac signal. But first, let's talk about how the transistor works. It is an active circuit element that contains semiconductors inside and consists of a total of 3 legs. These three legs are collector, base and emitter respectively. A low current applied to the base leg allows high currents to be controlled on the collector and emitter. This situation can be compared to the way a valve works. It can control the flow of water at high pressure with a small force. Transistors reflect the voltage value applied to the base leg to a 0.6V-0.7V lower voltage on the emitter, due to the diode like structure inside. In addition, each transistor has its own beta values. Beta value is used when calculating the current value that the current applied to the base leg will create on the collector. However, the beta value may change and be affected by many other conditions such as temperature, voltage value, current amount. For this reason, before working with transistors, the circuit should be analyzed and the needs should be determined, the desired situations should be investigated through the datasheet and the circuit should be built accordingly. Otherwise, the circuit may not work or transistors, other circuit elements and power supplies may be damaged. Going back to the circuit, emitter fallower is an application that does not give voltage gain to the incoming signal, but gives power gain. In our case, since the signal is strong enough and at high voltage, the emitter fallower circuit was deemed appropriate. Thanks to this circuit, the signal will not be distorted, since no gain is given, and the signal will oscillate on the full intermediate voltage. In this way, the ADC (Analaog to Digital Converter) in the Arduino nano will easily read this signal that travels between 0V-5V. If the signal was not placed on the intermediate voltage. The signal would have negative and positive voltages, so the negative side of the signal would not be read by arduino.

**Arduino Nano & Analog Calculation**



Moving on, the signal read on the ADC will have a value between 0 and 1023. The reason for this is that the ADC can read 10 bits. When the signal is converted to digital, it is divided by 16. Because the DAC (Digital to Analog Converter) on the receiver side is 6 bits. DAC will be explained in more detail later. If we examine the variables in the code, the char data type is constantly used. The reason for this is to increase the speed of the processor. Because a sample is collected from the audio signal, the processor must immediately receive the data, process it and transmit it via the UART port. For this reason, the code performs simple and short operations. That’s how we collect sample enough to mimic the real voice from the receiver’s DAC side.

**Esp8266NodeMCU & UDP**

 After the data is transmitted, the esp8266NodeMCU receives the data via the uart. The esp8266NodeMCU is the processor that forms the heart of the transmitter. Let's start by examining the codes. In the first step, the name and password of the wifi network to be connected are added to the code. Afterwards, a suitable port is selected for UDP communication. When selecting a port, it can be selected from numbers between 0 and 65000. However, since the first 1234 ports are usually full, it is recommended to choose a different port. When it comes to the setup section, the microcontroller tries to connect to the wifi network at 1 second intervals. After connecting, the receiver starts data flow through the port entered with the IP address of the esp8266NodeMCU. When it comes to the loop part, a packet consisting of 256 analog data is defined and reading is performed until the packet is full. After the packet is read, it is written to the port via UDP. After transmission is finished, the package is terminated and the cycle begins again. Like the Arduino nano board data type again mostly char and short. Because, the data should transmit fast enough to decrease the delay and calculations.

**Short Explanation About UDP**

To understand the sending of data, it is necessary to better understand the UDP process. UDP is a protocol that allows data to be sent over the internet to be compressed and transmitted faster. UDP occupies a big place in our daily lives. It is used to transmit audio and video data broadcast over the Internet.

**RECEIVER**

**esp8266NodeMCU**



The receiver side starts with the esp8266NodeMCU. While the setup, internet and port connection methods of the code are the same as the transmitter, the loop part constitutes the critical point. First, it is checked whether the incoming package is present or not. If there is a package, size information is taken and if it is equal to 256, each index of the package is read sequentially in the loop and sent to the Arduino Nano card on the UART port.

**Arduino Nano**

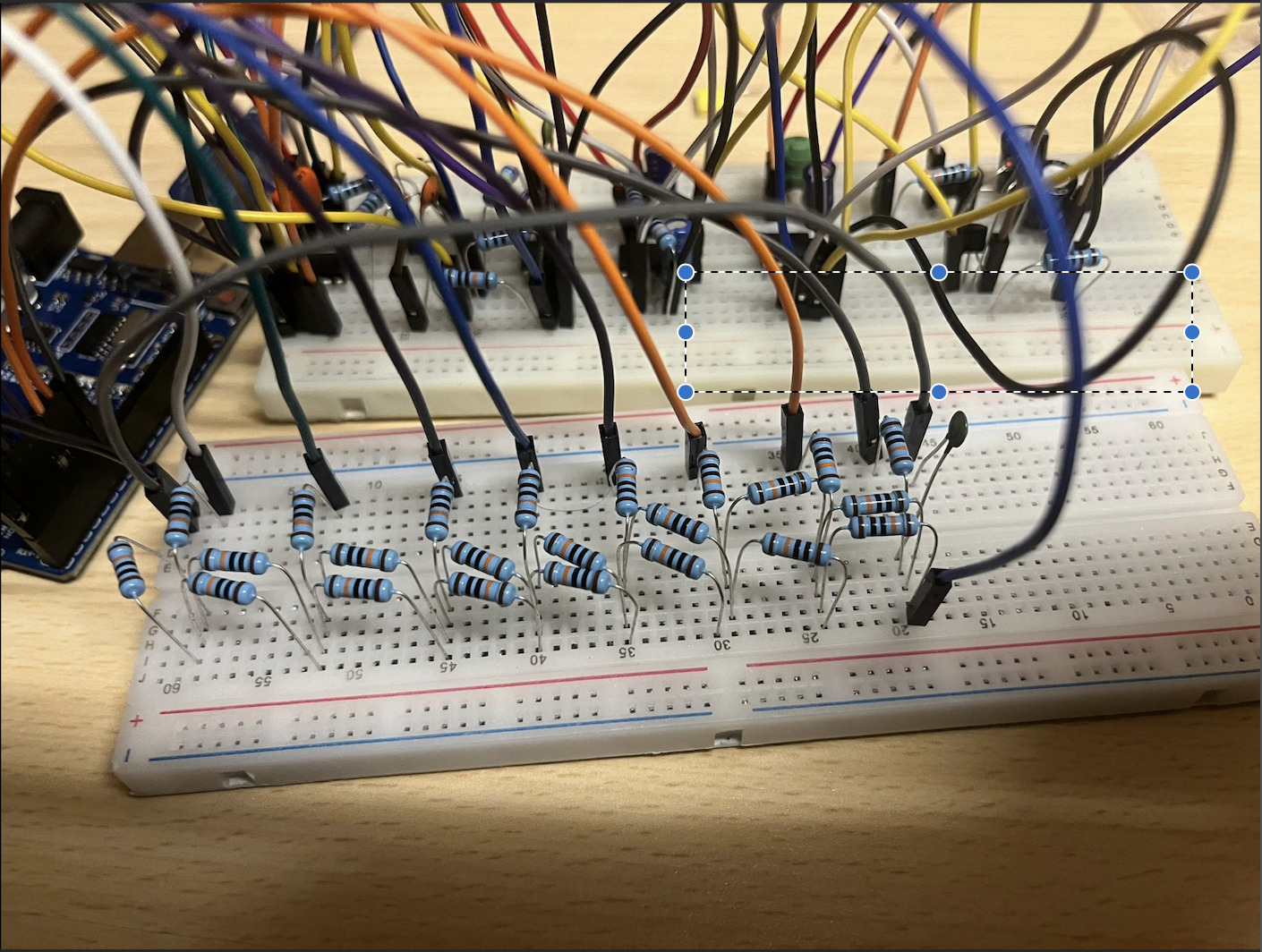


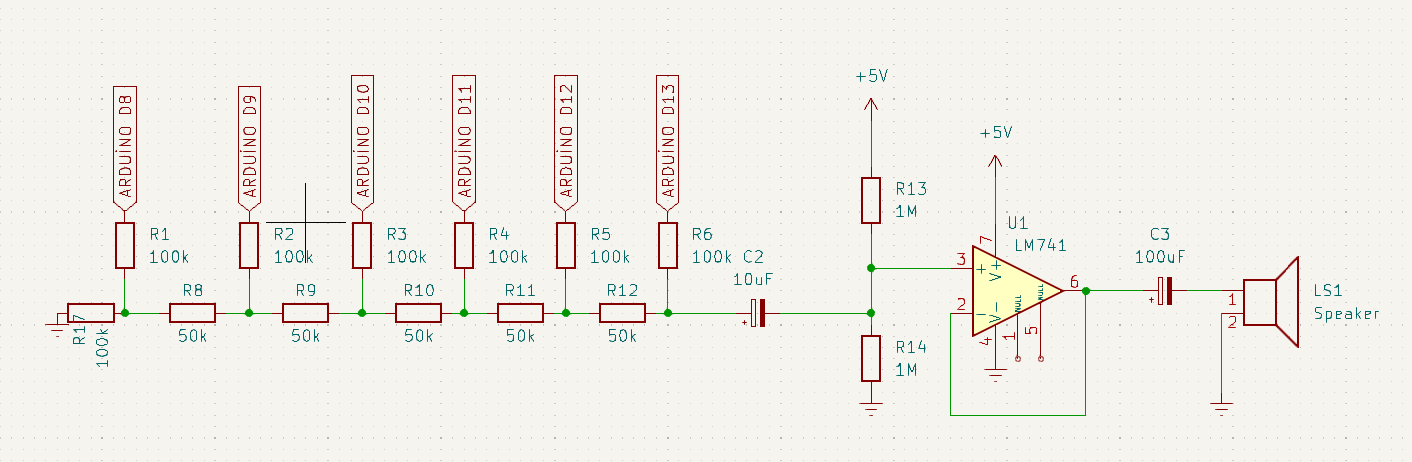
The data arriving on the Arduino Nano board is PORTB, a method type that covers all pins between 8 and 13 on the Arduino. In this way, 6 pins can be opened and closed according to incoming data. The incoming data here is the analog data itself read by the transmitter. The reason for doing this is that the pins in PORTB are connected to the DAC. As the pins are opened and closed, different voltages are created on the resistors on the DAC and the voltage value represented by the digital data becomes analog.

**DAC (Digital to Analog Converter)**

DAC is basically just a voltage divider built with resistors. The DAC in the circuit has a value of 6 bits, and when the AC analysis is performed, the equivalent resistances of the circuit can be easily calculated according to Thevenin theory. In this way, the voltage value of each bit when its logic state is HIGH can be found. Considering the Arduino processor, the DAC we created can easily generate signals up to 5KHz. This frequency range is sufficient to understand the sound. In our tests, the sound was understandable and it worked successfully. If the bit amount is increased, the sound quality increases proportionally. However, Arduino could not show sufficient performance in performing higher bit operations. In our experiments, we tried to increase the DAC to 8 bits. The DAC worked successfully independently of the receiver and transmitter, but caused signal distortion when connected to the rest of the circuit. The reason for this has not been resolved. However, there is a high probability that the source of the problem will be found with detailed analysis in the future.

**Circuit Part**





The signal generated by the DAC is not strong enough to drive a speaker. For this reason, a buffer circuit is needed. However, Arduino provides a very low voltage value (less then 5V). For this reason, it could not work efficiently with the common UA741 style opamps. Opamps operating at lower voltages were needed. However, since it caused an increase in cost, the emitter fallower circuit was used in a similar way in the solution emitter circuit. Current gain must be applied to the signal for the speaker to work. For this reason, high current must be passed through the circuit. However, Arduino can pass a maximum of 70mA on itself. Since the Arduino could be seriously damaged if these limits were exceeded, 10mA was passed through the emitter leg of the transistor. The results were quite sufficient. Sound could be heard from the speaker.

So in the end of the project we build an receiver and a transmitter circuit which works sucesfully for the needs of the project.

A screen with blue lines on it

Description automatically generated

**Discussion and Conclusion:**

For the website and the codes of the Project, flask and the database codes can be improved by adding different types of methods. And also the image of the website can be developed with html codes. For instance, different photos and menu bars can be added. In addition, the page that voice files are shown can be changeable. By doing this, users can reach the wanted file easily. However, in a given time this changes could not be done by us. But, in the next periods of the time, it can be implemented. On the other hand, the main aim was achieved. Voice files can be uploaded to the website by clicking on the button which located in website and also it is shown on the website so that, users can download the files with only clicking on the file names.

As a result, it is quite possible and economical to make a radio that the university can use. The cost of the equipment to be used will not be high. The circuits and software consisting of receiver and transmitter made within the project constitute a good example of this situation. In order to achieve its goal in the later stages of the project, it needs to focus on the transmitter circuit. Because the radio to be installed must host the broadcasts made in the past and be turned into a server that will host the interface, that is, the software, that the users will use. In addition, it is a requirement to add a firewall to the esp8266NodeMCU that will be connected to the wifi network. Otherwise, the security of the wifi network and data in the device will become a threat. What our work has contributed to us in the process until the product at the end of the project is as follows: problem solving, learning theoretical knowledge, experimenting, practical application, presenting and documenting the work and product.

**REFERENCES**

* <https://tr.wikipedia.org/>
* <https://www.internet-radio.com/>
* <https://www.accuradio.com/>
* <https://forum.arduino.cc/>
* <https://www.falstad.com/circuit/>
* <https://www.arduino.cc/>
* https://www.udemy.com/