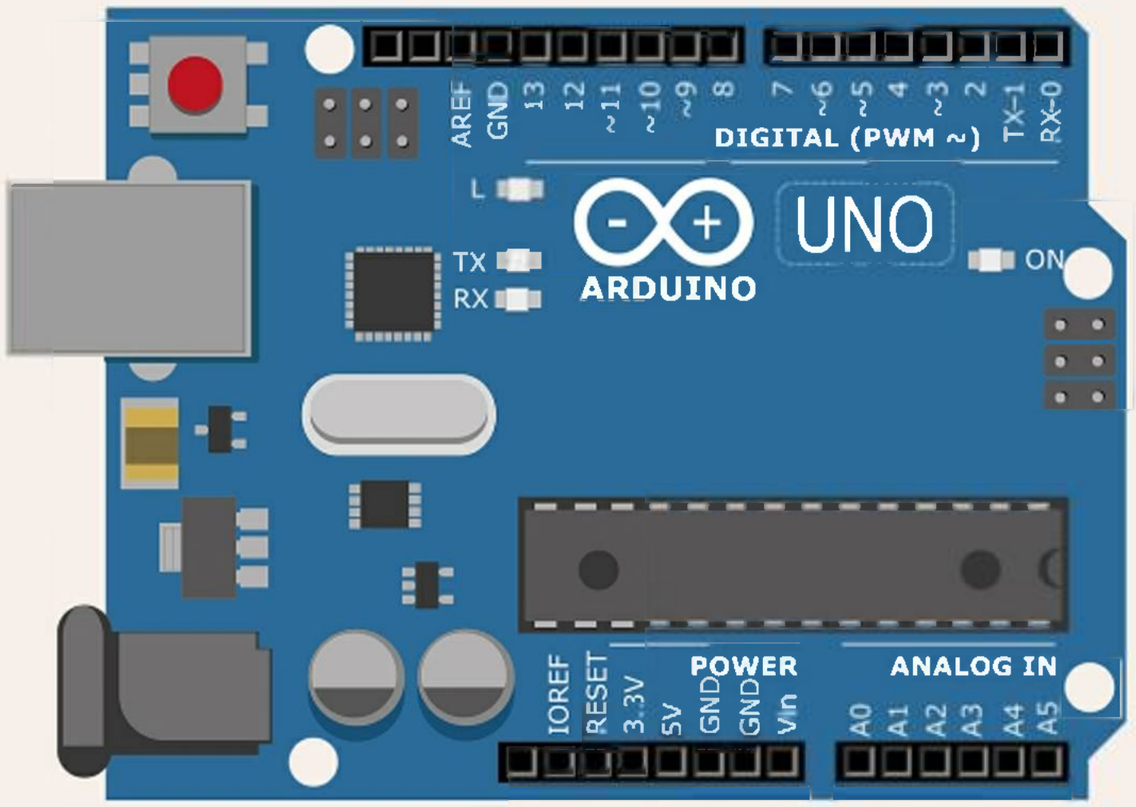
DEVELOPING CUSTOM ARDUINO AND WEB USING IOT PROJECT

A Practical Guide to Memory Management and Efficient Programming to Real-Time Industrial Data Monitoring and Control



DEVELOPING CUSTOM ARDUINO AND WEB USING IOT PROJECT

A Practical Guide to Memory Management and Efficient Programming to Real-Time Industrial Data Monitoring and Control

By

Emporio Hancock

TABLE OF CONTENTS

THIS IS WHAT WE ARE GOING TO DO

VIEWING THE COMPLETE SETUP ON MY WORKBENCH

THE IO CONNECTIONS THAT WILL BE USED

CONNECTION TO MAX485 MODULE

TEMPERATURE HUMIDITY SENSOR

SENSOR CONNECTION ON MY WORKBENCH

SOME BASICS OF THE ARDUINO IDE

WRITING THE SKETCH FOR THE MODBUS COMMUNICATION

ADAPTER AND WHERE TO PURCHASE

HOW TO CONNECT THE ESP8266 TO THE ARDUINO MEGA

PHYSICAL CONNECTIONS ON THE WORKBENCH

UPDATING THE FIRMWARE ON THE ESP8266

THE FLASH DOWNLOADER AND FIRMWARE FILES

WIFI MODULE USING THE FLASH DOWNLOADER

DOWNLOADING AND INSTALLING THE BLYNK LIBRARIES

RECEIVE DATA FROM THE ARDUINO

THE MODBUS CODE IS MERGED WITH THE BLYNK TEST CODE TO CREATE A NEW SKETCH

MODIFYING THE BLYNK APP PROJECT

REVIEWING OUR BLYNK IOT PROJECT

AN OVERVIEW OF HOW THE THINGSPEAK IOT SYSTEMS WORKS

FUNCTIONS IN C

VOID KEYWORD IN C

DIVIDE C PROJECT INTO MULTIPLE FILES IN C

HOW TO CREATE A LIBRARY IN C

ARRAYS IN C

ARRAYS EXAMPLE IN C

HARDWARE AND SOFTWARE REQUIREMENTS

HC05 BLUETOOTH MODULE

DIFFERENCE BETWEEN SOFTWARE SERIAL VS HARDWARE SERIAL IN ARDUINO

FIRST CIRCUIT - AT COMMAND MODE

SECOND CIRCUIT DESIGN FOR FINAL USE PIN32 INCLUDED

PROGRAMMING ARDUINO FOR PC WIRELESS PROGRAMMING

REQUIRED AT COMMANDS TO GET ARDUINO AND BLUETOOTH READY

PRACTICAL ASSEMBLY AND SENDING CODE FROM PC WIRELESSLY

CODING ARDUINO FOR MOBILE WIRELESS PROGRAMMING

HOW TO PROGRAM ARDUINO USING A MOBILE DEVICE BLUINO LOADER

PRACTICAL PROGRAM ARDUINO WITH A MOBILE VIA USB

PRACTICAL PROGRAM ARDUINO WITH A MOBILE VIA BLUETOOTH

DOWNLOAD AND INSTALL ARDUINO SOFTWARE

DOWNLOAD AND INSTALL CIRCUIT DESIGN AND WIRING SOFTWARE

INTRODUCTION SCOPE OF LEARNING

FIRST STEPS WITH AN ARDUINO

INTRODUCTION TO ELECTRICITY AND DIGITAL ELECTRONICS

IMPORTANT COMPONENTS OF ELECTRONICS & DIGITAL ELECTRONICS

THE ARDUINO BOARD (HARDWARE)

THE ARDUINO SOFTWARE (IDE) & PROGRAMMING BASICS

PROJECT 1 A FLASHING LED AND AN SOS SIGNAL

PROJECT 2 TEMPERATURE BASED LED LIGHT

PROJECT 3 LIGHT-DEPENDENT CONTROL OF A MOTOR (BLIND MOTOR)

PROJECT 4 GAS DETECTION ALARM

PROJECT 5 PASSWORD PROTECTED MECHANICAL SYSTEM

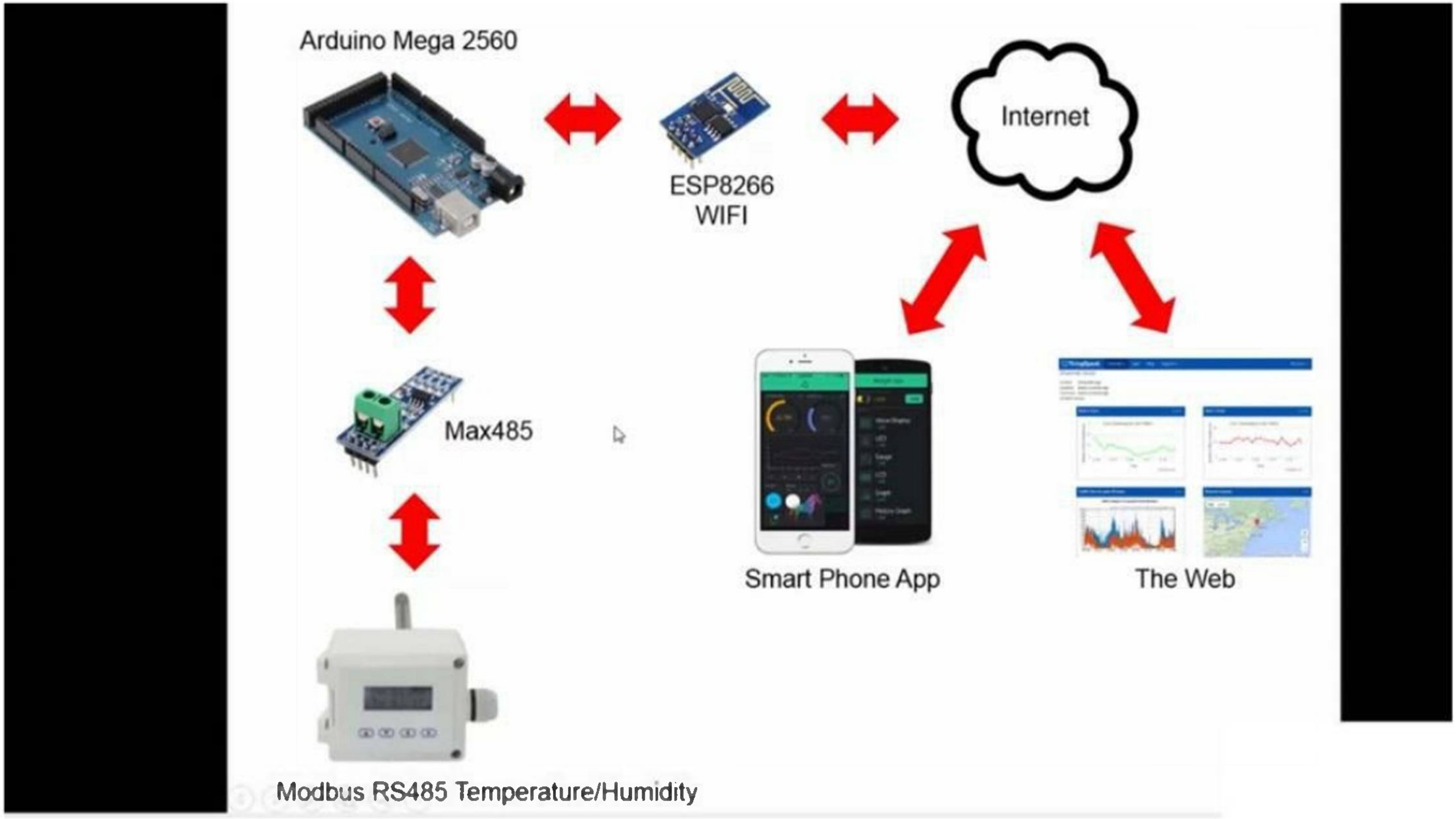
PROJECT 6 REMOTE CONTROLLED UNLOCKING MECHANISM

THIS IS WHAT WE ARE GOING TO DO

The Arduino meger 2560 This is the heart of all projects. This is the guy on the brain that's going to be controlling our entire project. Now usually in Arduino based projects you would see used most of the time and already we know you know it's the most popular of all the different types but we're using an Arduino mega which is a little more advanced and has some more features and it's a little more expensive but it will make life very easy. When building this type of project I'll tell you why the Arduino mega has for you Art or serial ports where the Arduino Uno has only one. And the fact that this has four will essentially make life very easy because what this already we know Omega has to do is multiple simultaneous serial communications and you need multiple serial ports for that. So it's going to have to be communicating via our as FOID fiber one of its ports to them or bust sleeve then on another port to essentially a wife and Madill to the Internet and then on another port through the programming cable to give us feedback and debugging messages right. So those things have to be happening simultaneously. So let's see what we do next. All right. So onto this Arduino Omega 25 60. We're going to first connect a Mock's 45 model so this will actually be connected to one of the serial ports on the Arduino Maggart and this marks forty five module will essentially do level shifting from TTL of $5 to the ORUs forward five voltages right and will allow the Arduino Magar to basically talk on a max Asare on an R for five network. And what is going to communicate with this guy here is some more. Most are for five temperature and humidity sensors. And this will be our Modbus sleep device

and the Arduino Magar will be a Modbus master device. And what it will be programmed to do is to constantly pool this device the Modbus will ask for five temperature and humidity sensors for one reading of temperature and one reading of humidity and constantly store up those readings in its memory. So that is the first thing that we're going to do and we would probably spend an entire section on it. After that is done and we get that up and running and were getting good values here because this device has an LCD display so we can compare what we're getting here to what we're getting in there and ensure that it's working well and after that we are actually going to then connect the Arduino mega to an ESB to 6:6 Wi-Fi module and this wife I-mode you will as you probably could tell from the name will allow the Arduino to actually connect to a wife network and therefore connect to the Internet which is what we want. We're actually going to connect the Arduino to the internet I'm seeing now but it just always sounds very cool to me that you could actually do this. So this model here you can see there's a little antenna on top there and so on and we program the Arduino to use this module to connect to the Internet the public Internet. Right. And I'll be connecting of course to the Wi-Fi network right in my office. And if you do the project will be your wife and that work Right. Once is connected to the Internet. Then we're going to do two things. Number one is this. We're going to download a smartphone app from a company called Blink and then configure that as well as configure a cord in the Arduino so you will view essentially temperature and humidity data on the smartphone itself. Right. Which is really cool. After that we're going to rewrite the code. And we're going to then use a service called things Peat which is a Web site which is an I.T. Web site and then program the Arduino again to get the data onto our Web page for it to be viewed, who is actually going to graph it and so on. Right. So that is essentially what we're going to do with this is the goal of our project to get Modbus data on the web onto a smartphone app. That's what we're going to do. Now we're going to build this gradually. I'm go-

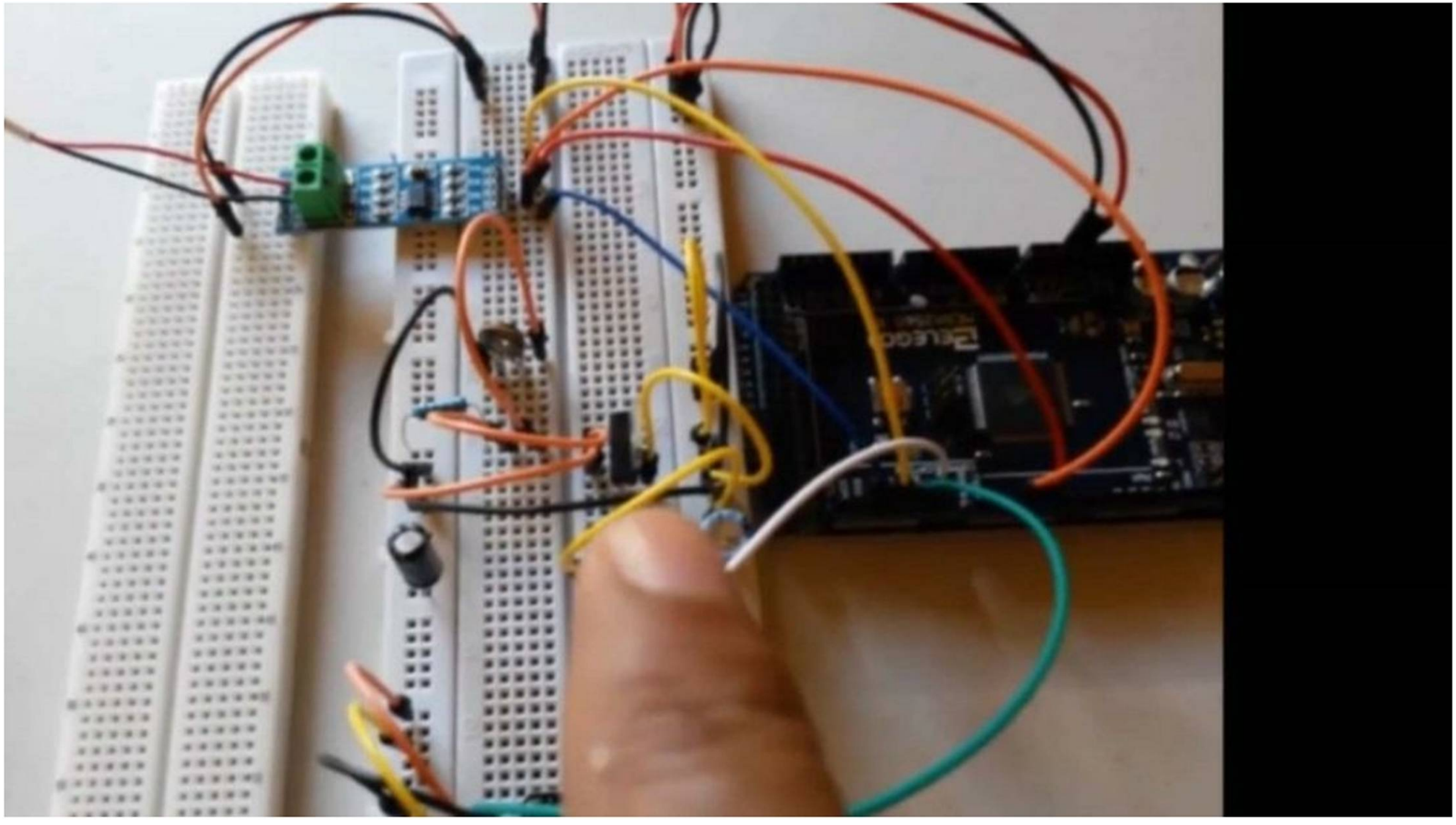
ing to build the code gradually. I'm not going to just give you a thousand lines of code and say here please understand this. That's fine.



Now we're going to build it and I'm going to show you what each line of code does and what its purpose is right. So you know we all have an understanding not just copying and pasting corda.

VIEWING THE COMPLETE SETUP ON MY WORKBENCH

All right. So I know it looks messy like a jumble of wires but let me just point them out to you right. So this is our Arduino MEGO right here. You can see this is our programming cable. This is a power supply powering the Arduino maker. Good. Now let's move on to the more buzz part. So there are some wires going across to this bread board here and that is the max for 85 you will that I showed you. This is a pair of why is it going here all the way to our bus. 45. Temperature and humidity sensor right there. It's not powered as yet nothing has power right now. But I just wanted to show you that on my desk. So this is how we do max four and five to our senses here. That's the main part of our project. Right. Right. Great. Now let's go over here. This little guy right there is our wife. I will speak to 6:6 and he can see why he's going there.



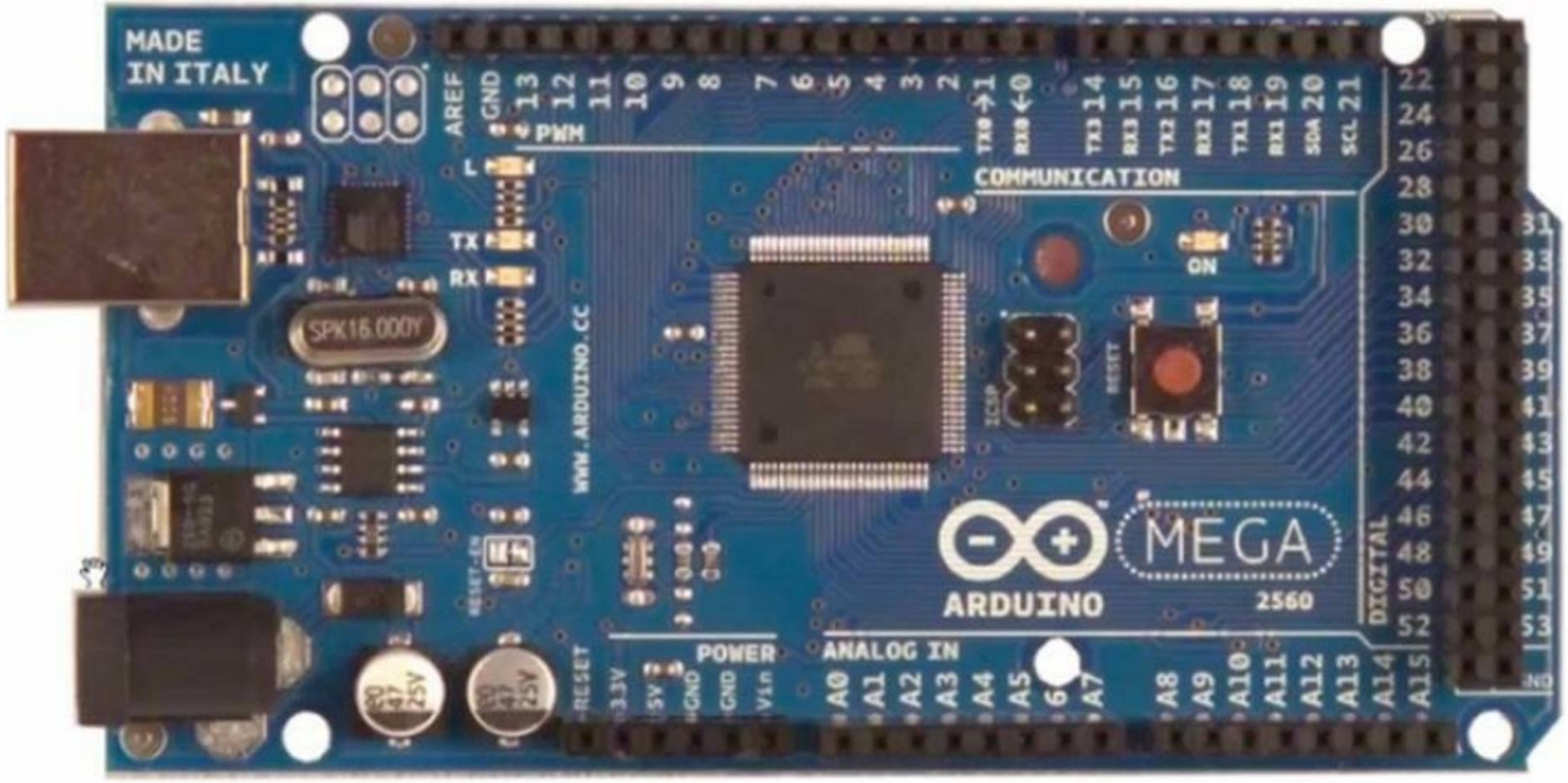
Are you seeing some sort of discrete component in the middle there? I'll explain those in detail. But essentially what that is is to break down five ways to 3.3 volts. That's all it does to supply 3.3 will power to this guy right here. Because he doesn't use fireballs or use 3.3 volts and you can see like a little antenna right there that's why fi antenna. So this is our Internet part of that. So we know in addition to communicating

via mobile Square will get it to communicate via a wife through this. Yes, the two 6:06 wife monu and the data it collects here will then be sent through his wife. We find more firstly to the smartphone app and secondly to the web using an I.T. IATSE is a public art server. So this is what it looks like. All wired up on my desk.

THE IO CONNECTIONS THAT WILL BE USED

So in this section what we're going to concentrate on are the hardware and wiring aspects of the connection of the mega to the max FOID 5 module to the Modbus slave which is our Modbus artist FOID 5 temperature and humidity sensor. So we're going to concentrate on hardware. In the next section we look at the code and software and so on. So you will see two resources associated with this lecture , one is a link to this Amazon page. So what I actually am using in this project is not a genuine Arduino mega it's a clone but it works really really well and I bought it at Amazon because I tried to buy everything on Amazon because I have Amazon Prime and it's much cheaper it's 14 ninety us here and on the Arduino side it's like 50 bucks. So this is an option for you to buy. I included the link here if you want to buy it here you can buy your Arduino mega anywhere you want. This is just my advice. So that's on the Amazon page right there. The other link is essentially It's a downloadable file which is a data datasheet for the Arduino Maga. And I want to just have a look at that data sheet and look at the pins on the mega that we will be using.

Arduino Mega 2560 Datasheet



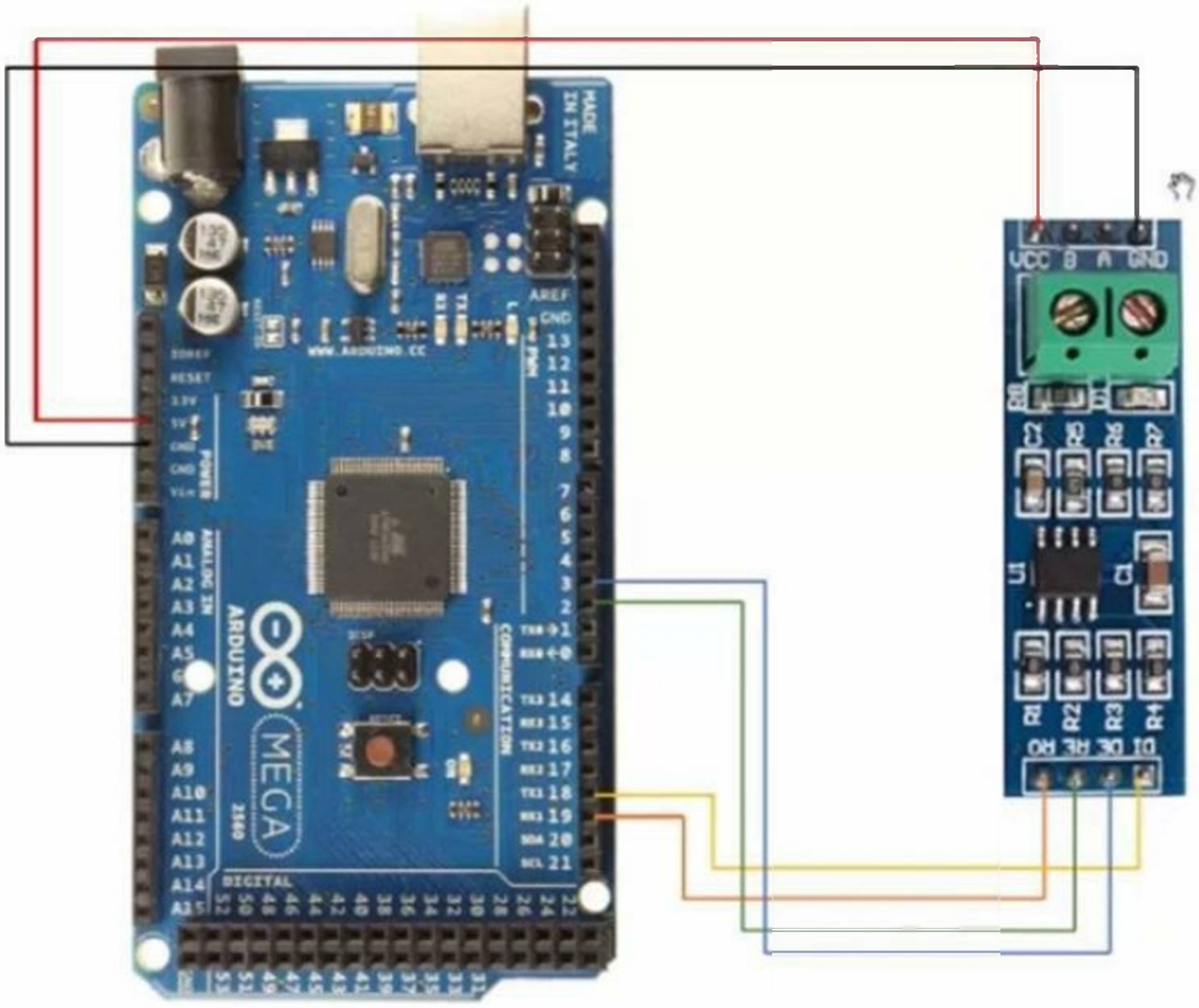
So let's have a look at that. Now there we go. All right. So I got this from a robot shop. And it's a pretty brief data sheet. We look at a few things there. But what I want to concentrate on is this really crisp clear image that they have excellent excellent image you can see everything really clearly right. So we have our programming port right here. And of course we have a power supply here. Now I want to just mention some-

thing on the spot as applied. All right. So usually in most Arduino projects you can power the Arduino through the USP and it works just fine. But with this project the Wi-Fi mode you take some current and it takes some power right. So you should use the external power supply on the Arduino board. And what I did was I didn't actually order this from Amazon or com. I went to my local electronics store and I asked them for essentially a 12 volt DC power supply and 1.5 amps. And I actually carried the mega ball and ensured that the plug in the connector actually mated properly with this right. So what you want to get is a 12 volt DC 1.5 power supply and plug it into here for this project. Otherwise you run the risk it may work but you run the risk of not giving the wife module enough power to connect. So you just want to get that for this project. OK, good. So 12 volts 1.5 arms you can get one you can get away with 1.0 as well. Now there are a lot of pins on this Arduino mega most of which we don't care about. There's a bunch of analog pins to connect devices. They're a bunch of digital pins to collect and connect devices. We don't have any devices that are directly connected to here. So we don't really care about all of these. I'm not going to go through all of these. What do we care about a lot? A lot are these guys right here. Zero 1 0. So this is the serial port zero essentially. And there's one story arc: one takes one serial port, one x 2 takes to serial port two, x tier serial port three. All right. So there are four serial ports in total. As I said in a previous lecture, this one here we're not going to touch at all zero on our exit is actually connected to the communications here with the programming device which is the laptop right to us cable. So we're going to not touch these at all because we're going to be using something called a serial monitor. Some of you may be familiar with it, some not. But essentially it's for debugging. All right. So while the program is running you can send serial messages back through this connector to something called a serial monitor which is part of the Arduino software and get you debugging messages and ensure that things are running the way they should. So we're not going to touch that. We are going to use Tia X-1 or X-1 these two to connect to the max r as Max 4 5 connector. So

Modbus communications are going to be happening on the pins 18 and 19 and then TAXI to an R x 2 which has been 16 and 17. We're going to be using this serial communication to go to the Wi-Fi module the DSP to 6:06. So we're going to be using this one here right. Just for debugging TXI one on our one for our is FOID five communication TXU and our X-2 for Wi-Fi communication. All right. So that is what we're going to be using. We're also going to be using pins 2 and 3 for more for the Modbus flow control and I'll show you how that works. Of course we'll be using volts and ground because we need to power the Mock's four to five you as well as the Wi-Fi module. So we will be using power because this is output power from the board itself. And that essentially is it. It will be I guess you could say wasting a lot of these pens but really and truly we need these multiple serial ports. So they are doing it. Magers are a really good choice for this and it's great though that we're not going to be using a horse set of pins because it makes the wiring messy right. So that essentially is they are doing a mega And that's our plan for pin usage and IO usage on the unit itself.

CONNECTION TO MAX485 MODULE

So what you're seeing right here is the wiring diagram that shows a connection of the Arduino mega 25:16 to the max 45 chip. Again this makes 45 sorry you will allow communication to the more bizarre is for five temperature and humidity sensor viruses for five. So this is a PDA document which is also attached as a resource onto this lecture so you can download it, print it out and have it for yourself. All right so let's have a look at it as you can see we're not using a heck of a lot of pins right. So let's go to power so you can see there are four pins on the side and the four pins on the side here for this module. So there's Vcc and ground. This is five holds power ungrown you can see this red wire goes to 5 volts. This block goes to ground. Now this is not the same color coding I use in my on my actual connection to my desk the physical connection I just use these different colors to discriminate one from the other discern one from the other.



So that's power. So we put the device with a 5 volts grid. Pretty straight forward. Now let's go to transmit and receive. We're using port number one which is our X1 anti-X 1 pins 19 and 18 respectively. Our X goes to our X1 goes to our all receiver which stands for receiver. We will take a look at our just now and then t X1 goes to D which is the driver in. So it is on these lines that data will actually be transmitted. So these are

the data lines going as the inputs on this and it will actually go all out on it and be here. This is Ian b r s FOID 5 that goes to the sensor. All right. So the receiver does the receiver pin and the driver in the eye is the transmitter pen. So essentially what we do is receiver to receiver and transmitter to transmitter t x goes to D and R x goes to r. Well OK it seems very logical right now. These two pins D-E on our control pins now are as FOID five inherently tends to be what you call a half duplex sort of communication. So if you send data and receive data on the same two lines. So this d n r e controls the flow of data right here. So Ari stands for receiver uneatable enabled when this pin is lower so the receiver on this R is 4 5 is enabled when this the r e pin is low OK and when. So in receiver more that's when the Arduino wants to wait in response from the more buzz for its five devices or a sensor. It will drive the receiver enable pin law to allow data to be received on the receive pin. However when it wants to transmit and send a Modbus Querrey to the Modbus sleeve it will transmit on Also the driver Annibal is enabled when pinners high it will drive DC high and then push that data out there. So what it does is it goes transmit then we receive the transmit. So it is constantly changing the basic voltage levels on these pins to control the flow of data. So in other words our D and D we call data flow control. They control the flow of data on this particular device. All right. So those are the two purposes of those pins. All right as I said Ian be as you are at 45.

TEMPERATURE HUMIDITY SENSOR

Now before I get into the meat of the content of this project it is something I forgot to do in the previous project. Does it tell you that there was a link to the Amazon sales page for the remarks FOID five Moggi. Right And I have the link right here. So this is where essentially I bought the max for five more that we saw in the previous project. Right. This is for five converters. And it was a pair of them for like three 19:9 us right. So I just bought the pair right here. Of course you can get this from lots of different places. I just bought it on Amazon. So I've also included the link to this. And this lecture as well in this project as a re-source just in case you know you missed the one in the previous project. OK, good.

So let's get into the clues that don't get into the meat of things here. What I want to do in this project is to actually give you an idea. I'll show you what our Modbus for ID five temperature and humidity sensor looks like right now. I've included a resource link and the resources to this particular page if you want to buy this. You can use any more bizarre is for a five temperature humidity sensor or you can just use any more buzz

FOID five sleeve device that you have where you can get cheap or whatever right to do this sort of testing. So this is where I got mine from D to now but it was a kind of expensive 119 U.S. but a good investment is a really good device. This particular one has like four five as well as Modbus TCAP it has Ethan that the one that I have does not. That has been sort of discontinued but this one sort of takes place. And let's look at the data sheet for it. This is a data sheet and I also attach this as a resource on to this particular lecture. So let's look at it here right. So what are we talking about a Modbus four five device. We want to have certain pieces of information right. And that'll be several things. First of all we need to know what the border is. What are the port settings that are used to communicate with this device? So what I did was I just sort of scrolled down through here and I saw right here it says default COMM port parameters ninety thousand two hundred board the Tibbits no parity one stop. So all of the port settings right here do a default.

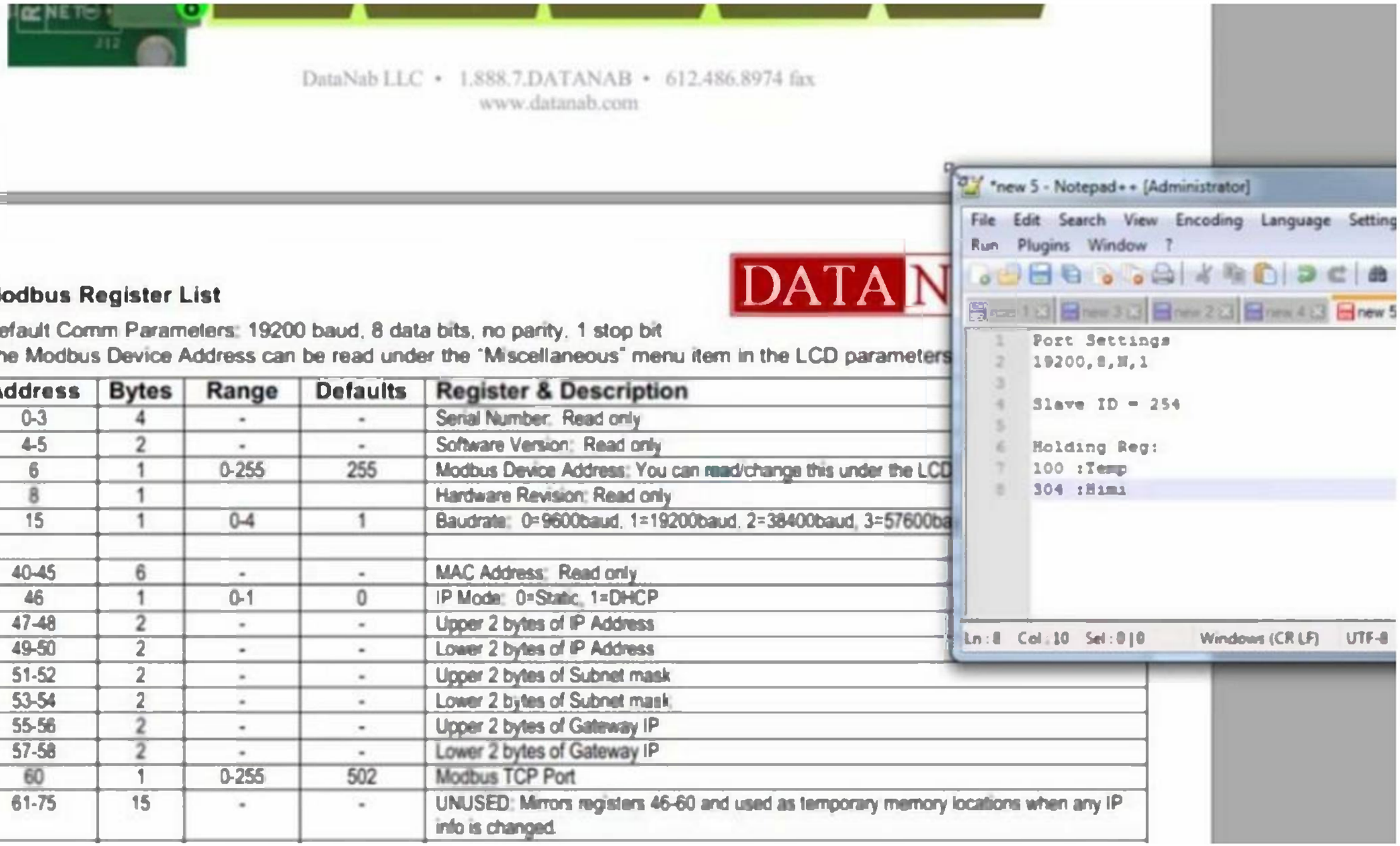
Modbus Register List

Default Comm Parameters: 19200 baud, 8 data bits, no parity, 1 stop bit
The Modbus Device Address can be read under the "Miscellaneous" menu item in the LCD parameters.

Address	Bytes	Range	Defaults	Register & Description
0-3	4	-	-	Serial Number: Read only
4-5	2	-	-	Software Version: Read only
6	1	0-255	255	Modbus Device Address: You can read/change this under the LCD "Miscellaneous" menu
8	1			Hardware Revision: Read only
15	1	0-4	1	Baudrate: 0=9600baud, 1=19200baud, 2=38400baud, 3=57600baud, 4=115200baud
40-45	6	-	-	MAC Address: Read only
46	1	0-1	0	IP Mode: 0=Static, 1=DHCP
47-48	2	-	-	Upper 2 bytes of IP Address
49-50	2	-	-	Lower 2 bytes of IP Address
51-52	2	-	-	Upper 2 bytes of Subnet mask
53-54	2	-	-	Lower 2 bytes of Subnet mask
55-56	2	-	-	Upper 2 bytes of Gateway IP
57-58	2	-	-	Lower 2 bytes of Gateway IP
60	1	0-255	502	Modbus TCP Port
61-75	15	-	-	UNUSED: Mirrors registers 46-60 and used as temporary memory locations when any IP info is changed.
100	2	-	-	Room Temp Reading in DegF. This reg can be written to adjust temp calibration offset
101	2	-	-	Room Temp Reading in DegC. This reg can be written to adjust temp calibration offset
121	1	0-1	1	Temperature units for LCD display: 0=DegC, 1=DegF
185	1	0-4	1	Baudrate: same as register 15 above.

So that's what I am going to use in this particular project. I'm going to leave it as default and the next question is of course what is the unit ID or sleeve ID of this device and what are the registers I want to read. So they didn't see it here but I'm looking through. I looked through the registers and description and I saw it says Modbus device address. You can read or change the LCD on the menu. Right. And the default for the

address is 255. OK so I think I had set mine to 254 or something. Right. The actual address so the unit ID address will be 254 and the registers I want to read are this one 100 which is the room temperature in degrees Fahrenheit and 3 or 4 which is the humidity in percentage percentage relative humidity. Right.So one hundred and three or four. No these registers here. They didn't see whether their input registers or holding registers. But I sort of got a hint when it says you for the device address you can read or change this right. So you can write to these registers as well. So I did a test and found that it was holding registers and sometimes devices are like that they don't say exactly what the register type is so you have to do a test. So if I were to sort of sum up. All right, let's get mine there. All right I would see isko port settings. The ride is 19 200 The Tibbits no party 1 stop there and the slave ID is equal to 2:54 which is what I set it out and register holding registers. One hundred is the temperature and three or four is humidity. Right. So that's the information I need. I know the address of the device and the address of the device I'm going to read from. I know the registers only type and I know the port settings. That's all I need to get data from the device.



DataNab LLC • 1.888.7.DATANAB • 612.486.8974 fax
www.datanab.com

DATAN

Modbus Register List

Default Comm Parameters: 19200 baud, 8 data bits, no parity, 1 stop bit
The Modbus Device Address can be read under the "Miscellaneous" menu item in the LCD parameters

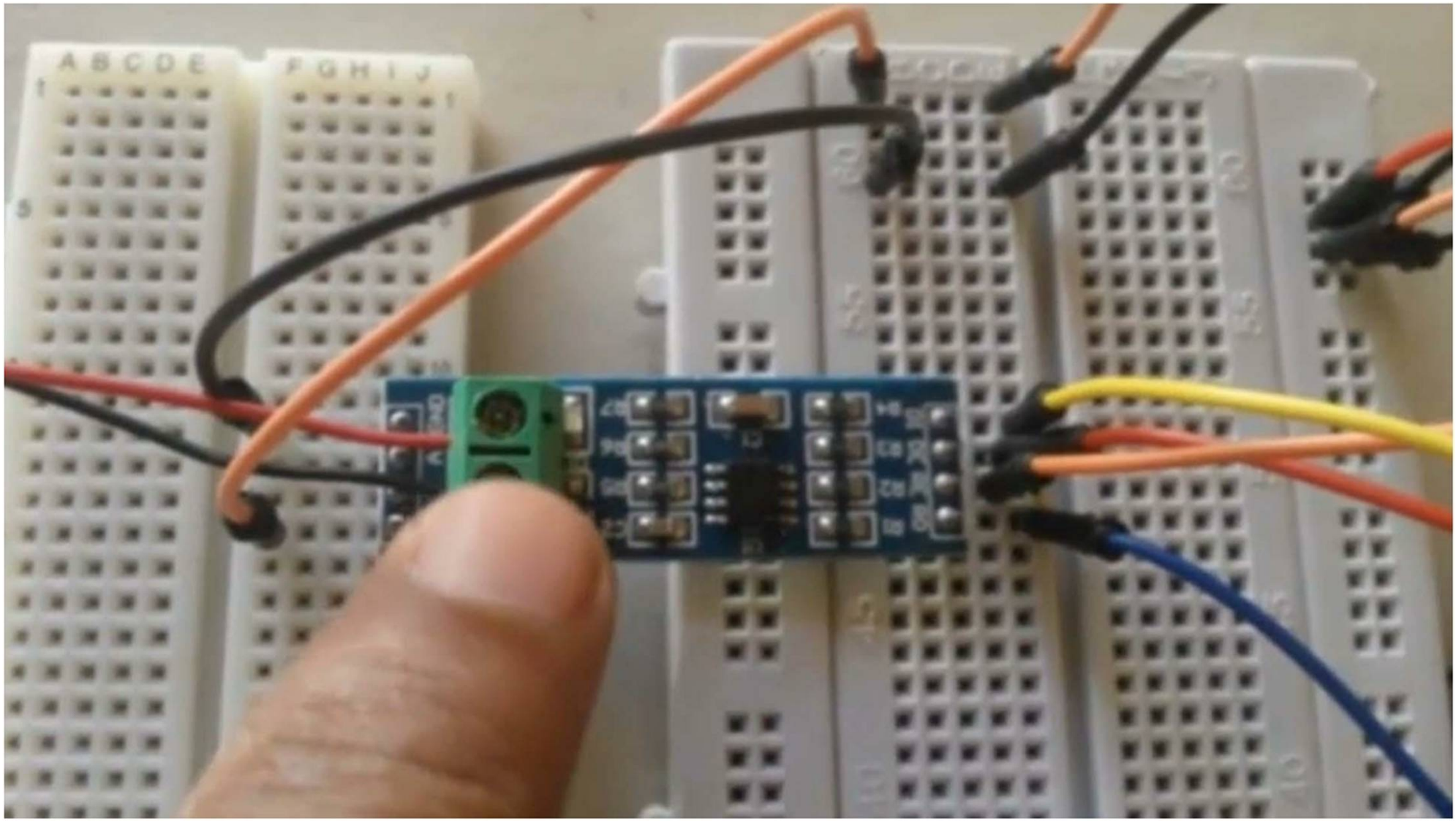
Address	Bytes	Range	Defaults	Register & Description
0-3	4	-	-	Serial Number: Read only
4-5	2	-	-	Software Version: Read only
6	1	0-255	255	Modbus Device Address: You can read/change this under the LCD
8	1			Hardware Revision: Read only
15	1	0-4	1	Baudrate: 0=9600baud, 1=19200baud, 2=38400baud, 3=57600ba
40-45	6	-	-	MAC Address: Read only
46	1	0-1	0	IP Mode: 0=Static, 1=DHCP
47-48	2	-	-	Upper 2 bytes of IP Address
49-50	2	-	-	Lower 2 bytes of IP Address
51-52	2	-	-	Upper 2 bytes of Subnet mask
53-54	2	-	-	Lower 2 bytes of Subnet mask
55-56	2	-	-	Upper 2 bytes of Gateway IP
57-58	2	-	-	Lower 2 bytes of Gateway IP
60	1	0-255	502	Modbus TCP Port
61-75	15	-	-	UNUSED: Mirrors registers 46-60 and used as temporary memory locations when any IP info is changed.

All right so we have these readings here now. And now I have the information that I need to move forth and read data from this device. All right, let's have a quick look at the inside here. You see these two minute blocks right here. If I just make this a little bigger you'll see that their power 24 volts needs to be powered here. And ours for 5 is down here. So there's four white five wires coming in here and I'll show you how this

looks on my desktop as well on my workbench. But that's the device essentially. So it has power. And it has an R S for its 5 connections through which data will be exchanged with the Arduino with its max for five Mordieu.

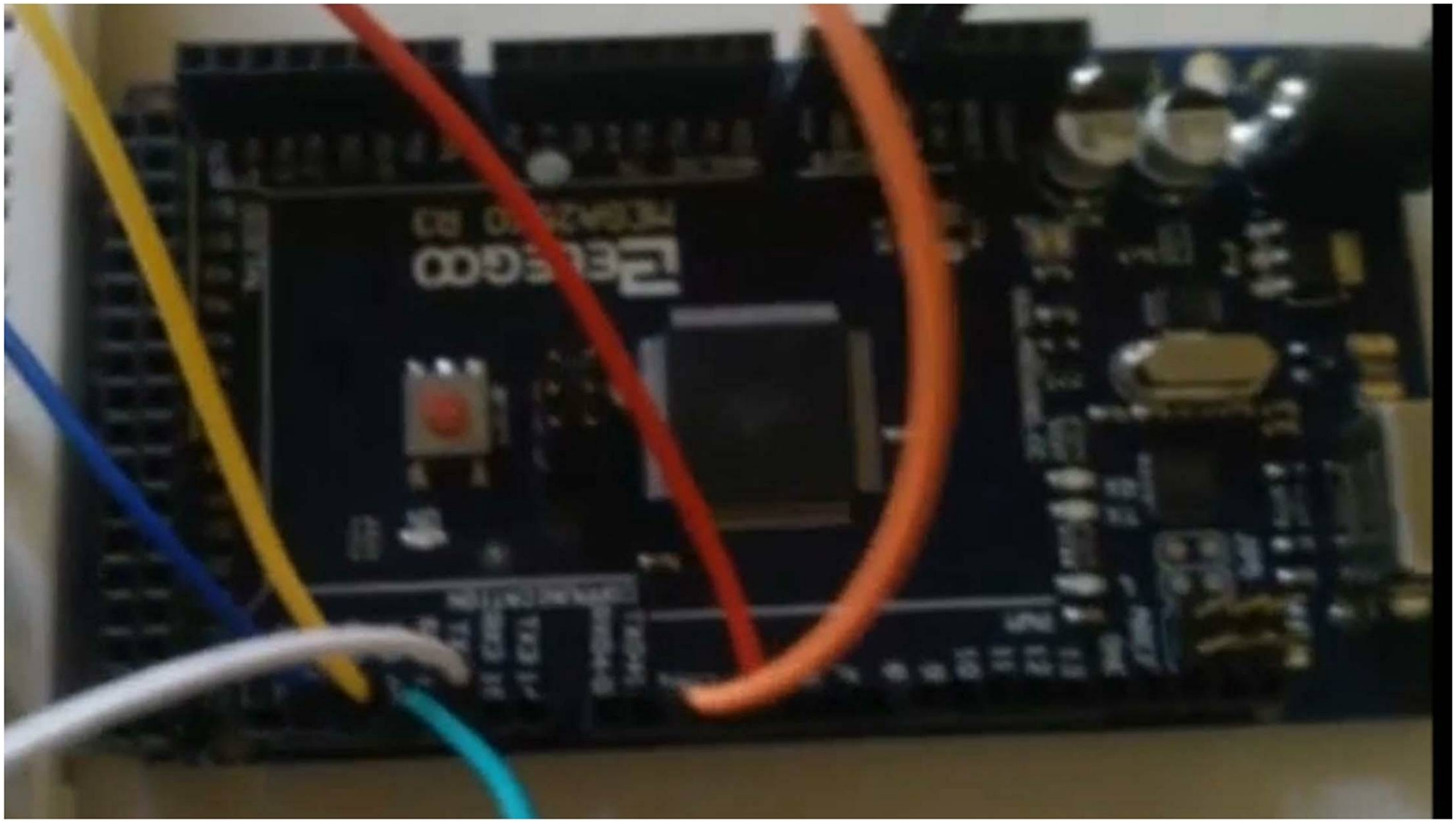
SENSOR CONNECTION ON MY WORKBENCH

Was the wiring diagram on how to connect our arguido here to the max for it. Five more do you. And then of course eventually the mob boss is asked to drive the ball with the temperature and humidity sensor. Right. So I just wanted to show you that wiring on my workbench is So let's take a look here now so it will be difficult to see but you should be able to see these pins right here. Zoom in a bit. That's the five volt and ground pins coming out of there and they sneak over here and then they jump to then jump to them and they jump there and essentially go to the Vcc and run right on this end on that end of the marks for it.



Five more to you then. B. Which is or is FOID five.This snake is a pair of wires to my Modbus sleeve essentially. Right.Rs 45 slave. So that is the connection to the front here. Now on this side no we have also transmitted to receive this yellow wire and red wire yellow wire and blue wire go all the way back now to

my T X and our X one here. Right. You can see going right back into the into the connectors then we have pins 2 and 3 on the Arduino orange and red wires going to be flow control which is d e r e.



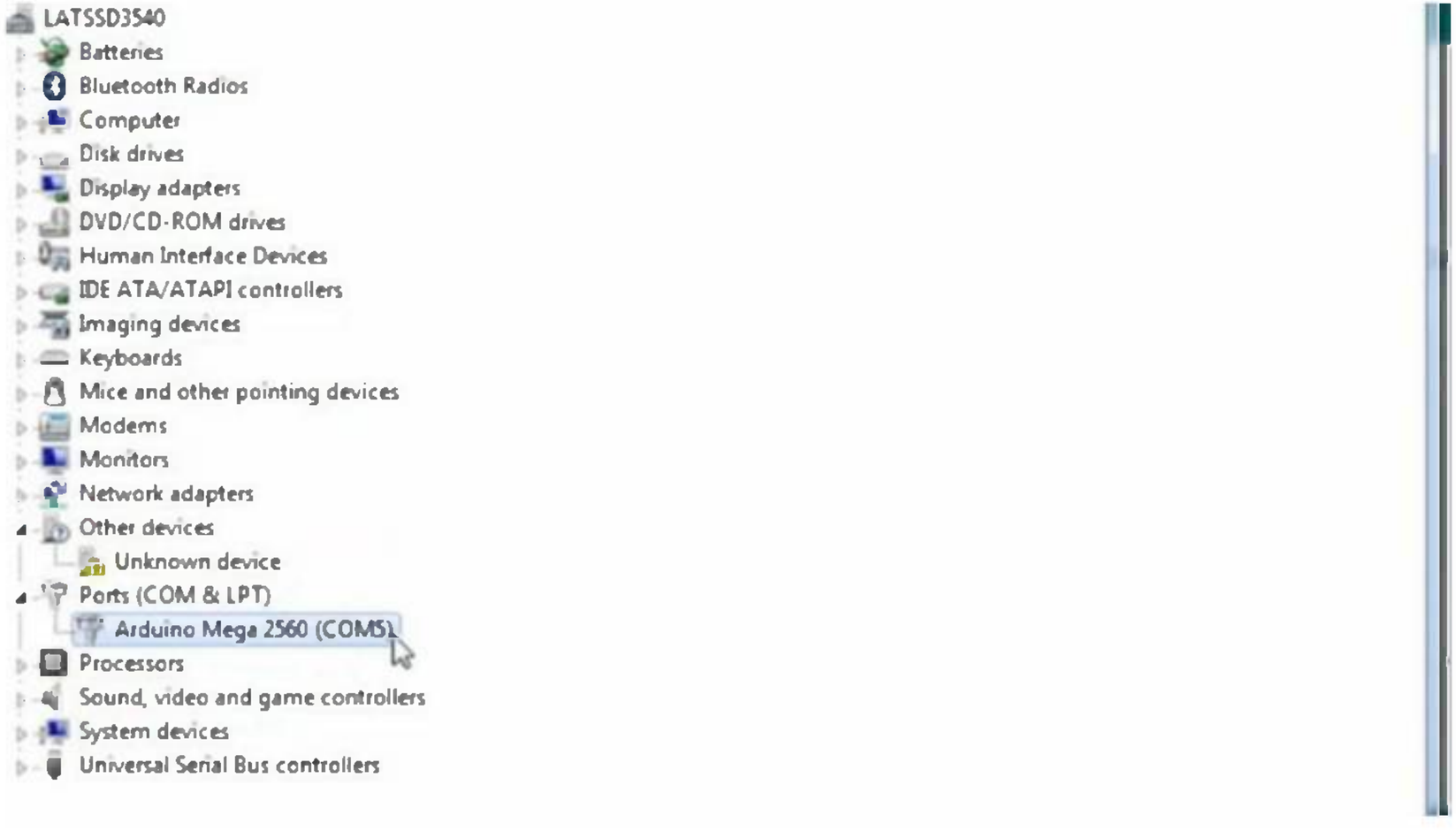
So that's essentially following the wiring diagram and should be in a previous lecture. OK so we have all the power. We have our data lines and control lines essentially going towards this end here and power is not in there. And only hours for five sneaks all the way over to this guy right there. So it goes to the arrest for 5 connectors which are in the device. You can see Iris 45 right there. And the wires go straight into this terminal block. Then we have the power for this device snaking all the way up there onto my 24 power supply on my desk. If I close this and you see the LCD display it will light up once we power this device into temperature and humidity. And that is it for all of us is a 45 minute walk on my work bench on the part of this project that it is.

SOME BASICS OF THE ARDUINO IDE

All right we're on the Arduino Web site right. And some of you all will nobody or do we know some of you will not know. I don't have the time and this course really to go through the basics of the Arduino software but I will still while I'm actually building the application still point out a few things for newcomers and so on. But if you are new to Arduino and new to the interface this site is excellent because there are a lot of tutorials you can get started with and the software is essentially free so you can go to software downloads and you will be able to download the idea. This is what I have right.

I just clicked on Windows Installer and it's the Arduino 1.8 point five version. Right so we have no idea there. So you can go ahead and download this and install it and so on. So I will go to my Arduino application that I have installed here so this is the Arduino ID or integrated development environment. Again even if you don't know C code the Arduino software comes with some examples as well as on the Web site itself.

There are these resources and all these tutorials to give you step by step examples. So he can take a look at those to actually lo If you're very very new to this.All right. So this is the actual Arduino idea here. Right. No. Before we jump into the idea, what I just want to show you is something about the compote. So I currently have my Arduino mega connected via US beat to my laptop. Now when you connect to the laptop what gets created is a virtual COM port. So if I go to just type in device manager here device manager and if you look at the ports you see when I plug it in Arduino mega twenty five sixty five. So here I know that communications port five is what is connected to the laptop into the Arduino itself right now. The communications board on your laptop might be different. It might become six come seven. The main thing is that you check it here and you use that when you're configuring the Arduino ID.



All right so we know that it's com 5 which is going to minimize that. And if I go to Tools port comm five is already selected because I selected it from before. But if your brand is a brand new installation you have to go here and select the proper COM port. Right. In this case it might be different for you. In addition to that what you have to do is go to board and select the correct one. In this case I have an Arduino mega 25 60 and I've selected it right there. So compote as well as the actual board that you're using because there are lots of types of Arduino out there right. So we're using the Arduino megger 25:16. OK so compote then choose the board and then you are actually ready. Believe it or not to download an application. So I opened a brand new program. Right. And programs in the we all the local sketches. So you see in this sketch. April 24. So all Arduino applications or sketches have two functions that must be there.It's set up a loop so set up is where you do all your initialization. I loop is the main program that runs over and over again just like the appeals application that runs over and over again. So I have my ID open and what I'm going to do is compile this program even though it does nothing. So this is my button to compile. So I'm going to compile a compiling sketch if I have arrows that will show me arrows right. No arrows so far that's great. And to download. They call it upload but I'm going to call it download. I'm going to click that and you're seeing done uploading down here. It's actually in the Arduino right. No. OK Sue. We've just downloaded our food supplication which of course does nothing but we've establish connectivity to the Arduino which is excellent. So if you're as you said if you are not familiar with do you know if you're not familiar with the art you know Arduino idea. This is a good first step to just verify that you're connected and you could download an application. Now one small small point good to file preferences display line numbers is usually unchecked. I want you to check it. Having line numbers displayed is really really cool because when you get an error it says line number so-and-so. And if the line numbers are displayed then you can easily track that error and find that error. Ok I don't know why they actually have it too.

WRITING THE SKETCH FOR THE MODBUS COMMUNICATION

All right so we're going to continue from the last lecture and complete the code for this Modbus part. Remember we're writing an application just to do the Modbus part and we'll add to it as things go along All right so we sort of reached this part here where we were developing the set up function which of course executes once at startup. All right I just have two more lines to put in there. We're going to pieces right now. There's a little more advanced Right. But let me tell you what's happening. The Modbus master library is actually developed to work with already known arms for five shields and genuine Arduino shields right which is which actually I have but it goes all over the Arduino go you know. Now we're not using the emacs FOID 5 module which is much cheaper for a lot of people. So we have to do a bit of modification. Modbus master has two functions and they are also called pre transmission and post to transmission but they use some different pins as opposed to two and three. We're using two and three to control the flow. So what we're doing is something called callbacks were overriding the default functions in-order and telling it to use our functions, our custom Prete transmission and pooch's transmission functions as opposed to theirs. OK so that's what these two lines do. No, not the transmission. Use this function. So are what are called callbacks. All right. OK, good. So that's it for our set up there. Now we're going to move to the loop function that says put your main code here to run repeatedly. Yes. So let's begin. First things first some variable dec-

laration. OK so these two variables help support the Modbus called we're going to write. All right. Let's put the line in. And here's our first significant piece of code. It says read temperature in the comment.

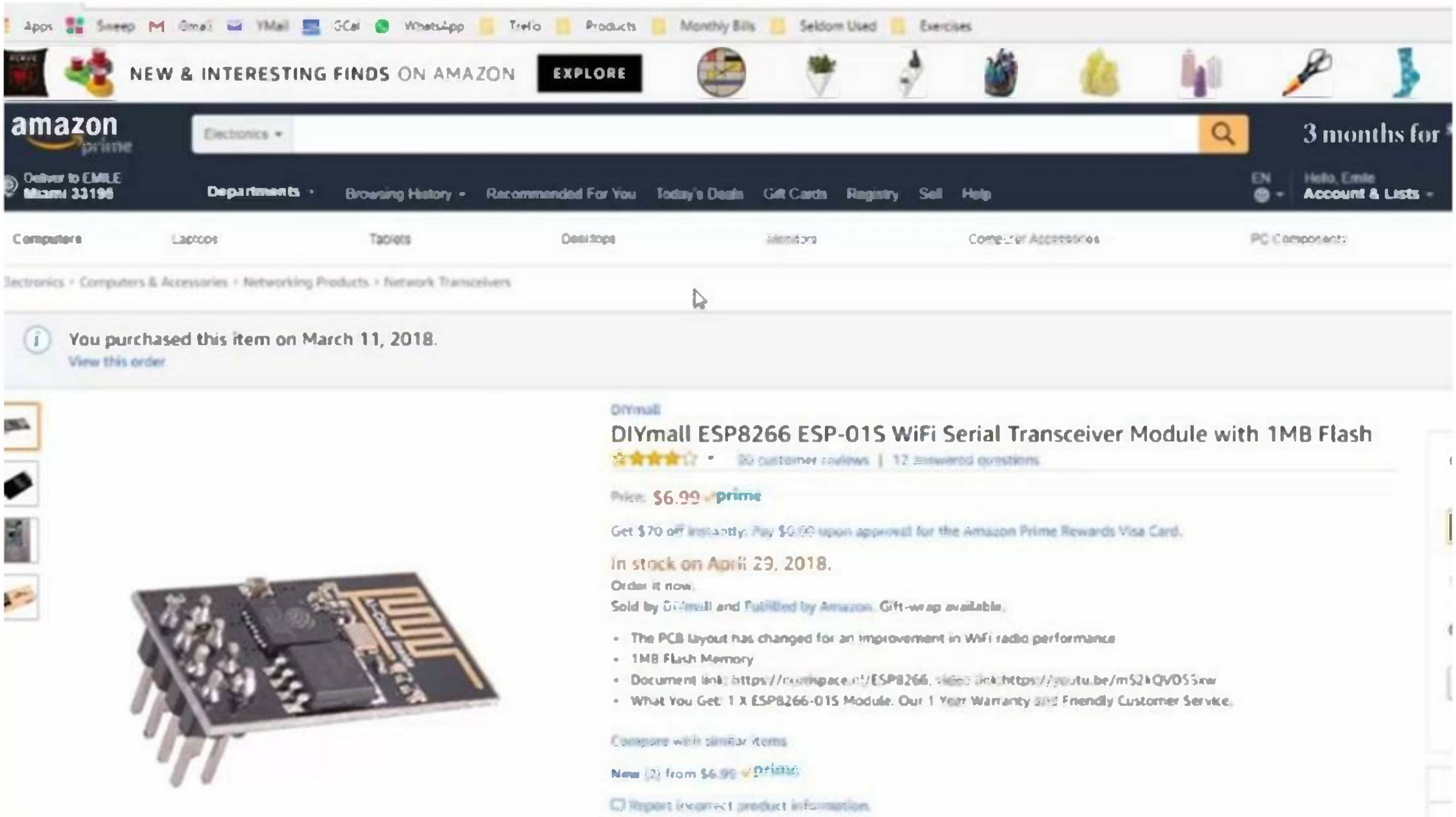
modbus01 §

```
51 // ****** STANDARD ARDUINO LOOP FUNCTION ******
52 void loop() {
53
54   uint8_t result;
55   uint16_t data[6];
56
57   // Read temperature
58   result = node.readHoldingRegisters(100, 1);
59   if (result == node.ku8MBSuccess)
60   {
61     Serial.print("Temp: ");
62     Serial.println(node.getResponseBuffer(0x00));
63   }
64
65
```

So let me tell you what happens here. Result. So Naude is doing a command here. No. Which is the Modbus master object.No not rerolling Halling registers read register. One study from one hundred and read only one. So in other words we're only reading one 100. Why 100. Well if you recall in our notes 100 is the temperature and three or four is the humidity and they're both holding registers. So a reading temperature and if the result is equal to the note key. It may be successful if it's successful then cereal print November cereal is not cereal. This is this cereal here that cereal one which is the most buzz is the writing command messages to the cereal monitor. So see print Printemps it just Princie would turn and then cereal print line print. Means print and put a carriage to and after. No would not get a response but for zero right just get zero here. In other words since we only read one register in the response buffer the result will be in place 0. Item 0. So get a response buffer so this gets the value that comes back which is a temperature. So great then we really just put a small delay in between. We don't want to confuse the device. Then we read humidity and it's about the same thing over again almost exactly. The only thing we're changing here is the address which is now three or four because humidity is three or four zero percent humidity cereal brand and get response. And then I'm going to put in another Dilley. I'm just going to wait five seconds before it reads again. So in other words what this Arduino is going to do is to pool temperature and humidity every five seconds and print the result on a cereal monitor.

ADAPTER AND WHERE TO PURCHASE

So we are on to a new section in this section. What we're going to start looking at is the wife aspect of our project because remember we have to connect our Arduino to the wife at work for it to have access to the Internet. Now I had mentioned the various building blocks in the introduction and this is the ESB is 2 6 6 Wi-Fi modules right. So this is the Amazon page for it. Essentially I bought this on Amazon as usual and the link here for another part I'll show you in a bit. Both of them are resources attached to this project. So if you want you can just go to those resources and get the links. If you want to order them.So this. Now there are several models of the ESB 2 6 6 this is the 0 1. So what you're going to get is the 0 1 model. This is sort of like a watered down version of the real one but it works very well. So this is where you would go to one of the locations you would go to Amazon to purchase it. It's very cheap, 699 at least on Amazon. No. I also put just another part which is here. So let me show you why I did that. Oh before I forget I forgot to mention that in the last project in the previous section I forgot to mention that I attached to the code file or sketch file Modbus 0 1 though I know which is. And I zipped it onto the project itself onto the lecture itself so that you can download that code and have it there. All right. But we're going to modify that code eventually. All right.Back to this now.

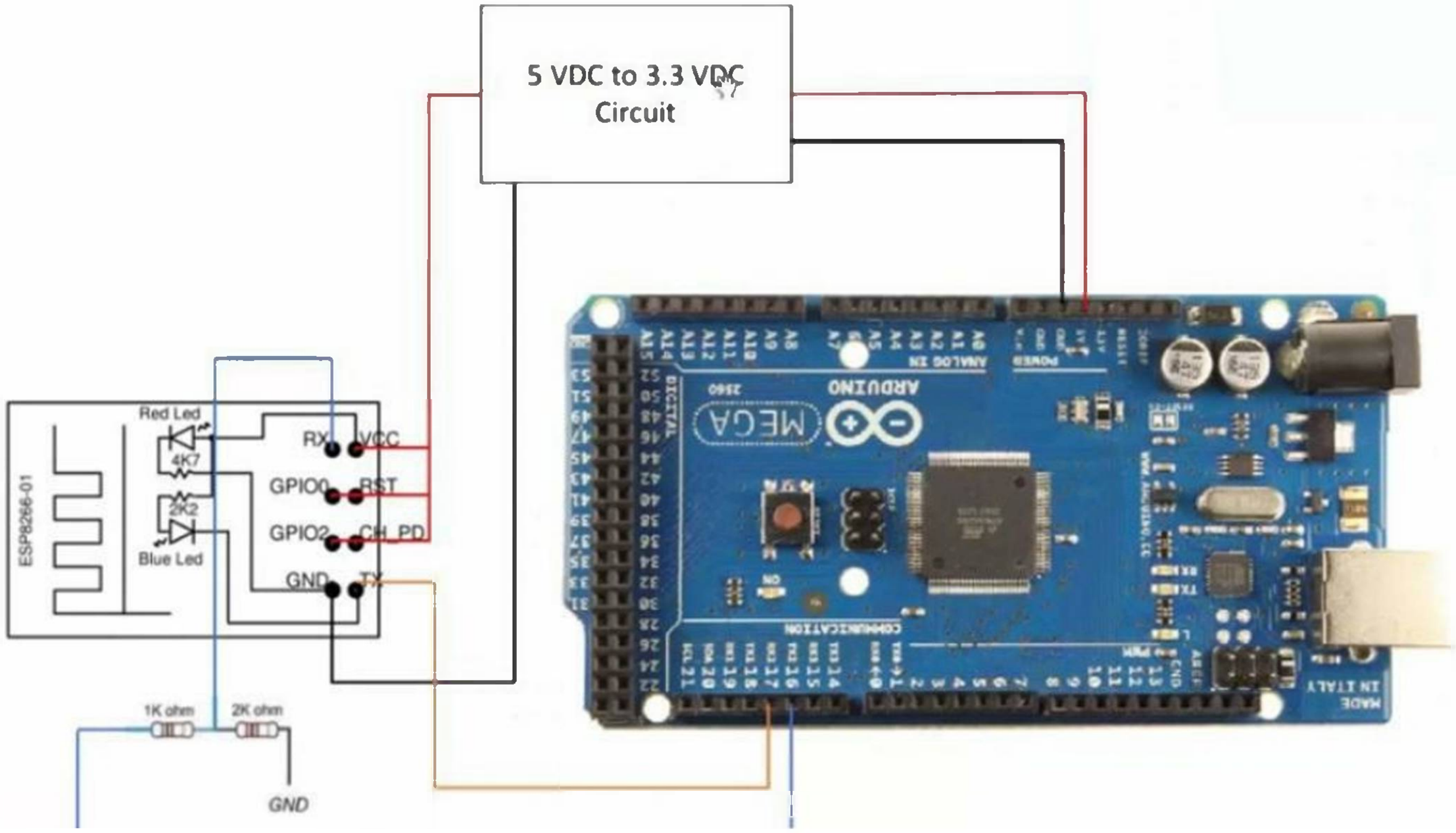


See these pins down there right. Those pins don't fit well. So the last breadboard is all So the moment for jumpers or jumper wires and so on. So I think I can get a version of that. OK this is what it looks like. What I bought was what is called an ESB to 6:06 brick old board breadboard adapter. Right. So the actual 6:06 goes into here. And this fits nicely now onto a breadboard very very nicely. So I also bought this part but this

is just sort of like a connector adapter. It doesn't really do much except allow the ESB to 6:06 to fit nicely on the breadboard. So I bought these two components to put them together. Right. So we have those two there. And in the remainder of the section we're going to look at the wiring diagrams for the connection of the Arduino to this ESB to 6:06 module and see how that is wired. There are two things that we're going to have to do. We're going after Did the Foom way and this more do of course then right Arduino applications to connect to Wi-Fi the Internet and so forth. So to do that.

HOW TO CONNECT THE ESP8266 TO THE ARDUINO MEGA

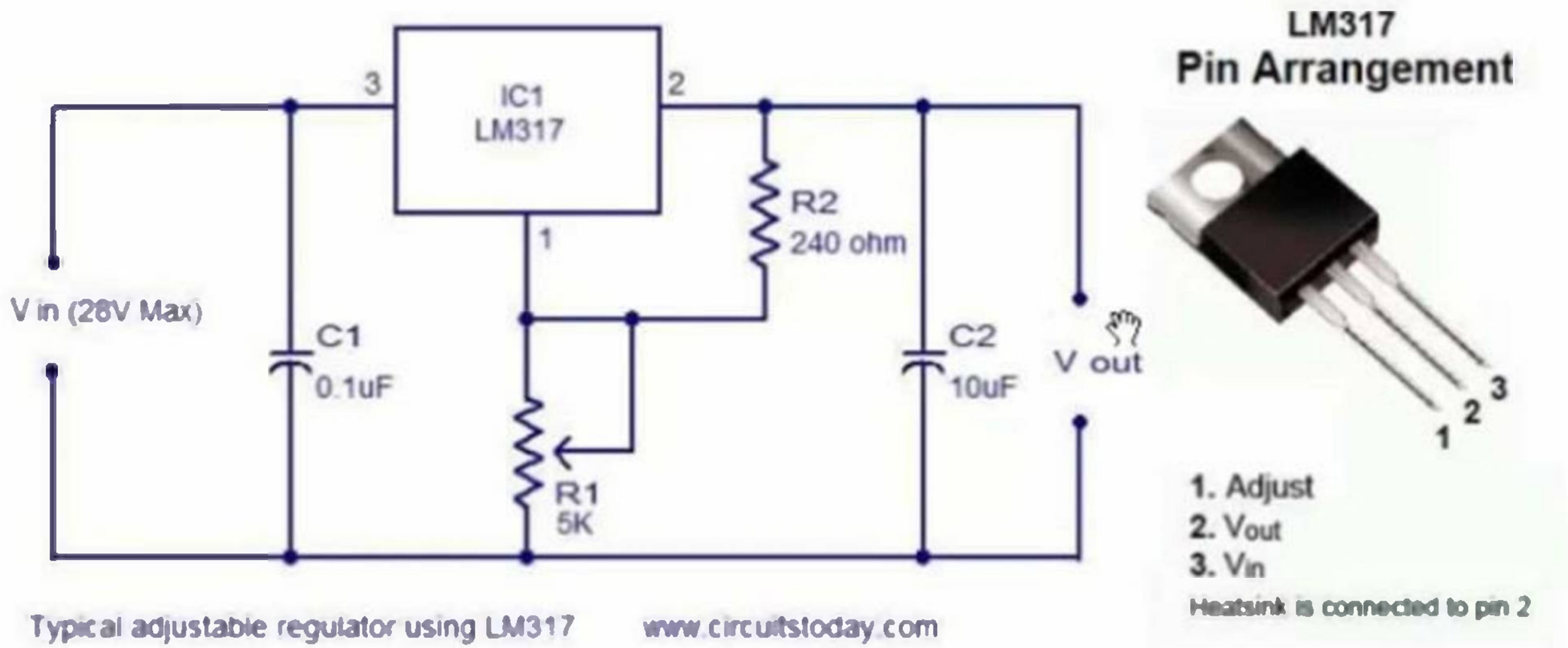
Both of these are attached to this lecture as resources so you can download and keep these diagrams. Print them out if you want. All right. So you're seeing here not many actual pins on the Arduino nega are used. All right, which is a good thing.Less work. So let's just look at the pins on ESB to 6:6. Now our exes receive T-Rexes. Let's make it a little bigger. Shrink down and of it our X is received the X is transmit Vcc is power. GNB is ground Aristedes reset. See each PD is chip select and GPI is general purpose. Eyeopener 0 and G.P.O. 2 is GP is general purpose but I'll put pen to it. So those are the pins right there. And something to note of course this will look at this block here but this block here is a five or 3.3 volt D.C. Circuit.



So 5 volts come out of there and get converted to 3.3 volts and are delivered through that to this board. If you put 5 volts onto this board you could cause it to blow or you could cause it to feel Now there's a 3.3 volt in on the Arduino. However the 3.3 volts here on the Arduino has an output but can supply enough current to the ESB to 6:06 to work properly. OK so you have to take it from the fireballs and break it down on some

projects. They even use a separate power supply to power this Wi-Fi unit here. All right. So let's look at that here. So we have five volts and ground going into a circuit ground coming back out connected to ground here. Then we have 3.3 volts going to Vcc and under normal running operations. All these pins reset chip select GPI 0 and two are pulled high 3.3 valves pulled high GPI zero. We will pull that load temporarily to flush this unit and give it a new food. But I'll show you that in subsequent lectures. The TXI line here is connected to the R x 2 line member we use no using port number 2 on the Arduino to communicate with the ESB to 6:6. Remember that port number one is being used. Our X1 anti-X one for Modbus communication. So we have TXI to our x. No, I know with the MOX FOID Fido for fiber that t x x and x are X but this works a bit differently in that commands are actually sent to this field to configure itself whereas at max 14:5 there's no commands that need to be sent. So it is TXI to our X and then the transmit here goes eventually to the R X but there's a strange sort of voltage divider one key and two key ORMs essentially X on. This will emit 5 volts and this voltage divider breaks it down to about 3.3 or three volts so that it doesn't blow the air here. So this is very important don't think you can bypass this you will blow this guy. So you will cause this to malfunction. So these resistors here of course were just when I got to the local electronics store. So essentially this is the connection that you would use to connect your Arduino Maggo to the ESB to 6:06 the dash 0 1 model. The general purpose is but I'll put pens used essentially for programming. There are other uses but for now we will just use it for the programming. Normal running on programming which I'll show you in subsequent lectures. OK, good. Now let's take a look at this. Five all these three points are the D.C. Circuit which is this. Now I went to my local electronics store and I got all this stuff: the Elham 3 1 7 which is this guy here, the capacitors, resistors and so on. Now they are models you can buy on Amazon that will do this shifting from five or so three volts 2.3 volts. But I just decided to do it this way. And essentially what

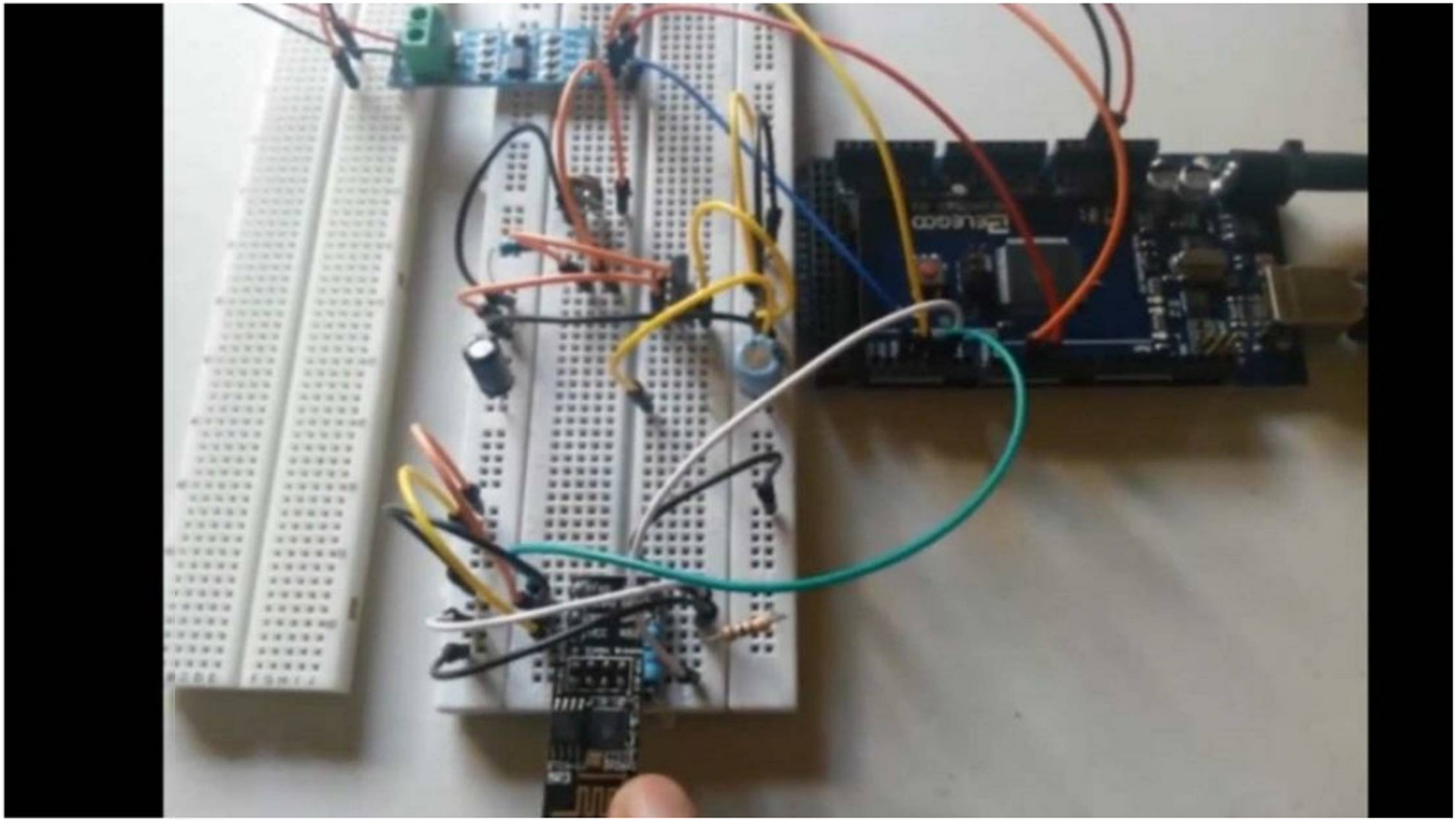
you do is you set this up the Wii in my V and has 5 volts. In this case volts on my wii all it would be the 3.3 volts coming out here. So what it did was I set this up and I put in my 5 volts right here and then put multi-media right here.



And then I just did this potentiometer And here this variable resistor with a screwdriver and this sort of went up and down until I got three point three balls coming out of the output. And that's how I set it up. So it's a pretty simple look at the 1 1 3 1 7 is a very very common voltage regulator. So that essentially is that circuit. So you have all of these you can get these components off the shelf. I mean I don't think you need to order this from Amazon or anything like that.

PHYSICAL CONNECTIONS ON THE WORKBENCH

It's the 6 x y model. So let's have a closer look at it. All right. So down here now you're seeing the break of the adapter that I bought here which is an old breadboard solderless breadboard adapter called the bridge called it up to and then the ESB 2 6 6 module fixed right into it right there. The pin actually goes down into the slots right there. So this allows our feet to take place very very nicely. Now let's look at some of those wires there. So the green and white wires they see right here comes from T-Rex. You look there you'll see it comes from T-Rex to one or X-2 goes across into the receiver. You can see through the jumble here but that goes across the connection there. And all I know is pole wires here go along this track into our circuit here.



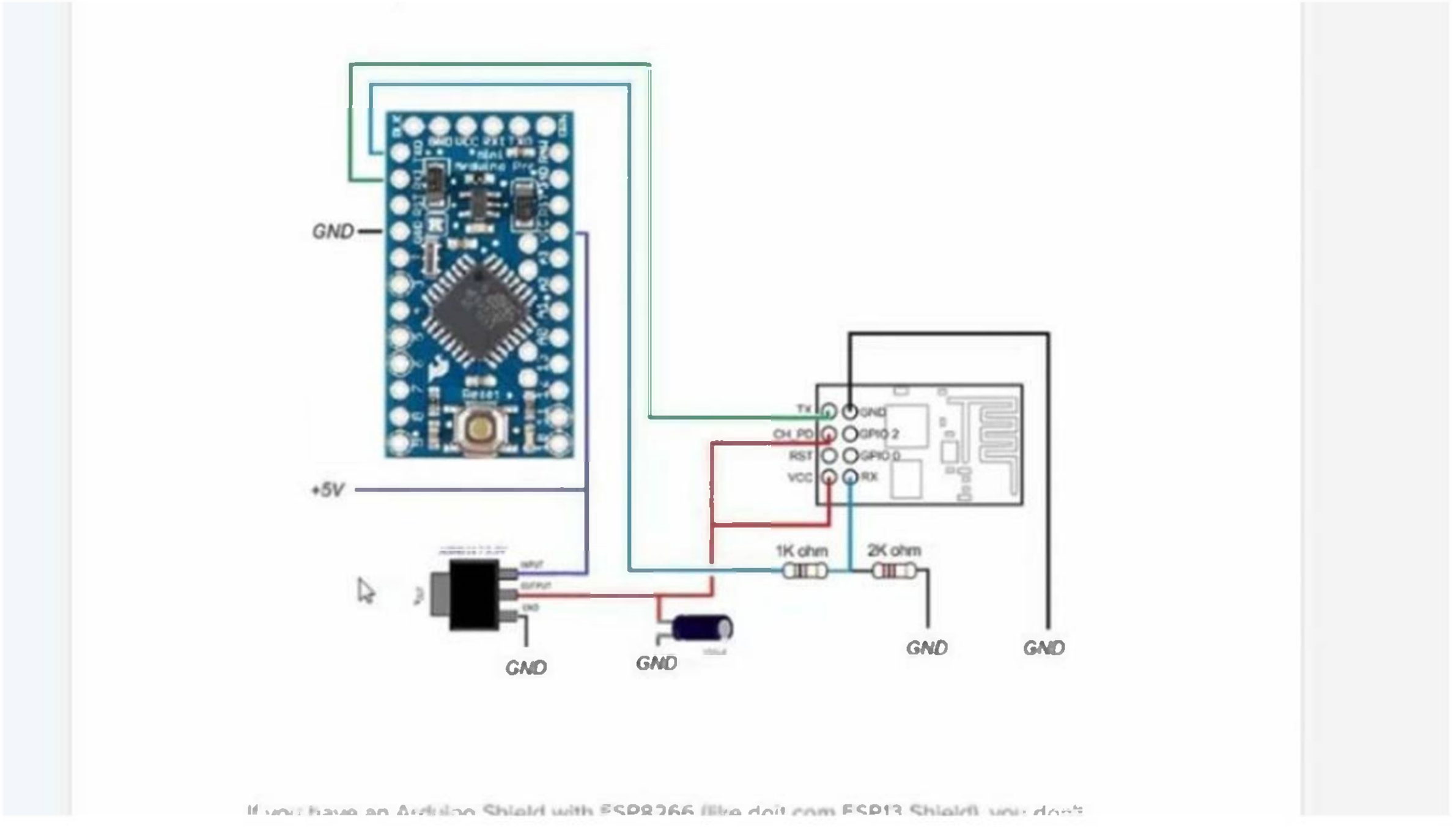
So this is all of five older 3.3 boards that get right there which then powers 3.3 balls onto this unit right here on the VSP 2 6 6. I know you can see all the details but I just want to give you an idea of what it looks like on the desktop. So you know looking at this circuit you will have an understanding of what is what. This is the

Arduino Max FOID 5. So it was a 500 3.3 circuit and this is the ESB to 6:06 with the. So the last bread boarding adapter. OK so stick it right there. And of course let's not forget all our sensors.

UPDATING THE FIRMWARE ON THE ESP8266

In the last section we looked at the hardware aspects of connecting the Arduino Maggo to the ESB to 6:06 in this section. What we have to do is some work on the ESB to 6:6 specifically we have to download firmware of a particular version into this device. Now there are different versions of Foom rather than run on this device and it could actually download from the net and download it to the device itself. The device we needed to download food was 1.1 point and I'll get into the details of that into this device but why. Let me show you. All right so this Web site is called Blink not cuddly and key urLS w w w dot dot C C and essentially Blinkx makes a smartphone application that can run on Android and it can run on Apple phones as well. IPhone and what they do is that their application can easily connect me to connect and be an I.T. application. And what they've done very very nicely is that they have created libraries for vigorous devices like the Arduino Raspberry Pi SparkFun and others. So they have actually created a library for Arduino that works with their smartphone application. But as a little caveat to that so I'm going to get started let me just show you what it's about. We're going to go through all this in detail. What I want to show you here is this. So they provide code for example code that you can run and you can. What I'm going to do is choose Arduino Maggo together with the shield that we're using. And essentially these notes up here stating that they're their code and their libraries will work with certain versions of firmware on the ESB to 6:6 And if you go down here they sort of list all the different versions and it says well it seems to work with all of these

versions. We observe that 80 versions of one point work best. So this is the version of the firmware on the ESB to 6 6 so that's why we need to download and update the ESB to it.

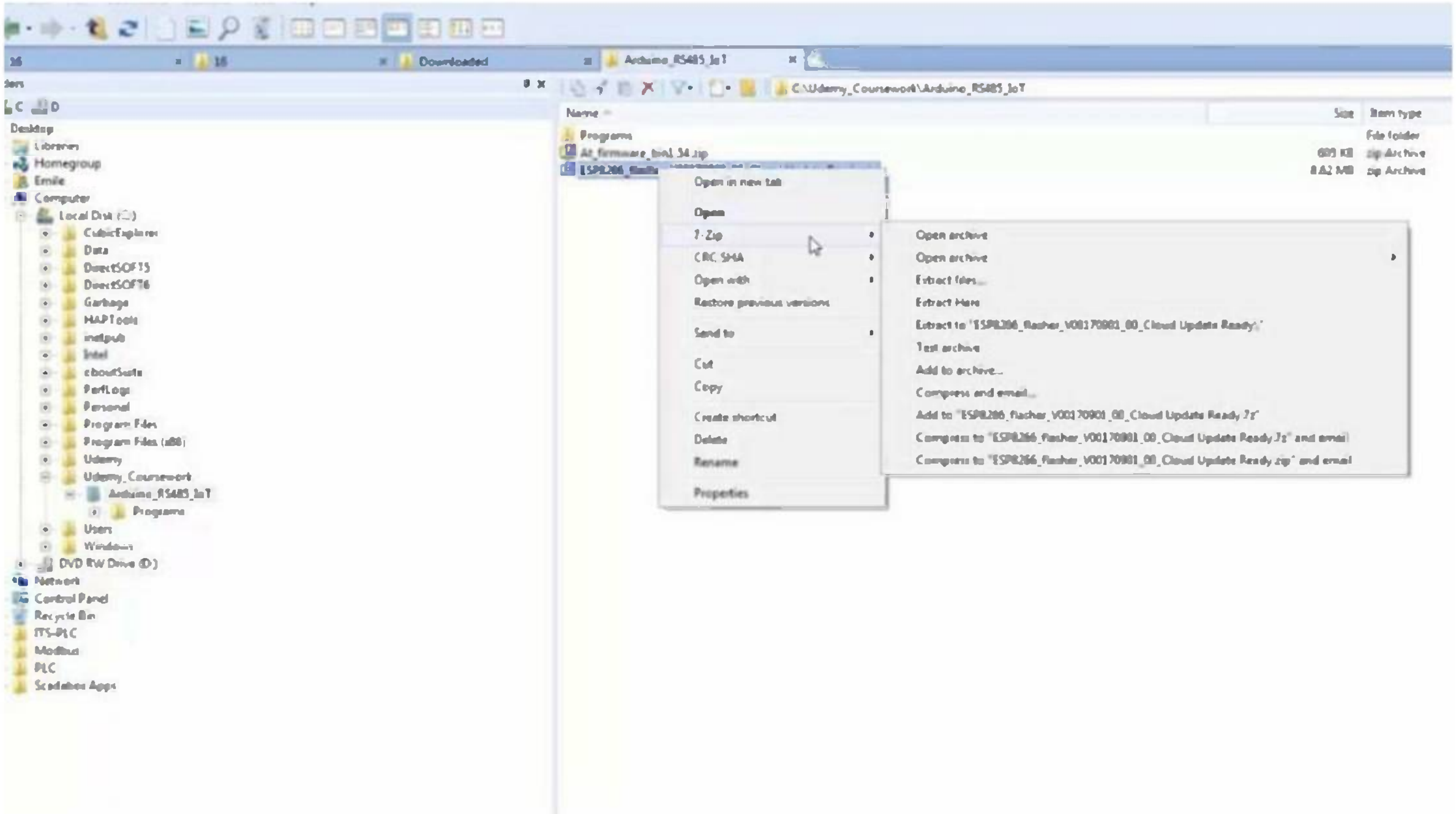


If you have an Arduino Shield with ESP8266 (like doit.com ESP13 Shield), you don't

It has to 66 with version 1.1 from where it can work with Blinkx so that we can see the values of temperature and humidity on our smartphone. All right. So a little bit complicated, a little bit of a nuisance but I'll walk through it and I'll be just fine. So that is the plan for this section. And I just wanted you to know why we're doing it. It's because of the blank application and in subsequent lectures in the section will actually download the firmware and download a special application that you can use to flash the ESB to 6:06 and all of the new firmware. And yeah and that's what we're going to do.

THE FLASH DOWNLOADER AND FIRMWARE FILES

I told you why we need to upload or download new firmware into the ESB to six six. All right. Now, the two things that we need to do that no one will be needed from in the first place. And number two, we need an application that can actually do the flushing because the Arduino I.D. I can't do that. Right. We have to get special software. So associated with this lecture, you will see two links, right? One of those links will take you to the SBA to six flash download downloader software and which is this one here. And the other link will take you to this page right here, which is the actual firmware, even though it says one point five point four, it's really one point one. But here's what I want you to do. So you reach this page and go to download. It's a Google Docs page and where I'm going to see if it is my Udemy coursework, Arduino Rs4 FOID five Iot. Remember, we created this folder previously and I want to keep everything in there safe. OK, and similarly for the food web page, you go here and you click download and it's going to download the file in a similar manner as before. As a file before. And you just follow through. OK, and there's my You Causevic directory and these are the two files I just downloaded. There are zip files. OK, here's my programs, our modules zero one application that we created in the previous section and let's go.

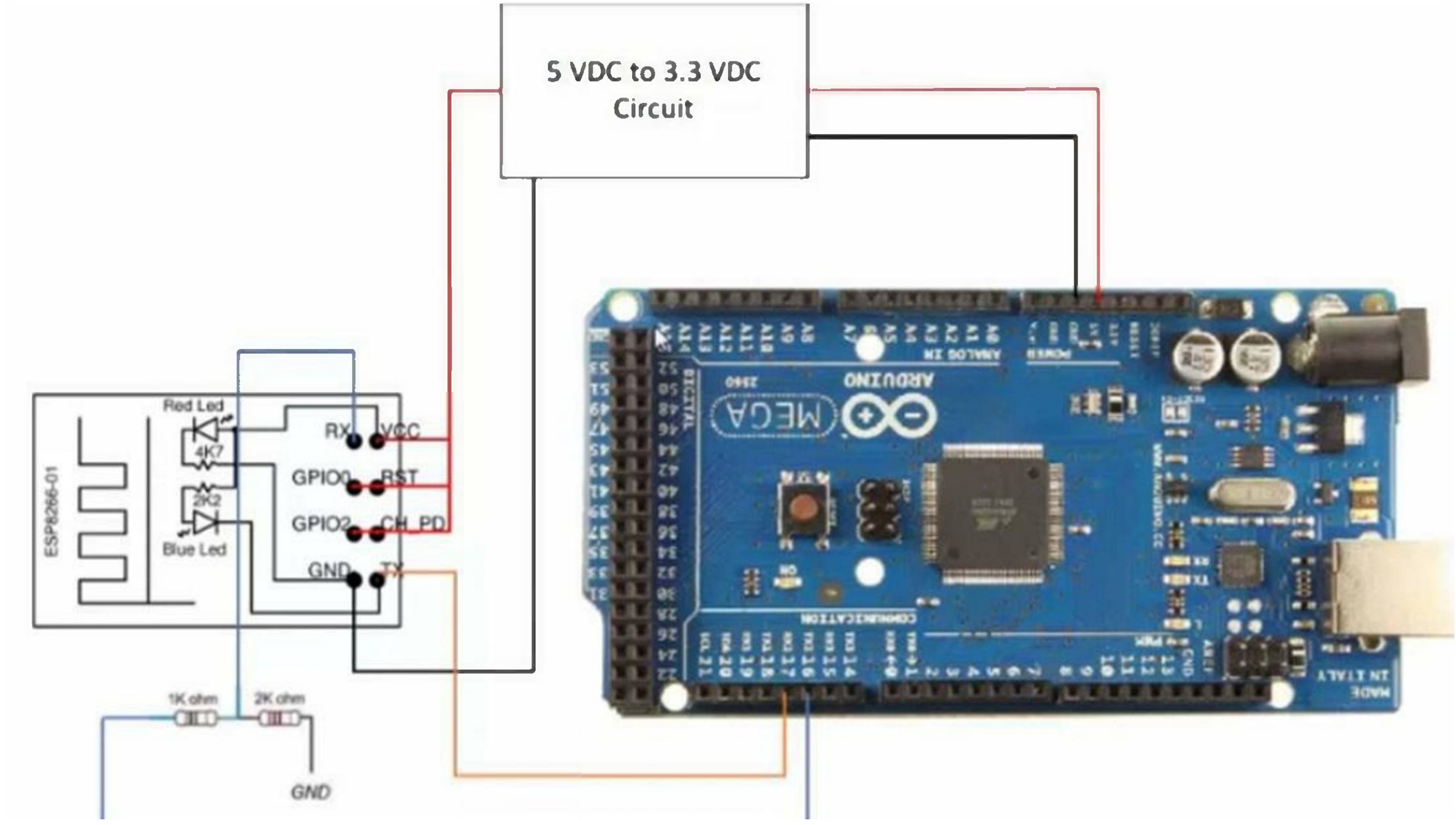


So let me unzip this. Extract here. All right, so these three files are associated with the flash downloader. So what I'm going to do, I am going to create a folder and I'm going to call it Flash. Call it yes. P did two six six flash downloaders. And I'm going to take these three files and put them in there. Good. So that's my flash

download. Now let's go to the firmware and unzip that as well. Extract here. And I have these three files right here. So I am going to create Yes. P ID two six six firmware. That's a folder.

WIFI MODULE USING THE FLASH DOWNLOADER

So what I want you to do first of all is to create a new sketch with just voice set up voice loop just a blank empty program. And what you're going to do is that you're going to download that application into your odd Arduino. So the Arduino essentially has just this program that does nothing. OK so I'm going to click there. Compiling sketch. And there we go. Good to go. All right. So we just downloaded a blank program into the Arduino. Then what are you going to do is you are going to power down your equipment. No power at all on it. Now this was the original circuit that I should have. And what I want you to specifically notice is a GP I was zero is connected to 3.3 volts so it's pulled high transmitter here on the ESB to 6 x is connected to our X to receive here are X and then R X here is connected to ticks. So it's like TXI to our X and our x xx. Now I want you to modify the circuit temporarily while it's powered down to do this.

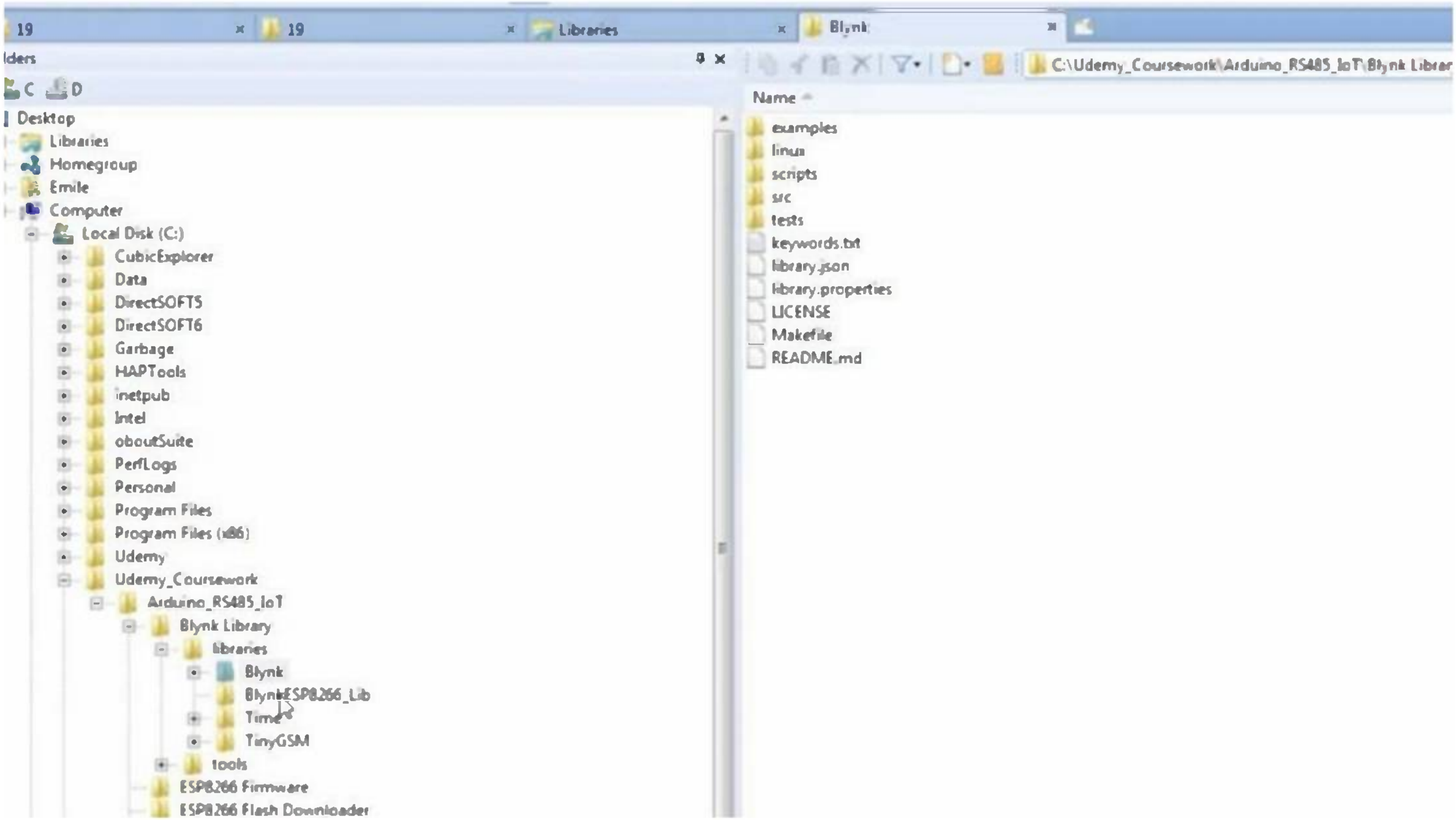


Now this PDA here is attached to this lecture. Don't get them mixed up because I put a note here about connections for downloading firmware. Look at what happens here the GPI 0 0 is connected to high or 3.3 volts. Here it's connected to ground. So what you're going to do is you're going to ground the spin and then you're going to flip around TXI Anarchs and this one TXI is connected to our X here and this TXI is

connected directly to TXI on this. So we're going to T-Rex on our X to our X instead. Once you make those changes what you're going to do is then power the system back up. Once you power the system back up you then go to the actual Flash software. OK so what I'm going to do now is actually make that change power down and power back up. All right. I made that change on the device itself and I'm going to power back up. I know you're not seeing me do it but I just did that. So now my circuit looks like this got powered back up. I'm not going to use the Arduino ID. Instead I'm going to go to my software that I downloaded. Remember that we don't lose the flash download Plus the firmware. And I'm going to double click on Flash. This is my very very simple program. Now the Says com one right here. But the Arduino we're programming the ESB to 6:06 through the Arduino. So we're using the Arduino port which is comm 5. There are no speed settings. Right click on the bin and then I'm going to go to you to make coursework Arduino firmware and I'm going to fool my favs to choose this one with the Eat Em and of it to the two m's I'm going to choose that. OK now I'm going to choose. Download and it says field to connect Don't worry. It just needs a second try download. There you go erasing flash and it's writing the new Flash to it. Now this takes some time so I'm going to pause and then come back so that I don't waste a bunch of time on the project. All right. So it's done. You see 99 percent down here it says leaving field to leave flash mode that's fine that just pops up as a quick with this flash downloader. All right. So once that is done I'm going to close this. I'm going to power down again and then I'm going to go back from this circuit diagram to our original circuit diagram. I'm going to make that change now. All right. And then I'm going to power back up. So I've just powered back up and now I'm back in my original circuit.

DOWNLOADING AND INSTALLING THE BLYNK LIBRARIES

And we're going to go to this page here with all these steps and click on this button that says download Blinkx library and it has version zero point 5.2. All right. And it doesn't even have a link on how to install the library but I'll do that as well. It was a nice project. All right. So let's download this library here Blinkx release and we're going to see it in our Arduino or is it five. I coyote directory or folder. Right Sue. I'm going to give you some coursework and this is the blink release right here. So I'm going to extract all of the files. And those are the files there right. Soooo what I'm going to do. I'm going to just put it in this folder here called for free. So take libraries and tools and cut and paste it there. Good. Nice and neat.All right. So this has to do with some directories or some folders tools and also libraries. So what you are seeing is needed here. ESB to 6:6.

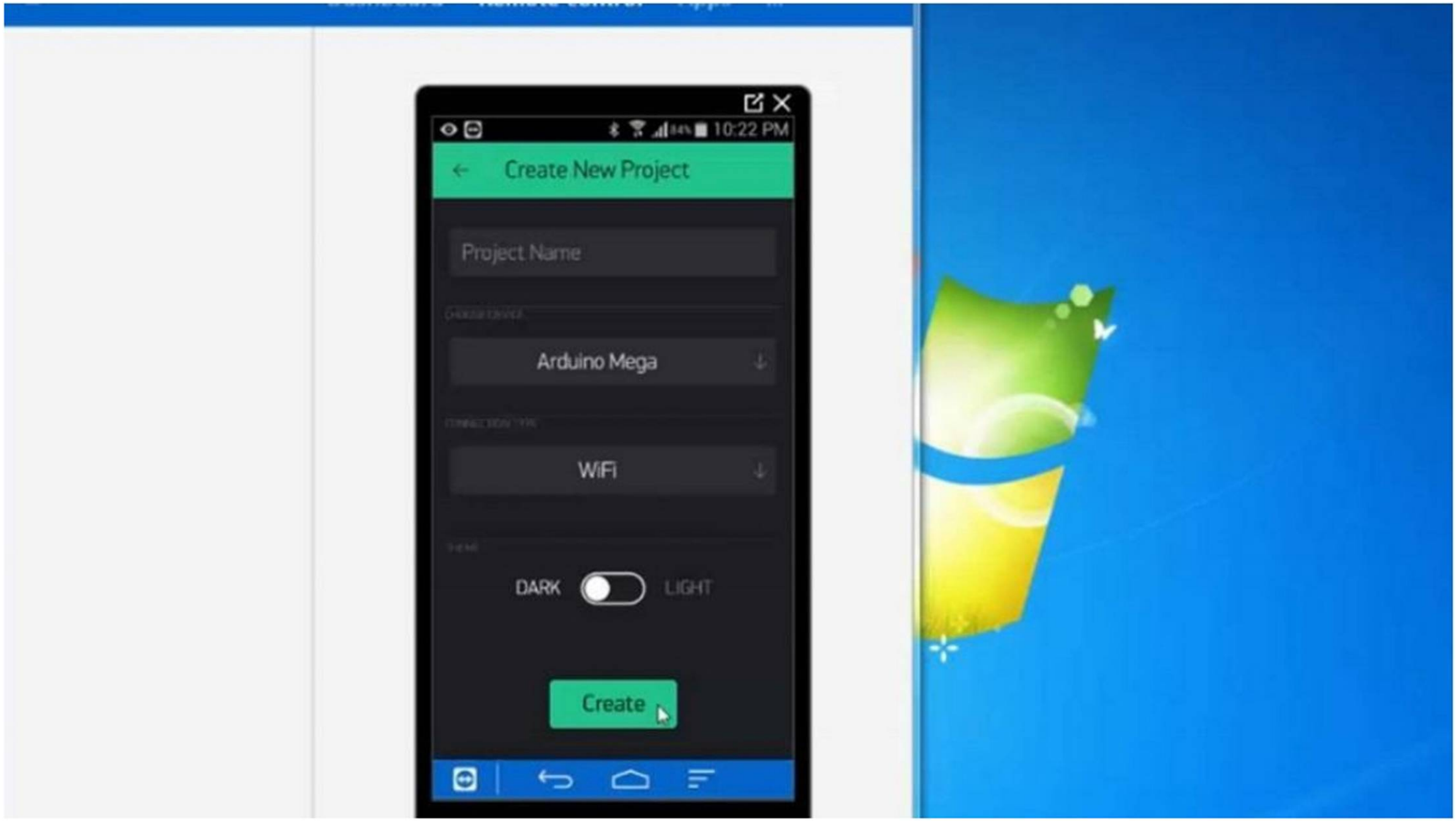


So what I want you to do is know that I've installed the libraries already so I'll just show you these two here. Blank and Blank. ESB to 6:6 underscore LRB and you can also copy all four. All right. And then you want to go to the desktop library documents Arduino and under libraries you're going to install you're going to drop all those files. So you're taking these four folders here and dropping them under here right now. Libraries

document Arduino libraries dropping them right underneath there. And once you do that the Arduino I.D. will automatically pick it up. So what it looks like. If we go to sketch, include the library. And if we go all the way down here you will see a blank. Yes. It to 6:6 underscore LRB once you see these two here you're good to go.

RECEIVE DATA FROM THE ARDUINO

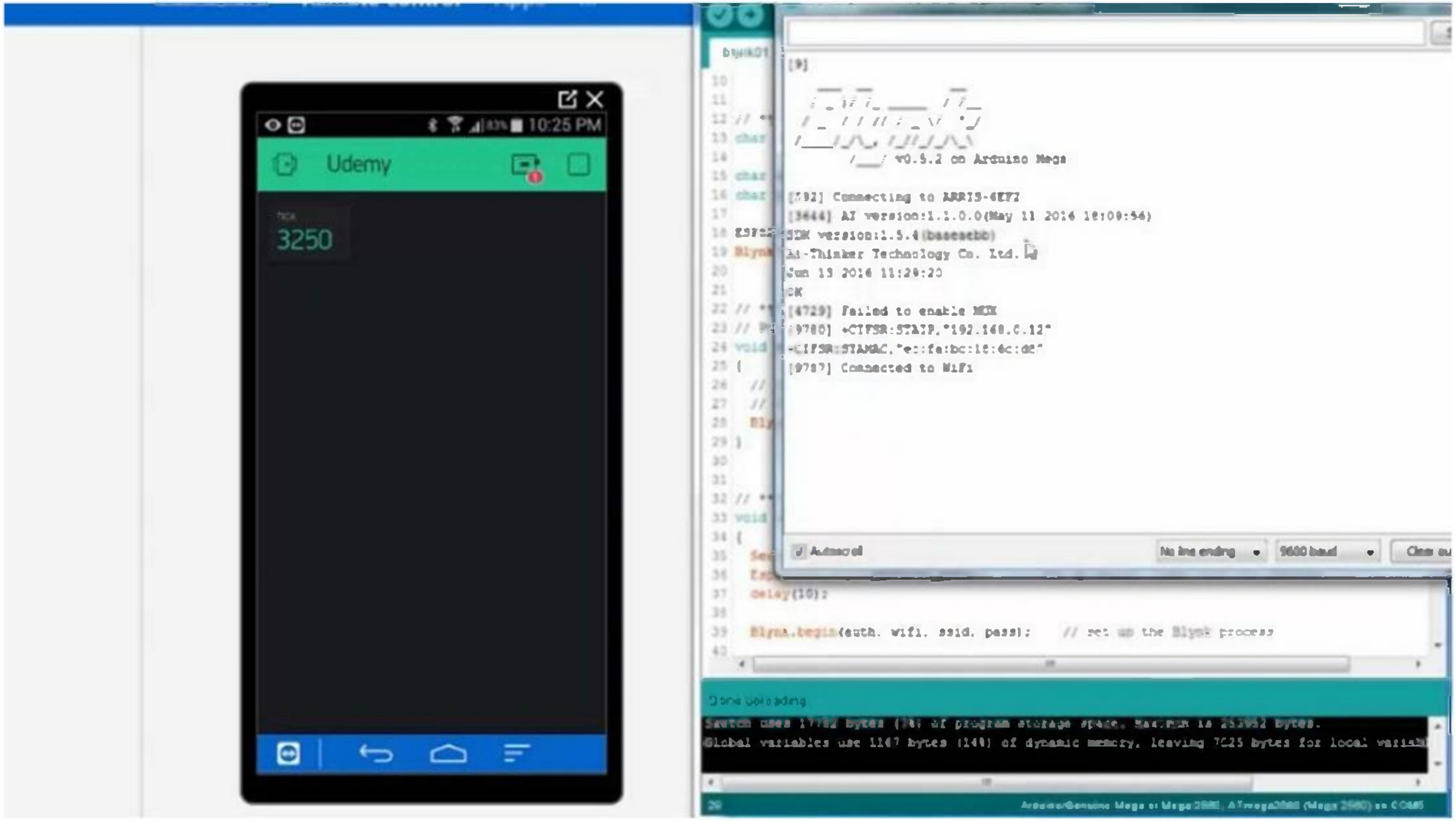
So you're saying I'm doing sort of like a remote control. So what is happening on my phone you can actually see on this particular screen. So great. Now you would actually go to the Play Store like I'm on Android. Right. So you would go to the police store right here and look for B-L white and key the blank up and install it right now. I'm not going to go through installing an app for you because you're supposed to know how to install it by now. So this is you blowing up right here. And when you're installing the blink it'll ask you for an e-mail address and a password and that will be your e-mail address where you will receive emails or codes etc.. So if I click on blink right what you will see happen is it will get it you will go to the screen and what you're most interested in is new project right now I've created a project already I'm going to swipe and go call you to me. That's the one I'm going to be using now when you create a new project if I go create a new project. What you do is that you select the hardware that you're using and you just kind of scroll down until one is highlighted. So we're using an Arduino megger. So we do that and then click And then how is connecting World-Wide fi. Right. So we're connecting via wife II. And once you click Create here this create button it will send an author talk to your email address which you would then use in the Arduino code.



So do we know code that's how you get the or to All right but I'm not going to create a new project here because I already have one so I have a project called you to me. Now how Blinkx works is Blinkx works via a project widget type basis right. So I have no widgets in this project so I'm going to add a widget here. Whoops. It is a great new project. So what I'm going to do. All right. So I'm going to click on this to high-

light it. So I'm now on the you to me project and I'm going to add a widget. And there are whole sets with different widgets here which is just like a nice graphic for displaying something. So I'm going to click on the volume display all right and then I'm going to click here. I'm actually touching this. I can touch here or use it on the screen itself and I am going to call this value here. Actually, first I'm going to assign it. I'm going to call it a virtual pin. It's not a physical thing called v 5. Now remember Wii 5 is what we were writing to in the Arduino code some choosing the five here click or key and then sort of choose V 5 and I want to update every let's see five seconds or let's say one second.

And then what I want to type in here is just Mellis standing for milliseconds right. Done. Actually I'm going to call this tick because it's a tick tick. All right so I have all my I have selected the five here I've named my widget V five is right here. So it's everyone's second and then I can go back here now and then I'm going to run my application and look what is happening here. The Arduino is actually sending this task over and over again to it. So that kid's been coding for a while so it's been going up. Now I can download a brand new opera so I can not do a brand you download which should reset that. So I'm going to do a download into the Arduino so it should be restarting. To my cereal monitor. It's reconnecting. That it stopped at 3 2 5 0 connecting to my wife. Right say new devices connected. Look , it started over again. 22 23 24. So you're seeing what is happening on the phone so this is actually happening on my phone screen. I'm just putting it on the computer screen for you to see. So that is our very first application and we've just seen how it works.



So we're just a simple timer tick. Now in the next section what we have to do is go on and take the Modbus code and move it with our application so that we're reading the values of temperature and humidity and putting it up here on the screen. So once again you create a project then you create widgets and they are

really great tutorials on the blink not CC website to follow to get the hang of projects and widgets. OK so I'm going to click here and I'm going to stop the transfer right now. All right. So that's the end of this section.

THE MODBUS CODE IS MERGED WITH THE BLYNK TEST CODE TO CREATE A NEW SKETCH

This one right here is Modbus 0 1 which we wrote just to control the Modbus communication between the Arduino and the sensor. And then this one was the blink test application blink 0 1 that we rode in the previous untested in the previous section that sent data up to the app on the phone. Now we're going to combine the both because what we want to happen is that we want the model data to be read then that data sent to the app on the phone. So I'm going to go file Cevallos and call this Blinkx Modbus 0 1 because it's Blinkx on us. All right. And of course I will attach the code to file a blank Modbus 01 as a zip file to either this lecture or the next one. All right. So let's start with defines. So this is all Modbus copy paste there good includes I have to include the bus master globose I have to add the more bus master Nu-Wood right as a global their support functions. I will need pre transmission and post transmission of course. OK then the set up standard set up functionless school here. All right let's put a comment here and let's set it up. Blink blink right. And then this is set up Modbus all key. So we do all of this stuff here. Copy and paste.

blynkmodbus01 | Arduino 1.8.4

File Edit Sketch Tools Help

blynkmodbus01 §

```
37 {
38   digitalWrite(MAX485_RE_NEG, 1);
39   digitalWrite(MAX485_DE, 1);
40 }
41
42 void postTransmission()
43 {
44   digitalWrite(MAX485_RE_NEG, 0);
45   digitalWrite(MAX485_DE, 0);
46 }
47
48 // ****** STANDARD ARDUINO SETUP FUNCTION ******
49 void setup()
50 {
51
52   // set up mo
53
54   // set up Blynk
55   Serial.begin(9600);   // set up serial comms with serial monitor
56   EspSerial.begin(ESP8266_BAUD);    // set up serial comms with the ESP8266
57   delay(10);
58
59   Blynk.begin(auth, wifi, ssid, pass);    // set up the Blynk process
60
61   timer.setInterval(1000L, myTimerEvent);   // set up timer event to call myTimerEvent function
62 }
63
64
65 // ****** STANDARD ARDUINO LOOP FUNCTION ******
66 void loop()
67 {
68   // there will only be these 2 lines in the loop function, put everything else in myTimerEvent
```

Done Saving

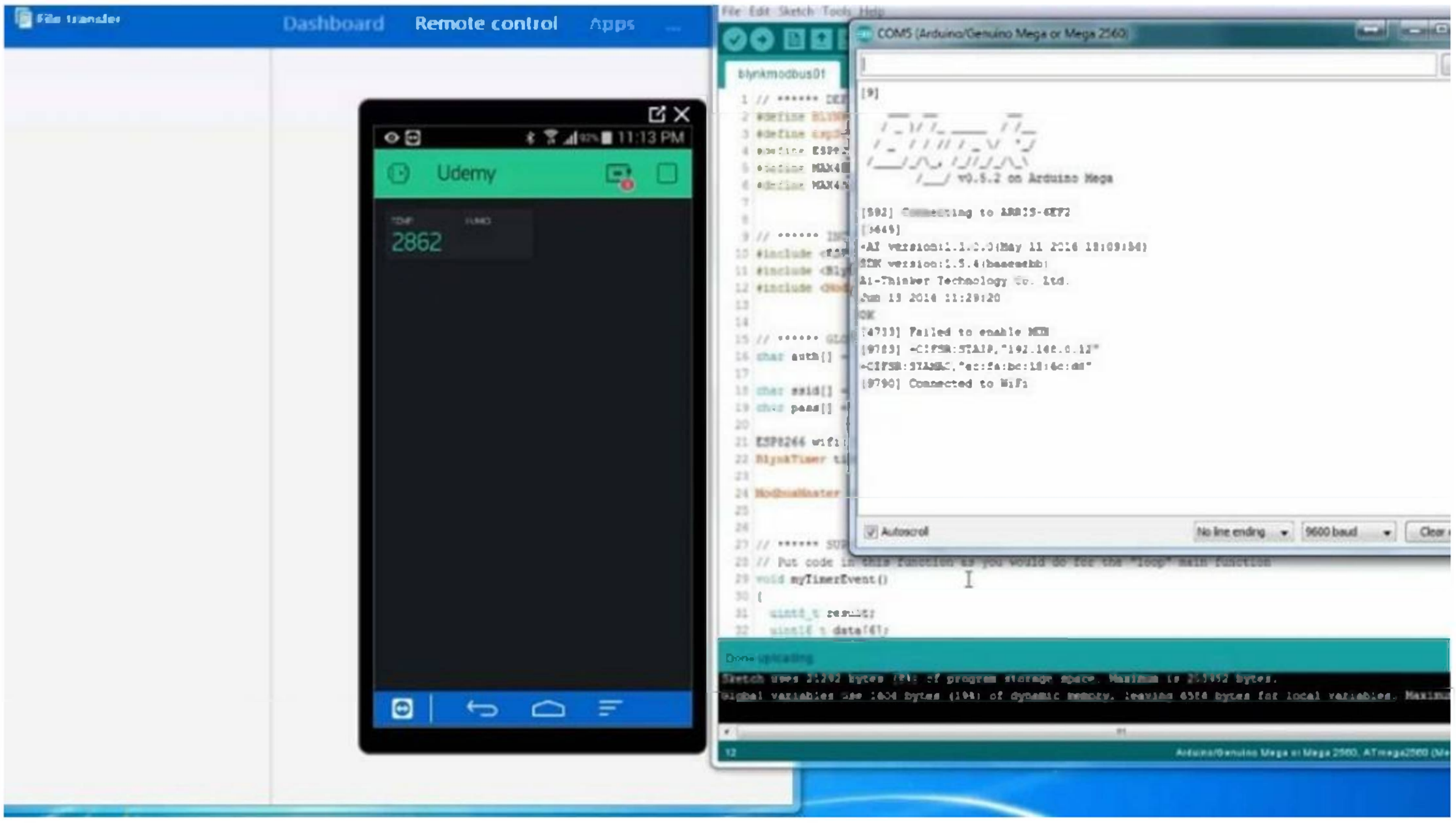
52 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega


modbus01
{
digitalWrite(MAX485_RE_NEG, 0);
digitalWrite(MAX485_DE, 0);
}
// ****** STANDARD ARDUINO SETUP FUNCTION ******
void setup() {
// make pins 2 and 3 output pins for Max485 flow cont
pinMode(MAX485_RE_NEG, OUTPUT);
pinMode(MAX485_DE, OUTPUT);
// Init in receive mode
digitalWrite(MAX485_RE_NEG, 0);
digitalWrite(MAX485_DE, 0);
Serial.begin(9600); // TX0/RX0 serial monitor
Serial1.begin(19200); // TX1/RX1 Modbus comms
// Modbus slave ID = 254
node.begin(254, Serial1);
// Callbacks allow us to configure the RS485 transce
node.preTransmission(preTransmission);
node.postTransmission(postTransmission);
}
// ****** STANDARD ARDUINO LOOP FUNCTION ******
void loop() {

Right now there's a bit of overlap right here. You see we have serial begin ninety hundred and then the serial begin 96 100 down there. So we don't need this line right here. We can actually comment that out And the serial one that began all of this just as before. And that looks fine right there. That looks fine right there. All right so let's move on. Now the loop function stays the same. So that means we what we have to do is put our Modbus code that actually does the reads in the my timer event. All right. So this is what I'm going to do. I'm just going to take all these comments here to need that key so all of my bus stuff is here. All of this right here. Copy and paste. Write Now we have to make some changes. All right so we just copy this here now. No I don't need any delay function because this is in a timer. All right. I don't any delay at all. So it's just going to read temperature read humidity but I don't need to printed out here and I don't need to print that out there. So what I do need to put is Blinkx right. And we're going to go with the five. So when we get the temperature we're writing it to be five in the blink app and we'll do the same here. Right. And we were right. Humility to V-6 care and we will remove this line right here. And that essentially is the mood of Modbus and the blink application the blink test application. So this is going to be set up what's going to happen let's put it. I think I want to put the time let's see every. Hmm let's see five seconds because I mean how quickly is this temperature and humidity going to be changing so this time will be a 5S every It will execute every five seconds it will read the Modbus data and send that temperature to v 5 in the blink OP and V-6 humidity to V-6 and a blink up using of course this authorization took on which links it to the project. OK let's compile and see if it compiles right. It's going. Yep yep it compiles no errors at all. All right, so we have our application in the next and the next lecture. What we're going to do is to make some modifications to the project on the phone.

MODIFYING THE BLYNK APP PROJECT

I'm going to call you made it to the it's going to call it humid and make it short and it is going to be assigned to virtual Wii 6 because that is what we assign it to in the Arduino code. Key. And every one second go again boom and run this key. So now while I have this there I am going to download our Blinkx Modbus 0 1 application and see what happens. Let's start a war. SEE real monitor. So there we go. Connecting. Word without error happens every now and then and I feel terrible marks connected to wifi. Waiting for the ping there's a ping.

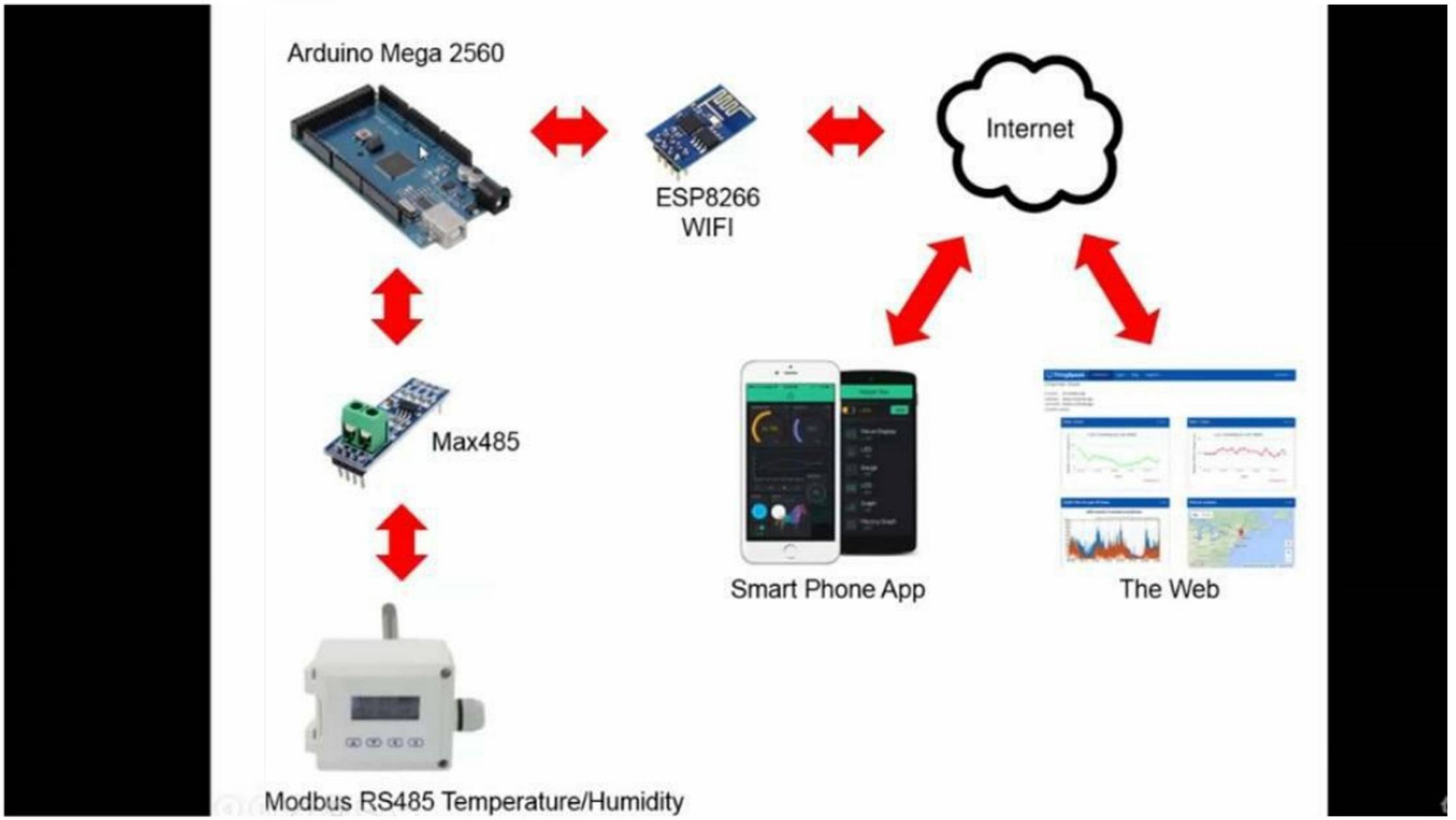


Let's see if any updates are taking place. Wow. There we go. Eighty six point seven hundred sixty one. Now what I'm going to do is bring up the actual ring of the actual display image of the display. All right so now we have the image of the display here. And I can use this. You have the image of the display together with what you're seeing on the forward. And we've done it right. So what you're seeing on the floor there is mimicking what you have there. So we have created our first iOS application using Arduino. We can view

this data on our forneon the up. It's going through the internet to this up coming from this Modbus device. We have succeeded.

REVIEWING OUR BLYNK IOT PROJECT

A lot of hardware, a lot of circuit diagrams, a lot of everything to you pretty quickly but has to get you up and running as quickly as possible to create this IMT project, our first IO t project. OK so going back now to our introduction where we saw this particular image right I took you through the plan. So what I want to do is just sort of solidify exactly what we just did. We established Modbus communication here and wrote the program just to test that. Then we establish communication with the Internet and of course with a blank. Now I want to speak a little more about the blank. Now remember what we did. We downloaded essentially the blank library into the Arduino device and did not blink. The library contained within it a lot of the intelligence that drove the Arduino because we didn't do much configuration for blink so that Blinkx library what it has embedded in it is the IP address of the Blink Suvari so that when it ran it knew how to send or where to send data it send it through the Wifi Network to the Internet to the Billings server IP addresses specifically.



And once it was in the blink server it was transferred to the Blink app and what bridged everything was connected everything was the auth took her authorization to remember that when you create a project you get an authorization token for it which is then used in the Arduino application. And that's the key. That's the unique identifier. Otherwise they would not know where to send the data and blinkSilvo would not

know which app on which project to send it to. Who's up. Because I have an account here with a username and password so it has no user. So the auth talk essentially says send it to this username and password and send it to this project. And that is the linkage so I just wanted you to understand that for that to be concrete in your mind. The Arduino live sorry the blank library contained all the intelligence to go with the blanks with entry up the authorization to on tool that where to go on what on what project to put it on. So that's where that authorization took place and it's so very important. So that essentially is what we did up to now. So we did this part here. We had our readings at the end of this section. You saw that we had lower readings going from all Modbus temperature humidity sensors all the way here through the blanks on to our right. And I had the image of my phone screen up on the screen and so on. So the next thing that we have to go on now is this part where we're going to put the same data here on the web instead and we're going to use something called Things speak.

AN OVERVIEW OF HOW THE THINGSPEAK IOT SYSTEMS WORKS

And in this section what we're going to be doing is actually taking our Modbus data and putting it on the web and we're going to use a service called Things speak which is an I.T. service. So we're done with our smartphone app. The code that we wrote for the blank application will not work for this thing. But there are some similarities. So what we're going to do is that both code that we have will work as well and that stays on change. But we're going to rewrite the code that activates the Wi-Fi transceiver module and send it instead to the things that speak Suvla where it will be displayed on the screen. On a word on the web interface. Sue, let me just give you a little insight into how things speak.

So with respect to blink we had it we had a blank library in the actual Arduino device and it was heavily tied to the app knowing things speak is a little looser. Things speak actually don't actually doesn't care about having an Arduino library or this library or that library. Anything can be sent to speak. And it's because it accepts each DDP request So this is a get each DTP request right. So it has a certain format you see, get his

sites and get on that census each TTP as API things P-Dog calls him that is sort of an update API key. Now the API key is similar to the authorization token in the authorization to Canara we saw in the blowing up right for the blink application. So this is called an API key and it's as I said similar to the authorization to a gun and then you're seeing field zero feel one equals zero. So this going to field one equals zero field to field three. So with a blink up if you recall I mean just add this here with a blink. We had to blink here. We had a project on inside that project we had many widgets. Right so the project was sort of the container and every widget represented a single value that we were sending up. So we had a widget for temperature and we had a widget for humidity with things to say.

GET https://api.thingspeak.com/update?api_key=FV4O8X30HRQK54QR&field1=0

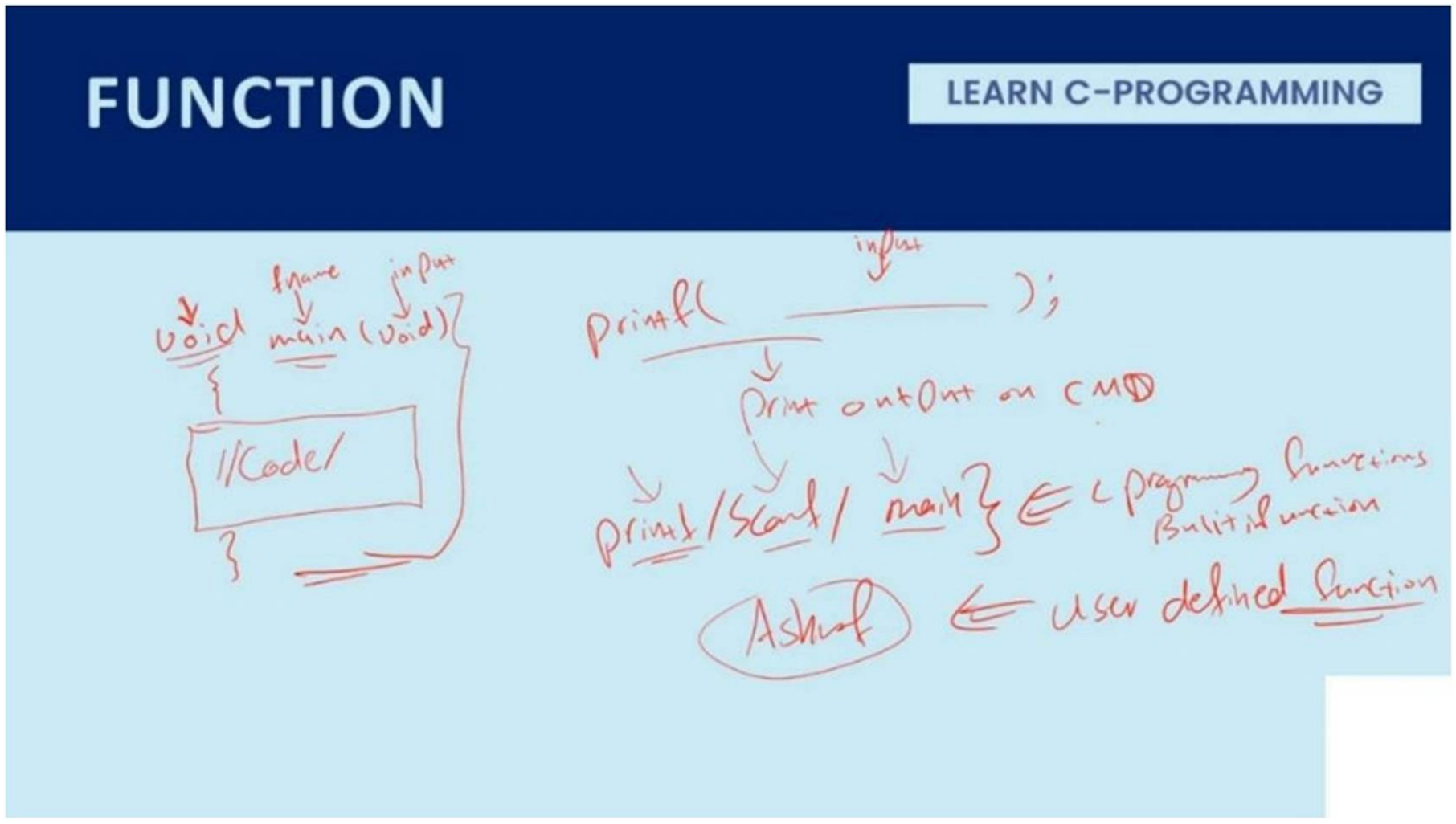
Thingspeak

It's a similar paradigm but the names are different. You create a channel of channels like a project and then in the channel there are fields and every field is of value. So we're going to be dealing with channels and fields in things that speak and I'll show you this when I show you how to create an account and so on and things speak All right. So that's the paradigm we're working with. So things speak and accept these sorts of statements. I get each day each DTP and so on. And those are essentially things that sort of are with the API key which is why the author talks on which is the link is your unique identifier for a particular channel that it updates it.

FUNCTIONS IN C

Hollande will come to this in unison, which is one of the most important lessons in this course. And in this lesson we'll talk about functions. The function is a block of code which only runs when it's called. It is used to perform certain actions and they are important for using code like define the code once and use it many times to use the function you need to call it inside your code. When calling a function, you may send some data or some input and it could return back and out for you. The most common function is the main function, which is the entry point that we have been writing since the beginning of this course. Now our main function is basically this. We invited like that void man to fantasies and to races. Now. This is the function name. And Men is reserved for that main function, which is the entry point between the two parentheses. We add the input. Now, if we don't want any input, we have the word void. Now this is the returning data type. If this function does not return anything, then we call it void. We add void and the returned as a type. And the code that you want to execute or reuse you use. You like it inside here. This is the code. Now, this is one of the functions that we have been using a lot since the beginning of our program to execute the function you will need to call it. The function provides you with the advantage that it is defined one time and can be executed many times. So it takes the same size in the memory. What if? How many times will it be called? When you call a function, you can send to it some inputs, as we already mentioned. And the other example for a function that we have been using since the beginning of our coding process is the print func-

tion. Now playing their function is basically a function that takes input between, that's all parentheses and the execution of this function all print output. On CMT. Which is the command line window. Now the print function takes a string as an input argument and prints it on the screen. Any project composed of one function or more. Which are basically the main print on scan i. Now, Brent, if. S.F.. Men. All of these are called. C programming. Functions. Or built and functions. Because we do not create these functions, we only use them. We call them by their name. But if you created a function, let's say that you call the function Ashraf. Then this is called user defined. Function. So anything that you create is called a defined function, anything that you use, and that's already created by C compiler for C programming language. It's called a built in function or a C programming function.



Now, to make things more clear, we need to talk about what you should do if you want to create your own function? Well, if you want to create your own function, you need to understand that a function consists of three parts and you need to memorize these parts. The first part is the prototype. And this prototype declares the function. So it tells the compiler that there is a function with this name and the specifications. Usually the prototype includes the return time, as we already mentioned. The functioning. Two parentheses in there and the input between the two parentheses. Then the line ends with a semicolon. This is called our prototype. Now this is the first part. The second part is called the implementation, and it's basically the place where you write the function code. It defines the function behavior and it has the very same shape as the prototype, as you can see, except for the same column here. We don't have semicolons. And instead we have that look at the places inside them. We ride the function called. And you can see that here. Here we have the first place. Here we have the second. And between them we use what derived our function statement or code. Now this is the second part, which is basically writing down the function itself with the code that we need to execute. Now the third mark is the function called. The function call includes the functioning. And any impulse between that to calibrate that two parentheses and return is something we need to store the data that this function returns inside of it. Now, if it does not return anything, we don't need to store anything. Let's do a quick revision. A function that you create consists of three things that you need to write down. The first thing is the prototype, which is basically a way to declare the function. It's a line that ends with a semicolon.

FUNCTION SYNTAX

LEARN C-PROGRAMMING

The function is composed of three parts

1. The Prototype
   - declares the function (banner)

     return_type Function_Name (Input_Type Input_Name,.....);
2. The Function Body/Implementation
   - Defines the function behavior

```
return_type Function_Name (Input_Type Input_Name , .....)
{
Function statements
}
```

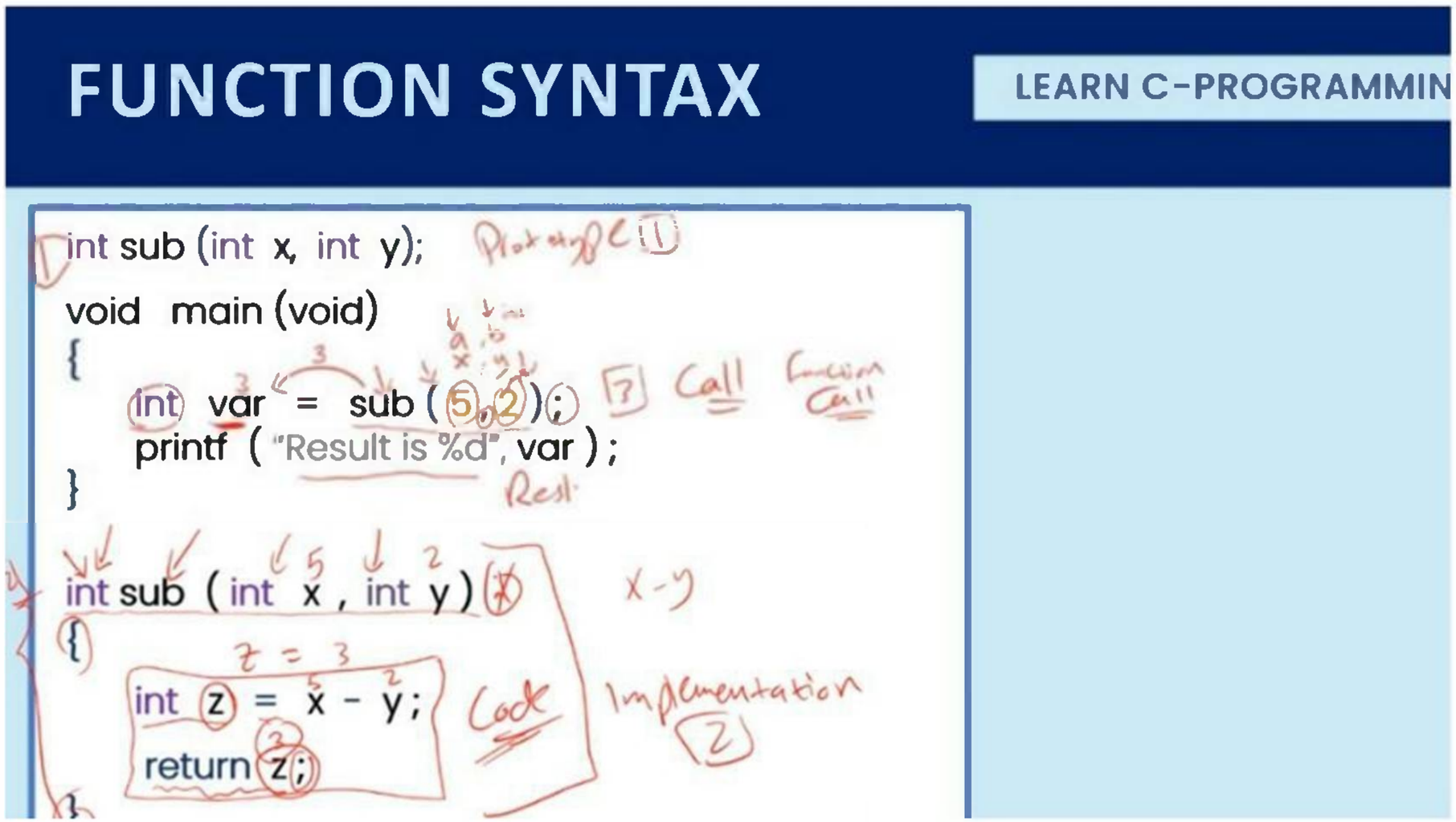
3. Function Call
   - Executes the function

     Output = Function_Name(Inputs);

And it includes the return type, if any, the function name that you define and two parentheses with an input. Between them. If there is any. Now there is no. And what you need to avoid and will come out of this and I suppose would lessen the meaning of avoid that. Although now after writing the prototype we need to find that implementation which is the function body. We use the same line as in the prototype, but then

instead of ending the line with a semicolon, we end it with two curly braces. This one. This one. Between the two calibrations. We arrived at the function court. The first step is the function call. Inside our main. We can call our function with its name. So function name and we send two inputs. Now I know that things might not be clear. So let's take an example to make things even more clear. Now this is a function that we defined and we are going to execute in a few seconds. The first part here. Is called prototype. This is a line. This is the return type. This functional router and integer. This. The sub word is the functioning. Which is the second part of the prototype, as we already mentioned. Now we have two franchisees, as you can see here and here, and these two parentheses will end with a semicolon. Now, this function will take two inputs. Now it can take more than two and puts the binding on our program and our codes. But you need to make sure that you have a semicolon between each of these inputs and you have the type and the name for each of these segments. So we have the data type for the first input integer and the name is X Acoma. Then the second input is Integer and its name is Y. The second thing, this is the first thing that we have. It's called the type, as we've already mentioned. Now, the second thing is the implementation. This is number two. And as we already mentioned, the first one is the same. It's a common thing between the prototype and the implementation, except for the semicolon. We have to remove it and after that we add two curly braces and between them we will add our code. Now, this is the code that you can't reuse. This is the function name. This is the term that I. These are the two inputs. This function subtract X minus y. So we'll create a new variable inside of the function. It will equal x minus y, so the result will be inside z. And since this function returns an integer, we need to return the mission value. And we end the line with a semicolon. So this is our code. This block is called. Implementation. Which is the second. Saying that you must do this is the first thing. Now, the third thing is the function call. Now, as you can see, the function call is simply calling the function with its name. Here we have the function. Then we have the two parentheses on the second col-

umn. But instead of adding an X and Y, we are adding two values to be passed to the value to that function. Sorry. And you need to add the symbol. Now, can you pass variables? Yes. You can replace this with X and Y or A and B or whatever you want, but you need to make sure that there are only two inputs. Since the function takes only two inputs and you need to make sure that the variables are integers and have values. Now, once you send the values to this function. You can't call it that, which is the third thing. We call this a call or function. Call function. Call. Now, the function call is basically calling. So it will give five and two instead of X and one and five will be here. Two will be here. Five minus two equals three. So it will equal three and it will return three. Now, when you call the function, it will return an integer. So we need to create an integer value.



A variable, sorry, we called it var to store the value that will be returned by the function and in this case it will be three. So var one equals three. And when we present this, it will present results. There Are three. This will be about what's in our console window. And here we call the function. As you can see in this line, inside the main function, you can call it again and again and again without having to drive the whole function. Each time you call it, you just write it once and you can call it, let's say ten times, 20 times, whatever you want. Now, to make things even more clear, let's implement this exercise in our notepad. Plus. Plus. Now to do this, we will start with the usual thing. We will include. The SDI or the. Now, the first thing that we need to do is avoid man. Void, which is our main function. Now. What we will do next is simply we will add the function prototype. Usually the prototype is a little bit above the main. So here. Function. Photo tie. So the function will return an integer and its name would be sub. It will take two inputs and X and Y and we will end it with a semicolon. Now the second step. This is the first step. The second step is that I think the function itself is underneath the man, outside the man and underneath, as you can see or below it, we are outside the man. Here. Life three. Sorry. Implementation. Now what you need to do is simply copy the whole line here from the prototype pasted here to remove the semicolon, as we already mentioned, and to calibrate this. Now we need to make sure that this is identical to this, except for the semicolon. You'll have to remove it. Now, this function will subtract X minus Y. A simple C code. And whenever a function returns something. It must contain the word return. It will return this value. Now let's go to the last step. The call. So let's say we want to call this function. Give it five and two. Now, if we did this and we executed the code, nothing would happen. This function will subtract five minus three. It will be minus two. Sorry to equal three, and the three values will be floating in the memory. So in order to control or to save the result from this function, we need to store it inside the variable. Let's call it var and it must be the same def API as the return data type. Here. Here. Here. And here. So it returns integers. So we need to receive the returned value

inside an integer. Now, once we have the returned value from the function, this is the function for. First, By its name. We can rent it out. Using a pen Epson. Results. The one. Person the. On air. We need to ask the bush. That's it. Let's save the code. Let's call it. Functions. Function this. Now let's compile it. GCSE. See and see. As you can see, it is all equal. Three. Now we can play a little bit with this call. You can pass variables instead of constants. Let's say that we want to ask the user. Went to values. So. First number. And we will take it with a scan of statements. Senators de. Let's call it. And first. Now. Let me define cool about that amongst integers. First number of people. Zero and two jobs. Second. Number of people. Zero. Now this is the second number. Variable that you want to store discount value on side.

```
#include <stdio.h>


// 1 - Function Prototype
int sub(int x, int y);

void main(void)
{
    int firstNum = 0;
    int secNum = 0;

printf("Enter first Number");
scanf("%d",&firstNum);
printf("Enter F Number");
scanf("%d",&se
                FP_OFF
                FP_SEG
int var = sub(Function                nction call by it's name
printf("Result equal %d",var);
}


// 2 - Implementation
int sub(int x, int y)
{
    int z = x - y;
    return z;
```

In the split seconds. So here's the first number. The second. The. Now we will call the function ourselves adding the two constants. We will add the first number. On the second number that they used up until. Then we will print out. That is the size that's on here. On the slush on ash and those spots on. Now let's call this again. Compile it. As you can see in the first number, it's set in and the second number to say four. That is almost equal sex, as you can see. It was implemented without any issues. So a function called counting constants and counting variables. As long as these variables are of the same type as the implementation and the prototype. Always make sure that you are using the same time or the same time, or else you might end up with syntax or logical errors. This is how easy it is to create a function. Those are defined functions. As you can see, this is the first step function prototype that it's here and at the time the function name and the two inputs ended with us. Second step is the implementation, which is the function itself. This is above the main. This is in or below the main. The same as the prototype, but instead of a semicolon, you need to curly braces and add your code inside it. The last step is the function call, which is basically calling, just like calling someone calling a function by its name, passing that input and getting the result. That is what it is and storing the result inside and the which is the variable that you want to store the results and.

VOID KEYWORD IN C

The void keyword is used for any function to give the meaning of nothing. For example, a function that takes void and returns void is in front of you. As you can see, this is the return to the type, as we mentioned in the previous lesson, and it's void. Now, it also takes a void between the two parentheses. The only goal from this function is to print like. Now this can be changed with anything, but this function will never tell any variable or any value, and it will not take any and both values. So for example, if we need to define a function that takes no arguments, we will vibe between the two parentheses.

VOID KEYWORD

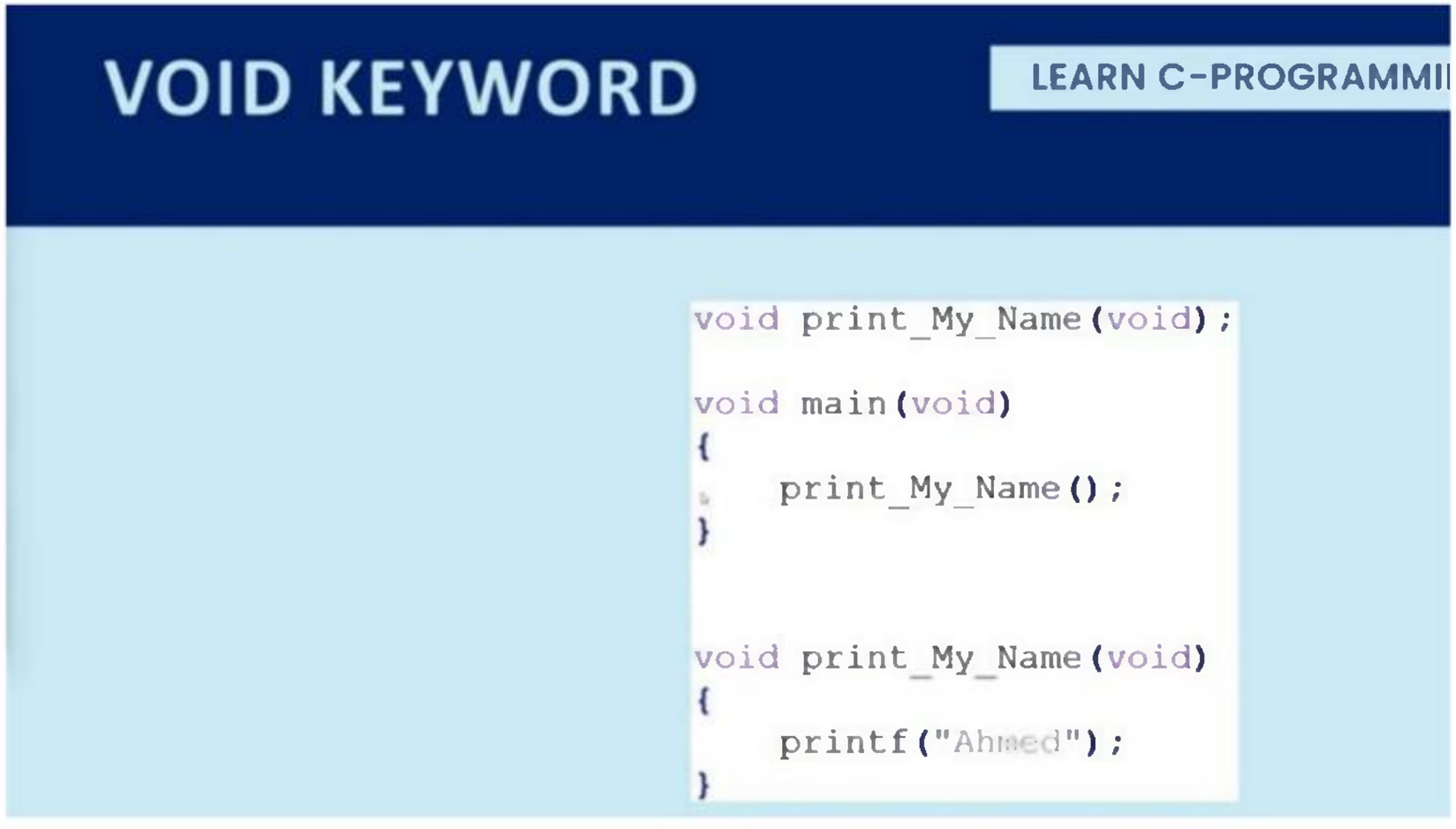
LEARN C-PROGRAMMING

- The ***void*** keyword is used to any function to give the meaning of No thing.

```
void print_My_Name (void)
{
    printf ("Mike");
}
```

The void. He will. Just like in this example, if we need to define a function that does not return any output we would like instead of the return time, the keyword void, and this is the void return type, the function can neither take input nor return output. And this is basically the whole meaning of void means nothing. So if you saw the word void anywhere, it means that this place will text nothing or it will return nothing.

Now, this is another example. This is our prototype for a function that returns nothing and takes nothing. That's why we have void here and here. Then all of this functions as a print by name. Now, as you can see, this is the implementation of this function and we use the same prototype, but we add two curly braces instead of the semicolon, and inside it we add the print statement. Now the last thing is the call and quoting a void function is easy. You just call it by its name without writing anything between the two parentheses, because it doesn't take any outlook, and without receiving a value from this and sorting to get into a variable, so to call a function that takes void the light, its name and type nothing between those parentheses. Just like in this line. And. If the function returns void like an odd example, then don't receive its output in a variable. Just call it as it is. Now let's do this in a practical manner.



Let's draw our own void function or less suitably by solving our own void function. Now let's add the hashing load that we are used to. This is all, man. And a study. I'll look at the server file and call it. Let's. Void keyword, see? Now inside here after using the hash include we need to call the void main function. Now the malfunction can be void. Our event, it's at the void. This is the syntax that we usually use. Now, let's try the prototype here. Boyd's friend's name. And boy. Now, the fine name, as we mentioned, is void keyword. Now, let's copy the prototype listed down here. And remove the semicolon. Took our live races and went off. Oh. And now to call. This is the first step. This is the second step. The first step is calling this function inside the main. So simply call it by its name, then name and add two parentheses without anything inside of them and it wants to return. I think you save. Fine. Open containing folder cmd. Let's copy the name. JCC, the name. This whole these two. Now add XP and those you can see execute this function. This is a function that takes nothing out of it. We call it a void function. This is how easy it is to implement it. So now whenever you see the word void, it means nothing. Don't put anything on Twitter.

DIVIDE C PROJECT INTO MULTIPLE FILES IN C

No Child Going to cover Dividing Sea Project into manifolds. Sometimes it would be recommended to split your project into some C files for issues of modularity and organization. All the C files we've compiled together to generate one output file. The general rule is to use or to call a function in a file. That is not the main file. You must know the prototype of that function. Now, let's say this with an example. First, let's make the main file include. And in here and. Man. Void. And here you need to write anything that has a functional statement. And for the sake of this lesson, we will apply the ADD function. We will add two numbers together using a function. So let's create a prototype of that function. So void. And. We'll take two parameters, X and Y. Now call me the prototype and place a test. We will do it without separating it into two files. Then I'll show you the separation. Now, let's save this. Let's call it 20. Many see fines, but see. Now. This is the prototype. This is the implementation. So this function will basically. Some actual numbers, x plus Y. And it will print. That is obvious. So some results. It was. Personally. And let's face it, that's it. This is all functions that will take inputs and quality, and I think it will present results here. And the last thing that we need to do is to call it. We will give it five and six. Now let's compile. I know that I'm typing faster because this is a code that we already did in the functions lesson. So now we will slow things down.

```c
#include <stdio.h>

void add(int x, int y);

int main(void)
{
add(5,6);
}


void add(int x, int y)
{
    int z = x + y;
    printf("Sum result equals %d",z);

}
```

```
Microsoft Windows [Version 10.0.22000.376]
(c) Microsoft Corporation. All rights reserved.

C:\C Codes>gcc 20manycfiles.c

C:\C Codes>a.exe
Sum result equals 11
C:\C Codes>
```

We combined the code, we executed the code and the result equals 11. Now, let's say that we want to organize our code and to make a new file for the function. So we don't want to write the function here inside our main file. So go and create a new file. Let's save it. Five Save us and let's give it a name. Let's see that we have 20. So let's call it a 20 out. Let's see. Because it is the same lesson. Now, here in this file, we

need to copy the implementation of our function. So could this institute. Now, since we are using a function, this file will not recognize the function. So we need to include. The library. That has this function. It's called a stereo. The same one that we call to you as to the old adage.Now. That's it. Let's try executing the same code. Now, we don't have any functionality in here, but you must have the prototype and the main function. Also try. This is where we fall. The function. The only thing that we copied to the other file is the function implementation. So that's a bit of advice as the function implementation demand file has the prototype on that call. Now to combine these two files, you can simply go here and right, this is here. No. I need to concentrate. As you can see, that's the first. Finally, 20 dollars. Bottles of tea and you need to find a second foreign name. So write one. Betsy then hit enter. Now, as you can see, it didn't recognise the first five because we have spelling errors, we have many c fives. Now once you hit enter, as you can see, it did come by the file. So if you're not ADA XP, you will get the very same result that we got before, even though we have our function implementation separated file. This is how you can split your code into multiple files. Now, this is one approach. The other approach is to create your own library. This is something we will cover in the next lesson. So let's clear this.

```c
#include <stdio.h>

void add(int x, int y); // prototype

int main(void)
{
add(5,6);  // call
}
```

```
Microsoft Windows [Version 10.0.22000.376]
(c) Microsoft Corporation. All rights reserved.

C:\C Codes>gcc 20manycfiles.c

C:\C Codes>a.exe
Sum result equals 11
C:\C Codes>gcc 20manyfiles.c 20add.c
gcc: error: 20manyfiles.c: No such file or directory

C:\C Codes>gcc 20manycfiles.c 20add.c

C:\C Codes>a.exe
Sum result equals 11
C:\C Codes>cls
```

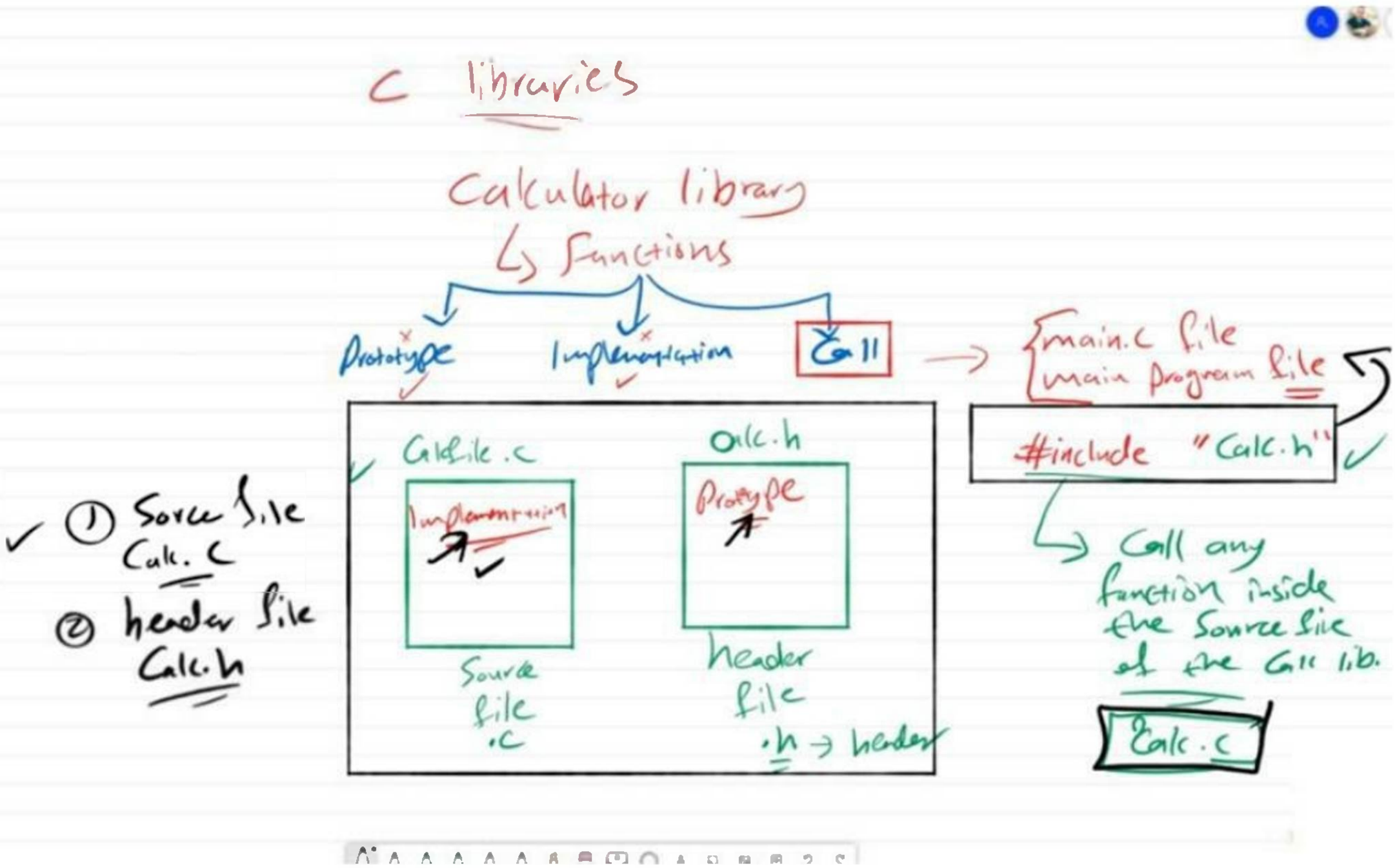
Now, another thing that I need to mention in this lesson is that we always get it easy, which is the default name generated by the compiled pads. Is there a way to change this name? Yes, that is okay. So let's call the combine again. As you can see, just as you combine up the first file on the second file, then add a space. If you add a dash, all stands for output. You collide any p e file name. So let's name it. Many files that easily. Hit

Enter. Now you can simply like many files the p e. Now we have this story spelling error and it will give you the same result. Now, we don't have it easy. We have the fine name that we added. And to do or to give this result, you just need to dash all then a space, then the final name. And in that, with that easy. This is how easy it is to customize the outlet file name instead of keeping it easy. Now you can customize it to get any file name depending on the project that you are working on. Now let's go back to our code and summarize what we did. We added the prototype to the main file. We call the function inside the main. Now, what is the implementation? The implementation is in the separated file. Here is the second C file. We call it with any name and we added the implementation for that function. But in order for this implementation to work, we need to add this library because we are using the print f function, which is a function that is not recognized by this file. So we need to include that as to the i o that each library to make our function recognize the print f function. So you can't separate that implementation, but you have to call the prototype before the main function. And you can easily call the function inside your domain. This is one way to do this. Again, the rules are simple. You need to combine or use these compilers to compile both files. And we already mentioned that you can change the ATC default output to anything by adding dash or as base. Then the file name. That's it for this lesson. I know that these are out of information to take, but they are very simple. If you want to separate your project, you can make another find and place all of the functions implementation into that new C file. In the next lesson, we are going to talk about libraries and how you can easily create a library in C.

HOW TO CREATE A LIBRARY IN C

In order to create a library and see, first you need to know what is a library? A library is basically a set of functions that do specific actions, and we usually create library use for sensors for, let's say, an LCD display for keyboard, for reading a temperature value or simply a calculator. So let's say that we want to create a new library and call it a call, collate or library. We already mentioned that the library is a set of functions. Now these functions are to be used. There are a few things that we need to write. We know that each function has three things. The prototype. That implementation. The call. Now, in order to create a library, we need to know the number of files that we can include. Basically you can include an unlimited number of files, but we will only concentrate on two types. This is the first file and let's call it the C or main, but C file. Now we can call it a calculator. This is the second file that we need to call and let's call it calculator dot edge. This one is called a header file. This one is called source file. The extension for this file is not seen. And for this file is not which. Stands for head of. Now we need to know what are the things that we will write inside each of these files. The hangar file will include the prototype. Photo type. This thing. So now we are done with this. The ceasefire will include the implementation. So we are done with this as well. Now the thing that remains is the coal. Coal is basically something we will do. And our main. That five, which is the main program file. Now to do this or to call these functions inside our mainframe, we need to include our library. So we write the word includes. But instead of writing these two signs, we will add the double quotation

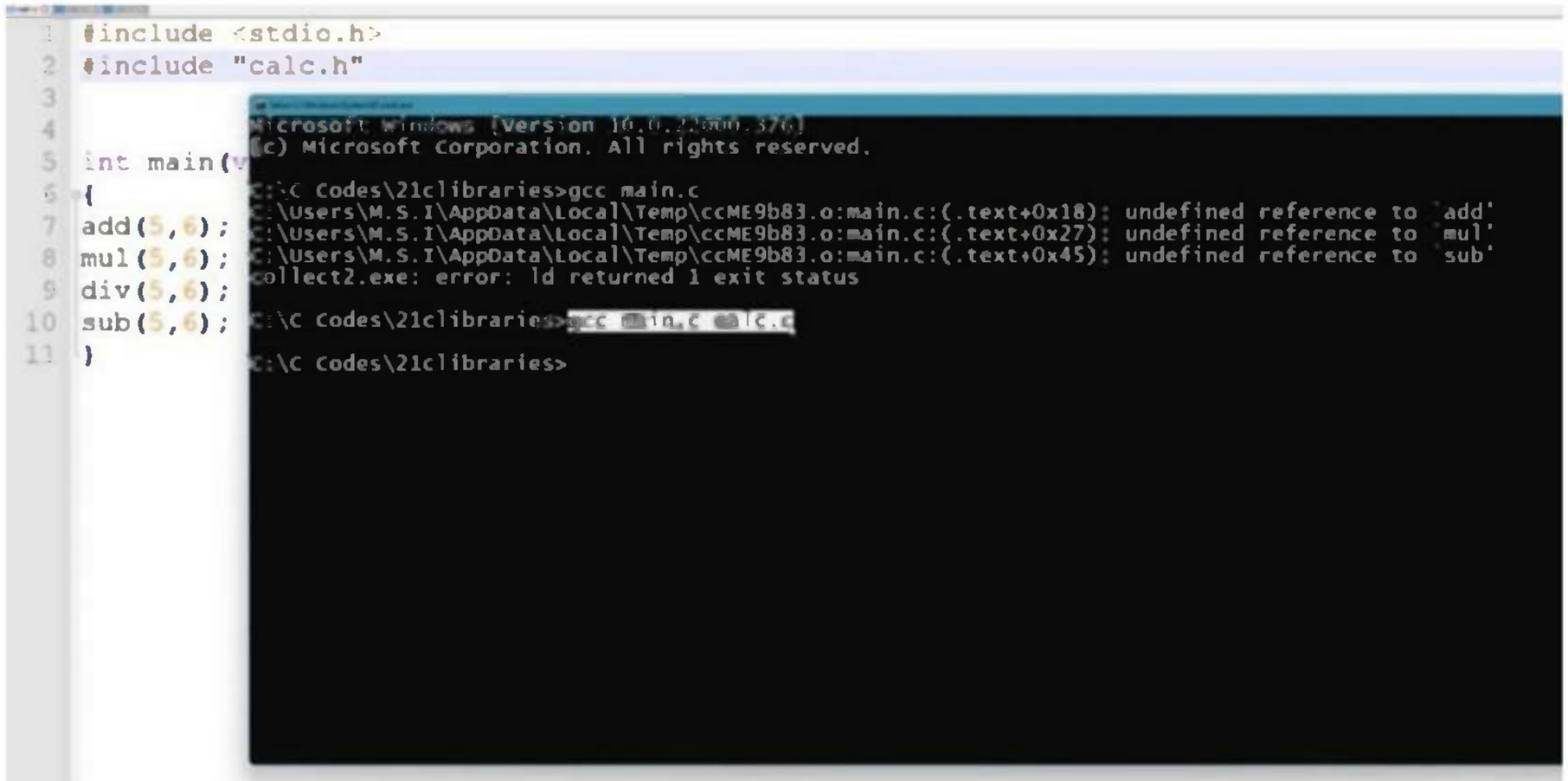
sign. And the only thing that we need to include is the header file. So on the calculator, the header, this line will allow us to call any function. Inside. The source file. Five off the calculator. Library. So if we did this, we wrote this line, this means that we can see any function inside that. Calculate all the C files. Which is the file that we created here. So to wrap things up, this is our library. This library has two things. First, a source file. Let's see. Second is ahead of the file. That. Now let's call them count. Count stands for calculator. Now the source file, as we mentioned, will include the functions implementation, while the header file will include the functions prototype. Now to call the functions we need to add the hash include which is this line. Inside our main. If we added this line, it means that we can see and interact with any function inside the galaxy, which is the source file, this file or this file? And this is basically how easy it is to create a function inside. Now or solid to create a library in C.



Now, to make things more clear, we will create that very same thing with the very same name calculator library inside our notepad and see how things will be executed. So let's do this. Let's go to our C compiler. Now this is our main file. So we need to do what we usually do. Which is included. And the man. Void. Once you do that, we need to save it. I'll create a new file. Let's see the numbers to make sure that we have the

right numbering for all of our files. Okay. We have it at 21. So let's call it 21. See. And I. Please. Now inside it. We need to create the man. See? Fine. This is our first five. Now we need to create another file and name. It can't be seen. And this is the source file. So source. Fine. Here we are. I implement functions. And we need to create another file. What should the header find? Functions. Prototype. Just like what we mentioned in our explanation. Make sure that it has the same name as the source file but ends with each and a C and click save. Now the implementation for our functions we need to add. A function that will sum to numbers and X and Y. And we need more. Then. We need four functions, actually. Ads on simplification. Division. On subtraction for the calculator. There are two numbers to multiply them, to divide them, to subtract them. Now, this is the header file where we wrote the function prototype. Copy this prototype to the source file. T It published the semicolon with two early braces. Now we need to start fighting our courts. As usual. This is nothing new. We are writing the implementation for each of this function, so instead equals x plus y. Both f. Some results. Equal percentage de. And it adds up to you. Call me the same lines. Here, here and here. Now, sort of some years later, I'd love the application and would need to add division. And here we need to add subtraction. Now we are done with the functions implementation inside the source file, but in order to use the print if we need to include the standard library. This is the stand up library. Now, we finished the prototype. We finished the implementation. We need to call them. So we need to add another hashing fluid, just like what we mentioned with double quotation like that. That. Now this means that our main function now can see the functions created inside the source file. So let's call an. Let's call the aunt. Function at five of six. Let's add slush and hash. To make sure that they will not be printed on the same line. Okay. Go back here. And it's called the ad function. The multiplication function. Another division function. The subduction function. Now we have to say, I think if you want to include the library that is created by the compiler creators, you can add that between the larger and smaller sites like that. But if the library that you want to

include is created by a user, you must add it between two double quotations, as you can see here. And this library was created by us. That's why it's edited between double quotations, and this library was created by the compiler, manufacturers or creators. So it is added to mean a greater or less equal sign. So as you can see, this is a good piece of information that you must keep in mind. Now let's save and let's try compiling our code. So here we need to write. JCC, man. Let's see. Now, as you can see here, we have a few errors. This is because we tried to compile only that first see file. Now I see. Main see. Karl. Let's see. As you can see it combined without any errors. As we mentioned earlier, in order to compile to see files, we need to make sure that we name the files here. Now let's execute that easily. Now, as you can see, some result equals 11. Okay.



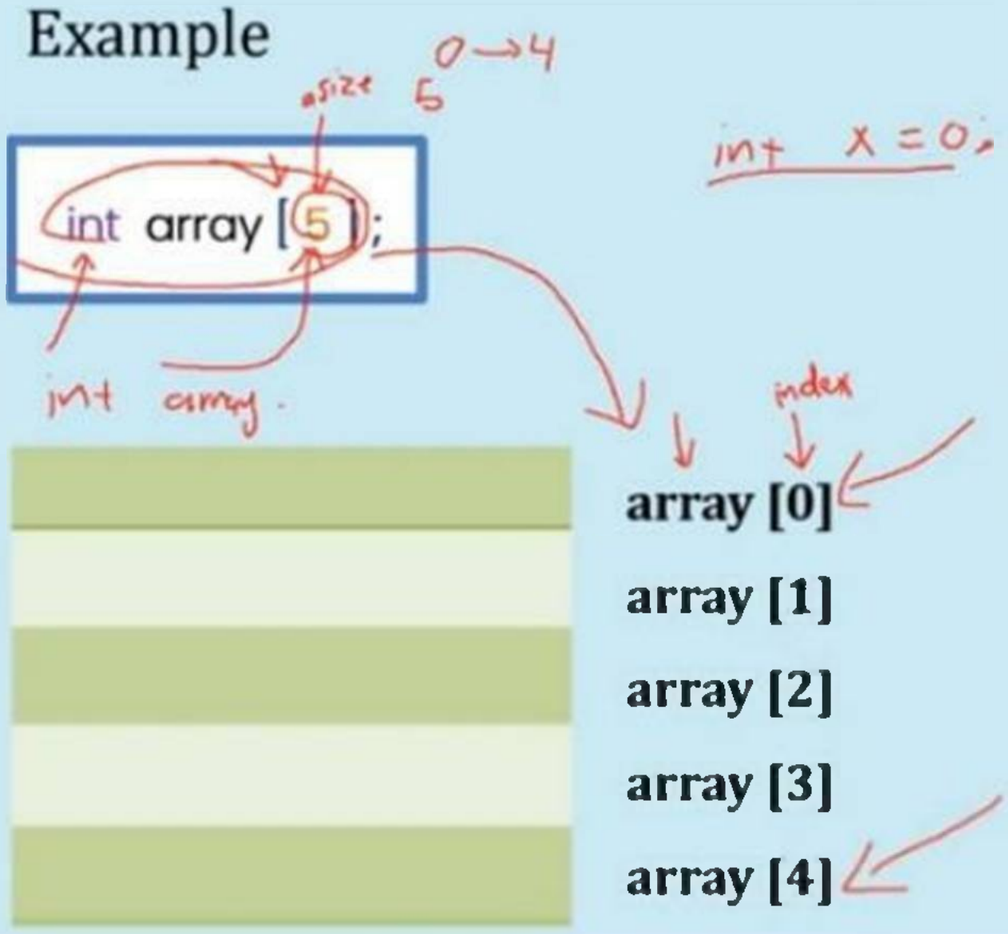
We have a problem and implementation. So here we have multiplication, asterisk, division and subtraction. Let's call it a game. End of the day, as you can see, 1130 zero minus one. That's it. Now, as you can see, our code works just fine without any issues. And we just created our first library. And see if you have written everything that I wrote and you created this library, then this is. The time that you must clap for yourself because you just created your very first sight in your library. I'm sure that this is a lot to take in from one goal.And try to repeat it more than once. Then you can go to the implementation and apply the same thing that I did. Let's summarize what we have covered in this lesson. In order to create a celebrity, we mentioned that you need to create two files. You cannot create more than two files. It's basically a design approach. But what we will create is a source file on the head of file. The source file until the file must have the same name with a different extension. The source files don't see the header files dot hitch. Now these two files are basically the library and we are creating a calculator library. The source file will include the implementation or functions and limitations. While the header file will include the functions prototype. Now the remaining thing is the call. In order to call these functions. We need to include the library name. The library header file is the thing that we must include in the main CFI so that I can include and add the header file calculator. Now you must add it between two double quotations because this is a library created by a user, not compiler related lightly. Stuff, including you can easily call the functions inside your domain. Because once you include this, it means that you have the prototype for all of these functions. Since the head up file includes the prototypes and you can call them easily. And this is what we did here. This is the functions implementation inside the source file. The header file calculates or the edge includes the prototype, and the call is inside the main. But before calling, we need to make sure that we are using the hash calculator at the edge. And this is written between two double quotations, not between larger and less than signs.

ARRAYS IN C

All of them are the same type. For example, let's look at this syntax. This is called the RH definition. The desert. Here is the intro. It's like defining any variable usually and defining it. We write indexes. Equals zero. But in this case, I'm still fighting this. We are adding these to. Brackets. And between them, we will add the array size. This is the size of our array. So this line will do this in our range.

INTRODUCTION TO ARRAY IN C

LEARN C-PROGRAMMING



It will create five variables. Their name will be an array and you can access them using the index. The number that we will write here is called Index. Now, the first thing that you need to know is that that day count starts from zero. So if it has five here, it means five elements. So it's 0 to 4. Now that index always starts from zero. It means that the rate has elements from element zero to element four, as you can see here. Relive

must be constant value and it cannot be variable. You cannot add other variables to the five here. You cannot do this. And every. And add X between the two brackets. This is wrong. It will give you a compilation. So array the bucket sticks. Only Coniston doesn't take value. And this is what happens when we execute this line. It will create five variables, and we can access them by simply writing Race zero. And we will see this in practice in a few minutes. Now. This line creates an array of integers of size five. Another example here is the array can be initialized during the time of definition. Using this syntax you can write out of five. And the two curly braces.Here and there and start filling them up. Now, since it's an array of integers, we will add integer values and you must separate them with a comma. As you can see here. The line must end with a cynical. Now, when you do this, it will trade this thing inside our memory. It will create a real zero greater value for everyone if the value of to agree to give it a value of three. Array three. Give it a value of four and therefore give it a value of five. And this is basically these values, as we already mentioned, will start with zero, one, two, three and four. And this is how easy it is to initialize an array during the definition. Now. There are some special cases when initializing the array with values less than its length. So here we have the length five and we only added two values. The remaining elements will be initialized with zeros. As you can see here, here and here. So it's like you wrote 1200 and zero. These two are the same. As long as you are not adding the rest of the values, the compiler will autocomplete them with the zeros. Now if you try. Initializing the array with values more than its length. This will lead to a compilation error. So let's say that this array has five elements. Let's go back. And we added another coma on six. Another on seven. This will most likely give you a compilation error because you are adding seven elements to an array of five elements.

ARRAY INITIALIZATION

LEARN C-PROGRAMMING

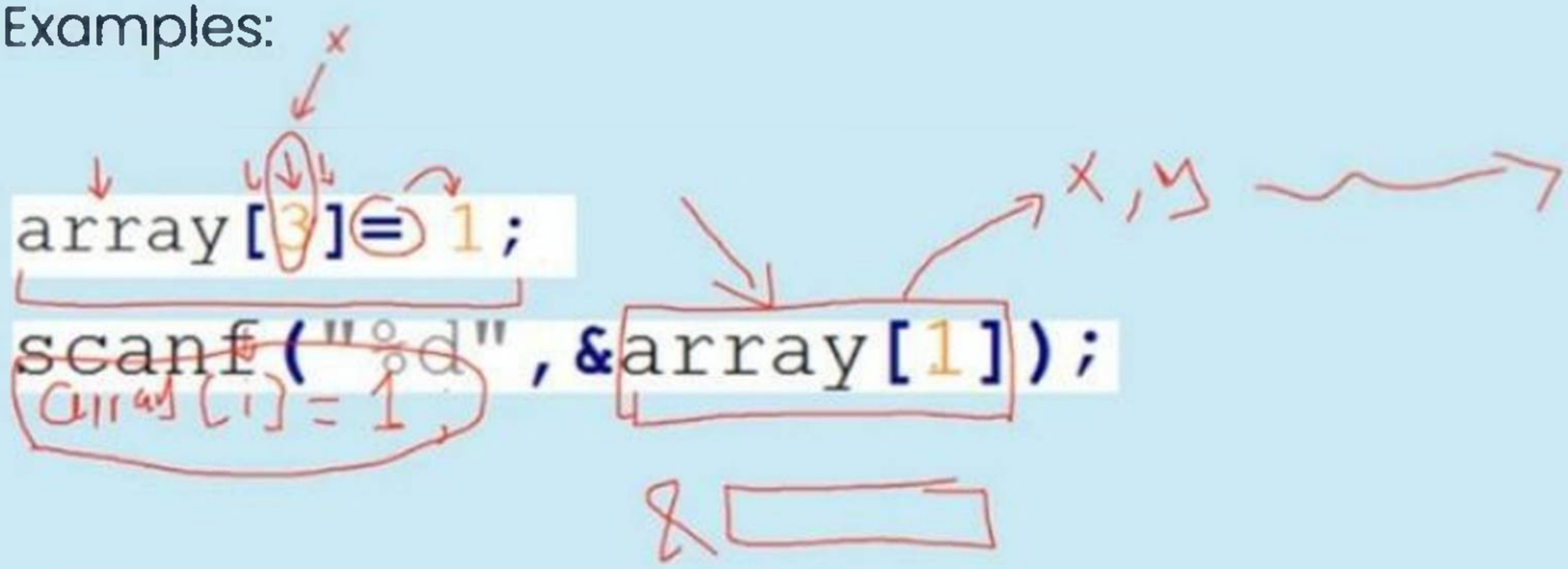
```
array[5]= {1,2};
```

So five elements. This must be only five. If you are just seven or six, it will give you a violation of. These are a few of the things that you must keep in mind when dealing with utterly. Now let's talk about accessing other elements. What you need to know here is that all other elements can only be accessed on the same statement at initialization. Like in this example. You can access all of the items and here we access all of

these items. Now, the point is. After the initialization, they can only be accessed element by element. So once, once initialized and you want to add a new value, you must add it using this syntax. You will revive the array. Name two brackets. The element number, then the equal sign, and then you can add the value. Again, that index starts from zero. So you need to take this into consideration. When you are writing this number, you can use a variable to indicate the relevant index. So this can be replaced with a variable, let's say x or like an array. I. And we can add it inside affordably. This can be a variable, but this is during the, let's say, accessing of array elements, not during the initialization. Now, if we tried adding a variable inside the initialization line like in here, it will give us a compilation error. But you can access array elements using a variable that won't cause any error. Other ways that you can access other elements is by scanning a variable directly inside on other elements. So arrangements are being treated like normal variables. So R1 is basically a variable. Just like x, y and any other variable. So you can easily access it by simply writing and the element. And just in this case, it's array one. You can even use them inside a print function. As you can see here we are printing element zero.

ACCESSING ARRAY ELEMENTS

- Examples:



And we percentage percentages just like a normal variable and sort of adding X or Y, we are adding the RH element that we want to print. Now, to make things more clear, we will do this in a practical manner. So let's get started. Hello and welcome. Now let's add cash. And then.Voice.Now. Let's save this. Let's call it. I really. Folks will be. 22. If I recall. Arby's. See? Now. That's it. Let's start by creating a new Ari. We already men-

tioned that you need to write another type. I have a name. Two brackets are their size. So let's get another all out of this. And make that number five. Now there is more than one approach to creating anything. You can't be like that, or you can directly initialize it with values. One, two, three, four, five. Or you can simply. Initialize some of the elements like one and two. As we already mentioned, this will add zeros and the remaining elements. This will add these elements inside our array, and this will not add an element inside our alley. The first piece of information that we didn't mention is that you cannot add a variable here unless the five. This is for Britain. And see now if we tried adding a variable, let's say an X equals five. Let's find out why. This? Now let's put all these. See? Now, as you can see. Now I combined them correctly. But we have a problem. Now who sees this problem? Let's go here and try to print. Let's add some values here. And to that idea that we just created with a variable initialize it less than the values that so badly. And I mean. Zero.If one person is the. And similar, are they? Of zero this are. And we want to present this value. Now go back. Try to compile and you'll see that we have an air of variable sized objects that may not be initialized. As you can see, it's indicating this.

```c
#include <stdio.h>


int main(void)
{
    //int x = 5;
// datatype arrayname[arraysize];
int array[5]={1,2,3,4,5,6,7};
//int array1[5]={1,2,3,4,5};
//int array2[5]={1,2};
printf("array element 0 = %d", arra
}
```

```
c:\C Codes>gcc 22Arrays.c
22Arrays.c: In function 'main':
22Arrays.c:8:25: warning: excess elements in array initializer
 int array[5]={1,2,3,4,5,6,7};
                         ^
22Arrays.c:8:25: note: (near initialization for 'array')
22Arrays.c:8:27: warning: excess elements in array initializer
 int array[5]={1,2,3,4,5,6,7};
                           ^
22Arrays.c:8:27: note: (near initialization for 'array')

C:\C Codes>_
```

Excess element in our daily life is, as you can see, it's not taking out of the elements since we have a variable here. Now to fix this place, this will have five. Comment this line and it should render value without any problem. As you can see, the other element zero equals one. And this is the first thing that we need to mention, which is do not add the variable inside the array initialization. Now. The second thing that we need to

mention is that. If you try to initialize an array of nets from these two. And work on this one. If you try to initialize and agree with more than its capacity. So here we have five elements and we added seven elements. Now this will cause an error as well. So just to see. Ari, as you can see, excess elements and Ari initialize it and it's pointing to elements six and seven. So it's indicating that these two elements are extra elements and your average size is only five. So to fix this, you need to remove them and make sure that you only add elements with the same number as inside. But all brackets. Now, the last thing that we need to mention is the other way or accessing elements after the initialization. To access them. You need the library name, two brackets and the element that you want to access. If you want to access the first element, it will be element zero. The second would be element one, because we already mentioned that the account inside the array starts from zero to less than the maximum value, which is five. So from 0 to 4. So if we want to access elements number three. We need to count 012. So here we need to write. This will help us access this element. We can send a new value to element number two here. Which is basically the three inside already. I'm kind of blessed with them. And this is how you can use axes and other elements to read the value inside it or to change the value. Now you can also print a value. Are they animals? Percentage D. And we can either rename. As usual, we can ask for any numbers. Let's say that you want to plant this value. So, zero, one, two, three. We need to add three here. Now let's go on this line of automated. Now we want to present two values. A great tool since we changed the value of three. Now let's compile. And execute. As you can see, the new value enhanced is ten and the value here is four. We printed values numbers two and three, which is ten and four. Now, if you want to print all of the three elements, you can. And these zero, one, two, three and four. Now I kind of go by combine. And execute. And here we have the one, two, two and four, five, one, two, ten, four, five. Now, this was replaced with three. Three was replaced with thin. And this line. Now, since we are printing using the very same line, only one thing is changed, which is the value inside the two

brackets. We can replace all of this with a formal. So we can either fall. And here we cannot. And X equals zero. X is this darn explosive less. Now we can cut this line base to tear the place. This place. Now, this is the thing that most people fall for, which is the condition inside the stadium. Should they put it six, five, four or less or equal or less? So in my case, the best thing that I prefer to do when dealing with arrays is simply adding X is less than and the number inside the brackets. So X is less than five. This will help us print all the values inside the array. Except for element number five, which is an element that does not exist because we have zero one, two, three, four. So always when dealing with for loops and arrays, add less and the number between the two brackets to make sure that you are printing all the numbers inside the battery. Let's execute this line. As you can see.One, two, three, four, five. Now you can make things even more interesting by adding another percentage to the year. To indicate that element that you are printing.So. We will add another variable. If we x here. On trying to combine the codes.The school, the sun. It will prevent this element zero from being one element. One is to add an element to those three out of the elementary is four add elements, four is five, which is a very interactive way of floating elements. Now, to make things more convenient, we need to start the elements with one set of zero so you can easily do this by adding plus one. And two between.Brackets or sorry parentheses. Now combine and run as you can see how relevant. One, two, three, four, five.Small convenient for the end user to see one. There is nothing called element zero since he doesn't know that an array starts with zero. So it's not user friendly to start with one and the user interface. But you must know deep down that it starts with zero. And this is how you can easily plan the arrangement. Now, in the next lesson, we'll take a good example of how we can take elements into the array and print elements on that display, just like this execution. Now, the last thing that I need to mention here is maybe some of you are asking why using arrays to begin to lo

```c
#include <stdio.h>


int main(void)
{
int x1 = 1;
int x2 = 2;
int x3 = 3;
int x4 = 4;
int x5 = 5;

// datatype arrayname[arraysize];
int array[5]={1,2,3,4,5};
//int array1[5]={1,2,3,4,5};
//int array2[5]={1,2};
//printf("array element 0 = %d", array[0]);

//array[2]= 10;

for(int x=0;x<5;x++)
{
printf("Array element %d: %d\n",(x+1),array[x]);
}
```

Why don't we use variables? Now we can use variables. You can go and create, let's say, five variables and give them these values. x1x23x45 and you can simply print them using five trend f statements. Now it was basically I created or let's say used in C to make sure that you don't waste your time by creating 100 variables of the same data type. Since we are creating five integer variables, why don't we create them with

a single line and a set of five lines? Why don't we print them with two lines and instead of printing them with five print f statements? Now this appears more convenient when you are creating an array of 100 elements. Let's say that we want to spend a hundred elements, not just five elements. It won't be convenient to create 100 integer variables and keep naming them x1x2x3x4x562x 99 or 100 and giving them values. The easiest way will be to create an array like this of a hundred elements and to print it using two lines instead of a 100% F statement. And this is why ugly was created. Now, if we try to execute this line. You can see that our compiler printed the first five numbers. And as we mentioned earlier, if you haven't added the rest of the numbers, it will automatically add zeros. So here we printed the hundred variable with only two lines using a four loop. And this is something you can do with arrays, but you cannot do with regular integers and regular variables. So you can either go with this approach or you can simply use one line. Of course, to create these five variables. And this is the main point of using an ally to save time, to make sure that you get the best results out of your, let's say that are management and to easily manipulate print take in values inside your array.

ARRAYS EXAMPLE IN C

So let's get started. But I include a civil to. And man. Boyd. And let's say. So the corner inside here. Going to. One, two, three. Another example. See. And as usual, let's run the file in here. Now let's get another end mark. Five. Now this is an array that will take five elements and we called it MOX. Now we will ask the user to install his model using a formal. So integer x. X is less than five, as we already mentioned. Exodus service. Now school f will be used the same as with variables. But if centers die. And Mark's.Of X. Now to make things more convenient, let's ask the users to enter more. And your mouth. Okay. Lots of this. And let's add another item percentage D here to indicate which mark you will be entering. And here we will add X plus one. Now this will ask the users to enter the say. Firstmark. Second Mark. Third mark. And he will keep entering values and the for loop will make sure that each annual value will be stored inside. One of the three elements we have five other elements. Now the second thing is printing the marks. So we'll copy the same for loop. We will remove the scan if we tell him that the first mark equals. The array. What should masks be? Yeah. So first, Mark. XM plus one. Now this will make sure that we are printing the values that are given to us. So we will enter five locks and we will print these five marks. The first percentage will be replaced with the marked number, which is one, two, three, four, five. The second percentage D will be replaced with the market self. Now.Let's combined records. Right, Ed, easy. And here you can see it's asking us to enter first.

```c
#include <stdio.h>
//23Arraysexample.c



int main(void)
{
int marks[5];

for(int x;x<5;x++)
{
    printf("Enter %d Mark : ",(x+1)
    scanf("%d",&marks[x]);
}

for(int x;x<5;x++)
{
    printf("%d Mark : %d",(x+1),mar

}

}
```

```
Array element 97: 0
Array element 98: 0
Array element 99: 0
Array element 100: 0

C:\C Codes>gcc 23Arraysexample.c

C:\C Codes>a.exe
Enter 1 Mark : 90
Enter 2 Mark :
```

Mark So it is like 98 is 70, 60. Now, as you can see, secondary markets, 83 markets, 75 markets 65 loss is 50. Just like what we interview. Now. It didn't start with two, not zero. So we need to add zero here. And you'll hear. And we need to add a personality here so the percentage can. So now let's start again. 90, 80, 70, 60, 50. And here they are, 90, 80, 70, 60, 50. Now, what if you want to print the average? It would be

a very easy thing to do inside this application since we are using arrays. And some sort of average. And so. Now in the fall, we will add plus equal some plus equal marks. Of X. So each new value of X that the user's entering, we will take the value and add it to the song. Once we are done with this for loop, we will have all the values inside the sum so we can easily go here and write about its equal sum divided by five. And we can't print coverage here. About equal. First there's the. Average. Now let's throw this out your screen. Now let's one.90, 90, 90, 90, 90. So as you can see, these are the five marks and this is the average, which is 90, because as you can see, I wrote all of them last night. And this is how easy it is to create a program that takes mobs from the users and princes average without having to create five variables, without having to sum these variables and without having to use five scan statements and five printf statements. So we only did this with a few lines of code. This is and this is how great arrays are. That's why you should use outrage whenever you feel that you have the same data tied to a lot of variables, just like our marks here or any other type of situation. Again, a raise will make your life way much easier. You can do the work without them, but you have to keep copying and pasting lines of codes, losing memory and losing time.

HARDWARE AND SOFTWARE REQUIREMENTS

The harder material required so that you can get started with the scores are Arduino on a Bluetooth module usually SC 0 5 will do the job. It's a very common, very cheap one and it's available almost everywhere so you can check and express eBay and Amazon . You can get it at a very cheap price. And if you face a hard time getting any of these items you can send me a message and I'll direct you to some places that you can give them. You also need a mini Big Board or a normal brick board so that you can hook up the components. You'd need some capacitors, one microphone and 60 involved some resistors, a hundred ohm five pieces of lumber wires or any type of files that you are using. Depending on what you usually use a USB cable for programming your Arduino for the first time you will only use this once.

Materials and Tools

Then you'll start programming it wirelessly. An android device with Android for so anything below that won't work and your Android device must have Bluetooth built and again Bluetooth availability. We already mentioned that and a laptop or P.C. so that you can use it to send the code wirelessly. As we mentioned that you will learn how to program Arduino via a mobile device or a laptop. Now let's go and check

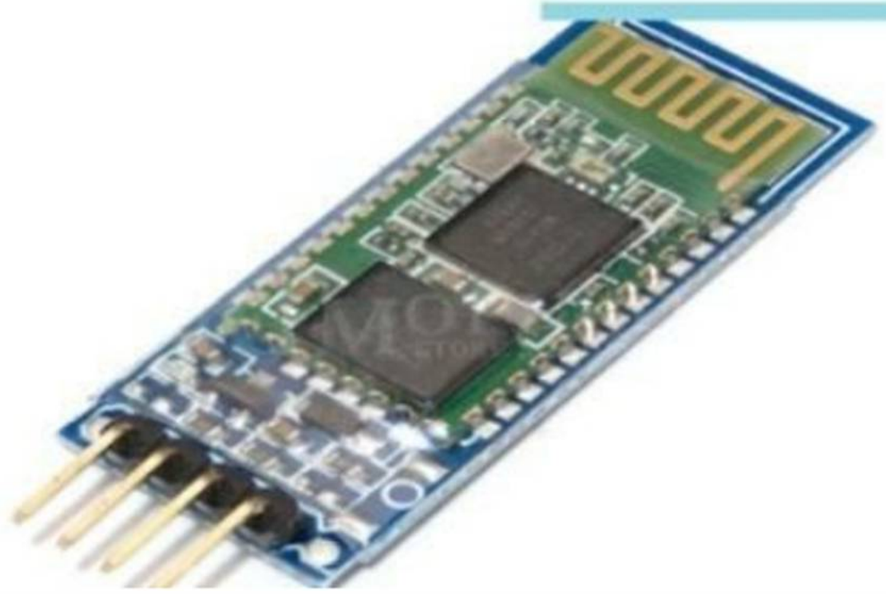
the software components. You'll need software called Window loader. This software is a free software that you can get from google play store. It's the software that we are going to use to write our Arduino code and send it to the Arduino board wirelessly. I will show you that in a minute. We will also need Arduino Ida E which is the free software provided by Arduino C.C. company for writing Arduino codes on a. We will need flying software which is a software for connecting your circuit or making circuit designs. It's also a free software and you will find an explanation of how to download this software. The section of the hardware and software material in this course. Now before going forward let's see these three softwares and how they lock in their websites and how you can easily find them.

And I leave the details to the lectures of how to download and install each of them. Uh this is a search engine. Now if you wrote Arduino either e download you'll get the download link for this Arduino I.D. software and you can simply download the required one depending on your operating system. Same for frightening as you can see sliding. Download it's the first link you can easily download for free from here. Now the first software from Google Play store and its name is Plano. So let's open up and check it out. As you can see we have flown an order. This is the software that we already talked about between a raw loader. It's an abdominal idea E for mobile devices and we'll talk about that and more details in the software section anyway.

HC05 BLUETOOTH MODULE

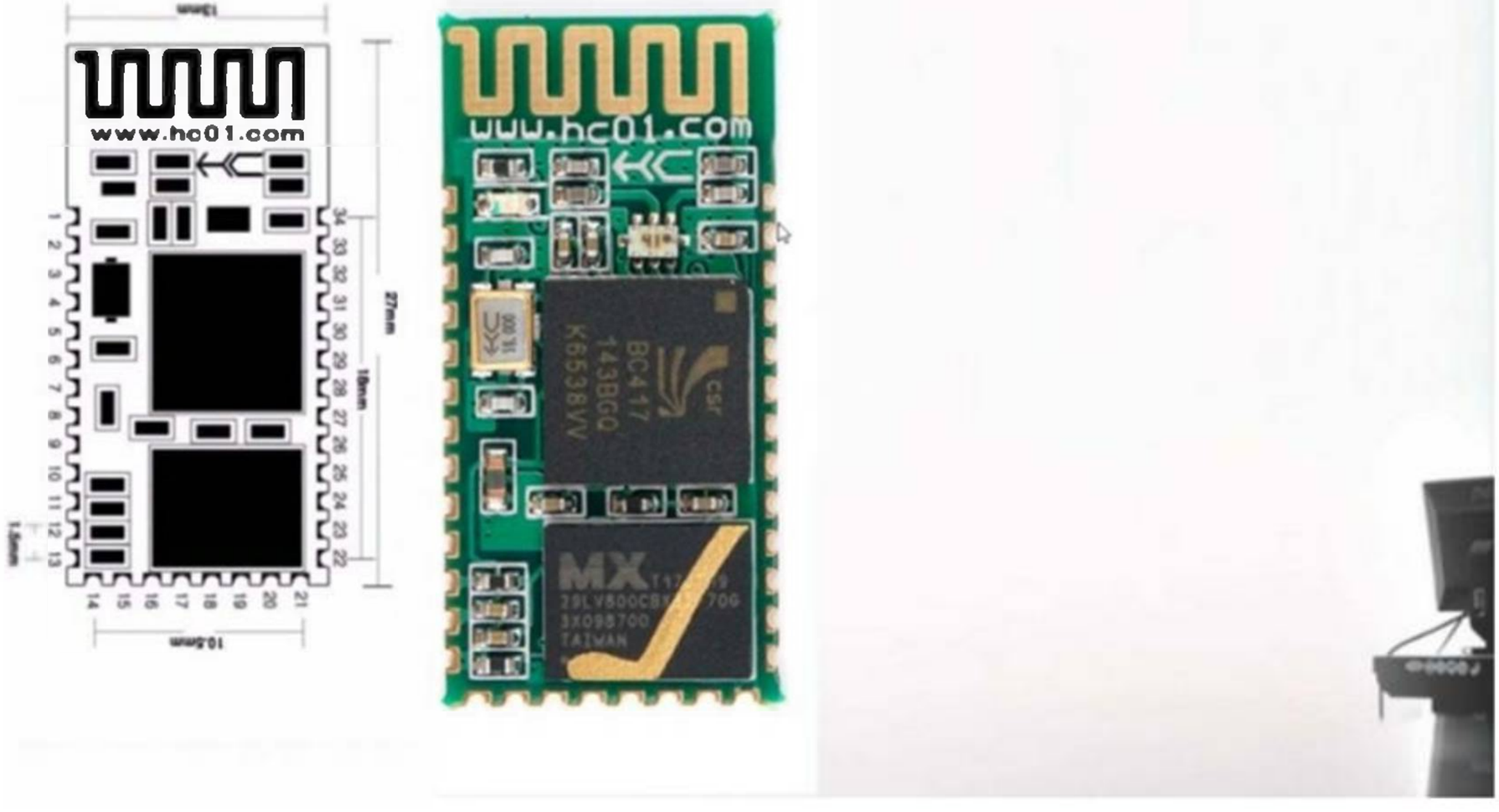
In this lesson we are going to cover that c 0 5 Bluetooth module. This project uses that c 0 5 Bluetooth module for communication. It's a very cheap and easy to find module. Now that c 0 5 comes in different versions and you have to pick the one that has a button on it. Make sure that you have that c 0 5 module and not the c 0 6 in shape. They look the same. They look identical but the difference is that that c 0 5 works as both a masseur and a client which is basically what we need. But the H C0 sex works only as a client and it won't work for us. This project will not work with that c 0 6 and again I have to mention that they look the same but they don't function the same. If you bought a module with the breakout board which is basically this one in the right on the left and the bottom right on the left this blue board is called a breakout board as you can see that green thing here is the module that we bought which is such a c 0 5. And if you want a module with the breakout board make sure it has a key terminal. Now if you look at this image let me use the pen.

HC05



If you look here you can find a button and this button is very important for us so you must go for a c 0 5 with the breakout board that has a button. Now some of these breakout balls don't have a button like this one as you can see there are two pins for the button but it's not there. So you have to buy it and sell it yourself or you have to connect two wives and hook them together when you want to push that button. This

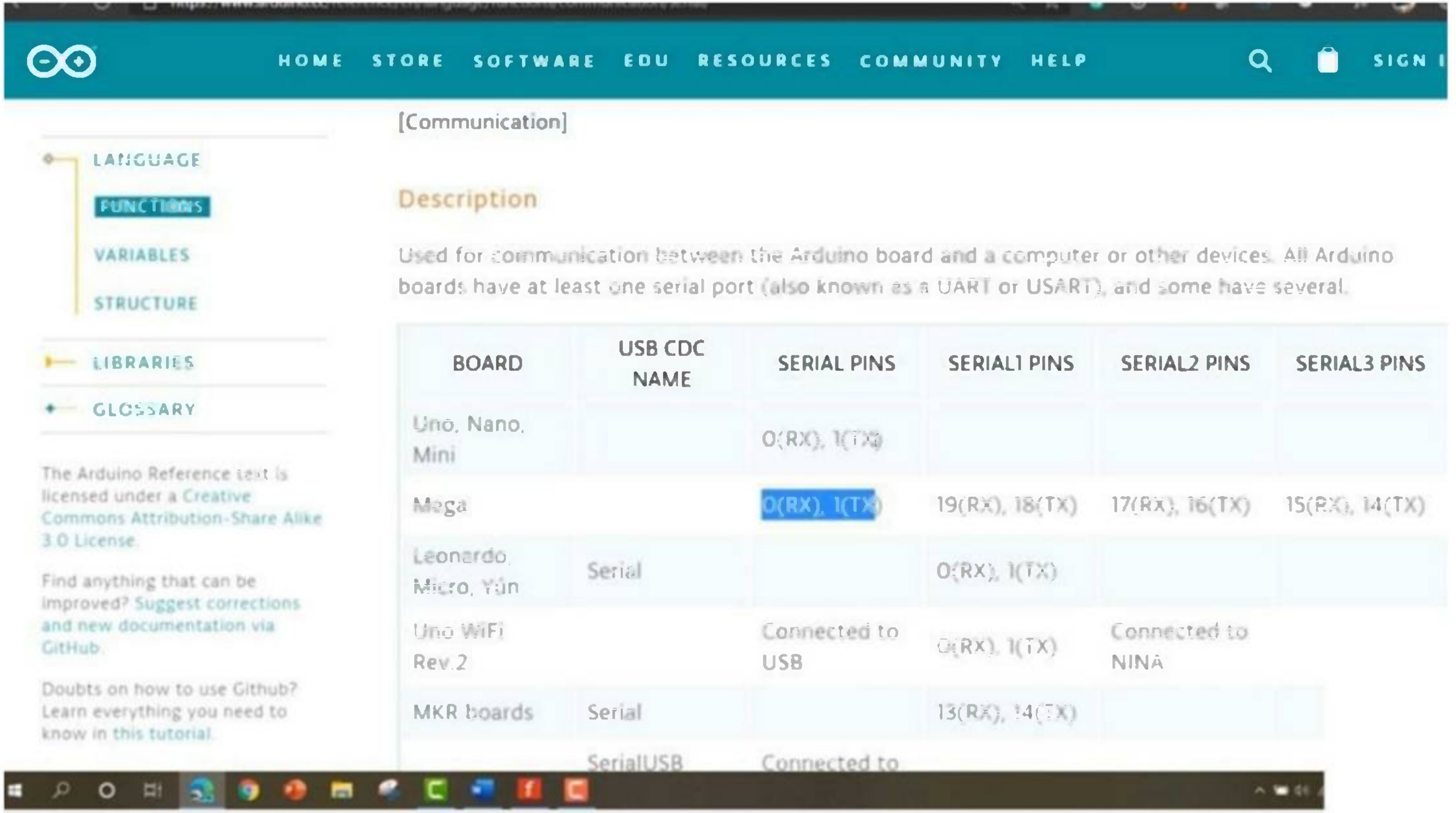
module won't work for us cause it doesn't have a breakout board and it doesn't have a pen. This is basically what we need now. If you come across a board or if you already have one that doesn't have a breakout board or a button you have to connect that button yourself. So you have to look for pin number thirty four which will act as a cabin then sell that bill of c 0 5 and hook it up to a button on a bridge board. This button is very important since we will use this to reset the board each time we applaud a code again. You have to do this yourselves and the final result will look like this. You will connect our two pin number thirty four and in the next slide. Let me show you as you can see here and this image. Pin 34 is this pin. So you have to look for it on your board.



And once you count from one to thirty four you have to connect a wire from here and connect to your board and use the Push button. If you don't know how to do it or if you already have the ball you can drop me a message or ask a question that you aren't able to and I'll send you a schematic for how to hook up that button. But since the module costs only for the alarms I think that you should go with the one that we already mentioned which is the one with the breakout and has a button on it. That's it for that c 0 5 module. Again you have to go for that C0 5 not let us see 0 6. You have to buy the one with the breakout board that has a button and if you don't have the breakout ball on the few already have the ball. You have to connect the 32 to a button because using this pin is necessary since we will set the board each time we applaud a code using this pen and the Bluetooth module that is set for this lesson. If you have any question regarding the HSC 05 please ask it in the current table. Thanks for reading. This is an educational engineering team.

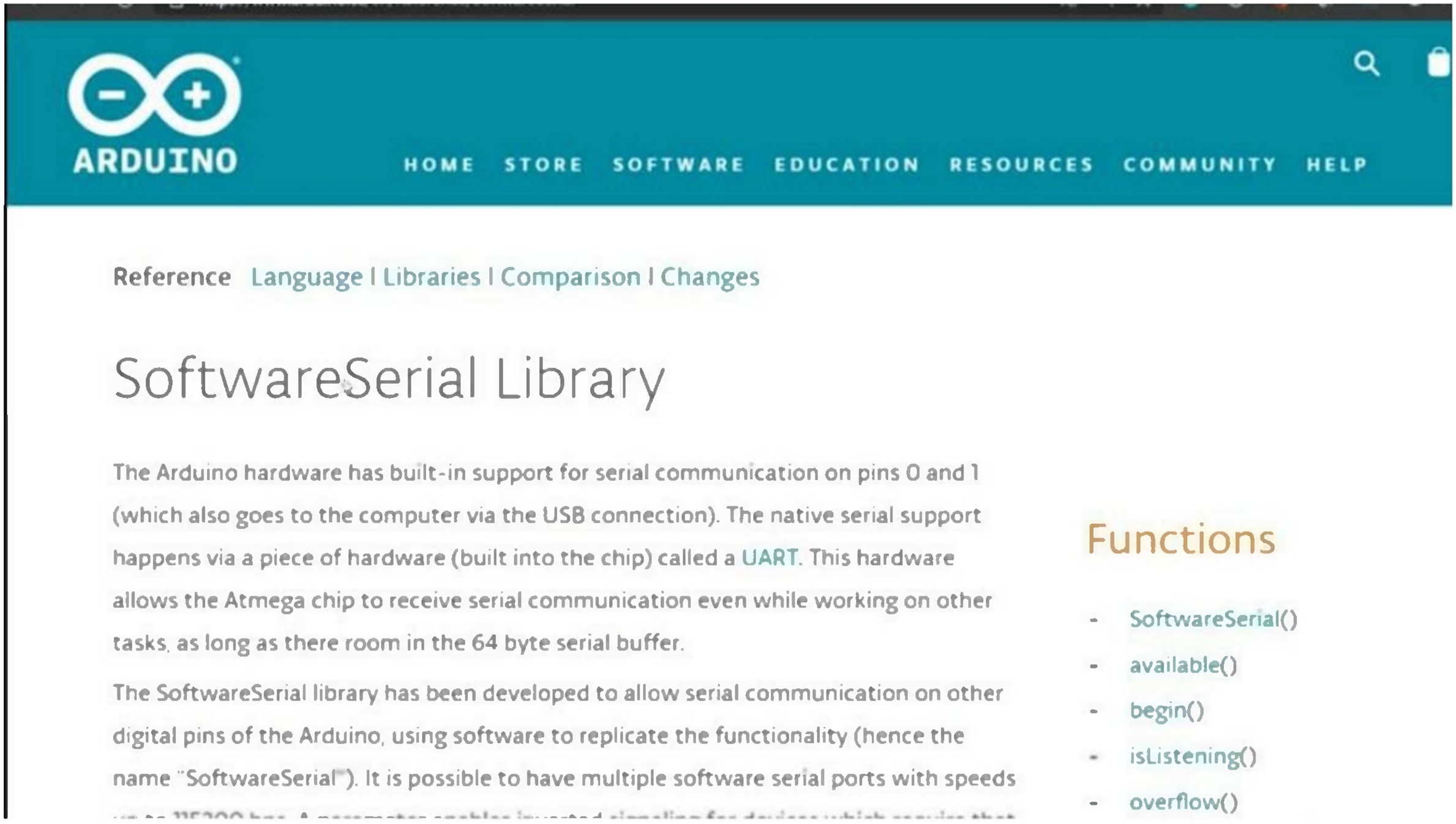
DIFFERENCE BETWEEN SOFTWARE SERIAL VS HARDWARE SERIAL IN ARDUINO

Hello and welcome to this annual lesson in which I'm going to explain the difference between software serial library and would we know serial. Now we all know that as we know hardware has built and supports forces communication on pins 0 and 1. And if you look here to this area let me zoom in. You can see that pen 0 and 1 has x 0 16 0 as labels. So this is the albino Syria. And this also goes to the computer via the USB connection then active Syria support happens via bits of hardware which is built in into that chip called Are You a art. This hardware allows us to make a chip to receive serial communication while working and other tasks. Now this does 0 and 1 pins. Serial is called hardware Serial which is explained here in this document and I have attached everything in this section. This Syria is basically communicating directly through these pens 0 and 1 are ex ante X and depending on the type of ball you can directly communicate with it and you don't have to assign bins for that.



So it's used for communication between the Arduino board and a computer or other devices all available to have at least one serial port which is a hardware serial and it's called serial and you have already seen it. If serial is available you can read from it you can print on the serial monitor and it's basically the Serial monitor that sends and receives that through your ISP. Now what is the difference between this hardware

serial and this software Syria. Now in the software scene the library has been developed to allow serial communication on other digital pins of the Arduino using software to replicate the functionality.

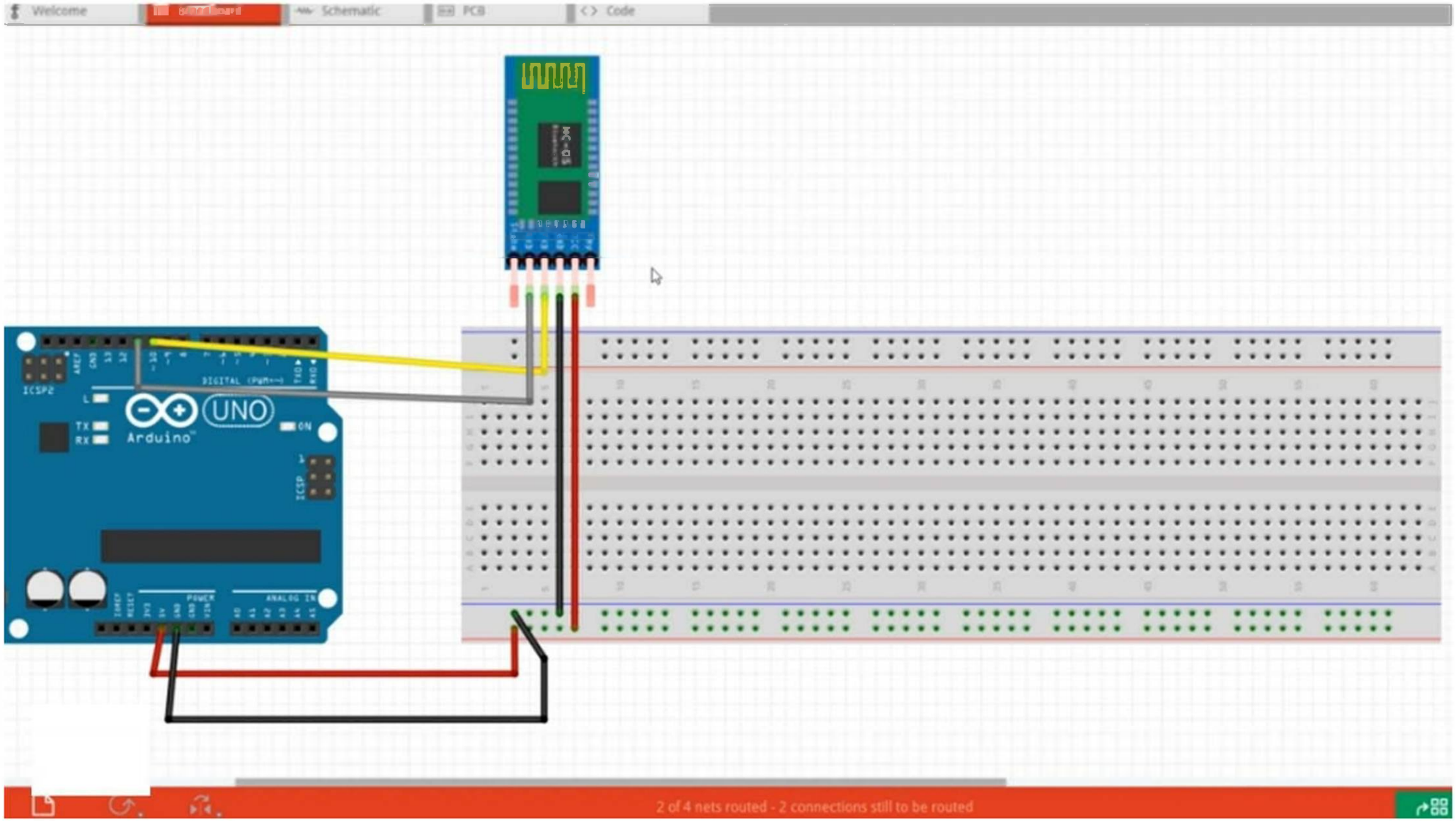


And this is why it's called Software serial. It is possible to have multiple or multiple software serial ports with speeds up to up to 11 or a hundred fifteen thousand two hundred feet per second. So what we all need to know is that this type of zero communication is basically a software mode that you can enable or disable so it has some limitations but I always include this in the lecture. I don't want to read them and there is an example but you need to know that there is a difference between this hardware Serial which is basically connected to zero and one pins in our arduino and this software serial that can be on any pin that you assign in the code. So the software serial isn't connected directly the way that they are sport. While the hardware series is connected they must be bought and it has two physical pins assigned for it which is child zero and one you can see here zero and one. While the hardware This is the hardware here while the software serial is basically a library that allows here communication on other digital bends of the Arduino it basically the use of a software library to replicate the functionality of the hardware serial to get more se-
rial ports from our Arduino. So this is the main difference: one physically exists or one does exist physically while the other does not exist physically. It's only a software replica of the original one. And when you are dealing with serial monitor and your Arduino IDC you are connecting through this hardware serial and you need to take that in mind and the next time you read or write a code and we are going to show this in our coding section but for now you need to know that there are two types of zero communication hardware serial and software serial hardware serial is labeled as RDX antiques on the Arduino ball which Arbenz 0 and 1 and is connected directly through the USP board while software serial is basically a software replicate of the hardware serial communication and you can use digital pins to send and receive serial data through the software serial library it has some limitation but it's a good replicate for the hardware Syria. Thanks for watching this lesson. If you have any questions please ask and if you aren't able. This is Ashley from the ed-
ucational engineering team.

FIRST CIRCUIT - AT COMMAND MODE

Hello and welcome to this new lesson in which I'm going to create the circuit that we are going to use to place the Arduino in 80 command mode. Now let's start by opening up rising as you know rising is a circuit design software that we are going to use to explain the connection for you know in order to place our adrenal in 80 command mode or in order to place our project in 80 command board. We need to connect the Bluetooth module in a certain way and overrate it so that it receives 80 commands from the Arduino idea. So let's go to the Arduino section or software. Now let's zoom out. Play Now we need to grab our adrenal gland. You can look for the Bluetooth module called C 0 5 We can light Bluetooth and as you can see here we have two modules. Let's try this one and look for another one as you can see there are a lot of modules in arduino and too easily access the one that we are looking for. You must click the right Bluetooth if you see. And here it is now. Let's rotate and zoom in. As you can see this is our Bluetooth module and this is our Arduino both now we need to connect more underground so you can get five false from here. To the board and the ground from here, change this block as you can see red and black. So now we have power in this. And now we can connect the ground to the ground. And five votes to form the PCC. Now we did the power connection. What we need to do next is connect T X and R X in this circuit. We need to connect X here to the software serial pen with the child which we are going to use in this case. And let's make it in 10 so connect Benton to t x. Let's show the coloring and connect being eleven to are X and let's show the cowl-

ing as well okay. Now the next step. You have two options. If you have the Bluetooth module with the key button usually there are some modules. Let me show you okay now if you have this module it comes with a button here. Then you don't need to connect anything else. All you need to do is connect the two powers depending on the tool to extend our experience. The key button will do the job for us and I'll explain this in a minute but if you don't have that and you have this module that doesn't have a key PIN then you have to connect pin 34 which is this pin the last one has two pin number nine in your board or to connect it to a button and a breadboard. So you have two options but I prefer going with this module the one that has the button because it will make your life much easier. Now let's explain the button thing as you can see this key thing. You either have to connect this key as you can see it's used to enable 80 commands to a PIN number nine or if you have a key here you don't have to connect anything if you don't have this pin nor have the button then you have to connect this pin Soledad to the wire and connect it to when number nine here why we need this pen why we need the key Ben why we need the key button there are all the same and we need them to place our project in the 80 command mode and how to do so it's fairly simple on what you need to do is basically press that key that exists in your Bluetooth module which is this one for a minute or five seconds after pressing it bludgeon your original ball to power supply.



Now once you do that you have to wait at least five seconds and you'll notice that the status lid is blinking in the two seconds interval not very fastly as it usually does. So once it starts blinking slowly in two seconds and tells Val you know that you are in the 80 mode and you can use the Serial monitor to send 80 commands if the Arduino did the response with ok it means that the 80 mode is on and you are good to go. You

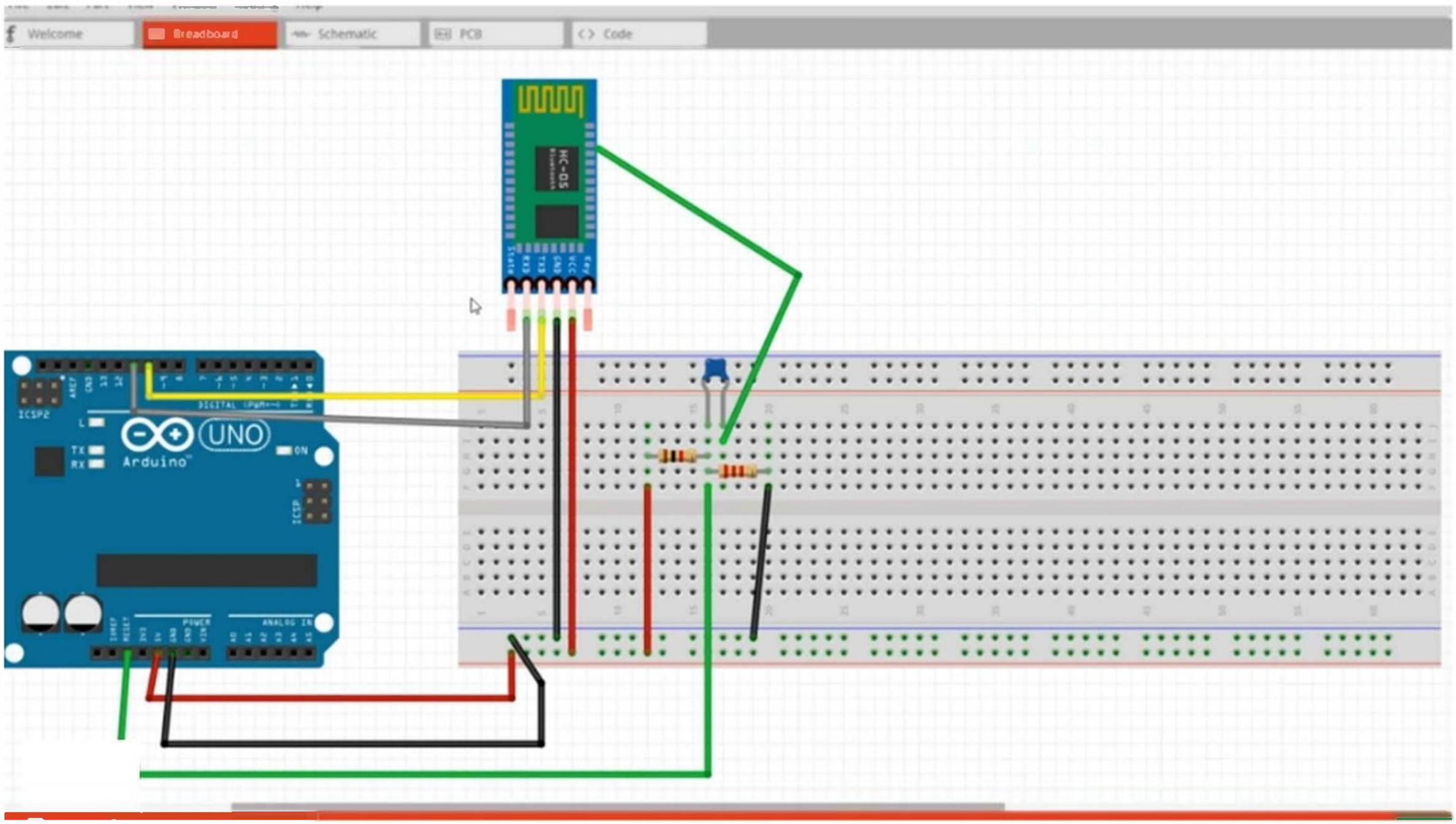
can start sending your commands again if you have this module you only have to connect the two about bins and the tutti accent are expands and this button will be used to place your project in the 84 month mode so you have to press it then plug in your Arduino with five seconds then remove your finger from that button and you'll notice that the status lid is blinking in two seconds interval then you know that you are in the 80 command mode that's it. This is how you can connect the circuit that will place the Arduino in an eighty four man board if you have any question regarding anything that asks that you want a board. This is optional from the educational engineering team.

SECOND CIRCUIT DESIGN FOR

FINAL USE PIN32 INCLUDED

Hello again. Now that we have placed our children on my board and since we have already sent all the 80 commands we need to connect our final circuit that we are going to make permanent using either a shield or a PCV. This circuit will include some additional connections. So we have the resources and ground connected here and we have the area ex ante X connected here and here to number two when Number 10 and 11 what we need to do now is basically connect being 30 to here in our Bluetooth shield and that pin must be connected to that asset. Ben and our algorithm argued why we would do this because we need the Arduino to reset the Bluetooth shield. Each time we upload a new code. So since Alwyn already has our set pin the Bluetooth shield will do the job and each time there's a new code uploaded that device will go and reset the board since the Bluetooth shield will send out a set signal to our admin. Now to do this we need to create a simple circuit. Basically it's a voltage divider circuit that will produce like three point three volts. So to do this we need two resistors. This is the first one and this is the second one. We also need a capacitor now that resistor. The first one is one kilo on the resistor. So we must change the value to one kilo on the second one. The two points to kill on these two resistors will take five volts underground so that kill on resistor must be connected to the five volts and the two points to kill on resistor must be connected to ground. Now let me change the coloring K black and red. I'm sorry. Now this cluster must be a hundred and four or point one.

Michael it's Sandra. For ceramic cover thought so you can change the value here but facing half time changing it. Okay now uh this is a ceramic capacitor so it doesn't have liberality. As you can see one of its legs is connected to the voltage divider circuit. And what we need to do next is to take a bend from here and connect that to it. Ben and our adrenal So here's here's the pen. That's Kenneth. That's it. Now the other side of the capacitor must be connected to Ben 32 here in our shield. And I'll show you this and the practical lesson but what you need to know now is that this pen is connected to Ben 32. Let's give it a green color. Okay now our circuit is abundantly what we have done here is that we connected that status pen this pen Ben. Number 32 in our Arduino on our Blue Shield to a capacitor and the other leg of the capacitor is connected to the voltage divider circuit that provides three point three Volt. And we took the Ask a line from this area to that is it.



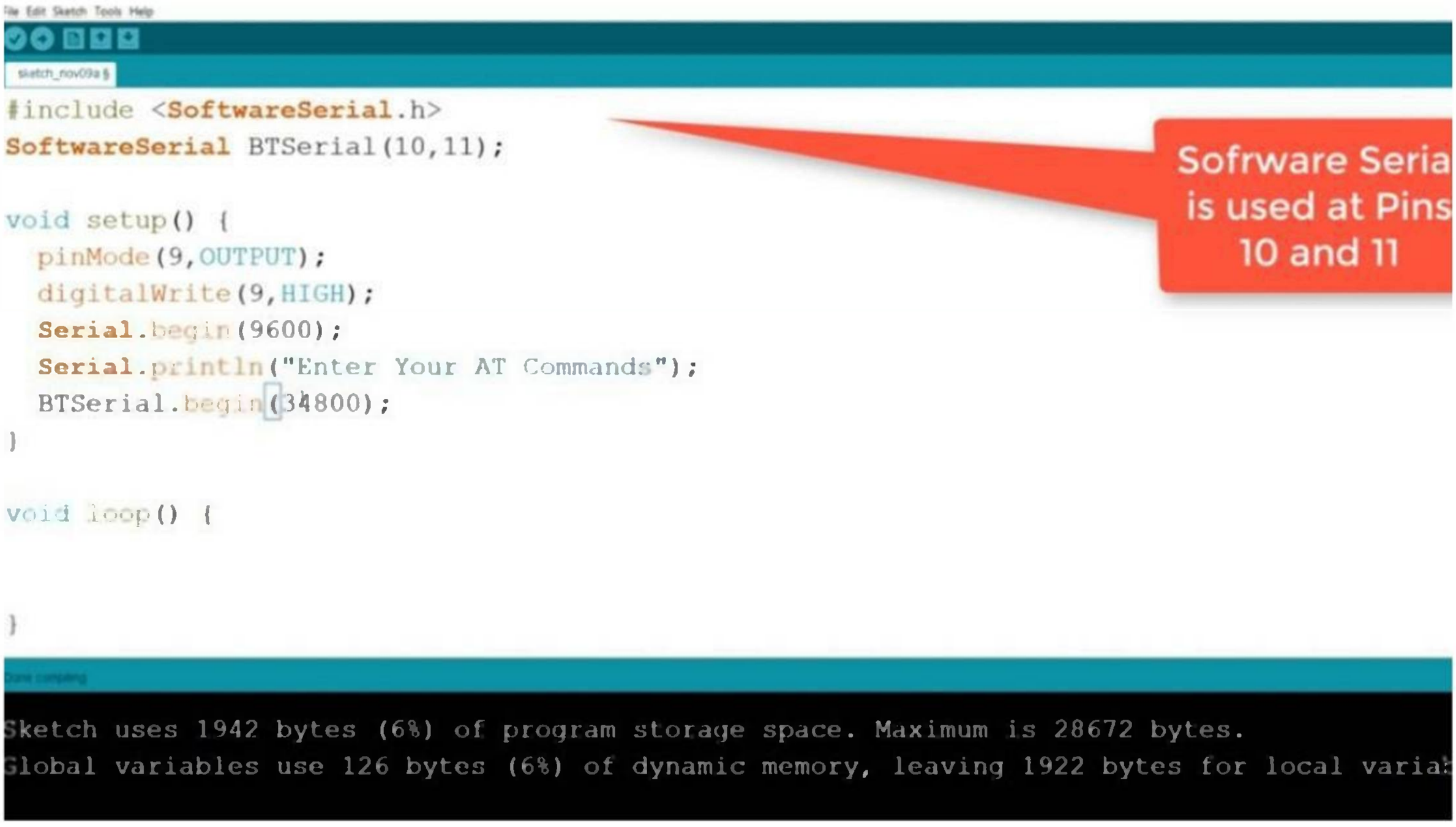
Ben and Arduino basically this will maintain Aricept oil and will send a set signal to Arduino each time there is a new code or a new code is uploaded to the board. So instead of manually setting the module on the project each time this will automate the process and it will make the wireless programming aware much easier. I will show you this connection in the practical lesson but that's it for now. If you have any questions please ask if you aren't able. Thanks for reading. This is Ashraf from the educational engineering team.

PROGRAMMING ARDUINO FOR PC

WIRELESS PROGRAMMING

Hello and welcome. Now before we start writing codes wirelessly to our Arduino we first need to program our Arduino with one single code so that we can easily edit or send new codes to that Arduino. So let's start by opening up Arduino IEEE in this code. We are going to initiate the Bluetooth module and send and receive some 80 commands using the Serial monitor. So this creates a new project now inside our project. First we need to include the software serial library software side of the text. And after that we need to initiate a Bluetooth module using the software Syria. So we can just copy the software and best it here and let's define the pins for this Bluetooth module. We are doing all that they are connected to pins number zero and one that we need to create Bluetooth. So we look here. So to our best schematics in the fly zone area we can easily identify two bands of these pins. This is the software that we talked about recently in Atlantic City software and hardware sizzle on the song 0 0 1 Bluetooth are going to connect to pay. Now after 9/11 you can compile your code to make sure that everything is Now let's remove these lines as comments and we don't need them in the setup. We need to assign bill number nine which is connected to the key to output cause this pain will pull that c 0 5 pin 34 which is the key being too high to switch this module to a T mode. Now we will send a high signal to that pin. There is that right pin number nine high. And we will solve the Serial monitor for Arduino to see what's going on. Using serial begin and what about the rate of 9600 hand

let's print someone something something in a serial murder print. And uh let's ask the users to enter your 18 months after that. Let's start that Bluetooth serial communication using the begin function and let's set that C default speed in 80 commands to three thirty four thousand eight hundred.



You can change it later if you don't like all of you or thirty eight thousand four hundred. This is usually the default but you can't change it with whatever you want. Now this is what we have done in the setup. We are assigned the output pin which has been number nine. That's connected to the key button and our Bluetooth shield. And we have sent the high signal to switch that module to 80 commands or 80 modes. We start with the serial monitoring and we ask those up to enter 80 commands. After that we started the Bluetooth communication at three eight or thirty eight four thousand thirty eight thousand four hundred baud rate. Now in the loop what we are going to do is ask if there is Bluetooth communication available then we need to read it and write it on the Serial monitor. So if Bluetooth Syria is not available then right whatever we receive to the serial window we are going to put Bluetooth serial 3D function. And again if serial is available let's go with this and Mr. T's if Syria is available then use the Bluetooth. All right function to read serial that's what this will do is the following. This will keep reading from XY 0 5 and send to Arduino Serial monitor while this F will keep reading from Arduino Serial monitor and send to each C0 5. So even if we don't have a computer connected this means that whatever code that was sent via the Bluetooth module will be directed to the Serial monitor and whatever code that we write to the Serial monitor will be sent to the Bluetooth module. So this means that we have both way or two way communications between the Bluetooth and the other no. So whatever we are connecting to that Bluetooth or that it's computer or phone it will send data and added we all receive it and disobeyed and the Serial monitor and whatever code that we send from out below zero monitor it will be sent via Bluetooth to the other device which is a phone or a P.C. so this is basically the code that we need to upload our Arduino board before starting the 80 command board after uploading that code. We need to do some 80 command manipulation and we will cover that in the next lesson. But for now that's it. We initiate that software serial for the Bluetooth module and we have assigned the pens that we are connecting the Bluetooth module to. We assign the key pin which is con-

nected to be number nine. In addition to output, we have sent high signal to make sure that it's in 80 mode and we started the zero communication and Arduino using 9600 both rate and we started the Bluetooth module at about the rate of thirty eight thousand four hundred. And here we are basically asking the Arduino to keep breathing from the Bluetooth module and send data to see a monitor. And here we are asking Arduino to keep breathing from its own arduino and monitor and send the data to the c 0 5. This is for the average record regarding that basic wireless communication and in the next lesson we are going to cover the 80 commands that we need and how we can send them. Thanks for watching this lesson. This is Ashraf from the educational engineering team.

REQUIRED AT COMMANDS TO GET ARDUINO AND BLUETOOTH READY

Hello and welcome. Now in the previous lesson I showed you how you can connect your Arduino using the first circuit design by hooking up ground and VCR C T X and other expense and the most important pain which is the keep in pain 34 and Bluetooth shield to your Arduino digital pin. After setting up that connection when you're pretty bored or on your PSB and before connecting your argument not your computer you must remove the power from the edge c 0 5 Bluetooth module by simply disconnecting the PCC pin after that go ahead and connect your Arduino to our computer and then block back the PCC pin to the Bluetooth shield. This would have vault your c 0 5 in 80 command mode after booting it an eighty command mode you must open up the serial terminal and into the following commands. We will enter them one by one but for now let's take a look at eight months.

1. Connect PIO11 to high level.
2. Power on, module into command state.
3. Using baud rate 38400, sent the "AT+ROLE=1\r\n" to module, with "OK\r\n" means setting successes.
4. Connect the PIO11 to low level, repower the module, the module work as server (master).

AT commands: (all end with \r\n)

1. Test command:

Command	Respond	Parameter
AT	OK	-

2. Reset

Command	Respond	Parameter
AT+RESET	OK	-

3. Get firmware version

Command	Respond	Parameter
AT+VERSION?	+VERSION:<Param> OK	Param : firmware version

Example:
AT+VERSION?\r\n
+VERSION:2.0-20100601
OK

The most common one here has eighty commands. And if you got a response which is okay means that the 80 mode is on 80 plus is set to reset the 80 command or the 80 model 80 plus version. Question mark will respond with the version of the device or firmware that you are using then it will. Right. Okay. 80 Plus or G L which is the first command that we are going to send here. In our example is basically setting your device

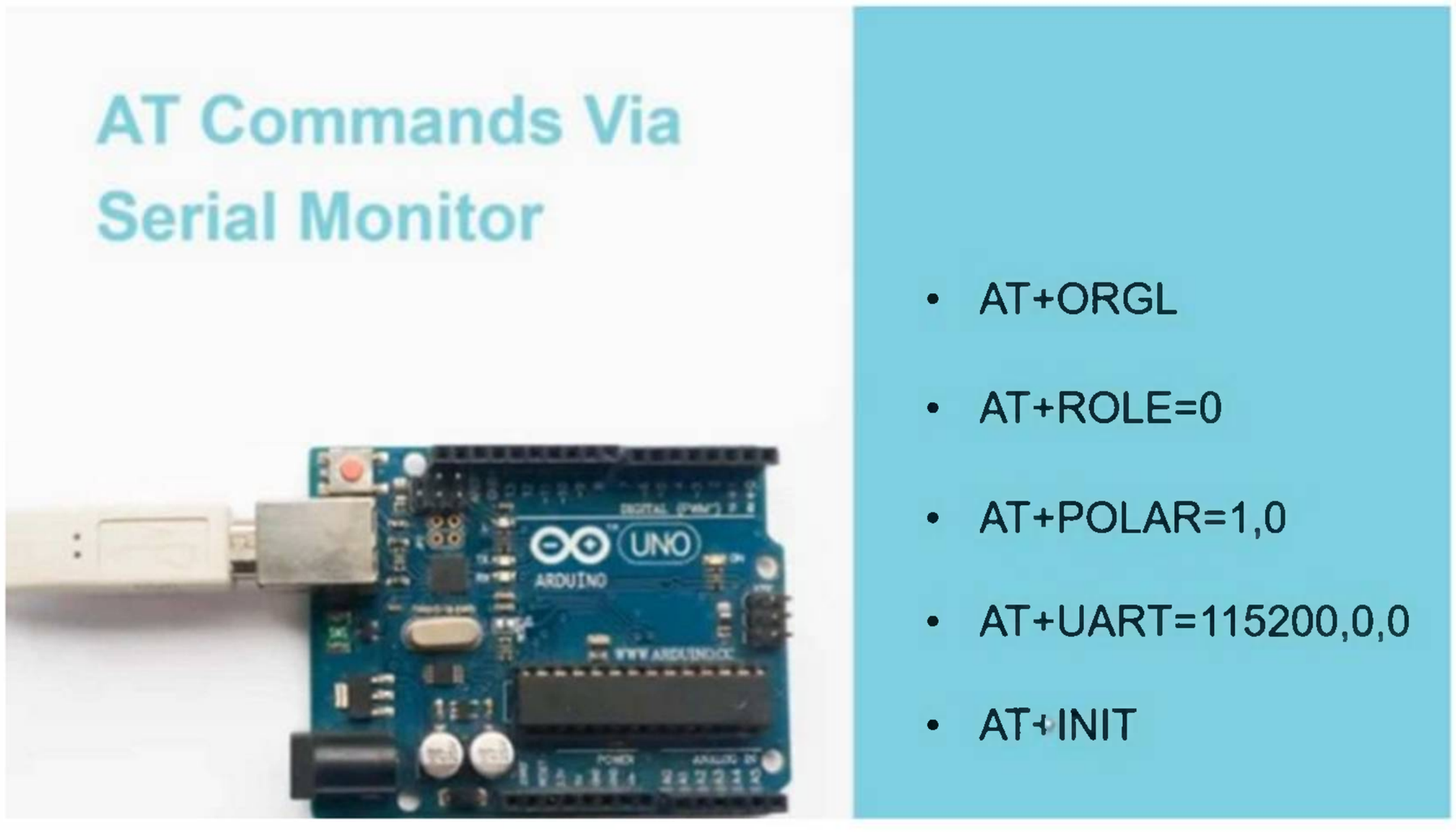
as a slave mode with being called one two three four and the device name is for the default. See 2010 6 1 with a broad rate of 3 thirty eight thousand four hundred bits per second. So when you write this line it means that you are doing this okay now in order to do this. You must first go and open up the arduino. I already mentioned that you need to program your Arduino and recover that and Arduino coding section and after uploading the code to your Arduino you must go ahead with this step which is basically opening up the serial monitoring. Now ignore this code you must open up the serial monitor by clicking here. And since we don't have a connection as you can see we must connect our ball and we will do this in the practical now the full commands that the four commands that we need to send are these commands 80 or geo. And we already mentioned this command. It means that by restoring defaults it will restore your device to the default settings which is setting it as a slave or in slave mode with a PIN code or password 1 2 3 4. It will give it the default device name and it will set it to both rates of thirty eight thousand four hundred. The second comment that we need to send is 80 plus R or E equals zero. While this command does sit or check module mode. So depending on the module Mode this will respond with okay and some parameters. Now if we send or if we have sent 80 plus full equals zero This means that we are setting our device as a slave. If we have seen one. This means that we are setting our device as a master. If we have sent slave Lou this means that we are setting our device in a slave lube which is a continuous slaving globe. If what we need to know is that 80 plus Ron equals zero means that we are asking Arduino to set the Bluetooth device as a slave device. Now the third command that we need to send and we must send them in the same order 80 plus Paula equals one and comma then zero. Now what this command does is simply set chick on a set or a chick lit. Input Output. So if we have a field locked here you can see that it takes two parameters. The first one is for the p i 0 8 drive list so the office sends one. This means that API 0 8 high drive lead is assigned now for the second parameter; it's for the p i 0 9 low and high drive lead.

AT Commands Via Serial Monitor

- AT+ORGL
- AT+ROLE=0
- AT+POLAR=1,0

If we have seen 0. Just like in our case here. This means that API 0 9 low drive lead is selected. So these are relevant to the Bluetooth shield and the lid. Input Output check and you don't need to know much about them other than this other than the fact that this will assign one of them to high and the other one too low. You can look up the sheet and look for p i 0 8 and 0 9 to see them and to take a quick look at their effect in

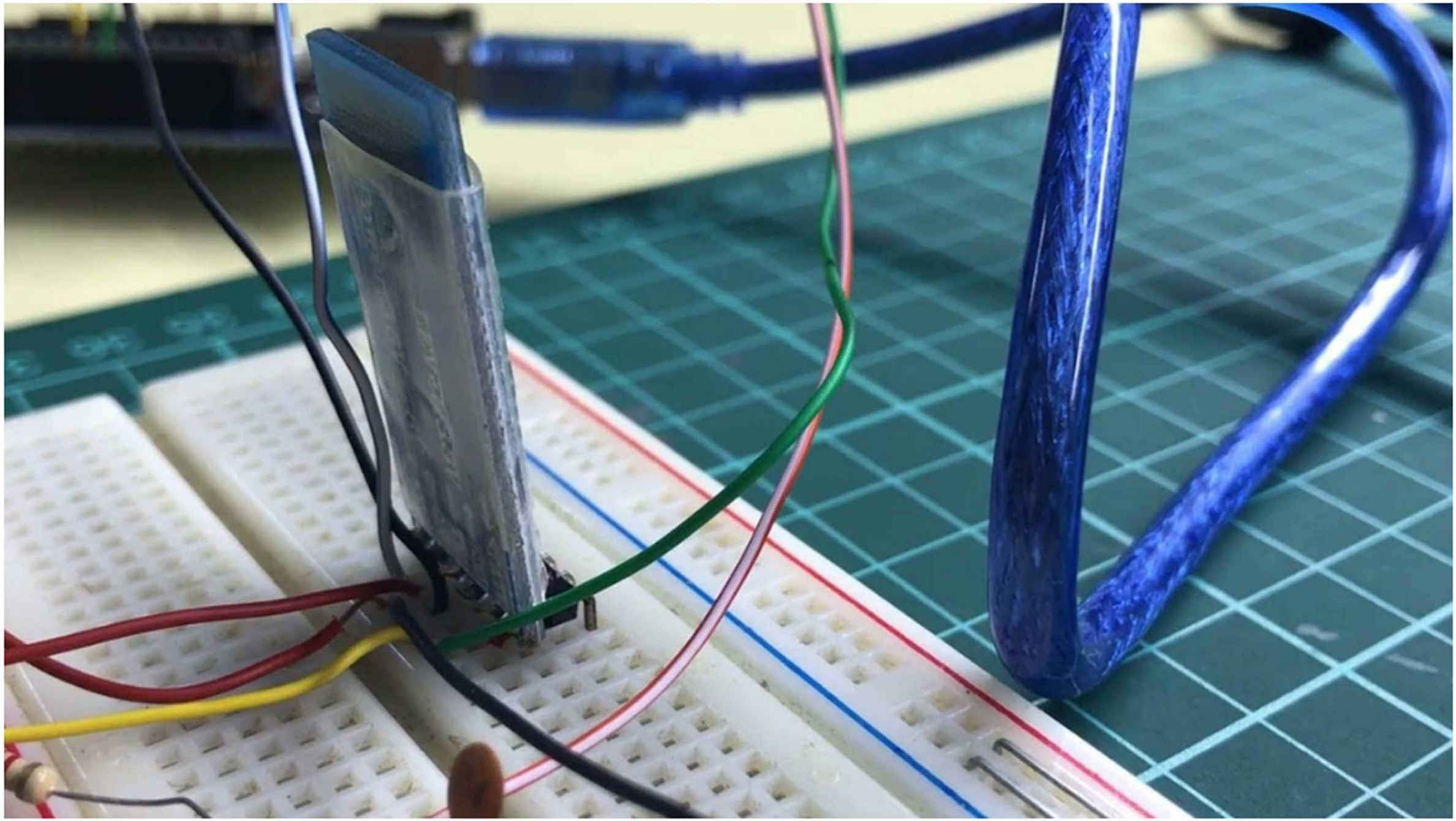
our code. Now the next command is 80. You asked a hundred and fifteen thousand two hundred zero zero. What this command does is simply sit or check serial parameters. So it takes three parameters. The first one is the board rate. The second one is the stop bit. And the third one is the parity check for errors. So these are the three parameters that we send using that you are at command. Now we come to the last parameter which is 80 plus in it which is initialize and what this command does basically is initializing the US BP profile library. If you get a response with okay it means that everything is done correctly. If you received a fail it means that something has been done wrong or something is missing with the connections so you need to check your connections and start from the first command going down to the last one.



So make sure that you follow these commands one by one and if you haven't. Make sure to check the previous lesson in which I placed all of the 80 commands with a brief explanation for each of them and with some examples to help you understand. Basically what 80 commands does is put our arduino and Bluetooth shield in a mode that cannot receive data from a computer wirelessly. And this is the whole goal from this course. So follow these steps and we are going to apply them in the practical section but for now this is what you need to do to get your Arduino up and ready for communication with P.S. wirelessly. Thanks for reading. This is Ashraf from the educational engineering team.

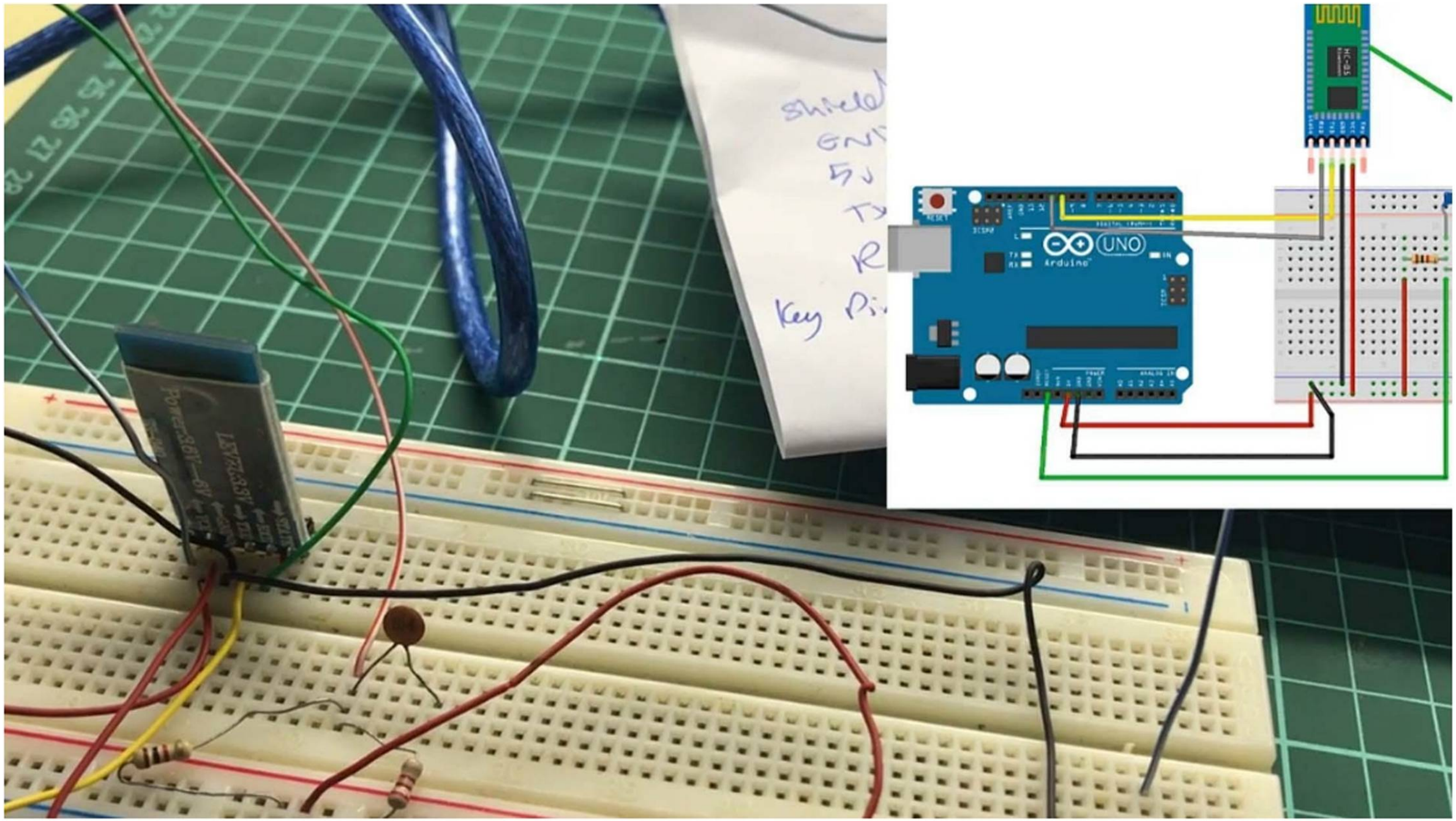
PRACTICAL ASSEMBLY AND SENDING CODE FROM PC WIRELESSLY

Hello and welcome. Now this is out of Grenoble and this is our Bluetooth module. I have connected. Let me show you. I have connected these pins. The red one is the power. And the black one is the ground. I have connected them to the other green board out which is theirs here. And they have connected the dots ex ante ex. To Ben's numbers 10 and 11. And we're onboard and now we are going to put this aluminum ball in the 80s command moon. Now in order to do so I have to block this USP while basically holding this button. We have a button down here as you can see this little button, this one I have to click on. Then I must block the USP while holding that button in place for at least five seconds. And when that leads and the Bluetooth module starts blinking. If the two seconds I know for sure that arm and the 80 command mode so that I can start sending eight months. Now let's do this. Let me hold the phone now. As you can see if we move this back a little bit I must click on this button. T this lid is banking every two seconds.



And this is how you know that you are in the 80 command mode. Now you must go back to our laptop and install these commands. Let's go to that. As you can see here let me turn this off. The key is after saying the Humboldt as you can see is asking guests until eight commands you must make sure that both an aisle and C are checked and you have the moderate 9600. Now let's start writing the 80 commands. The first one is

80. Click and tell as you can see OK means that we are in the 80 command now 80 plus or are G L to return to factory default 80 plus wrong to set the slave and Pastor mode and the already explained all of these commands and it feels E equals zero. And 80 plus more love. Equals one zero eighty. Plus you A R T equals the board rate and the bit. Now we have the 80 plus initialization then as you can see we have four or five bouquets which means that everything is fine and we have written all of these commands correctly. Now the next step will be smoldering being number thirty two and connecting that to the resistance undercover store circuit. Thanks for watching this lesson. See you next time. This is Usher from the educational engineering team. Now according to this schematic we need to connect pen 32 with the carbon stored. One hundred four point one microphone out capacitors ceramic capacitors. So it doesn't have any polarity. We need to connect to a voltage divert the second that has two resistors one point one one kill on two point two kill connected to B C underground and then from the mental point here we must take a wire and connect to that is set in the r Gwyneth side. So this will be the new edit which is the second permanent circuit you can hook this using a grid board or using a PCV shield. Now let me show you this in action. As you can see here we have the two resistors one is connected to ground the other one is connected to five volts and they meet here in the middle just like in our circuit here. These two meet here in the middle. Now this is our capacitor. It's also here and we were well Soledad a wire from this module to the other leg of the capacitor and at the end we will take another wire from here just the middle point that has two other systems and one to the store and we will take that wire as you can see here to the reset pin and all Arduino board and our Arduino board will be ready for Bluetooth programming. That's it for the Second Circuit connection. If you have any questions please ask in the background aboard.



Again you have to follow the First Circuit connection and to press on the button in your Bluetooth shield to put it in 80 commands and you need to enter 80 commands. After that you can apply what we have said in this lesson. Thanks for reading. This is Asha from the educational engineering team.

CODING ARDUINO FOR MOBILE

WIRELESS PROGRAMMING

Hello and welcome to this annual lesson in which you are going to write the code required for Arduino to start receiving codes from a mobile device. So let's start by initializing the serial communication with a broad rate of thirty eight thousand four hundred which is the default rate. Then let's add some delay for the board rate or for the cell communication to stabilize to make sure that we are receiving correct data. After that we are going to send some 80 commands and we already covered that in the previous section. So let's start by sending the first command serial the print line and inside these two parentheses we are going to write 80 plus names and let's name our Bluetooth module. Let's call it B wireless and give it a number. Let's add a hashtag. 0 0. Now let's add another delay and let's say double the rate. So let's write the same sentence with the delay to avoid wasting time. Now the delay can be set by using an art command and you must. It will take three parameters. First thing is the board rate. We will use one hundred fifteen thousand two hundred zero zero the first. This board rate is used when we are dealing with not all blue we know. And bingo.

```
sketch_nov0[illegible]

void setup() {
  // put your setup code here, to run once:
Serial.begin(38400);
delay(500);

Serial.println("AT+NAME=BWireless#00");
delay(500);

Serial.println("AT+UART=115200,0,0");
delay(500);


}

void loop() {
  // put your main code here, to run repeatedly:
```

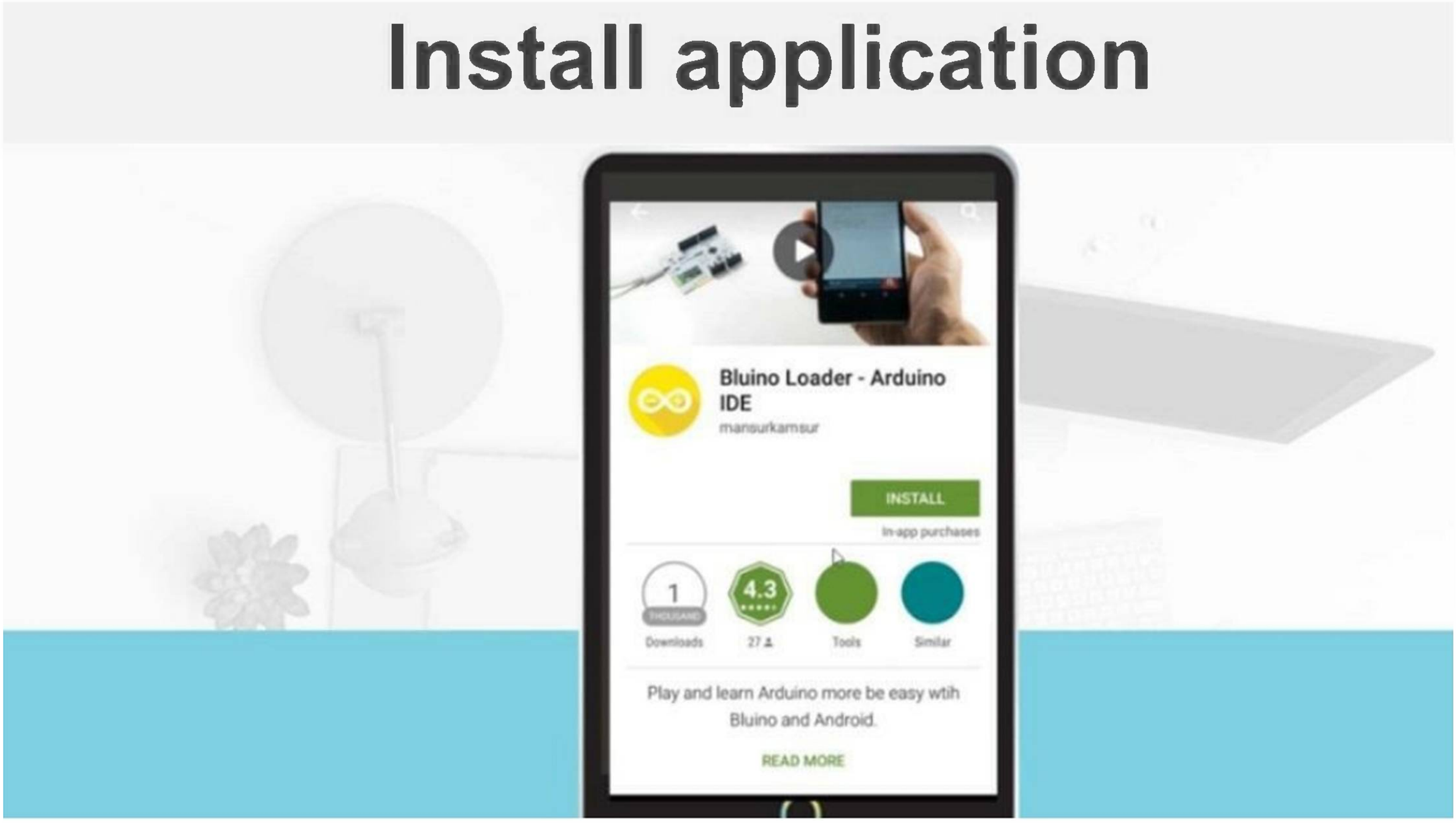
But if we are dealing with nano Leonardo micro or Pro MIni we must change this with seventy five thousand six hundred. So we must use this number but in case we are dealing with all mega we can use this number. These two bits are for Perry to check. And for even or odd. And we must add a delay after that so that we will wait for this order to be sent. Next step will be to set our ball out. And we already explained

this command in the 80 commands section. So just copy this and base it on 80 plus boola. These are for lead check tests. And it will take two parameters. One means high and zero means low. This is for the first latest check. And this is for the second one. And after that we must add a delay. This is the code. It's a very simple code that we need to send. First we initialize the board rate for the zero communication. Then we were assigned a name. We have set the board rate for value art communication and we have set the ball out at the command. Now what we need to do is simply upload the code to our Arduino board and move on to the next step. You can change the password if you don't want to use the Thunder glass world by simply copying this line and moving this sentence and changing it with P.S. WD. And choose a four digits password with four zeros for example. So now whenever you connect your adrenal Bluetooth shield wirelessly you have to enter four zeros. This is how to change the password or the default password. After this you need to hook up the circuit as already mentioned and follow our practical section of steps. Thanks for watching this coding lesson. Make sure to upload this code to your Arduino before moving on to the next step. Thanks for reading. This is Ashraf from the educational engineering team.

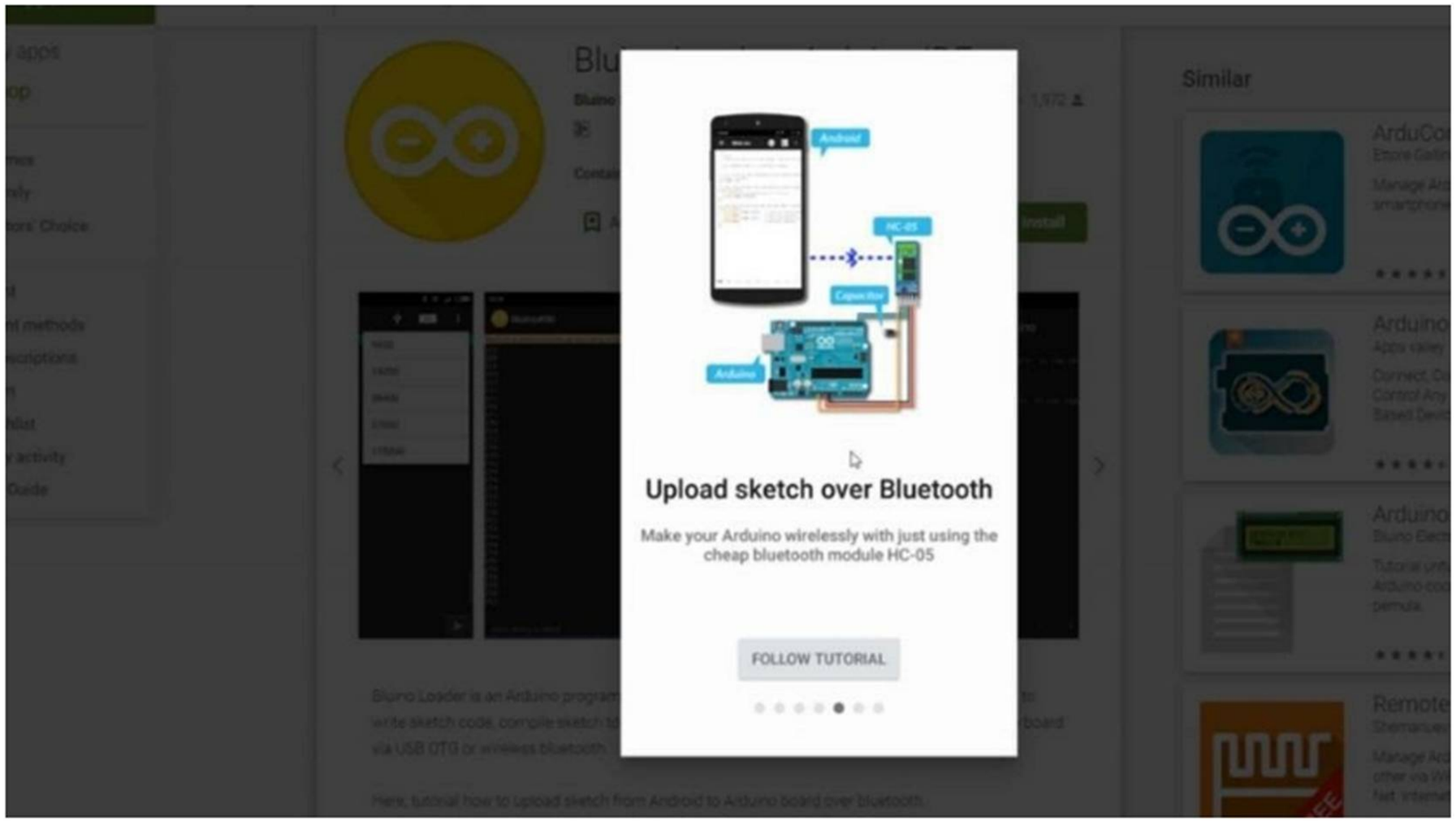
HOW TO PROGRAM ARDUINO USING A MOBILE DEVICE BLUINO LOADER

Programming Arduino using mobile now in order to program your Arduino using mobile. You have to install the Blue Willow loader which is the Arduino idea E for green boards. This app lets you play and learn Arduino more easily with Pluto and Android. Now this is the first app that I want to show you and it's for Android 4 or higher. It has a Bluetooth connection built in and you can get it from google play store. I will show you that in a minute and it's available for free. And there is a paid version. It's called Blue Willow loader pro. So depending on your needs this app is for working with the Arduino Ono. You can write, edit and upload sketches to Arduino Ono over bluetooth 0 5 the app is easy to understand. It's just like the Arduino idea. There are many example sketches and we are going to use one of them in this course. Also you can add some libraries by just copying and pasting the library folder to the folder of proving loader libraries. Your Arduino or Android file manager. So if you have a library that doesn't exist and blue in order just go to your file manager to look for blooming order and inside it look for a libraries folder and based your library there you can change them. A lot of things in this app you can change the theme, the editing color, the text size and a lot of other features as well. If you want to get a serial Monitor feature to remove the ads and that inside the app and to get a more premium feature you can buy the full version but for this reason

and for our examples we won't need it. And again you can simply upload a sketch you have done over Bluetooth so it doesn't require a fourth.



Now let me first show you where you can get this AB. Now search for the Google Play store, then go inside that Google Play store and buy Lowenthal Lauder. That's it. BLUE IN A loader as you can see it's an Arduino Aidid, a clone for Android. And this is how the Cold War Logs inside your Android device and it has many screenshots. This is for choosing the type of ball that you are connecting with. And this is where you can find examples. We will show you this in action. In another lesson. But for now let's check these properties. As you can see this is a serial monitor. It's exactly the same as the already there device. And as you can see here it shows you how to connect an Arduino with your Bluetooth module so that this app can categorize it and send codes to your Arduino board wirelessly.



Now again it has a lot of examples where you can use it in landscape mode. And here is a quick description for the app itself. As you can see blue blue in a low loader is an aluminum programmer software that runs on Android. Make it easy to write sketch code, compile sketch original text files and upload it to blue wino or various Arduino boards wirelessly or using a USB or 2G cable so you can also use our converter from your ISP to renew us B to upload that code as people. Now hazardous materials and it has other features that are listed here. I already mentioned most of them so you just need to click in a store after signing into your Google account and you will have the app on your Android device screen. Now that's it for the blue in the loader. Let's go back and check what we should do once we have the app installed. Now once you have the app installed you must open it up and go to the lift menu. And from here we need to upload samba sketch blink code into Arduino. So once we start extracting the tools and we choose an example to upload as you can see here we have a new file and open file differences to change the settings and info for the app. We can open up the blank AB and as you can see this is its code. You can write it yourself or you can open it up from the examples and I will show you that in the practical lesson within which we are going to program Arduino wirelessly. And as you can see it's simple and easy after that you need to scan for your own and Arduino device. It will look like this Bluetooth device.

Try upload sample sketch Blink.ino into Arduino

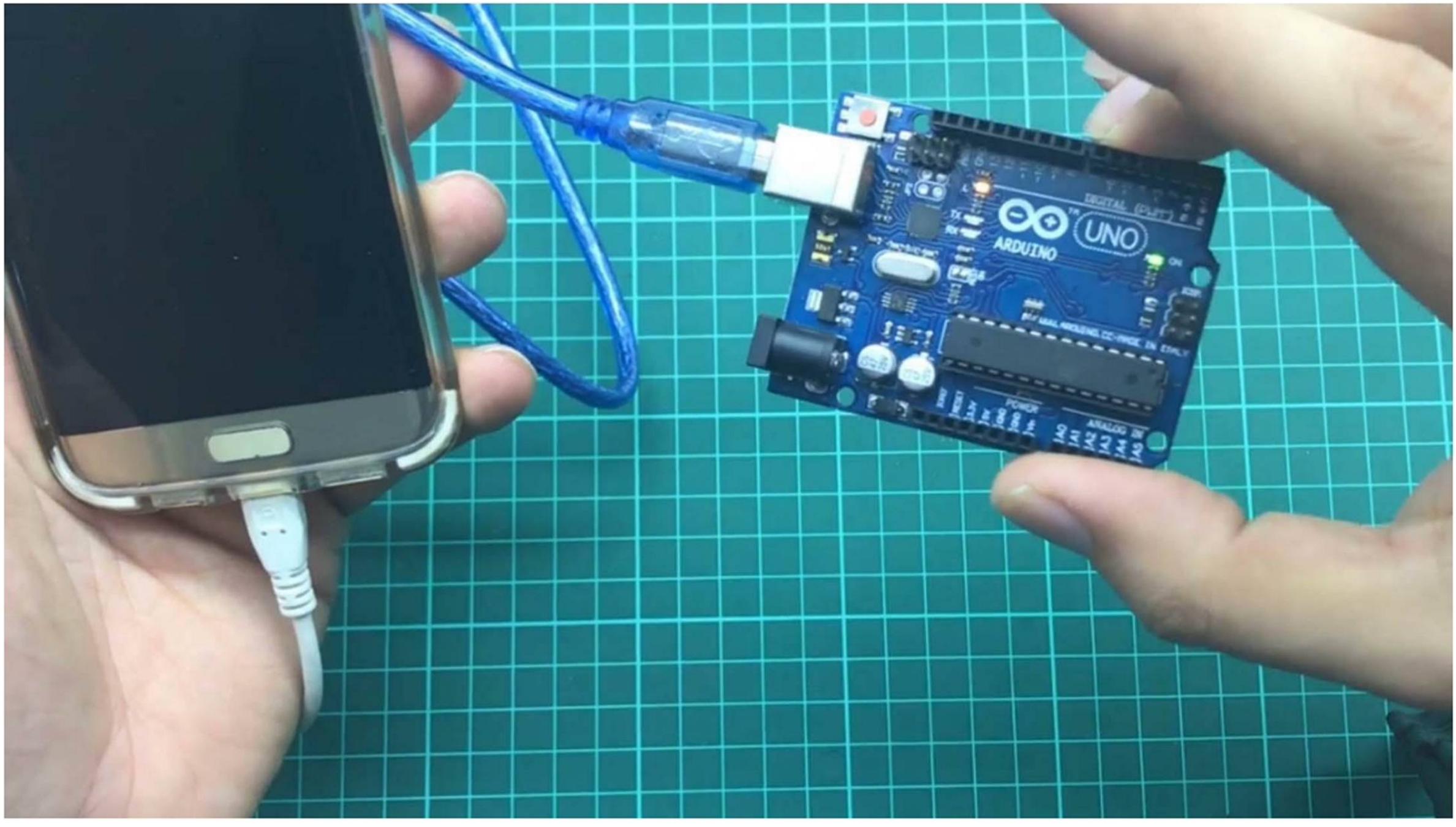
After scanning for it you will find it and you must enter the PIN code click Okay. And voila the code will be on your Arduino in no time. And this is it. This is how easy it is to use this app to upload a code to your Arduino board. It's easy if you don't have Bluetooth or if you don't like to program it wirelessly you can bring you out with no use or T.J. which is basically a converter that converts from your ISP cable to a mini

US cable that exists in most of today's phones. If you have a problem connecting your mobile device to your Arduino you can send me a message and I'll send you an image running for the TV converter for now. Let me just show you the autistic USB cable how it looks now just right or T.J. USP if you looked for the images as you can see this is a cable that takes the USB cable of Arduino or and converted to a mini US cable that can fit inside your mobile device. Now there are a lot of virgins. If you have a USB type C and your Android device you can buy the one with usb type C head or any other type. This will help you program your adrenal using USB cable and a mobile device without having to buy a computer. So this is a third option that you can consider when programming at will. For now that's it. In the next lesson we are going to show you this in action. I will install the USB and I will show you step by step how to open up the blank example and how to search for your Arduino via Bluetooth or your ISP and how to upload your code. Thanks for watching this lesson. This is Ashraf from the educational engineering team.

PRACTICAL PROGRAM ARDUINO

WITH A MOBILE VIA USB

Hello and welcome to this new lesson in which I'm going to show you what you need to start programming. How would we know using it on your mobile device? So we need to outline a board. It's obvious because this is the main goal of the discourse program. This little thing. We need a mobile phone with Android. It can be anything. We also need our USP or 2G cable. As you can see this will go to our mobile device and this will go to the Arduino cable and we need an Arduino cable as you can see so first things first. You must first blog your unwinnable. Using the normal USB board. Now let's leave this aside the other side of the US people must be plugged to this boat e.g. cable. Now we have our protegee cable ready. So we must blog it to our Android USP

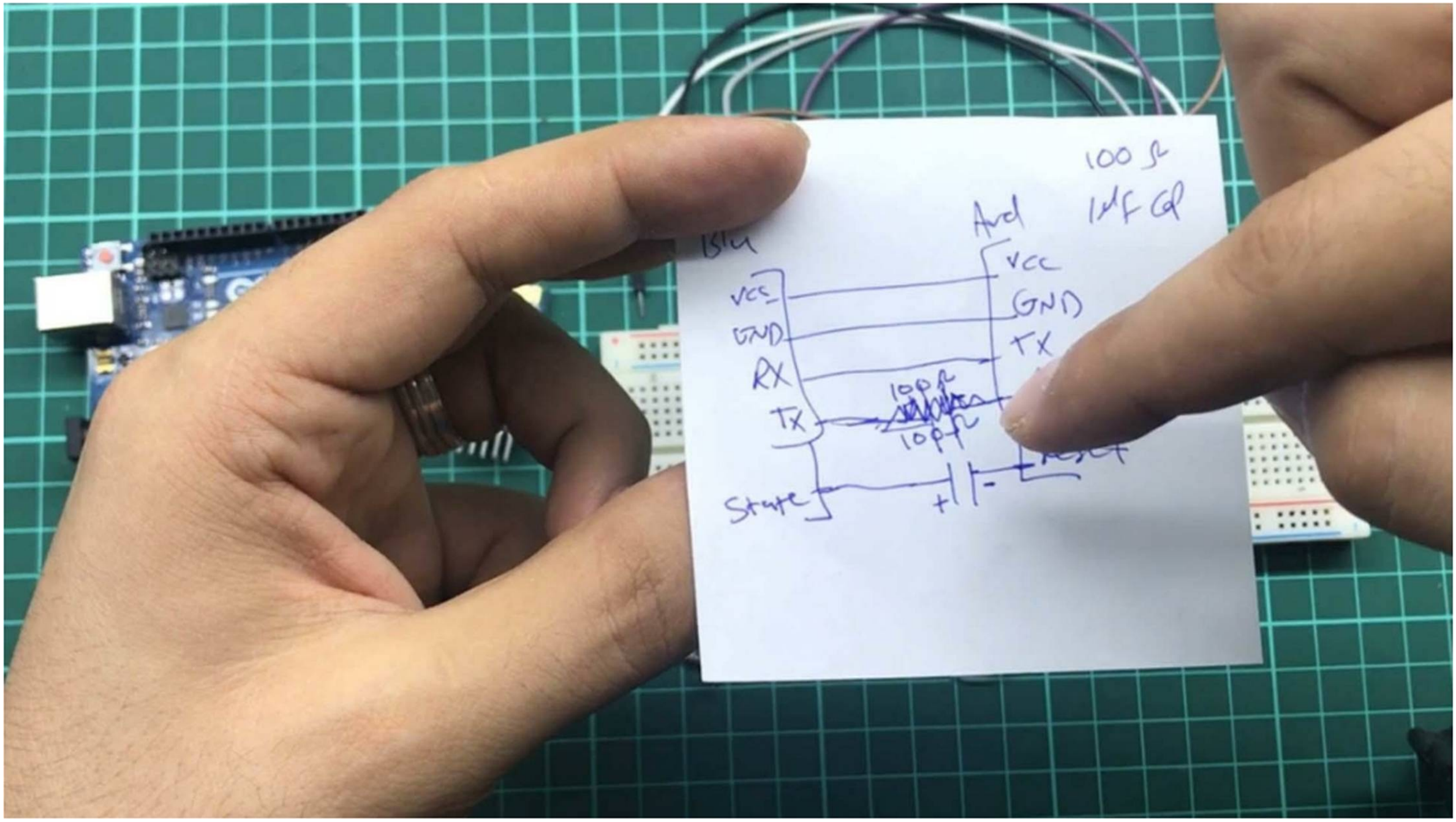


Now once plugged in you can see that the Arduino board is bound as you can see we have some cables we don't have a power source for. It's taking the board from the mobile device. So what you need to do now is simply make sure you go and look for blowing up which is the app that we explained how to download early. Now go to the preferences and click the board section from here toggle this to the other side to choose

the USB connection and choose your ball time. We are using all the boards. This is the first step. Now the drag from the left creates an open file and clicks on examples. Basics choose the blank. Example this is our blink. Example as the golden sweep in 13 out both high and low with a delay. Now click here to combine and move the cord to your Arduino. Now allow the blue in the loader to us that they must be OK now done uploading blink to our. And as you can see here and our Arduino board that it is Blink England 13 so that's it. As you can see we just brought this code and our Arduino boards blinking. We have done this in no time. Now if you are facing a hard time finding W.A. AB It's simple. You just need to go to the play store and write Luciano as you can see this is that we are using blue in loader out of the idea. It's a free app. You can install it and use it just like I just explained. That's it for this lesson. As you can see again it's blinking and it's taking the about from our mobile device so you don't need a computer anymore. Thanks for watching this lesson. If you have any questions please ask and if you aren't able. This is Ashraf from the educational engineering team.

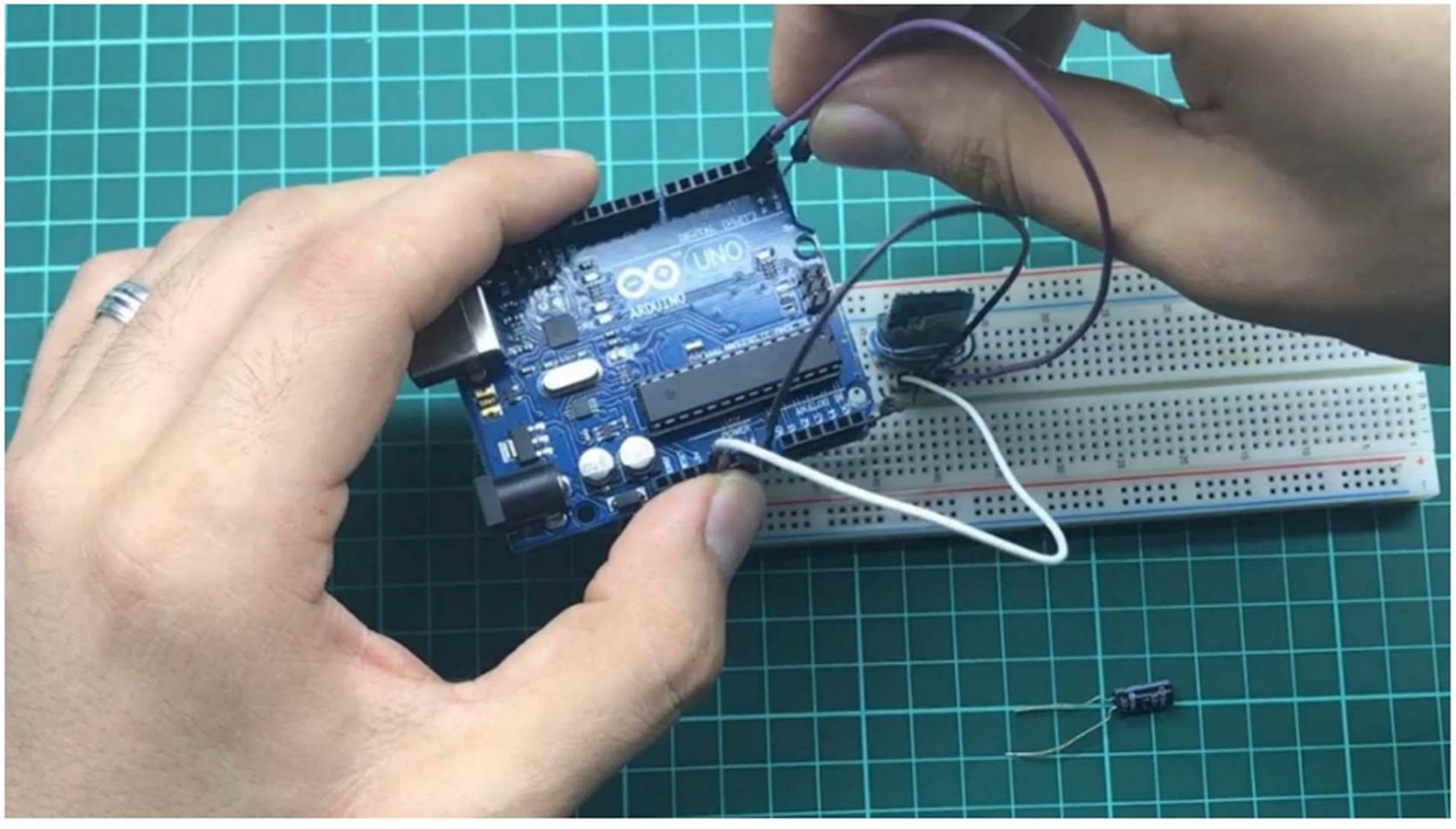
PRACTICAL PROGRAM ARDUINO WITH A MOBILE VIA BLUETOOTH

Hello and welcome to this annual US one in which I'm going to show you how to connect your albino and get ready for wireless transferring of the cord. So what you need is to be aboard a Bluetooth module as you can see this is actually 05. Some cables to connect your bread board to place your components: a capacitor and one resistor 100 on resistor. I added two sixty and forty on resistors to get 100 or so. These are the main components that we need and the circuit that we are going to connect is shown here. As you can see this is the Bluetooth side and this is the Renault side. This is here on the ground will be connected together to an argument on Bluetooth extended to X will be connected together directly while R X and the Arduino will be connected through 100 on resistor 30 x and the Bluetooth module.

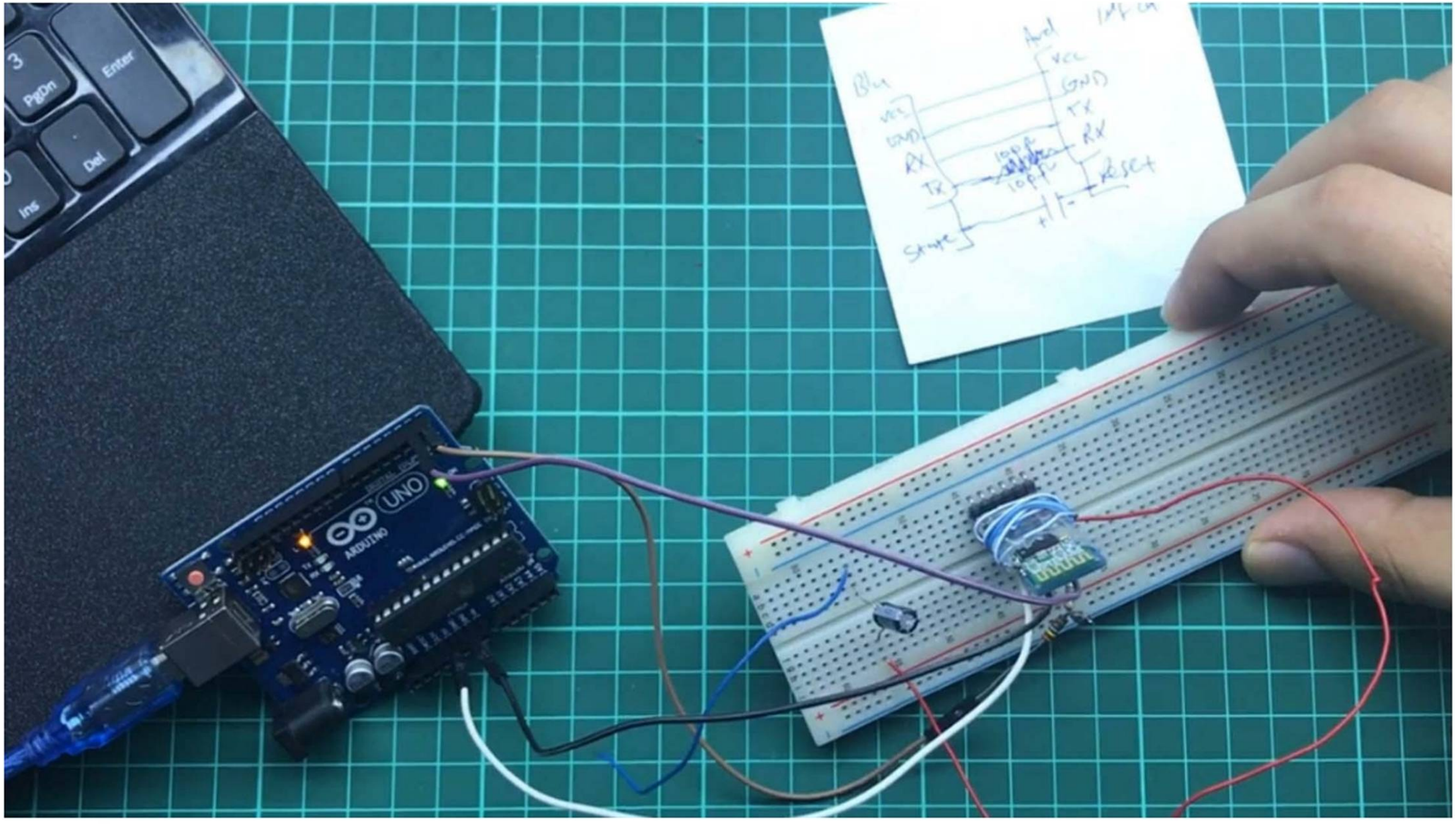


And last but not least we must connect better set and status pins with one flawed capacitor. So let's give the circuit connected here. Now as you can see this is our Bluetooth module and you can see that it has status X the X ground and BCSE So let's connect it here on our board. Now the next step will be simply connecting the wires to our Arduino board. So let's take two words, this black one foreground. So here we have the

ground. As you can see. So disconnect our tween the ground and or Bluetooth module ground. As you can see this is the ground for the module. Now let's connect the five volts again and let's connect them to the VCR see our board now that we have these two connected let's connect Betty X from our unwinnable or the R X and the Bluetooth would you. So this that I go way off as you can see that he X here is connected to the R X and our will to fill you. Now as we already mentioned we have RDX anti X connected through 100 on the resistor. So says this is that is a store we must connect to the T X and the Bluetooth for you just basically in the middle. The other side must be connected to the Bluetooth and I connect to that is installed directly okay. That's it.

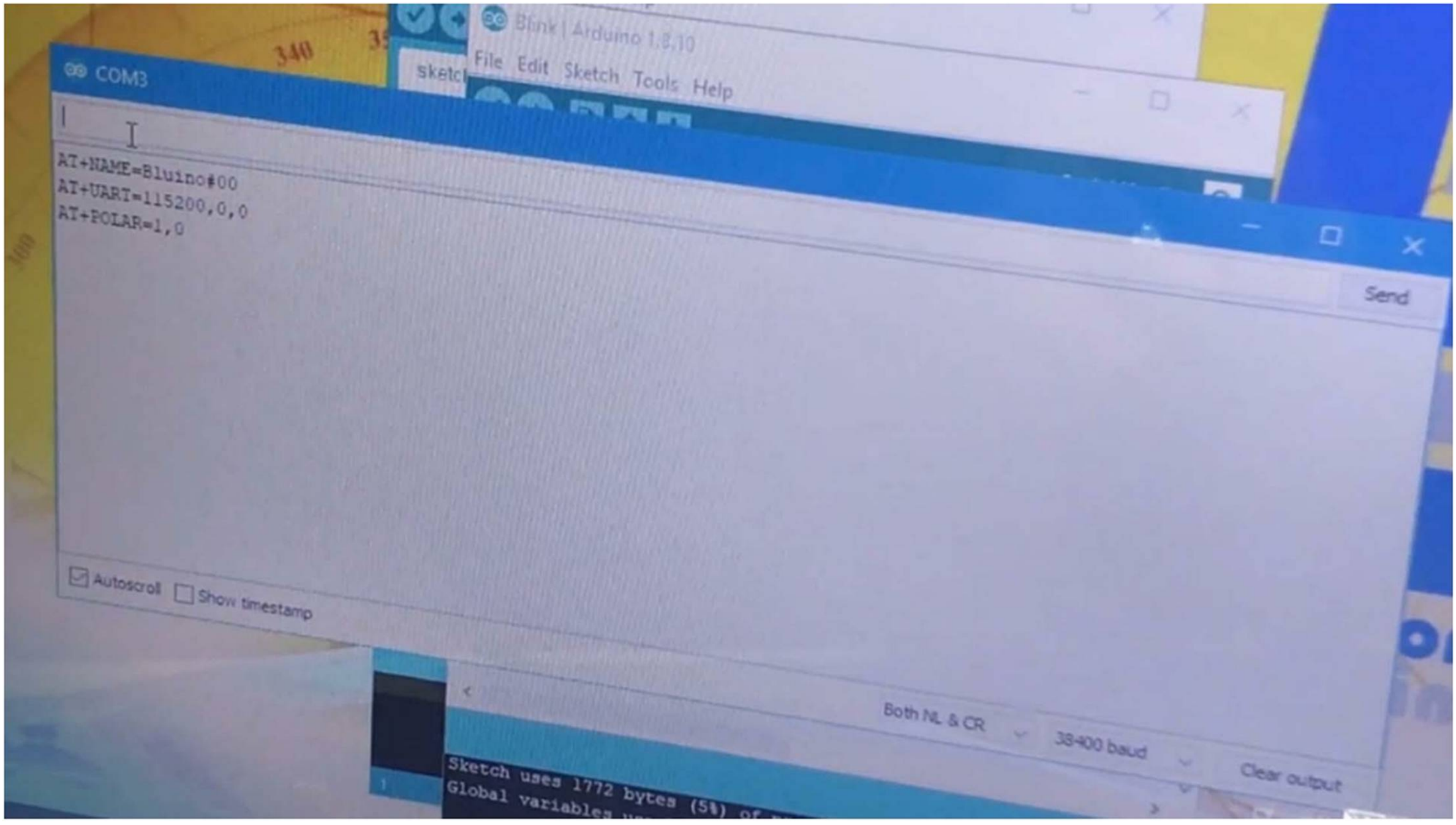


Now we have only one bill which is that of a set state of Spain so that a set pin which exists here which can be found here and all of the ball must be connected to the negative terminal like a buster. So this is the negative terminal and I think it directly here I can see now connect this thing that is it. Ben Now connect the dots at Ben again. That's it. Ben is connected to the negative terminal and that was this terminal which is the other terminal is connected to our and we know will spin. So let's bring in our wire so that we can easily connect it. You can use other wires if you wish. I can't replace this with a symbolic way of connecting this year. There's still this disconnect between the commerce store here and our board. Let's connect the negative terminal with that asset. Ben our board and let's connect the positive will well closing this sort of Dwyer with the state to spend an hour with you. Now this is our circuit. This is how you can connect easily again. Let's summarize what we have done here. We send ground to the VCR counter for the Bluetooth module the T X and Arduino which has been number one to that X and the Bluetooth. Would you extend the Arduino connected through a hundred on resistor to the Bluetooth module to X. That's how an albino knows it's connected using this blue wire through the capacitor to the state of spin and the Bluetooth. You. Now I think that this is the 100 all the one Michael flux capacitor. So let's change this. This one is point 1 Michael okay now so it is ready. What you need to do now is place your adrenal in something we call an 80 commandment. So in order to do so we have to connect it. They must be bought and for doing so we have to place on the button that exists on the you know on the Bluetooth module you can see it in the bottom left corner here can point for it. OK here. This is our button. This is the button that we are going to place to place or are we known in the 80 commandments. So I'll connect my arduino here no before connecting it to the USP from the other side. I must press on that button now I'm blessed on this button. Now wait for five seconds That's it. Now we have a wire and we know in the 80 command mode Let's reconnect. That is it. Ben Now if you open a cereal terminal or see the monitor screen. Let me show you what this is



Now you need to change this body rate to the body rate that we set as default and all we know choose from here both. L can't see up from here. Choose thirty eight thousand four hundred. Now you'll see that the 80 commands are here at. You name it, you asked for Ebola and if you did send eighty now everything is The

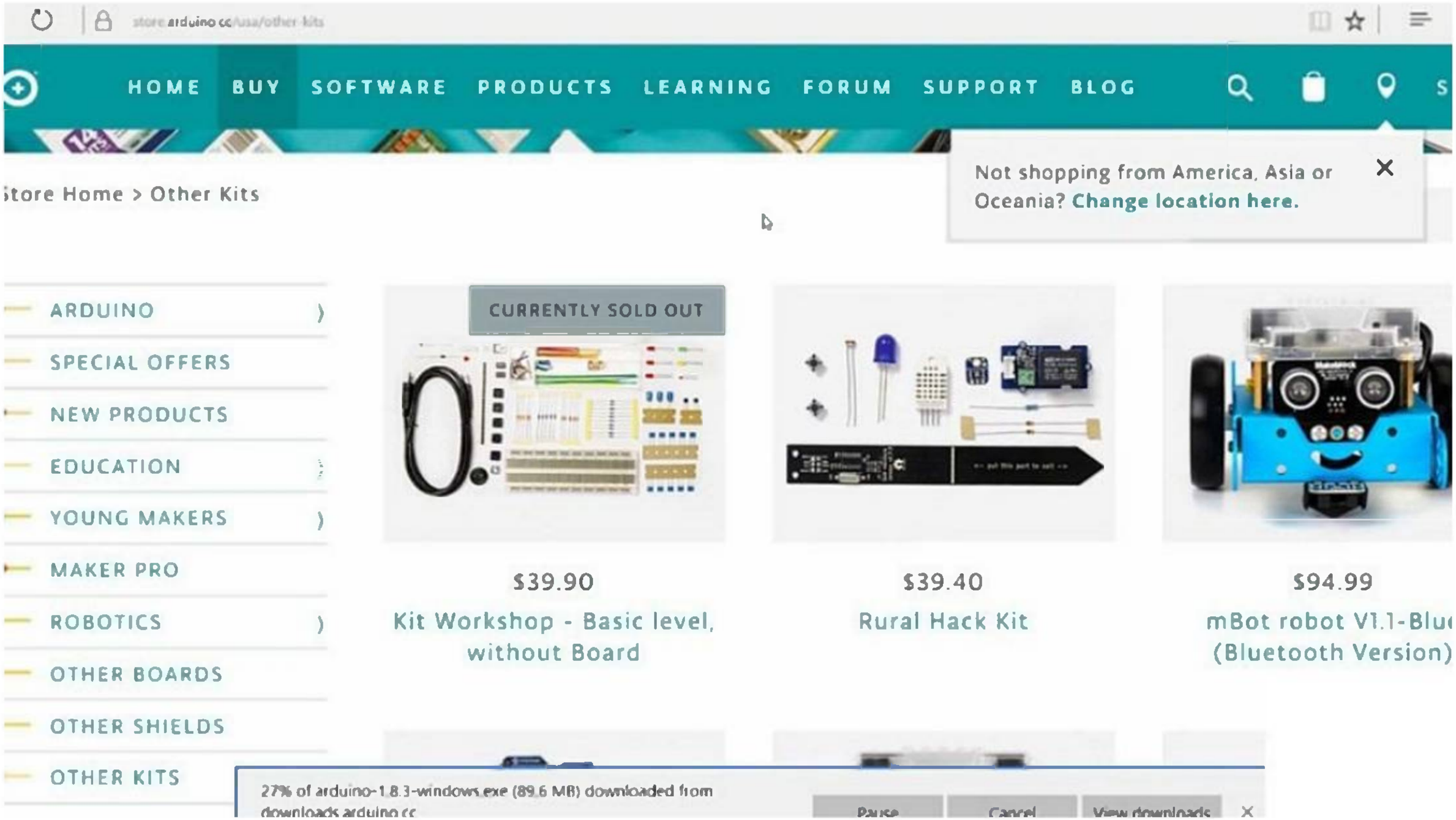
Bluetooth name for our algorithm module is named Blue. You know 0 0 and we will show you this in a minute using a mobile device for now everything's



We placed our Arduino in the 80 command mode and we sent commands that will change the basic configuration. Now the next step will be going back and looking for our Arduino using a bluetooth mobile device. Let me show you these steps now to do so. Here is an Android device as you can see here with me zoom out. This is our Android device and what we need to do here is look for Luciano. Here it is blowin now through drag from the left. You can check the preferences on the click board, change this to Bluetooth and choose on a board. Now we have Tobin, an example of a fine example of basic blink. Let's open up the blank tab. Now what you need to do next is click here to combine and send your code. Now it's asking you if you want to allow using Bluetooth click allow. Now in order to place our algorithm in the new mode we have to unplug it unplug it again. But for now let's choose blue. Now. As you can see our lids blinking just like the code that we have seen fear Bluetooth. That's it for this lesson. If you have any questions please ask the front board. Thanks for reading. This is an educational engineering team.

DOWNLOAD AND INSTALL ARDUINO SOFTWARE

Hello and welcome to this new lease on this education engineering team. And today we will teach you how to download the software requirements of this course which is Arduino Ida. First you must go and search for arduino I.D. using google or bank the first of doubt window to see which is the official Arduino Web site. You can either use that green web editor without installing any software or you can download that green I.D.. Now depending on your operating system you must choose one of these windows mac or linux for my case. It's a Windows Installer. Now it's asking you to donate or you can just download click on Save and you'll have to wait about three minutes for the download to finish. Now let's see the Arduino Web site. It offers a lot of products as you can see here. You can buy shares of any of these items from the official item no upside. These items include Arduino boards, aluminum kits and a lot of other interesting stuff.



Now let's see the other kids category as you can see this kit. This is all about it. It has a very okay and very interesting collection. You can also buy Arduino boards or Arduino shields. Let's look at boards and modules and see how much each of these boards might cost you. This is the official place to buy Arduino boards. You need to make sure that you are not buying from places that are a fake aluminum boards or I've been on

boards that are not manufactured but by the original Arduino company okay takes four votes to can also check that learning section for tutorials reference and things on your things to learn as you can see here the boards the entry level has almost Leonardo and starter kit nano mini micro the enhanced feature has the mega then timed things has other as yes I'm shield the wearables for making let's say a smartwatch of the printing section for making a 3D printer let's see the Omo the mega K this is our adrenal material on 0 1 a symbol 3D printer as you can see looks nice to cost you around seven hundred dollars.

You can also see the Arduino on board. It'll cost you 24 dollars or twenty five dollars Arduino Ono as you can see. This is the original stuff now in order to know if your board is original or not you can flip it. If it's a simple you or Germany then it's original. The ones that are assembled in China are not created by the shell out Green a company. It's created by Arduino associates and it must cost less about ten to one dollar out of the bushes. Such bolts that are not there don't have the same quality as the ones manufactured in the US. Now let's see the original finished downloading. Let's run a download Arduino is already installed so you have to uninstall the one that you have okay. Let me see you have to close an instance of Arduino on that you you have opened then uninstall the previous version as you can see it's removing the files if you already have Arduino installed in your computer you need to update it to the latest version if you already have the largest version uh then please skip this listen click on Next and install as you can see the installation started won't take long.

Store Home > Arduino Mega 2560 Rev3

ARDUINO

SPECIAL OFFERS

The MEGA 2560 is designed for more complex projects. With 5
pins, 16 analog inputs and a larger space for your sketch it is th
recommended board for 3D printers and robotics projects. Thi

You can check our resources section to download the code and the material to get links for the hardware material. Okay. Okay now hit close. Then go to the start menu, go to the latter then choose Arduino. This is our main Arduino idea now to make a quick overview. This is the button that you must click to verify that your code is written correctly. This one is used to verify and upload the code to Arduino. These are for orbit

and save this to create a new file. As you can see when you create a new file you have two main methods: the setup and the Loop the setup for your setup code. The loop is for your code that will run repeatedly in the file is common sense Skitch is used to upload verify the code or add new libraries the tools is used to choose the ball should the board give the ball inform choose double drama burn bootloader which is an advance topic an Arduino open the serial monitor or serial blotter. Fixing the encoding of the code or format it holds is where you can find things you are looking for. It's the answer. Now you can check examples. Basic digital analog communication controls and sort of display. Lets see that we want a basic example for blinking LED as you can see this is blinking red. Example. It's very simple and straightforward . It's commented so that you can understand what's happening and what's going on. So that's why our awareness is very popular because it provides amazing support. That's it for this lesson. If you have any questions please ask and if you aren't able. Thanks for reading. Wish you a happy learning. This is educational. Engineering team.

DOWNLOAD AND INSTALL CIRCUIT DESIGN AND WIRING SOFTWARE

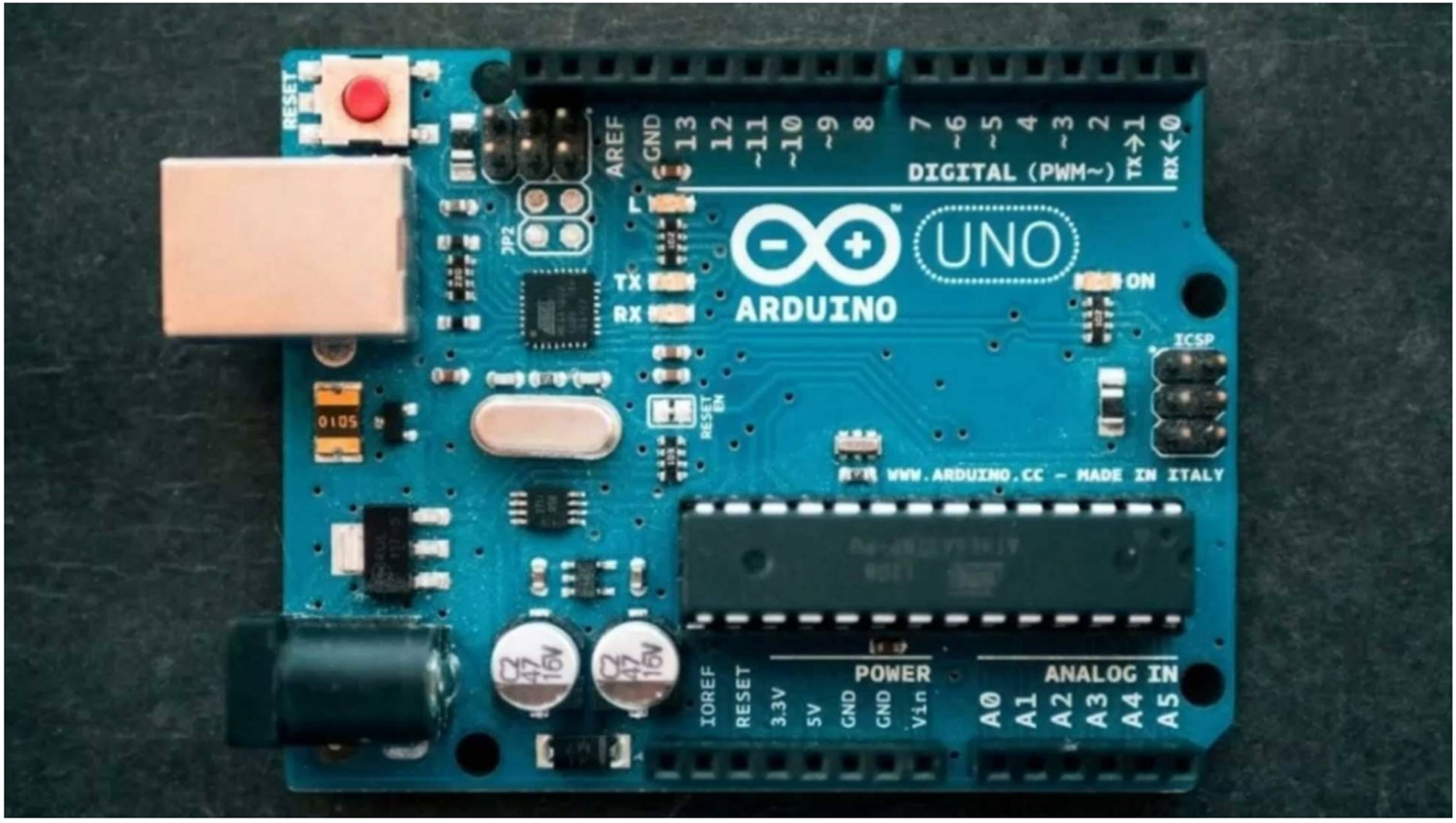
Hello and welcome to this new one in which we will talk about that simply and connect different components together. Now after purchasing the components that we mentioned in the previous lesson you can simply connect them together depending on the schematic that I will show you in this lesson to show you the schematic we need first to download a software called zing. It's a software used to connect different elements together without sharing them. As you can see this is arguing with Allard. This is the schematic and this is the ECB. We have a course on how to use it to create different circuits and how to use it to create BCBS in no time. Even if you don't have any previous knowledge you can check it out using our profile by slashing the education and engineering team. If you face any problem finding it please drop us a message and we'll be more than happy to help you find it and learn it with a maximum discount. Now click on moderation direction then click on the download choose your abilities system as you can see it's available for Mac Linux and Windows. It's free software. We need the windows 64 bit virgin as you can see this is the download link so I'll start the download. It's a fine one hundred eight mega 180 mega byte fine so it will take about two minutes to download. Now again if we went back to this page we can see that the software is used to create circuits with different elements. As you can see this is a lead and this is the Arduino Ono. I will take you in

a quick tutorial after the software is downloaded on how to use it but it's a very promising software for creating your schematics and creating a ABC design without any complexity.

Okay I'll bother with you and get back once the download is finished again. The software is used to create schematics and to create a PCV. This is obviously before. How would we know? And this is the lead that was connected here in this 3D view. So it will make life much easier. If you are a university student and you want these images for your presentation you can simply save these designs as Angie or rebadge them as an image. It's about saving software or your schematics as images as you can see here. We have a share button. Now once the dialogue is finished as you can see here click Auburn then you can simply double click the folder go here and turn slides in . It's easy to extract all the files. Let me cancel this and extract it to let's say the documents and your folder slides. Okay and we can open it using that for a lot. As you can see this is the folder where we have all the files for the flooding software. Will take a few minutes it's extract all of these files since it's fine. Okay now let's close this double click on Friday and XY and it will load the items in no time. As you can see, now it's loading the core parts. And as you can see here the software is loaded. This is the user interface. Here I will compare the Brit board as you can see schematic BSB and code. Let's create a new file. This is our new fine. Now we need to add the items that we will use in this course. You can search for the items using this search button or you can click on these icons.

INTRODUCTION SCOPE OF LEARNING

Welcome and thank you very much for choosing this course in this Arduino beginner's guide. You will find an introduction to the use of the mini PC Arduino in theory and practice. I, as an engineer, will share with you my knowledge from study and practice step by step in this course and this course aimed specifically at beginners. You will learn all the basics you need to know when working within Arduino.



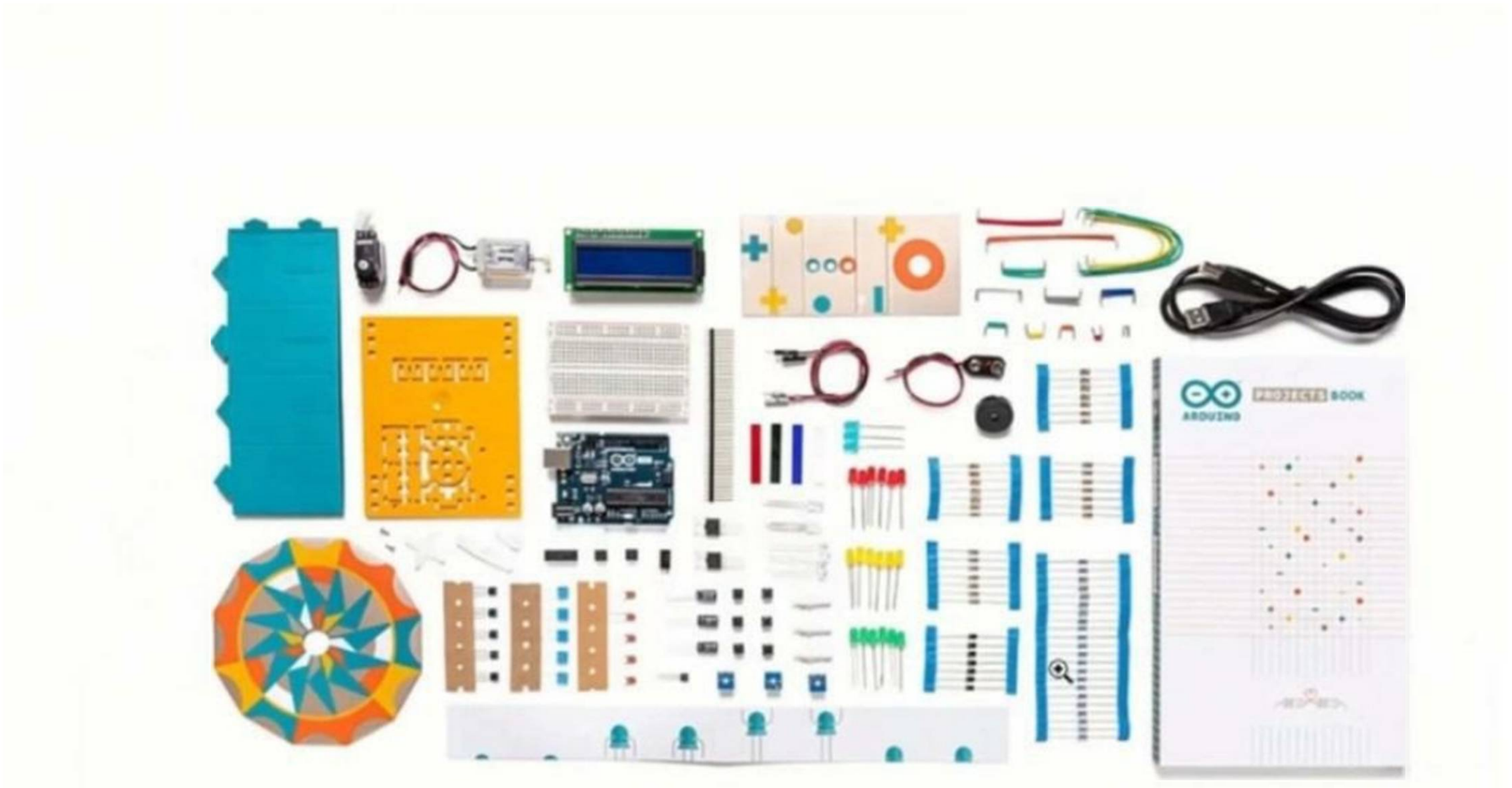
We will work exclusively with the Arduino Uno in this course, as it is ideally suited for beginners. In brief, this course will teach you the following in detail the basic terms and components of electrical engineering as background knowledge structure of an Arduino Uno board and how to use it. What is the Arduino I.D? What is it used for and how is it structured? Programming basics block based programming. Programming

bases. Text based programming. How to create a system with an original. And how to write the required program code. Hands-on learning based on exciting Do-It-Yourself projects, as well as signals with leg temperature based on the decontrol light dependent control of a motor. Gas detects an alarm. Password protected system, remote controlled system and much more. Be excited. Let's go.

FIRST STEPS WITH AN ARDUINO

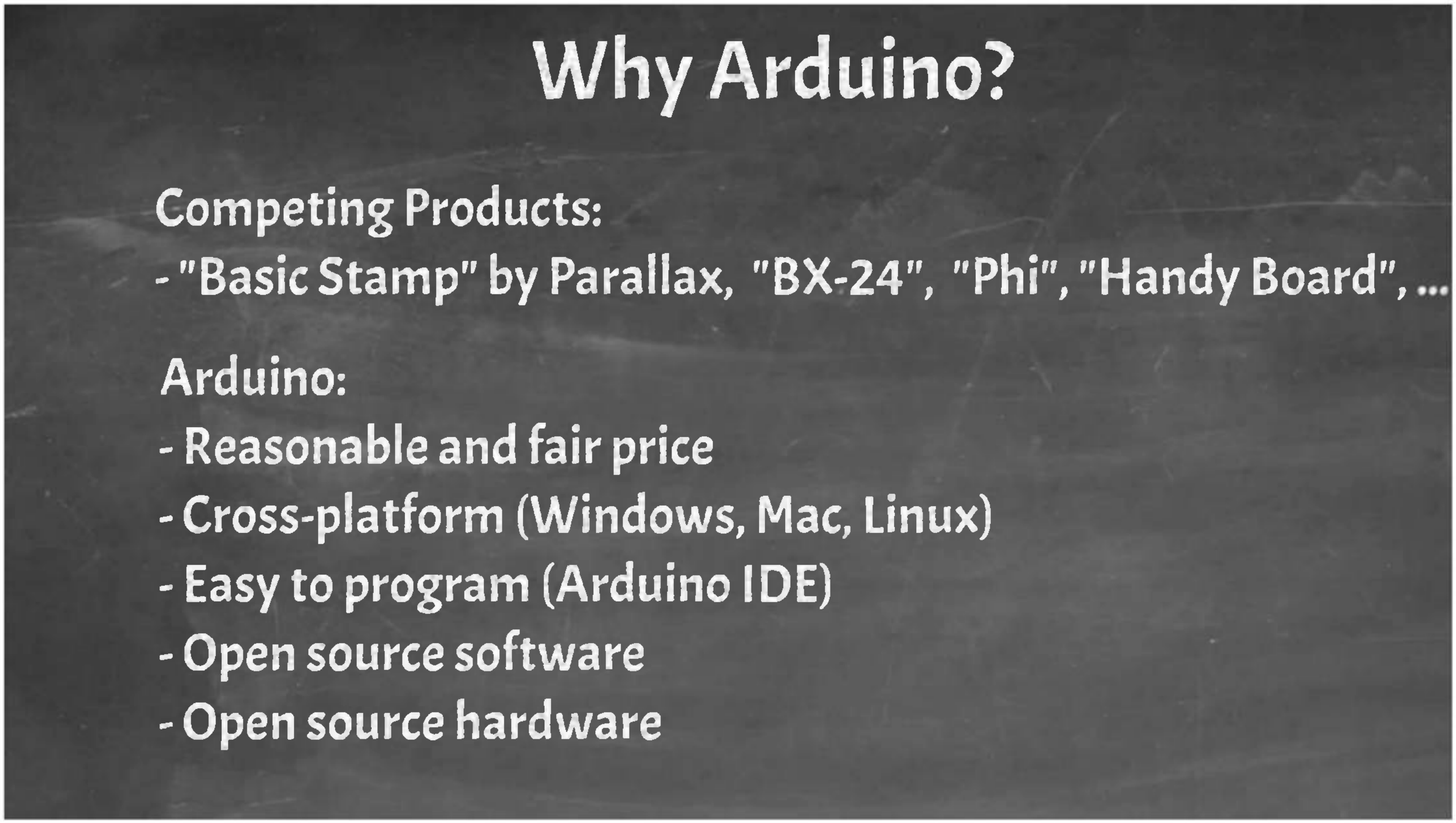
Arduino is an electronic platform consisting of hardware and software that is very user friendly and was created as part of an open source project. The term open source is generally characterized by the fact that software is freely available. Active participation of users is desired and there are no restrictions on use. Simply put, an Arduino is nothing more than a small and very simple PC or microcontroller that is capable of taking input signals, processing them internally and then converting them into corresponding output signals. An input signal could, for example, be sunlight falling on the sensor. The corresponding output signal could, for example, be controlling a motor. This mini PC can be purchased in the very poorest appearance of a circuit board equipped with electronic components either individually or as a set. There are different arena boards, modules and beginner sets. The following Arduino boards are recommended to get started with. Arduino, UNO, Arduino, Nano, Arduino, Leonardo Arduino, Micro, a good overview of all products and the way to order them can be found on the official website. W w w dot Arduino Dot CC you can either buy the Arduino products we need in this course via this official site or buy them via Amazon or eBay, by the way, and Arduino Uniport is available from about $20. Complete starter set from about $70. In this course, we will mainly deal with the Arduino UNO board and use it for the projects. We will also need other components such as LEDs, resistors sensors, for example, infrared sensor actuators, for example, a motor

for the project. Which components in particular are needed? You will find them the further course, and in each project clearly listed for this, it is recommended to buy the so-called Arduino starter kit for beginners.



Or also another start to set or in addition to the Arduino Uniport also sensor or a module set, which contains the required components in order for the mini PC to know what to do with the previously mentioned input signals and what output signals, we would like to have the Arduino board and its instructions. These instructions for the microcontroller are given by the user. That means by us. Thanks to a program code. Programming language is used for this purpose for programming and transmission. A special software is used, the Arduino software IEEE. This software can be downloaded online free of charge, for example, at Arduino Dot CC. Countless projects have already been realized with the Arduino microcontroller, and this mini PC is suitable for hobby projects for prototyping or even for scientific projects. The Arduino community is spread all over the world. It includes students, engineers, hobbyists, artists, programmers and so on. Millions of users have contributed to this Open-Source platform, and thanks to these contributions, a lot of knowledge has been accumulated to help professional and new users with their various projects. Adreno is specifically designed for users who need a simple and inexpensive platform for electronics and programming projects. Since Arduino is an open source project, users can change anything they want or customize any function according to their needs. Why choose an Arduino? There are also other microcontroller platforms and competing products, some of them that are similar to Arduino are called, for example. Basic step by parallax beaks 24 fi and handy board. And there are many other boards with similar functions. However, these microcontrollers use quite old fashioned programming methods. The community is not as large as for Arduino, and the instructions are not so easy for newcomers. The following is a brief list of why Arduino is so great and why you are right to choose Arduino and this course at a reasonable price. Arduino has a fair to a reasonable price, and this is one of the main reasons for its worldwide success. The Arduino Uno board, for example, is already available for around 20 bucks. Cross-platform, as with many major platforms, most microcontrollers only work with Windows. They lack support for systems like Mac and Linux Arduino. On

the other hand, it runs with all systems easy to program. Probably the most important point. An Arduino is effortless to use and program the software for it. Arduino IEEE is very user friendly. This helps especially beginners, but also young people or retirees to get familiar with the program very easily and playfully.

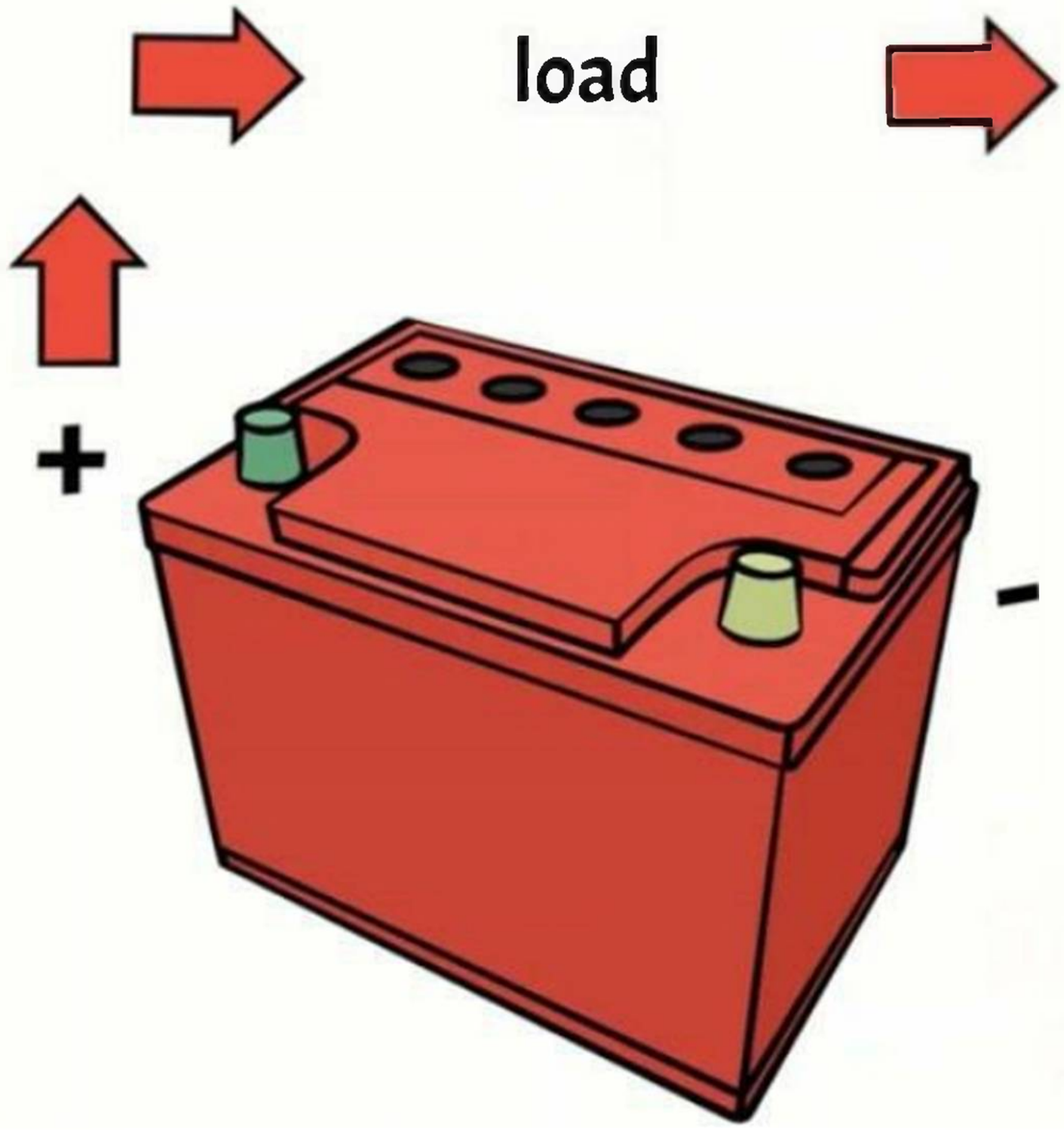


Nevertheless, Arduino also offers the possibility to perform complex projects and programming, so it is also a great platform for advanced users. Open source software, everyone can contribute to this great project. Every user can create new libraries. We will learn later with business and make them available to other users as well. Open source hardware. The Arduino hardware is also open source and can be modified by any user through a kind of blocking play system and through a so-called Brett board. Modules can be added in a variety of different projects and can be implemented. It is a kind of modular system.

INTRODUCTION TO ELECTRICITY AND DIGITAL ELECTRONICS

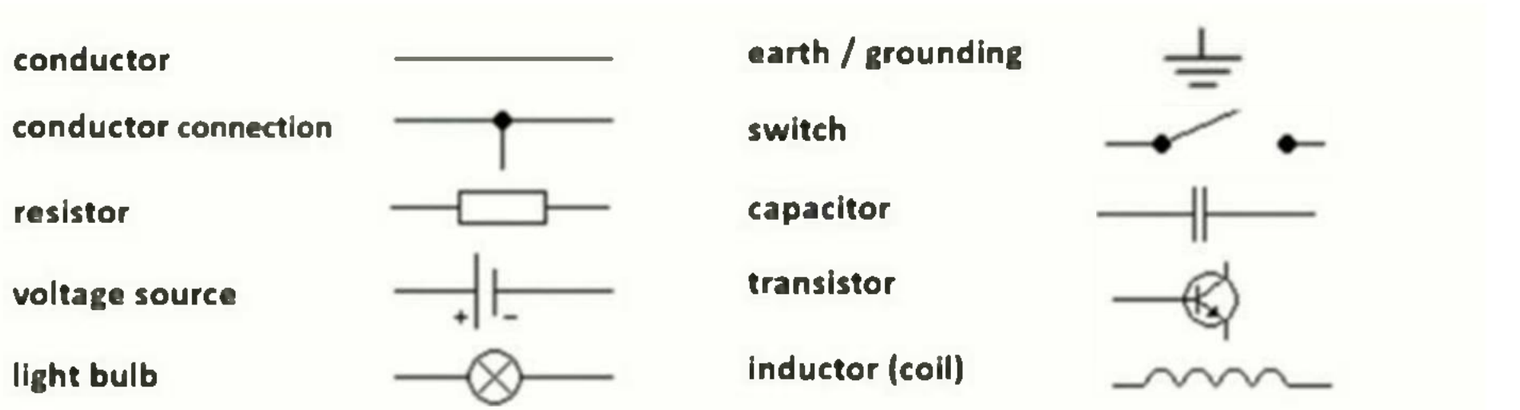
Before we go into detail about the Arduino board and the Arduino software, let's first take a look at the basics of electricity and digital electronics. Electricity is created by electrons flowing from a place with higher potential, higher energy to a place with lower potential, lower energy. It can be relatively well imagined by means of a waterfall. The water represents the electrons flowing from the top point of the waterfall. High potential high energy to the bottom point of the waterfall. Low potential. Low energy. The potential energy is transformed into kinetic energy during this process. That's why it loses this energy state in the process. But actually, this energy is transformed, as said before. Similarly, the electron wants to flow from a place with higher current, high potential to a place with lower current, low potential. Voltage is the unit of electrical energy generated by the battery, the battery or any other voltage source has two terminals. One terminal is called the negative terminal and the other terminal is called the positive terminal at the positive terminal. The voltage potential is higher compared to the negative side. Thus, the current flows from the positive side plus pole to the negative side minus pole.

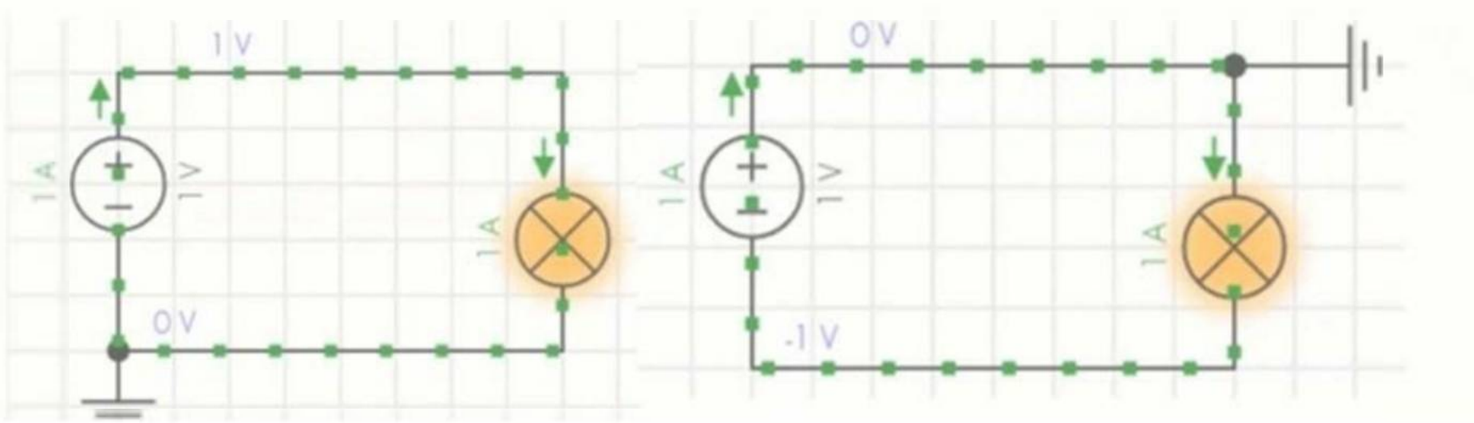
Technical current flow:



Considering the technical direction of current. You can think of a battery or other power generating source as fun functioning as a pump. A battery, for example, generates voltage or energy through an electrochemical reaction inside this voltage or energy flows out of the positive pole in the form of electrons. These electrons symbolize the water molecules that are pumped out to compensate for the lost electrons. The battery,

similar to a suction pump, draws the same number of electrons back in through the negative pole. What is a circuit? Simply put, a circuit is an arrangement of different components, with an electrically conductive connection between them for an electrical circuit or a circuit to work. You need an energy source such as a battery and a load, such as a light bulb, as well as connections between these two components, which are called conductors in electrical engineering. These components are represented in a circuit as the following symbols. For a lamp, for example, to light up, as shown in the following figure, the secret must be closed. That means there must be a connection between the two poles plus and minus of a power source. For example, the battery and the lamp. If this is the case, current flows from one pole of the power source, for example, the battery through the lamp and back to the other pole of the power source.



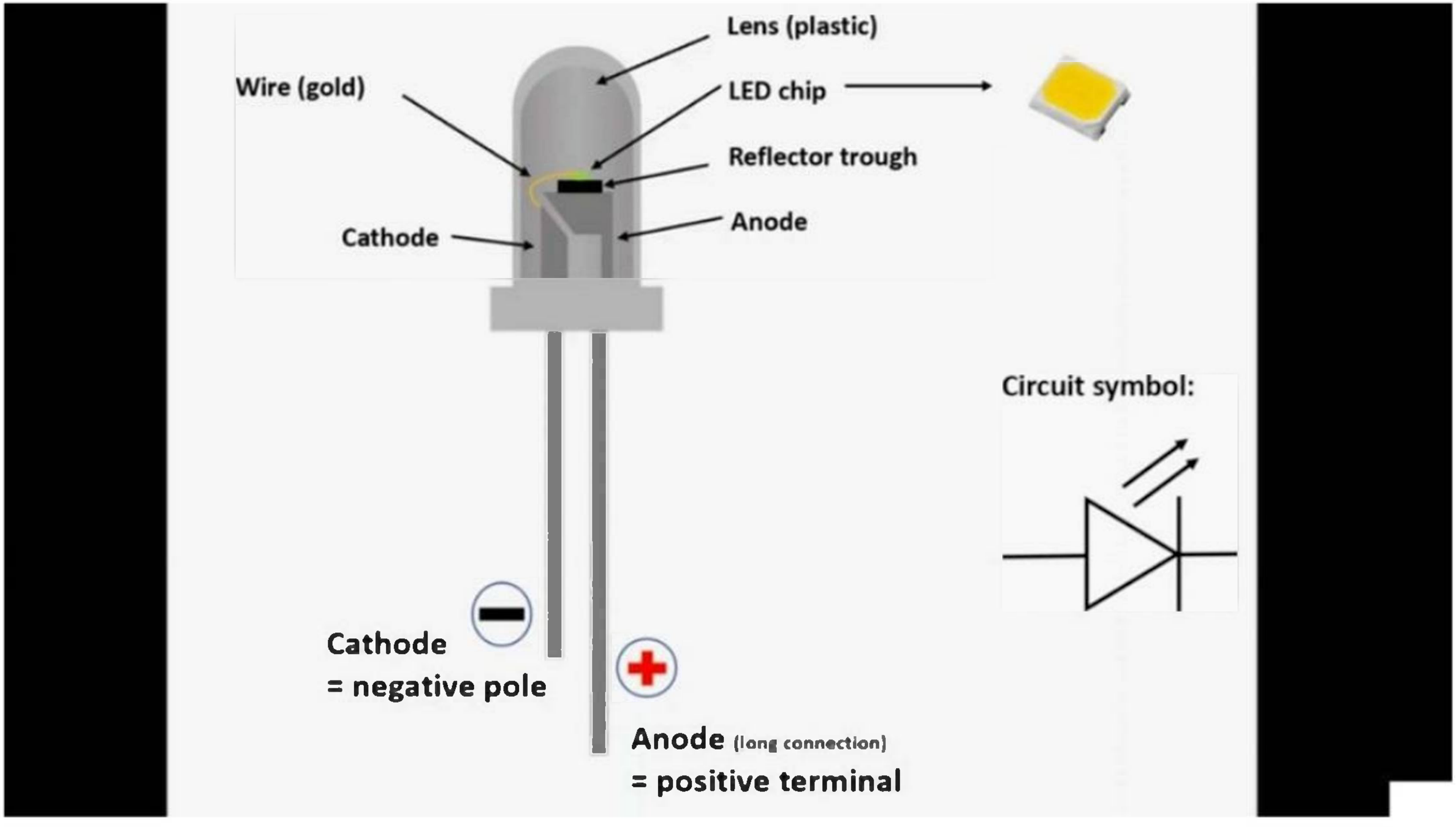


When this connection is severed, for example, by a switch, current no longer flows and the lamp no longer lights. In this case, it is called an open circuit. A short circuit occurs if the current can flow from one pole of the current source to the other pole unhindered and without first passing through an electrical component, for example, through an uninsulated spot of a cable on a metal surface. This is because the current always takes the path of least resistance. A circuit diagram is the basic concept of a circuit that can be drawn, for example, on a piece of paper or with the help of a computer program. Such a circuit diagram can also be made a bit more descriptive, a bit more descriptive. Circuit diagrams for the Arduino can be created at best, with the software from freezing torque, which can be downloaded at the given. You're all for little money. As Fritz's dot org, you can also find many references and instructions for using the software. As we can see on the picture in this project. For example, a relay and the module are connected to Arduino Uno wire colored cables. The colors of the wires each have a meaning that helps to make correct wiring. In this illustration, red wires stand for the five volt signal and all black wires for the ground signal zero one.

IMPORTANT COMPONENTS OF ELECTRONICS & DIGITAL ELECTRONICS

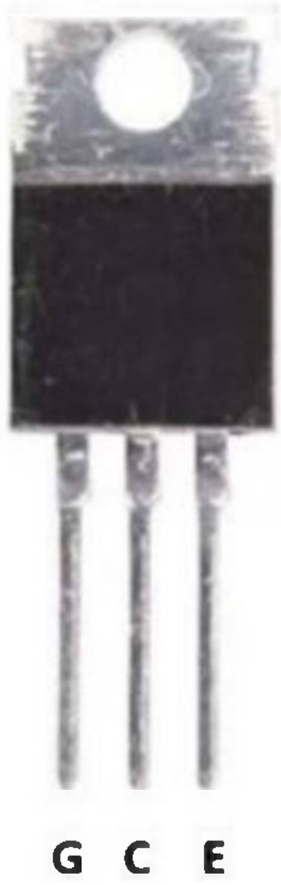
In this chapter, we will deal with the basics of electronics, a minefield of electrical engineering in particular. We will take a closer look at digital electronics. Pieces of digital electronics are simple switching operations. The computer is one of the best examples of these switching operations and of digital electronics. The applications that a modern computer enables us to do are achieved thanks to switching operations performed by millions of transistors. So what is the basic principle behind the PC? Surely you have heard this before. It is the so-called binary system which is based on the two numbers zero and one. Communication and digital systems take place with the help of these numbers or with the help of various combinations of these two numbers. Since the Arduino is basically nothing more than a very simple and stripped down mini PC, this principle is also applied here. The true binary numbers are mostly represented in today's electronics systems by the voltages five fold, one or high value and zero zero or low value. The restriction to only two numbers or voltage values seems to be very limiting, and it is very hard to imagine how a PC can achieve today's outstanding performance. Based on this system, however, this system and its simplicity makes sense. It simplifies the matter because it is extremely simple to recognize these two states. That means zero or one and two definitely distinguish them from each other. A diode is a semiconductor

component in electronics, but has the property of allowing current to flow in only one direction for which direction the other direction is blocked for the current flow reversed direction.

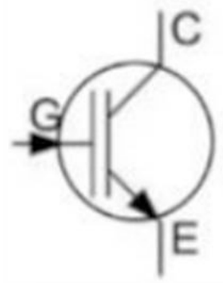


You can imagine a diode simply like a valve. The simplest application of a diode is the leg. The LED light emitting diode is a semiconductor device that produces light when it is energized. The light is produced by current flowing from a DC source to the diode and through it. Since then, LEDs are a semiconductor device. It also has a forward direction. This means that current can only flow through it in that direction. If an is connected incorrectly, no light will be produced. The color of the light and whether it is visible or not. For example, infrared is controlled by the doping and material used. Two major advantages of ladies are a long life. B, low power consumption compared to old fashioned lamps and lady can achieve a lifetime of several 10000 hours and has many times better efficiency. Why is that? Conventional light bulbs produce an enormous amount of heat in addition to visible light, which means that the energy expended is converted not only into light but primarily into heat. With LEDs, only a little heat is produced as a waste or byproduct, and almost all the energy can be used to produce the light. There are now different types of allergies. The simplest design is shown in the scented image. The heart, and also the actual semiconductor element of the energy shown is the tip, which is placed on the reflector, on the anode and amidst the light. The circuit symbol of an energy consists of the diode circuit symbol with two additional slanted arrows, which are supposed to represent emitting light. Transistor is a simple three terminal component that can best be thought of as a wealth that controls the flow of water in a pump. For example, if you turn the control of the wealth in a certain direction, that means open water flow increases. And if we turn it in the other direction, that means closing the flow decreases the wealth. In the case of the transistor would be diets, and the water would be the current electronics and generally simplified has a lot to do with switching elements, and transistors also behave like switches.

Insulated Gate Bipolar Transistor (IGBT)

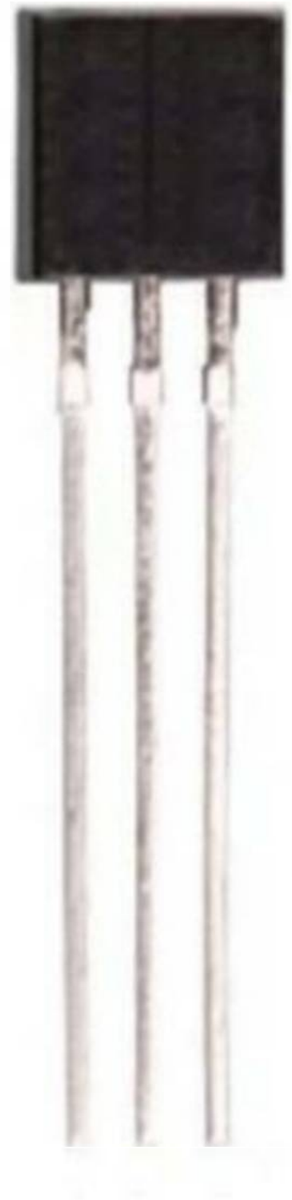


Circuit symbol:

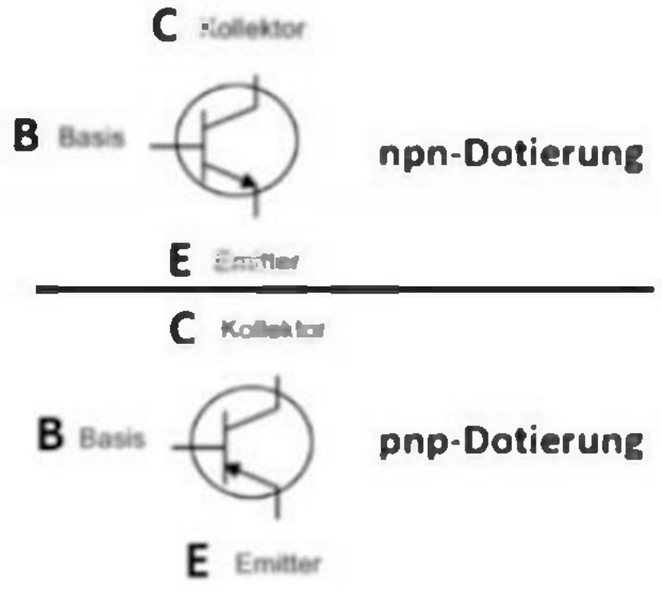


And other variants depending on the design

Bipolartransistor

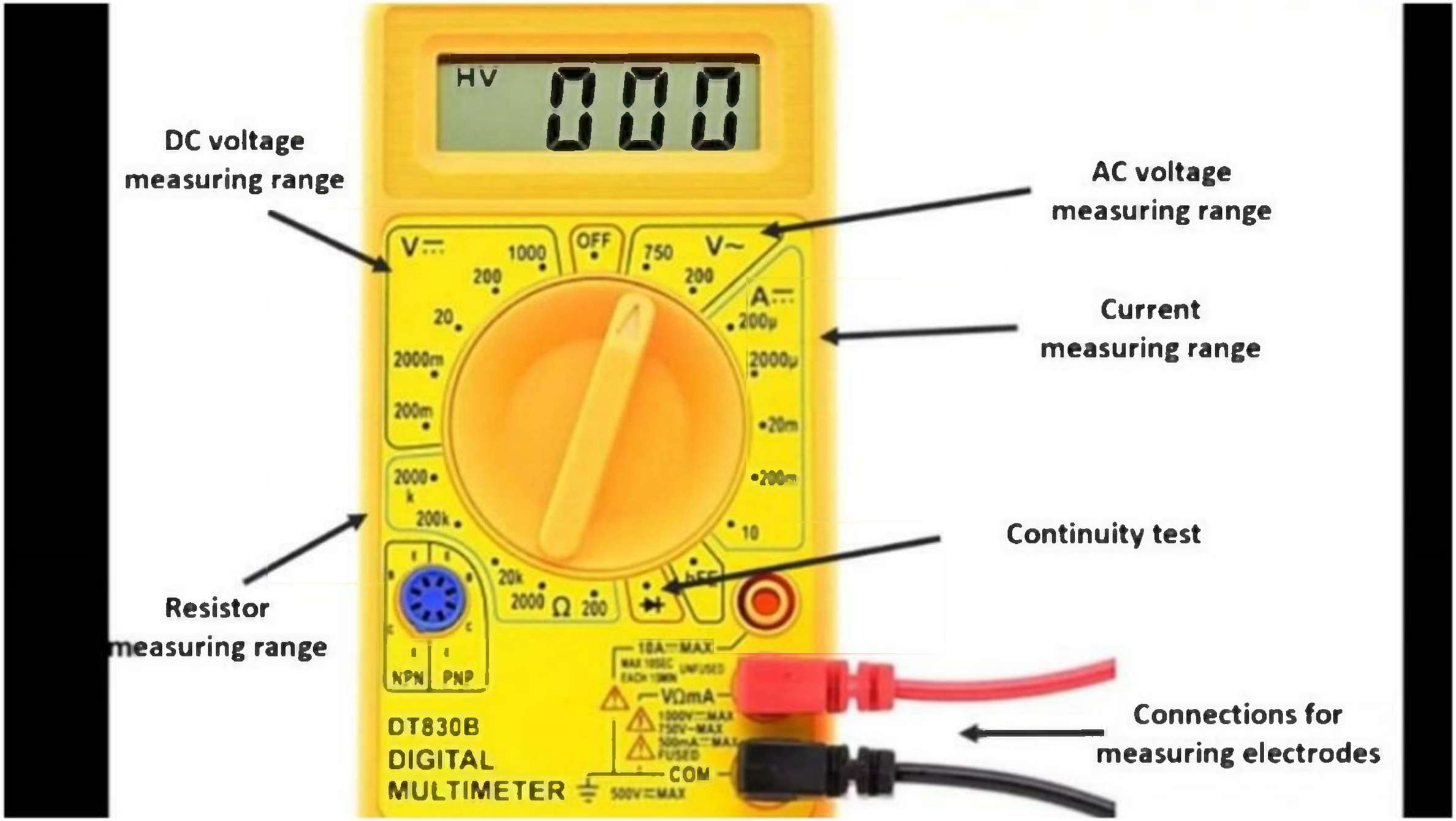


Circuit symbol:



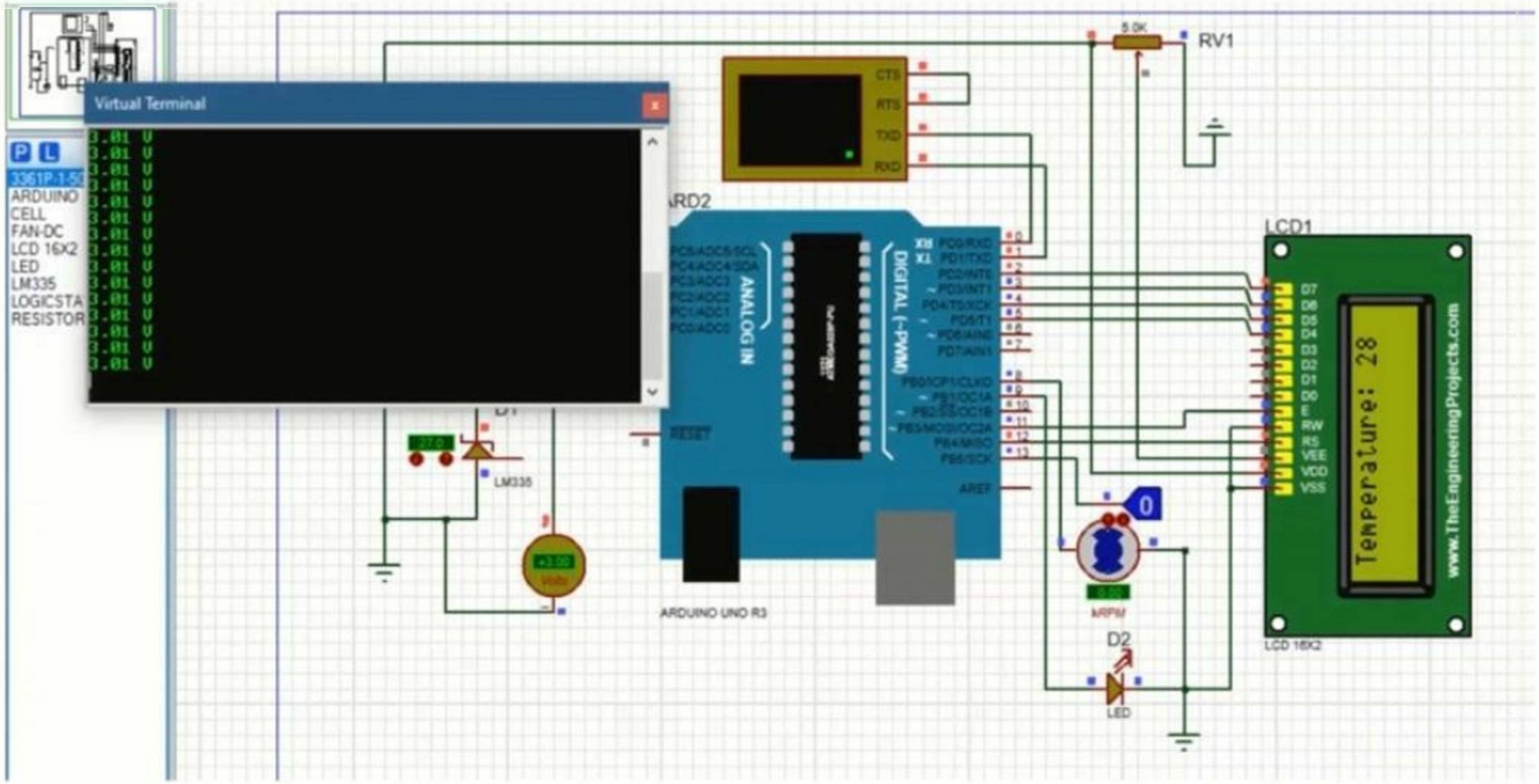
In addition to this switching capability, transistors also have the property of amplification, which would be equivalent to changing the welfare issue for the amount of water output. This amplification property is particularly important in the world of electronics. There are several types of transistors. One of the simplest being the bipolar transistor. Furthermore, there are, for example, the field effect transistor and the

metal oxide semiconductor field effect transistor. All types of transistors have special properties and are used in different applications. In simple terms, a capacitor consists of nothing more than two plates arranged parallel to each other and the dielectric between them, a dielectric is simply a weekly or non conductive substance. Solid liquid gas with charge carriers that are not free to move capacitors are generally considered charge storage devices because when an electrical potential is applied, they can store voltage energy in their plates. The ability of a capacitor to store charge depends on the area of the plates. The spacing and the dielectric between the plates. This ability to store charge is typically referred to as capacitance as the charge and the plate increases. So that's the voltage of the capacitor. And this continues until the capacitance is reached. Resistors are components that can be used primarily to, as the name suggests, apply resistance to something. In this case, their sister acts against the current and can be used to limit the flow of current into a component that is connected to the resistor. Basically, every conductor has a resistance that can be calculated depending on its length and cross section. In our case, we use so-called sheet resistors. Here, for example, there are the carbon former sisters and the metal, or also metal oxide foam resistors with these types, the resistance while you come from a Karami core with a layer of carbon or metal or metal oxide, the resistance. While you can be measured either with the help of a multi meter or directly on the resistor by means of the colored rings, each resistor has a color code consisting of five rings that reveal the resistance. Whether you read this color coding must be explained in detail and would therefore go beyond the scope of this chapter. After all, we want to work with the Arduino as soon as possible. You can either look up this coding online or bust a set using moved the meter to measure it. In electrical engineering, multi meters are often used as measuring instruments, multimeters with two terminals and measure voltage current resistance, capacitance and inductance.



But you can also measure the polarity of transistors and perform a continuity test with it. They continued to test us on whether a circuit is shorted or not. Multimeters can only measure one variable at a time, such as current or voltage, to measure multiple parameters. We need to use several individual devices. The figure shows a simple multimeter with the different measuring ranges, depending on what you want to measure,

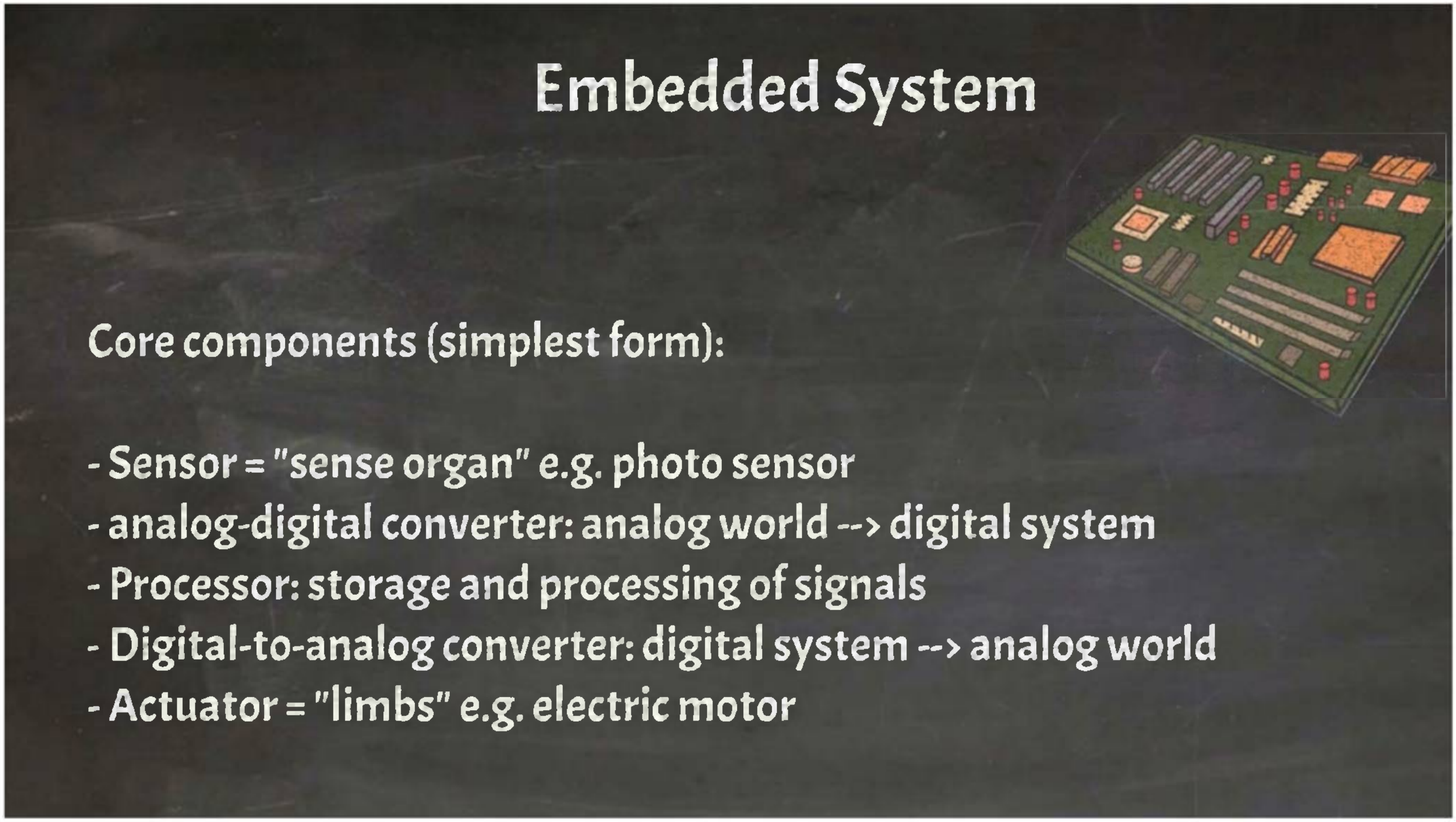
you turn the dial to the appropriate range when taking a measurement always taught from the highest possible voltage or amperage. To the highest or the highest resistance value and then turn the display setting down until a suitable value is displayed. This means, for example, that if you make a measurement on a DC voltage source and you suspect that whether you're between 20 and 200 volts, you first turn the setting range to 200 watts and then downwards. If you want to miss measurable pitch, you have to connect the measuring electrodes parallel to the voltage source or to the component you want to measure. As for a light bulb, for example, this would work like this. And if you want to measure the current, you have to connect the measuring instrument, the multimeter in series to the consumer. That means disconnect the line. Perhaps you have heard the term embedded system before, perhaps you have also often wondered what it actually is and what it is used for. In simple terms, an embedded system describes the presence of a type of computer in a technical system or on a circuit board, as in the case of the Arduino that carries out signal transmission or data processing of input and output signals. This processing is done by a microcontroller which is a tiny computer. This microcontroller is designed to perform certain functions, and it's basically nothing more than a tiny computing system consisting of a semiconductor chip. You can program the microcontroller using PC software to perform operations. As an example in the figure, you can see a system controlled by an Arduino Uno, the system switches the power of two devices, air conditioner and electric heater, depending on temperature and time during off peak hours. Data is obtained from the utility company. It tries to match the electricity bill with the room temperature. A budget limit for the electricity price can be set with a potential meter. Are we one? It also attempts to turn the air conditioner on at night and off during working hours six days per week.



In this secret, the temperature is measured using three five temperature sensors and displayed on the LCD. In this complex system, the Arduino switches the air conditioner and heater by matching the temperature, time and power. Three parameters This example should only show what is possible with an Arduino and that even complex systems are possible. Of course, we will start much simpler in one of the following chap-

ters and learn everything step by step. So do not be afraid. The system can be a stand alone system, but it can also cooperate with other systems to perform a common task in every embedded system. There are circuits that perform the functions and send or receive instructions. That means transmitting data in the form of voltage. With the help of conductive elements in its simplest form, an embedded system consists of the following core components: processor, sensor actuator and an analog to digital converter, as well as a digital to analog converter. We will look at these components in a little more detail. The further sensor is a component that can convert physical changes in the real world into an electrical signal that can be used by a computer or electrical system to process data. Think of it like a human's sensory organ with the help of eyes, ears and other sensory organs. Our brain can interpret the outside world and thus create an image of it. And similarly, you can imagine it with computer systems. In this example, the sensors would stand for the sense organs and the microcontroller for the brain. The electrical signals coming from the sensors to the microcontroller allow the embedded system or the microcontroller to interpret what is happening in the outside world and then execute a response or program given by a programmer for a scenario by means of code. Another important component of an embedded system is the analog to digital converter. This converts the analog signals sent by the sensors into a digital signal for this purpose, as we already know by now. The binary system is used. That means the two numbers one and zero. These binary numbers represent the language of the system in which a microcontroller can understand and react. The difference between analog and digital signals is, among other things, that an analog signal can be the carrier of several pieces of information. Whereas with a digital signal, one can assign a unique piece of information to each signal and analog signal would therefore confuse the microcontroller. To put it bluntly, processors are the heart of any embedded system. A processor performs all tasks related to the received data. This component therefore receives data that stores it, processes it and tells the system in what way it must react to this data.

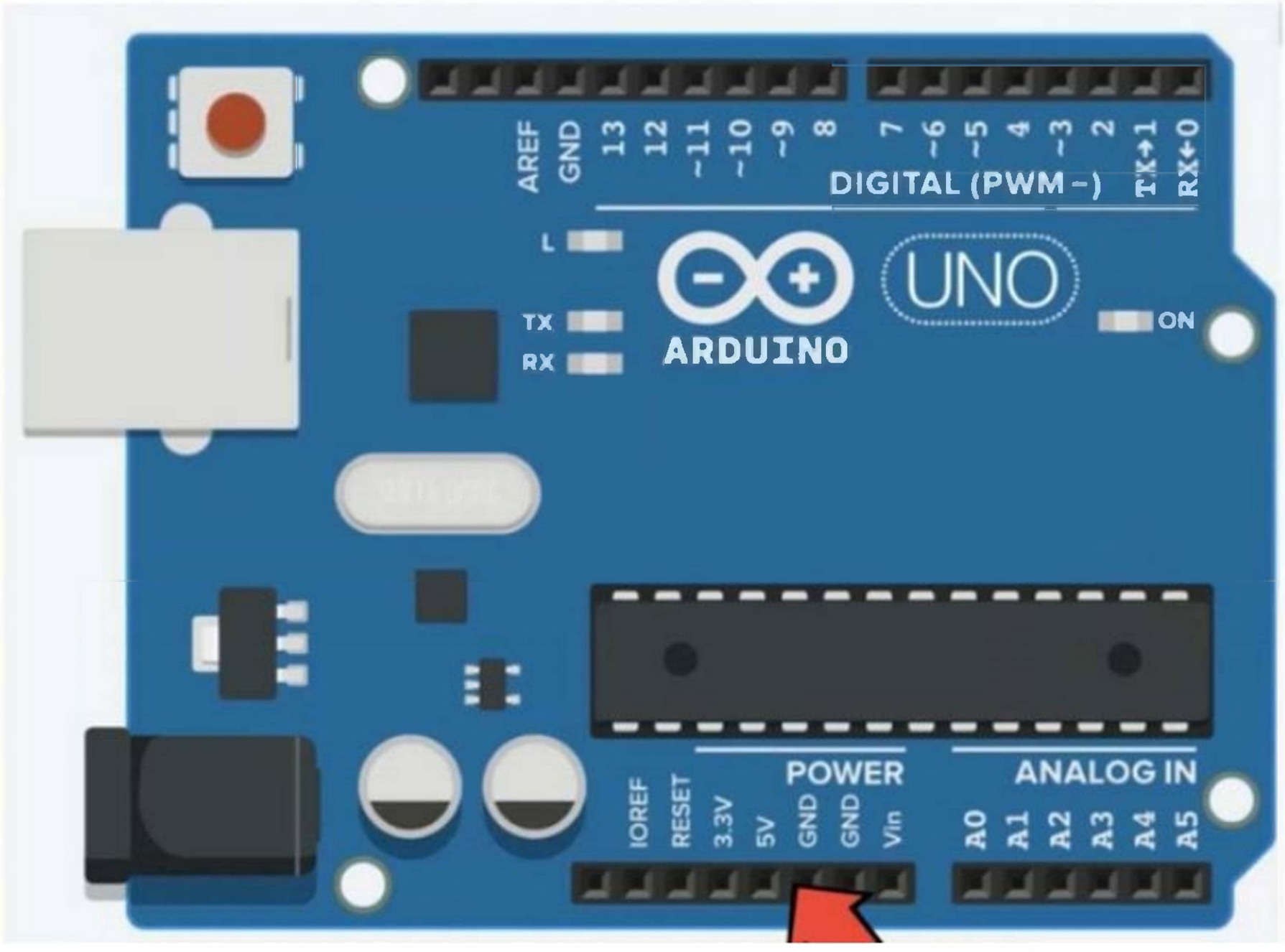
A digital to analog converter is basically just the opposite of an analog to digital converter. It converts the digital signal sent by the microcontroller back into an analog signal. So why is that digital signal converted back to an analog signal? Simply because an analog signal can be understood by a physical device or actuators? An actuator, for example, an electric motor converts the analog signal received from the microcontroller and the digital to analog converter into a physical action.



There are mechanical, acoustic, chemical, thermal and optical actuators that can perform physical actions in the real world, according to their design. This is how embedded systems interact with the environment. From this, there are a few steps that we need to take when creating or designing an Arduino system. First, we think about what task our embedded system means that our DNA should perform, for example, raise the blinds when the sun rises. Then we think about which sensors we need for this. For example, a light sensor. In addition, we need to think about the program code for this task. Don't worry, we'll get to that and we still need an actuary to perform the task. For example, a motor. Of course, we also have to build the board with the components, according to a previously considered secret.

THE ARDUINO BOARD (HARDWARE)

In this chapter, let's take a look at the hardware. That means the board of the Arduino, each pin of an Arduino board is marked with a number or a label. The board works with five involved in what follows, we will look a little more closely at the components of an Arduino in this case, the Arduino owner. We will now take a closer look at some of the most important components of the Arduino board. Step by step. Digital pins can supply a five voltage or zero volt similarity. This can also detect whether a voltage is present at the pin and whether this is five fold or zero, logically. The latter is the case when there is no voltage. You can define in our program code whether opinion should be used as output or input. We will see how this works later. An internal already is connected to 13 of the Arduino. This ADT can be useful in many situations. We will deal with it again later. Another lady is connected to the power pin to indicate if the Arduino is receiving power. The eight mega microcontroller controls the board, controls all input and output signals and serves as the digital control center of the Arduino. It is the processor of the board and later also contains the program code transferred by the user. Five analog pins and a look in are used to read analog voltage and convert it to digital voltage. This is done with the help of an analog to digital converter, which we have already learned about.



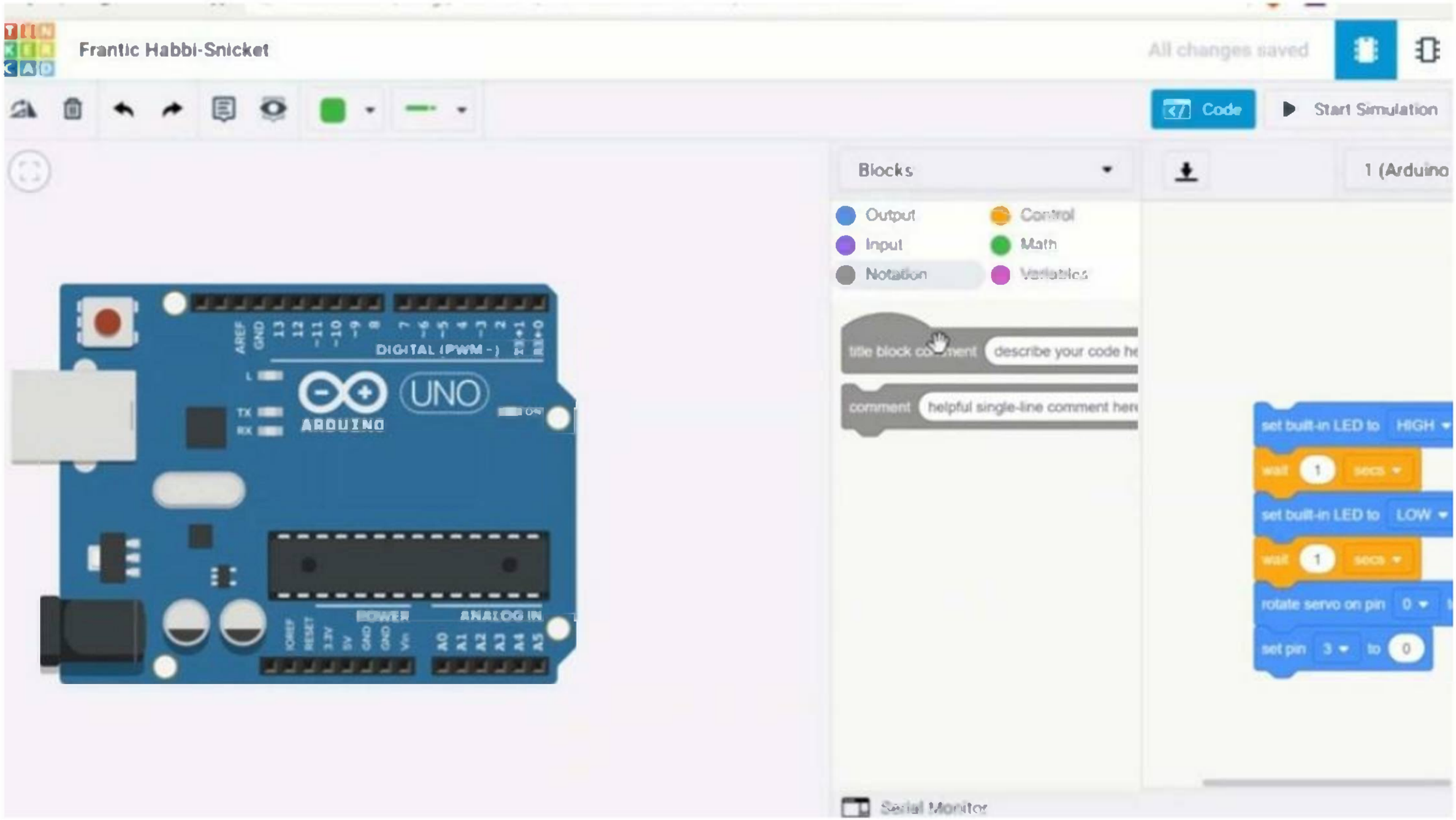
The two pins, TNG and five volt are used to supply power to the circuits in a project. The 3.3 volt power pin is also a valuable key in the stands for ground, which is the negative terminal of the board. The board can be powered by a USB cable or power plug. Arduino can work with voltages from five to 12 volts. In no case should a higher voltage be applied. But two pins labeled ticks and risks are connected to allergies and

indicate when communication is taking place. That means whether a signal is being processed or not. For example, this is especially essential for troubleshooting, which can be simplified considerably. Finally, the Arduino can communicate with the computer while you sport, for example, to transfer the program codes to the processor. You can reset the code at any time with a reset button. This button stops all functions that the board performs and restarts it. We will neglect the other elements and connections for now as we will come back to them later or in the practical project. We will understand the practical use of the connections in the next chapter. We turn away from the hardware and take a look at the Arduino software. Stay tuned. After the basics, exciting and great projects will await us. But first, a short explanation about the use of a so-called breadboard upright board is the best way to build a circuit as soon as it becomes a bit more complex or contains several parts. With the breadboard, there is an area for the power supply of the Breadboard Plus and minus imprint and areas with letters and numbers. The pins that are in the row letters eight to eat and F-250 are conductive, fully connected to each other. This means, for example, H1 and A1 or H5 and i5 and G5 are conductive fleets connected to components, and cables are plucked into the respective pins and thus connected to each other.

THE ARDUINO SOFTWARE (IDE) & PROGRAMMING BASICS

The Arduino Integrated Development Environment, commonly known as the Arduino Software I.D., consists of a text based editor for writing the lines of code and using the section toolbar. Several menus. The software can be connected to a microcontroller of the Arduino board to upload coats to run a program. Download the software at Arduino Dot CC. In this chapter, we will look at how to write the program code in this Arduino IDE, which you can later execute on the Arduino board. But first, let's take a look at an alternative method when programming an Arduino. Before we go into more detail about the Arduino idea and learn how to create the program code in it, we will first get to know the method of block based programming. This is simply an alternative to the more complicated text based programming that follows later block based programming. It's the simplest form of programming. This is mainly useful and great for people who have no experience with programming because you can achieve success quickly and easily. You can imagine it as if you put building blocks on top of each other in the software, each of which has a specific function. You just have to put these building blocks together in an orderly and meaningful way to get the finished code for beginners. This type of programming is very useful to learn the basics of programming methods in general operation. The best way to get started with book based programming on an Arduino is to use Autodesk's Tinkercad software. Tinker Cat is an online platform where, among other things, you can

quickly and easily program on Arduino using the block based programming, which are presented after creating an account at Tinker Cat dot com. You can get started. Through block based programming, we mainly get the following three advantages. As a beginner, we don't have to be afraid of small but essential errors in the syntax. The program structure. Second, we can thus concentrate on the main task without worrying about the programming interface. And third, we can become more familiar with the basic structure and flow of a text based programming through the given blocks, code blocks are divided into different categories. These categories are also color coded for better clarity. The following categories are available for selection. Category output These blocks are used to instruct the actuators what to do while the microcontroller. So we control the output signals through this category input with the help of these blocks. We bring the data from the census that means the input signals to the microcontrollers. Category annotation comments. The blocks that can be found in this category do not directly affect the Arduino code, but are used to indicate what the program code actually does.

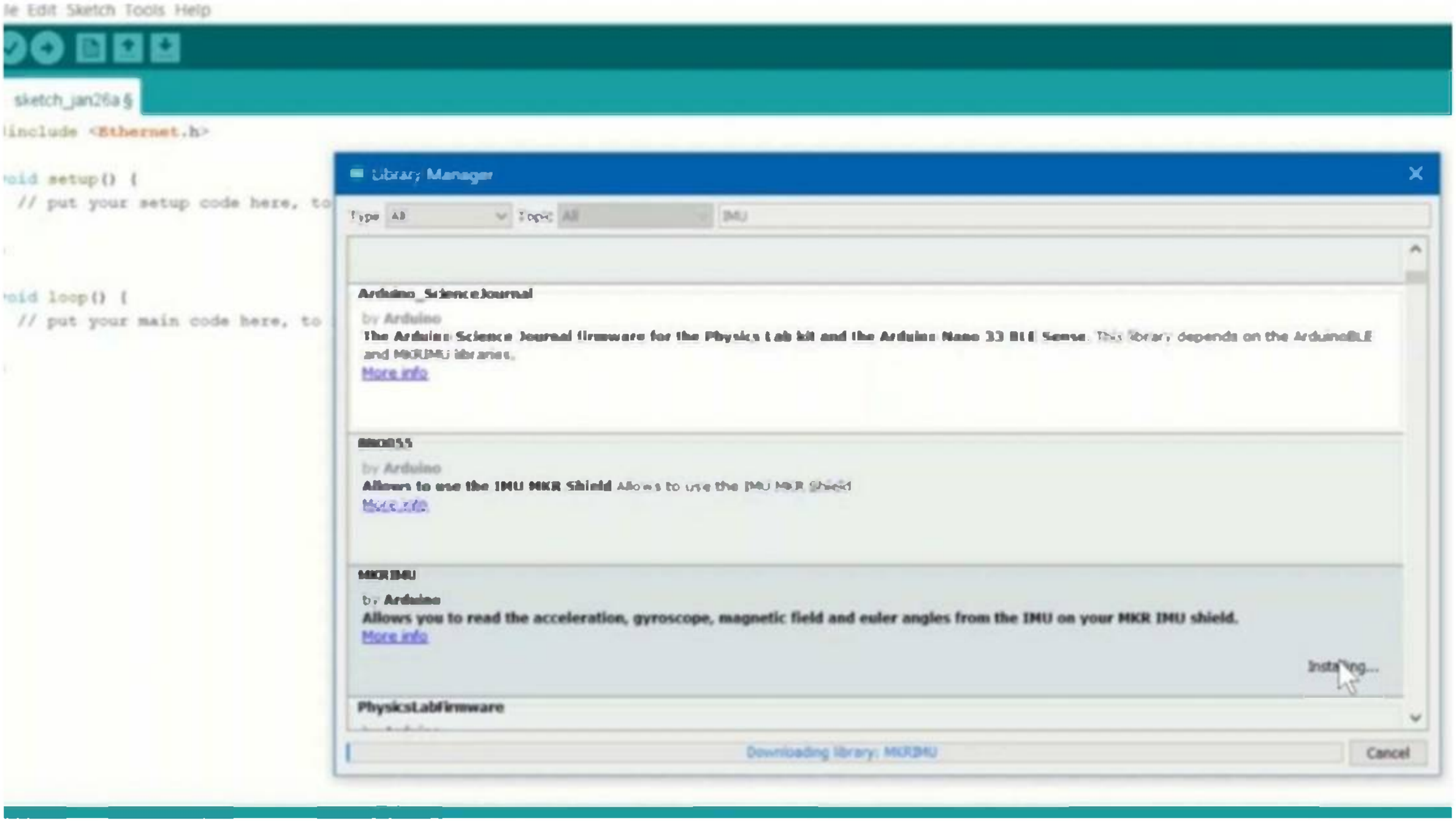


These books help the user to understand the program code category. Control control structures help to enable the microcontroller to make decisions based on the data it receives. Category variables Variables are changing values that the program uses to execute mathematical functions or to store data. When we use blocks from the different categories on Tinkercad, they align with each other like in a flow chart. But let's

just take a look at a relatively simple example. For example, we would like to control an LED using block based programming and Tinker could. In our example, we connect an elegy to two of the Arduino. Furthermore, we put a resistor between the negative pin of the lady and the negative terminal of the Arduino board TNG to control the amount of current flowing through it. This resistor will help us control the amount of current flowing through already and will prevent the lady from burning out. For example, if we now at the first block from the category output and tinker cut as shown in the image, we can use it to turn on the LCD. It also automatically implements the following circumstances in the program code without us having to program them separately. First, Pinta is defined as output per second.

Pinta can no longer be used as an input pin. Third, in this code set up, the lady is connected to 2.2 of the Arduino for the actual switching on of the other. Just playing around with pink occurred and the possibilities of block based programming. That's the best way to learn. Meanwhile, we continue with the preparations for this course mainly discussing text based methods of programming. In the Arduino software, you use the built in text editor to write code for the Arduino, a code written with the Arduino software. It's called Sketch. The editor includes the following functions, among others. Cut and paste and search and replace the message area provides the ideas response when code is written. Such a response can also be an error message. For example, the console provides text based output messages provided by the Arduino software, for example general information. And Arduino code can then be saved with the file extension. But, you know, when it is ready. In the lower right corner of the window, the configured Arduino board and serial number are displayed. The toolbar buttons give you options to review and upload programs, create sketches open and safe, and open the serial monitor. Using the serial monitor, you can see what information the Arduino is sending to the PC MEPs, the communication between Arduino and PC. In the next step, let's familiarize ourselves in detail with the elements of the program for this purpose. Let's take a look at the common bar with icons located in the upper part. The small checkmark is used to check the entered code for errors before compiling. Compiling means that you or the program translates the programming language into the machine language of the computer compiling then starts automatically with a click on the checkmark. With the error located to the right of the check mark and pointing to the right, you can upload your code to the configured Arduino board. It must be connected via a USB port for this. With the arrows pointing up and down, you can open or save a sketch with the open error. You can also find example sketches. The button next to it, which looks like a document, is used to create the new sketch. And the small magnifying glass on the right side of the program opens the serial monitor, this is used as already men-

tioned to monitor the communication between Arduino and PC software or vice versa. Libraries represent an extension that allows us to quickly and easily give the order to renew additional functionality. It is basically nothing more than a coat that has already been written by eager members of the community. Especially for Arduino beginners. This is a huge help in terms of time and effort. You can use a library by importing it. This is done in the Arduino software ID in the menu at the top under the item sketch. Here you select the library and then select the library you want to use. You will get hashtag include statements at the beginning of the code. Alternatively, you can just write them directly at the beginning of the code. If you know the name of the library. The library manager is used to install new libraries in the sketch tutorials, open the program and click on the sketch menu and then include library and then select Manage libraries. In the library miniature, we find a list of libraries ready for installation. Now we want to search for example, for the IMU Library. To achieve this, we simply type in the abbreviation IMU in the search field. After that, we can select the version of the library.



I am using the abbreviation of Inertial Measurement Unit and is the name for a measurement unit sensor network that is used to measure acceleration and rotation rates. After we have selected the latest version, we can click on the install button and then have to wait briefly for the new library to be installed. If we switch to the include library menu item again, we can check whether the library is now present and the in-

stallation was therefore successful. New or needed libraries can also be found online. These can be downloaded and installed as compressed files. Most libraries can be found on GitHub. GitHub dot com GitHub is a management platform community for software development. Once you have downloaded the library, you can load it into the program in the following way. In the Arduino I'd go to sketch then include Library. Then select Edit Zip Library. Then we are prompted to specify the location of the desired library, navigate to the location of the downloaded file and select I will tell you which libraries we need for our subsequent projects at the beginning of each project. If we then click again on the sketch tab in the upper menu bar and then on include library, we can if the process was successful, see the installed library in the lower area of the drop down menu. Now the library is ready to be used. The serial monitor is used to display data sent from the Arduino to the computer. Here it is important to take the correct ball great. So the portrait bottom right so that it matches the rate you find in the sketch environment at Serial Dot Begin. What this means, exactly you will understand better when we get to the practical projects, so it's best to go on first if you don't understand something right away. By the end of the course, it should be clearer. So that we do not have to work exclusively with block based programming. We would like to get to know text based Arduino programming in this chapter as well. This type of programming is a bit more difficult because we need to know the exact syntax and functions. Arduino code is written in C++ programming language. Therefore, this chapter will give a basic overview of the structure of a text based Arduino code, as well as introduce the most important functions well, use and structures. After working through this chapter, we also already get to the very practical programming and implementation of some great Arduino projects. We've read the code and ordered that you know, it's in the text based editor in a so-called sketch. This contains the complete code, which is then transferred to the Arduino microcontroller with the upload button right arrow. Before that, you have to click the check mark to compile the code for any code written for an Arduino. There

are two essential components. The first of these is to set up the coat inside the following curly braces of this function is executed only once and all relevant and essential information and structures for further code are listed here. For example, we tell the microcontroller here which pins are used as inputs and rich pins are used as outputs. The other essential component is loop, the loop function creates a loop. This means that the code inside the following curly brackets will be executed again and again, and not only once any task that the microcontroller is to perform is written into here. So the basic code is written here. Let's first familiarize ourselves with the general program structure syntax. When programming Arduino code. You can imagine the antics like the punctuation marks and paragraphs in the text, for example, after a sentence, you make a period. But when programming an Arduino, you make a semicolon after each line of code. In addition, we must adhere to the following structure. Curly braces are used to start and stop a function when the function is executed. The code inside these braces is executed. A semicolon tells the code that the current line of code is finished. Tooth lashes are used to write the comment to better understand as a human what the code is doing. All lines of code that start with these characters are ignored by the microcontroller. Multi-line comment can also be started with a flash, followed by an asterisks. When you are done with this, you set the string in the opposite way. That means first and asterix and then slash all lines of code between these characters are also ignored by the microcontroller, with the hashtag defined. You can assign a name to a constant variable with hash tag include. You can include an external library into the code.

{ } Curly braces are used to start and stop a function. When the function is executed, the code inside these braces is executed.

; A semicolon tells the code that the current line of code is finished.

// Two slashes are used to write a comment to better understand as a human what the code is doing. All lines of code that start with these characters are ignored by the microcontroller.

/* A multiline comment can also be started with a slash followed by an asterisk. When you are done with this, you set the string in the opposite way, i.e., first an asterisk and then a slash ***/**. All lines of code between these characters are also ignored by the microcontroller.

With **#define** you can assign a name to a constant variable.

With **#include** you can include an external library into the code.

If a coach should not work once all the software gives an error when compiling, you should always check first whether you have used all syntax elements correctly. For example, check whether there is a semicolon after each line of code or whatever. All comments start with two slashes, or whether all the necessary brackets open and closed have been set. The following operators are used when programming in code to de-

fine logical comments when. The following operators are used in programming to define logical comments when programming Arduino code. Two equal signs mean equality of two variables, for example, X equals equals Y, X and Y, both equal. An exclamation mark followed by an equal sign means unequal, equal less than sign means less than another variable agreed to then sign means greater than another variable. A less equal sign means less than or equal. Another variable agreed to equal sign means greater than or equal to another variable. The percent sign you can get the remainder. Therefore, mathematical operation and asterix is used for multiplication plus is used for addition. Minus is used for subtraction as Lesch is used for the division. Equal sign is used to assign a value to a variable to end. Signs are operative for logical end to vertical signs are the operative for logical or plus plus means error 1:59 variable minus minus means subtract one from a variable plus equals the abbreviation for X equals x plus y minus equals. For example, is the abbreviation for X equals x minus y. Every contains multiple values for one variable, bullion stores, the binary state of variable, true or false byte stores, a bite value chain stores one character float. Stores of four provide value and decimal form, double stores and eight byte value also in decimal form into stores. A four byte number long stores, an eight byte number size T stores the size of a variable in bytes string stores, a text unsigned followed by, for example, or long or other command hopes with negative numbers. An unsigned conder consideration is performed. Void is used for function declarations that do not return a value at the end of the function. When programming for an Arduino data or values can be either a constant or a variable, here is the difference: a constant is a fixed value. That means a data element to which a value has been permanently assigned. The constant high means that the microcontroller should apply five volts to a pin. Additionally, which one has to be defined? The constant low, on the other hand, means that the Arduino should apply zero world to a pin, in addition which one has to be defined. The term true is used to define that a certain statement is true. The term false is used to define the particular statement as false in-

put defines that the pin to be determined is used for an input signal. That means that the microcontroller should read which voltage is present at this point. Output defines that the pin to be determined is used for an output signal that means that the microcontroller should apply either zero or five volts high or low to this pin input pull up. It's used to connect an internal resistor to a pin. LRT built in is used to control and connected to 13 of the Arduino describes a fixed numeric value. A variable is a data element in the Arduino program that associates a name or letter with an assigned value. Defining a variable is called a declaring variable in the programming language. You can perform all kinds of mathematical operations with a variable. For example, Using this line of code, we declare an integral variable that is named X and has a value of 45. When we have declared the variable in this way, we can use this variable in our program. Such a declaration must always take place first. Otherwise, the microcontroller will not know the variable. Now we can calculate with this variable, for example, at the value 10 to this variable. This would then look like this. This line of code says that the new value for X is equal to the old value of X plus the constant 10. We can also transfer the value from one variable to another. We do this by writing the new variable on the left side and the original variable on the right side. For example, we want to store the value of it in a new wearable, for example, in the variable. Why then do we have to write? Once a variable is declared, it is associated with the stored value throughout the program. If we try to assign this name to another thought to type, the IEEE will give an error message. Also essential is the scope of the declared variable. This simply means that if we declare a variable at the beginning of the program, we can use it anywhere in the program. However, if we declare a variable only in a specific function, then the variable can be used only in that function. In the following code, we can declare three variables for this purpose as an example and consider what the scope of these variables is. So now we have declared three variables named X, Y and Z.

sketch_jan26c §

```
int x=0;
void setup()
{}
void loop()
{
int y=10;
}
void new_function()
{
int z=15;
}
```

Scope X is a global variable and can be used in any of the functions declared at the beginning of the program. Code Y was declared in the scope of void loop, so it can only be used in that scope and set was declared void. New function. So again, it can only be used in that specific function. So always pay attention to which variables you declare in which place. To combine the individual variables, operators and constants into a

function or a working structure, we need expressions that create the control or command. The most important ones are as follows. If is used for checking a condition and is used to perform an operation when that condition is met, LS is used for the action to be performed if the condition is not met, as if it's used when a second condition is to be checked. If the first condition is not met, break stops the code in a loop. Continue, restarts the code in a loop while is used to create a small loop within the code. This is executed until a defined condition is met for its use to create a loop that is executed with a defined number of operations to while is used to create a small loop that runs until a condition is met. Goto makes the program continue in a straight line. Return will return a specific value at the end of the function. Functions are basically nothing more than abbreviations for a code segment that you would actually have to write again and again for a certain action, since some actions are needed frequently. It makes sense to bundle them in certain expressions. The functions are simply declared like variables. In addition, functions bring some other advantages. Some advantages that functions offer are the code remains organized and structured. Debugging the code becomes straightforward. That is, if the code doesn't work, the code is efficient and clear. The code is straightforward to understand for a new user. As an example, we can create a function that adds to numbers. In this code, we have declared a function called test function. At the beginning, we used the word void, which means that the function does not return a value but only performs the action. That means X, X and Y and then stores them and sets them. If we want to output the revenue for that, we need to construct the function as follows. This function is of the integer data type. It adds X and Y then stores the result that means the value in set and then outputs the value of that which is stored in an integer variable called a. We have started the functions in both cases and the word loop area. We did this because we want the function to be executed continuously. That means in a loop. If we wanted to execute the function only once at the beginning of the program, we would have put it in the void setup scope. Some very basic and important, as

well as already declared and thus ready to use functions utilized in Arduino programming are digital. Read to read the digital input to get to the right to right to a digital output pin mode. To assign an order, make a pin connector of the port, either an input or an output pin and agree to read the analog input and look right to right analog output. Stop every tone of a passer with no tone to start a tone in the bus or reduce tone to read a pulse on a pin. Use pulls in. Pulls in long. It's used to read long pauses to shift the pile of data we use either shift in or shift out. Random to find a random number within the bounds. To make the program wait for a certain time, we use delay the number we put in the brackets then describes the waiting time in milliseconds . For example, delay 1000 means coat weights, 1000 milliseconds equals one second. To make the program breathe in microseconds, we use delay microseconds. Tariq, the time that has passed since the program was started, micros, microseconds and millions in milliseconds. To get the absolute value of a number, we use apps to specify constraints, we use constraints to find the maximum of two numbers. We use Macs to find the minimum of two numbers we use to calculate the power of a number. We use power to find the square of no use square to find the square root of a number. We use square roots. To calculate the value of a bit, we use it to set a specific bet to zero. We use it clearly to reach a single bit from a number. We use it to set it to one. We use it to convert a number into bits. We use the right to get the leftmost byte of a number. We use a high byte to get the byte on the far right of a number. We use Lopate to attach an external interrupt function to a pin user, attach interrupt to remove an external interrupt function from a specific point. We use detach, interrupt to stop the internal interrupt use interrupts and to stop them. We use no interrupt. Byte is used to convert a value to a byte char is used to convert a value to a character variable float is used to convert value into a float variable. Int is used to convert a value into an integer variable. Long is used to convert the value to a long variable, and string is used to convert a value into a string in the following. We will learn how to use some of these functions using example projects, so don't worry if you don't

know specifically how to use the functions, operators and conditions yet. Before we get to the project, let's take a brief look at how to connect the port to the PC and load some program code onto the Arduino board to connect the Arduino board with the PC or the Arduino IEEE software. We first have to connect the Arduino board with a USB cable to the PC. Then we open the menu tools in the Arduino IEEE in the menu bar and select the correct board type in our case. The Arduino Uno and the menu board. In the next step, we have to make sure that the correct use port of the PC is assigned. We can also determine this on the tools in the Port submenu. This menu item can be found directly below port. Here, the port must be selected, which has Arduino or a similar designation. For example, also genuine, the Arduino is connected to this port on your PC.

Basic functions:

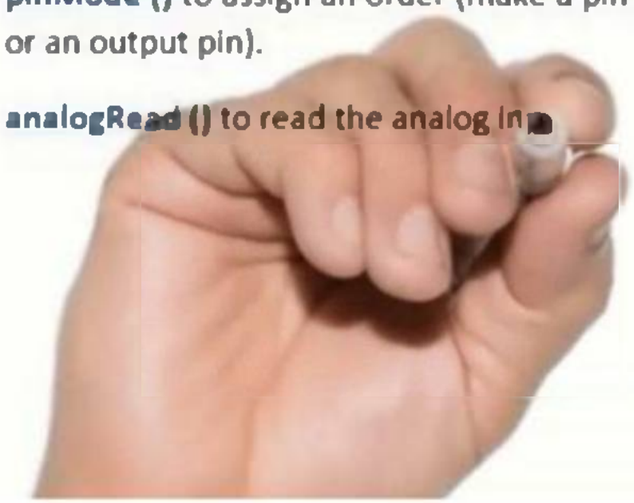
Some very basic and important, as well as already declared and thus ready to use functions utilized in Arduino programming are:

digitalRead () to read the digital input.

digitalWrite () to write to a digital output.

pinMode () to assign an order (make a pin connector of the board either an input or an output pin).

analogRead () to read the analog inp

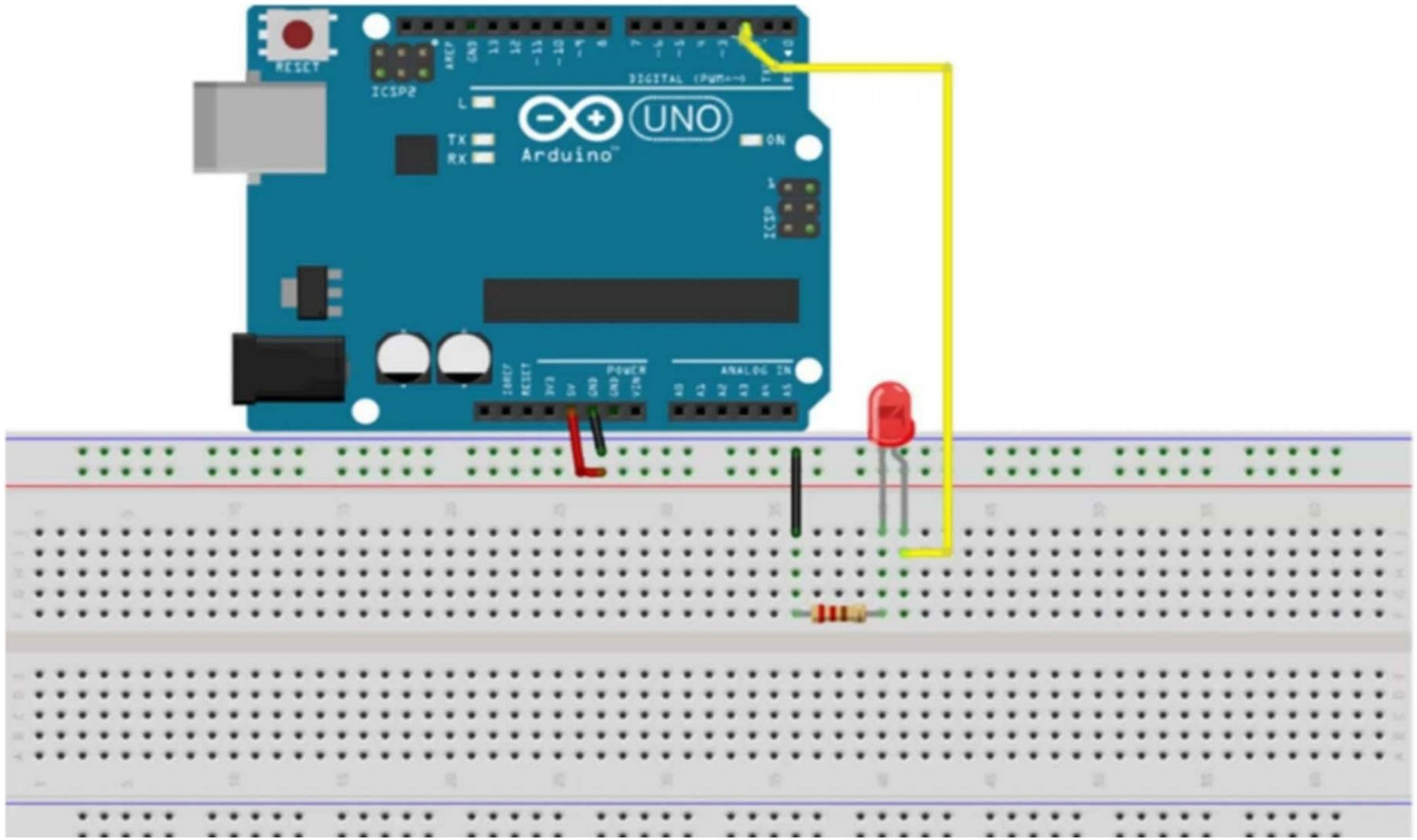


Now, the Arduino board is probably connected to the PC or software, and we can start by writing the first program code, compiling it and loading it on the board. We do this as follows. First, we write the program code or copy it into the text editor of the IEEE. If you have a program code already complete in itself, just delete the already existing text and then save the sketch. Then we pressed a small check mark to check and compile the code. If no errors were found, the message appears that the compilation was completed successfully. This may take some time, depending on the length. Finally, reload the coat onto the already new board by pressing the arrow, pointing to the right, then the Arduino will start executing the program code. Before that, you can open the serial monitor to monitor the communication between board and software.

PROJECT 1 A FLASHING LED

AND AN SOS SIGNAL

Project one, a flashing LED and an S.O.S. signal in this project, we will control the state of light. For this, we will use an Arduino known to switch the energy on and off with a delay of, for example, three seconds duration of the landis' glow or five seconds switched off state required components. One Arduino Uno. One breadboard for jumper wires led one 200 ohm. Or is this Tor? We connect the lady to the Arduino using a re-
sistor. The cables and the breadboard.



As shown for this, we first supply the breadboard with power. We connect the RET cable to the five volt pin of the Arduino board and the other end of the cable we plug into the breadboard, as shown in the picture. We also connect the black cable to the key and deepen of the Arduino board and the other end of the cable we plug into the breadboard, as shown in the picture. Then we place the LG Shortall to the resistor and the

resistor is shown and connect the lady with a black wire to the ground of the board. We need the resistor to limit the current here. We use ohm's law and the Formula R equals U divided by R stands for the resistance, U for the voltage and I for the current. Finally, we need a yellow cable that can also be another color which goes from the energy to the board pin to. Program code for the Arduino IEEE. We first declared the wearable and the deep pin and the sign tore the pin to which we have connected the lady pin to. Here we enter the setup code to be executed only once. We want the lady pin to be defined as an output pin, that means to receive an output signal so that the alert lights up. That means. Here we enter the main code to be executed in a loop. First supply deepens with five volt switches and on. Then three thousand milliseconds, three seconds. Then supply deepens with zero world turns off. Then reached 5000 milliseconds, five seconds.

```
// We first declare the variable "led_pin" and assign it to the pin to which we have connected the LED (pin 2)
const int led_pin=2;

void setup() {

// Here we enter the setup code to be executed only once. We want the led_pin to be defined as an output pin, i.e., to receive a
output signal so that the LED lights up, i.e. :

pinMode(led_pin,OUTPUT);
}

void loop() {

// Here we enter the main code to be executed repeatedly (in a loop)

digitalWrite(led_pin,HIGH); // first supply led_pin with 5V (switches LED on).
delay(3000); // then wait 3000 milliseconds (= 3 seconds).
```

The Loop then executes the program code rapidly. That means the lady rapidly switches off and on with the previously defined delay and of the program code. As an exercise, try switching the lady on and off at this point so that a signal is sent. It's a signal three times short, three times long, three times short, long signal two seconds short signal, one second distance between short and long point five seconds distance of five seconds between several. The solution will follow shortly. Now, the solution follows, we only have to change the part in white loop in the previous program code for the S.O.S. signal. This could look like this. A short signal is defined in the following, for example, with one second along one with two seconds elvedi on between the signals, there should be 0.5 seconds for disconnection of. The three short signals follow first.

```
void loop()
{
// The three short signals follow first:

digitalWrite(led_pin,HIGH); // first supply the led_pin with 5V (switches LED on)
delay(1000); // wait for short signal 1000 milliseconds (= seconds1)
digitalWrite(led_pin,LOW); // then supply led_pin with 0V (switches LED off)
delay(500); // wait for e.g. milliseconds500 to separate (= seconds0.5)

digitalWrite(led_pin,HIGH); // supply led_pin with 5V again (switches LED on)
delay(1000); // wait for short signal 1000 milliseconds (= 1 seconds)
digitalWrite(led_pin,LOW); // then supply led_pin with 0V (switches LED of)
delay(500); // wait for separation e.g. 500 milliseconds (= 0.5 seconds)

digitalWrite(led_pin,HIGH); // supply led_pin with 5V again (switches LED on)
delay(1000); // wait for short signal 1000 milliseconds (= 1 seconds)
digitalWrite(led_pin,LOW); // then supply led_pin with 0V (switches LED off)
```

First supply, the alleged weapon with five volt switches already on wait for short signal, 1000 milliseconds one second. Then Supply Ella deepened with zero world switches, Eladio off. Well, for example, five hundred milliseconds to separate point five seconds. Supply deepens with five faults again switches already on. Wait for a short signal. Thousand milliseconds one second. Then supply deepened with zero volt switches, any of. Wait for separation. For example, five hundred milliseconds points to five seconds. Supply of the Dippin with five volts again switches a lady on. Wait for a short signal, 1000 milliseconds, one second, then supply deepens with zero volt switches off. Wait for separation, for example, five hundred milliseconds, 0.5 seconds. Then the three long signals follow. First supply and the deep end with five volt switches already on. Wait for a long signal. Two thousand milliseconds. Two seconds then supply an LTE pin with zero world, which is an LCD of. Wait, for example, 500 milliseconds for separation point, five seconds. We need this paragraph two times more.

```
// Finally, three short signals follow, just like in the first section:

digitalWrite(led_pin,HIGH);
delay(1000);
digitalWrite(led_pin,LOW);
delay(500);

digitalWrite(led_pin,HIGH);
delay(1000);
digitalWrite(led_pin,LOW);
delay(500);

digitalWrite(led_pin,HIGH);
delay(1000);
digitalWrite(led_pin,LOW);
```

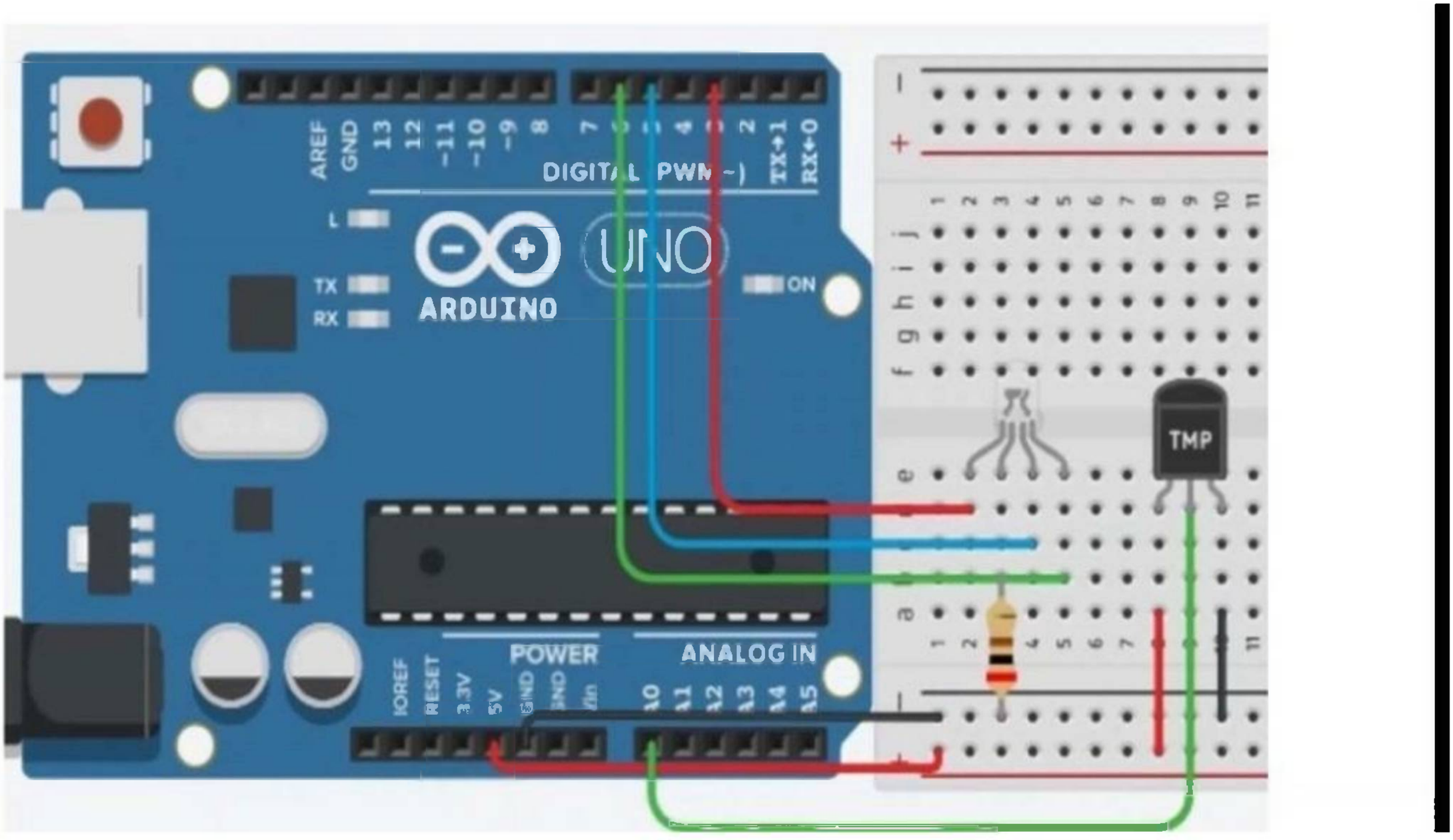
Finally, three short signals follow, just like in the first section. To separate between several signals, for example, wait five thousand milliseconds, five seconds. The loop function then executes the S.O.S. signal rapidly and permanently. When you try the code, don't forget the rest of the code structure as in the first project. Just replace the part inside the void loop from the first project with the code from here. Excellent.

We have successfully completed the first protection. Let's move on to the second project, this one will be a bit more difficult.

PROJECT 2 TEMPERATURE BASED LED LIGHT

In this project, we will control the state of an archipelago based on the temperature well, you detected by a temperature sensor when the temperatures are high, the lights would turn red. On the other hand, when the temperature is low, the lady should turn blue when the temperature is optimal. The light shall be green. Required components. One Arduino, Uno one Brett bought jumper wires, one RGV, a 3D one 200 Ohm resistor, one three five temperature sensor. But Elham, 03:05 provides an analogue output voltage that is linearly proportional to the temperature in degrees Celsius. The temperature ranges from minus 50 to plus 155 degrees Celsius to convert the analog voltage into a temperature. A scathing factor of 10 movable per degrees Celsius is required. The assignment of output, voltage and temperature in Celsius can be read from a linear function. For example, the 580 volt signal of the sensor corresponds to 50 degrees Celsius, and Agip can shine in three colors, namely red, green and blue. The lady has two more connections than normal energy, and the color of the light depends on which connection is supplied with current. To control the energy you need pins three six five, which are actually digital pins that also allow pools with modulation. You can recognize this by the small print, and we've also at nine, 10, 11. First, we supply the bread with power again for this, we connect the red cable to the five pin of the Arduino board and the other end of the cable, we plug into the breadboard, as shown on the picture. In addition, we connect the black cable to the T and deepen the Arduino board and the other end of the cable, we also plug into the bread board that's shown in the

picture. Then we at the LNG and the temperatures into Tempe, make sure that each leg is correctly seated in the connectors. You need to be careful here not to break any of the legs when bending them into place. The two outer legs of the temperature sensor are supplied with power by connecting a black and the red wire to the plus and minus pole of the breadboard. We connect the middle leg of the temperature sensor with a coloured cable to the connector, a zero of the Arduino.



We connect the legs of the RCMP, a lady with coloured cables to the digital pins. Three, five and six of the Arduino board. Furthermore, we need a resistor to connect one leg of the archipelago to the negative pole of the breadboard and thus also to the pole of the Arduino. Program code for the Arduino. IEEE, we first declare our wearables by assigning them to the respective connection pin. The abbreviations are b key to stand for red, blue, green and temperature, respectively. Connection for it, glow of the PIN three, connection for blue, glow of and PIN five. Connection for green glow of energy at six. Temperature sensor is connected to the PIN. Here we enter the set up code to be executed only once. We want all pins connected to the ready to be defined as output pins and the pin connected to the temperature sensor to be defined as input pin. First, we need the above command for the serial interface that I read nine thousand six hundred thirty seconds. This starts the communication between PC and Arduino board, and the temperature is transmitted to the serial monitor in the IEEE Board rate nine thousand six hundred. Definition of pin is PIN three as output pin definition of the PIN B. That means the PIN five as output pin definition of PIN G. That means PIN six is output pin definition of pin tee. That means pin a zero as input pin. Here we enter the main code to be executed repeatedly in a loop. First, we define the following code: the function temp, which we will need in a moment for this purpose. The microcontroller is to read the value in PIN a zero. I'm 35 and I look at temperatures in the. Shows us the temperature and the surreal monitor. In the following we create and if and several as if conditions which controlled the LRT in several steps, depending on the temperature of the sensor, therefore we use look right instead of digital right here only. Zero volt or five volt would be possible. See first project.

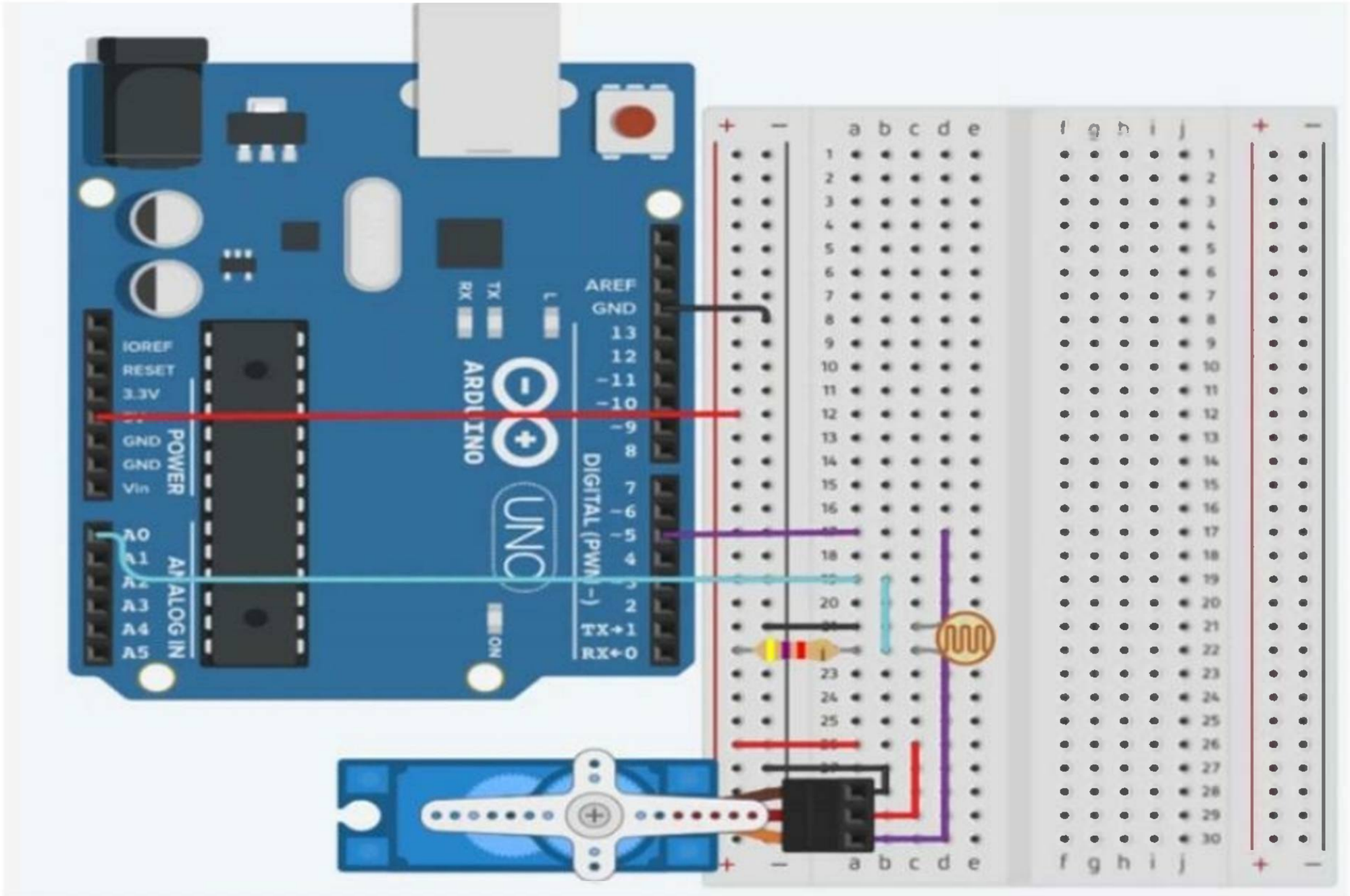
```
void loop()
// Here we enter the main code to be executed repeatedly (in a loop).
{
// First we define with the following code the function temp, which we will need in a moment. For this purpose the microcontroller
read the value in PIN A0.
 int temp = analogRead(t); // our LM35 is an analog temperature sensor
 Serial.println(temp); // shows us the temperature in the serial monitor (mV !)
// In the following we create an "If" and several "Else if" conditions, which control the LED in several steps depending on the tempera
of the sensor. Therefore we use "analogWrite" instead of "digitalwrite" (here only 0V or 5V would be possible; see first project).
 if (temp > 300)
// When the output voltage rises above the value 300mV (approx. 30° C according to the scaling factor), the following values shoul
applied to the pins
 {
  analogWrite(r, 255); // a strong red color; values gradable from 0 to 255
  analogWrite(b, 0);
  analogWrite(g, 0);
}
```

When the output voltage rises above the value 300mm 300-year-old approximately 30 degrees Celsius, according to the scaling factor, the following values should be applied to the pins. A strong red color? Well, you scurried up from zero to 255. If the voltage is above the value of 250 million volt, approximately 25 degrees Celsius, on the other hand, the following values should be applied to the pins in the following anal-

ogously a medium red color. A faint red color. A strong green color. A mixture of blue and green. A strong blue collar. The Loop then executes a program code repeatedly that means the temperature is measured and interpreted continuously, and the signal is passed to the deep end of the program code.

PROJECT 3 LIGHT-DEPENDENT CONTROL OF A MOTOR (BLIND MOTOR)

In this project, we want to control the outlines of a window with the help of a servo motor and an elder sensor. This should happen depending on the amount of light coming in from outside. Required components. One Arduino Uno, one breadboard, one photoresistor, the R-Centre one several motor, nine champ of wires, one resistor 4.7 kg As the name suggests, a photo resistor can be thought of as a simple resistor that has the special feature of changing its resistance value, depending on the amount of incident light. The less light falling on the sensor, the higher the resistance becomes, the more light falling on the sensor, the lower the resistance comes. The sensor is based on the photoelectric effect. As usual, we first supply the Brett Port with power to do this, connect the black and the red cable from the tee and deepen and the five volt pin of the Arduino board to the positive and negative pole of the breadboard. By the way, it doesn't matter which TNT pin of the port is used.



Then we insert the photo resistor and the several motors, as shown in the picture. There is still one resistor missing and the rest of the wiring you can do as shown in the picture. Program code for the Arduino I.D.. First, we include the required library for the several more tours in our program code. If the I.D. gifts an error message when compiling, you must first install this library while the library manager. Then recreate the

several objects so that we can control the servo motor. Then we declare the connection points for several motors and sensors for tourists too. Then we have to declare variables for the position of the cerebral and for the properties of the photoresistor. Very able for saving the survival position, very able to store the survival position at max light.

```
// First we include the required library for the servo motor in our program code. If the IDE gives an error message when compil
you must first install this library via the library manager.

#include <Servo.h>


// Then we create a "Servo Object" so that we can control the servo motor.

Servo myservo;
// Then we declare the connection pins for servo motor and sensor (photoresistor)

const int servo_pin=5;
const int sensor=A0;


// Then we have to declare variables for the position of the servo and for the properties of the photoresistor

int pos = 0; // Variable for saving the servo position
int light_pos =0; // variable to store the servo position at max. light
int max_light = 997; // This is the value we define as maximum light incidence
int intensity=0; // Light intensity at any position
```

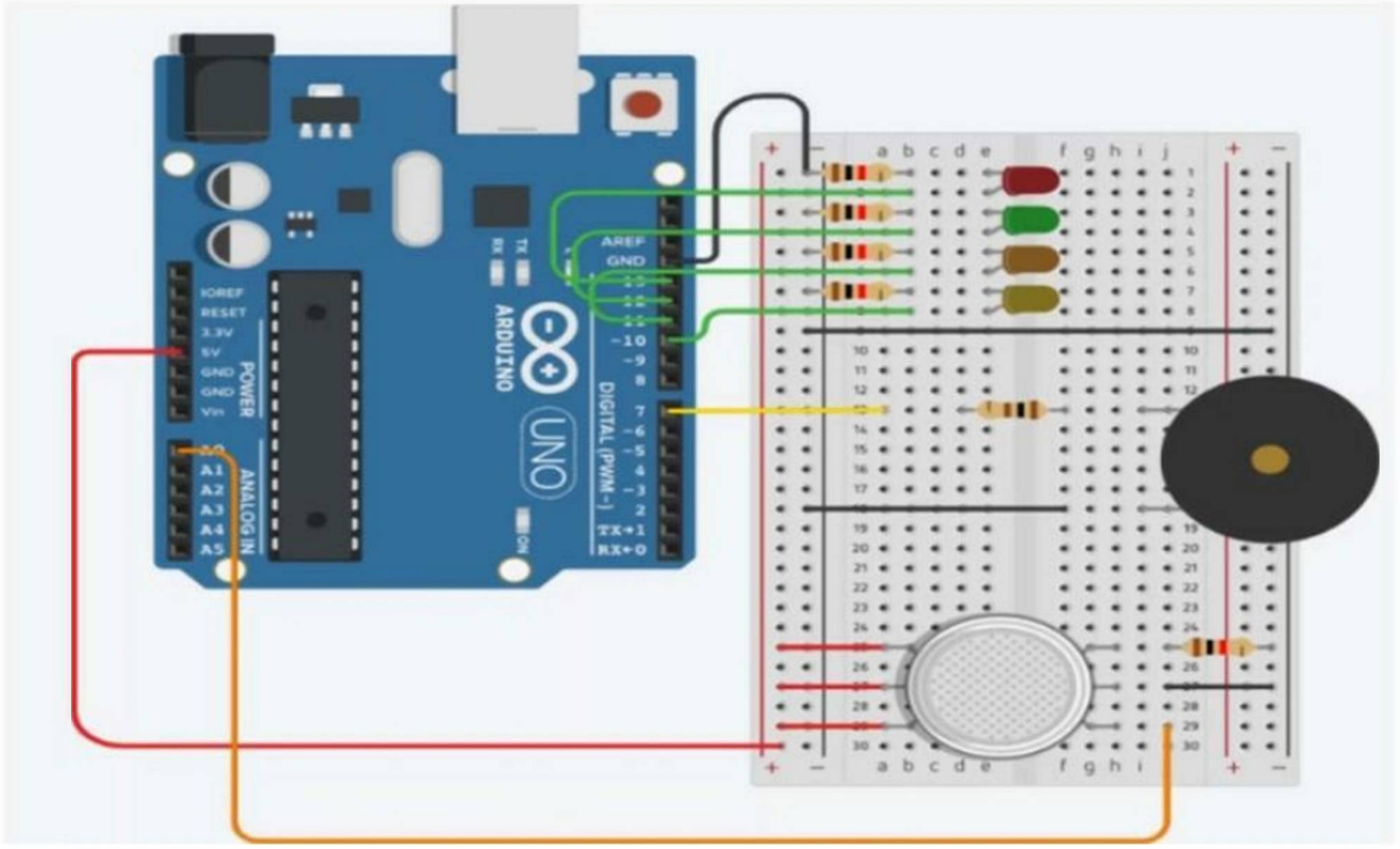
This is the well you would define as maximum light incidents. Light intensity at any position. Here we enter the setup code to be executed only once. Here we want to connect the server with the server object and start the serial communication. Connects to several to pin five with a several object start, serial communication said board rate to the seller of the serial monitor. Eight zero is defined as an input pin. Here we enter the main code to be executed repeatedly in a loop. For each position, from zero to 180 degrees of the several modes to execute the following code posts plus equals one means pass equals posts plus one. That means posts equals zero plus one equals one. Said position of several motors and you will use them. That position for several zero to 180 degrees. Reid sends a value and overwrite variable. Check if maximum light incident is reached, if this condition is true, it means that the output value for the variable max light equals nine nine seven is greater than the value that the sensor is currently outputting.

```
// Read sensor value (light incidence) also in reverse loop
for (pos = 180; pos >= 0; pos -= 1) // position from 180 degrees to 0 degrees
{
myservo.write(pos);
 intensity= analogRead(sensor);
   if (max_light>intensity)
 {
  max_light = intensity; // store new value for maximum brightness
  light_pos = pos; // save the servo position at maximum light incidence
 }
  Serial.println (intensity);
  delay(50);
}
```

That means the maximum brightness would not have been reached. Store new values for maximum brightness. Safe to several positions. Shows us the intensity in the serial monitor weighs 50 milliseconds until the cerebral motor has reached the position. Right center, well, you light incidents also in reverse loop position from 180 degrees to zero degrees. Store new values for maximum brightness. Saved the survival position at maximum light incidents. In the following section, the server was to move to the position that was saved if the following condition is met. Check if the now new store value of the variable is not equal to the initial value of the variable darker, the maximum light. Go to the defined position of maximum light incidents. Shows us the sense of, well, you and the serial monitor. Wait, 20 seconds. Check whether the light intensity has changed or not. Float equals floating point numbers. Checking the change of the light intensity. Wait, three seconds. Reset initial variables. End of the program code.

PROJECT 4 GAS DETECTION ALARM

We will build a gas detector that will sound an alarm if it detects a gas leak. The alarm will sound until the gas leak is stopped. In addition, LEDs are triggered depending on the amount of gas that the sensor detects if there is a lot of gas leaking. All four ladies should light up. If there is little gas, only one of the four lights should light up required components. One Arduino Uno on a breadboard, one gas sensor, one buzzer, 14 jumper wires, five resistors, one kiloton, four Lydian gas sensors, one resistor, one home at four percent four energy. We connect all the components and the Arduino together on a broad breadboard, as shown on the picture. Make sure that you use the correct pins.

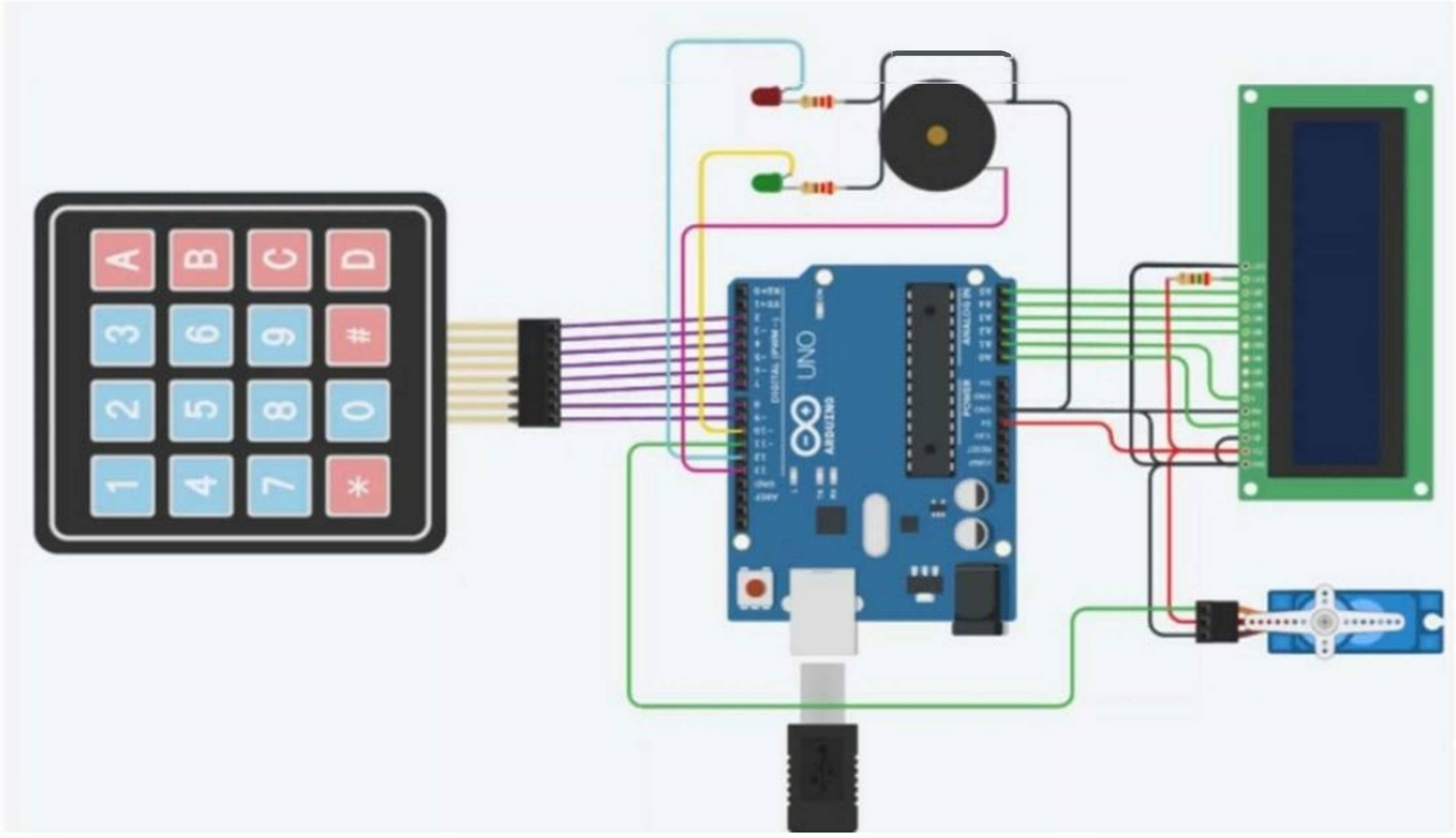


You can also arrange the components differently if you like. However, the circuit should remain comparable. But then you might have to change the variables or names in the following code program code for the Arduino I.D.. We first declare our liable for the gas center assigning the connector pin A0. Here we enter the setup code to be executed only once. We want pin a zero, which is connected to the gas sensor to be

defined as input pin. Also, we want all prints connected to the buzzer, or it is to be defined as an output pin. Define the guest center as input. Defined perception as output. Define low deepen as output. Then we start the communication with the serial interface data rate nine thousand six hundred per seconds with the following code. This starts the communication between ordering a port and PC, and that data is transferred to the serial monitor in the. Here we enter the main code to be executed repeatedly. First, we declare a variable that should reach the center. Then we generate the tone at seven with 220 Hertz for 100 milliseconds. Tone is activated as soon as the sensor registers measured while you. That means guess no matter in which quantity. Then wait 300 milliseconds. Then no tone should be applied to 2.7 seven anymore. Then 10000 milliseconds should be waited for. In the following we create and if and several as if conditions which give us different control of the ladies, depending on the gas value measured by the sensor, we use digital right here. If value is greater than 75, then switch on all these. Otherwise, if between 50 and 75, then. Switch on only three LEDs. Otherwise, if between 25 and 50, then. Switch on only to early this. Otherwise, if between zero and 25, then. Switch on only one lady. End of the program code.

PROJECT 5 PASSWORD PROTECTED MECHANICAL SYSTEM

In this project, we will create a system that is protected by a password. It will remain locked until the user enters the correct password. When the correct password is entered, the cerebral motor will move and open the system. We will assign different passwords to different uses. Each user will have their own user ID and password. The system will only be unlocked if these two security features match and are correct.



Moreover, after multiple incorrect entries, the red light and buzzer shall be activated and the entry shall be locked for 30 seconds. On the other hand, if the input is correct, the green agenda should activate the required components. One Arduino Uno on breadboard one Keep It one LCD display various jumper wires three resistors two LCD one busser one several motor. We link all the components and the Arduino to-

gether, as shown in the image. We don't use a breadboard this time because almost all the connections are between the Arduino and the individual components anyway. But you can also use a breadboard for this if you want to. Before we start with the program code, we have to install the required library for the Keep It, Keep It by Mark Stanley and Alexander Breivik. We do this either via the library manager or research the corresponding zip file online using Google and find it. For example, here Playground Dot, Arduino Dot CC and loaded into the Arduino IEEE. You may need to do this for the display as well. Program code for the Arduino I.D.. First, we include the necessary libraries for the servo motor, the Keepit and the LCD display in our program code. Then we create a server object so that we can control the servo motor. In the following we declare our variables for the green and the red lady, as well as for the buzzer by assigning the connector pins 10, 12 and 13. We then declare the number of rows and columns of our keyboard field for each. With Kemp, we define the keys on the key pet that can be pressed according to the road and columns as they appear on the keyboard.

```
// First we include the necessary libraries for the servo motor, the keypad and the LCD display in our program code.

#include <LiquidCrystal.h>
#include <Keypad.h>
#include <Servo.h>

// Then we create a "Servo Object" so that we can control the servo motor.

Servo myservo;

// In the following we declare our variables for the green and red LED, as well as for the buzzer by assigning the connector pins 10, 12, 13.

const int gled=10;
const int rled=12;
const int buzzer=13;

// We then declare the number of rows and columns of our keyboard field (4 each)

const byte numRows = 4;
const byte numCols = 4;
```

Then we need a coat that maps the connectors of the carpet with the connectors on the Arduino rows zero to three columns zero two three, the following code initializes an instance of the keypad Chris. The following code creates a variable for the LCD display with the numbers of the pin interfaces assigned to the LCD display. In the following, we assigned the user IDs and passwords. In addition, we declare the following

variables. Here we enter the setup code to be executed only once. Start surreal communication. Initials, LCD display. Wait, five milliseconds define green lady variable as output, define a red LED variable as output, define busser as output, and set the position of the text on the display column row. Show text. Enter the code. Set the position of the text on the display column row display text to open lo Now we need code for the servo motor. Several motorists connected to 11 to several motorists should move to position five degrees.

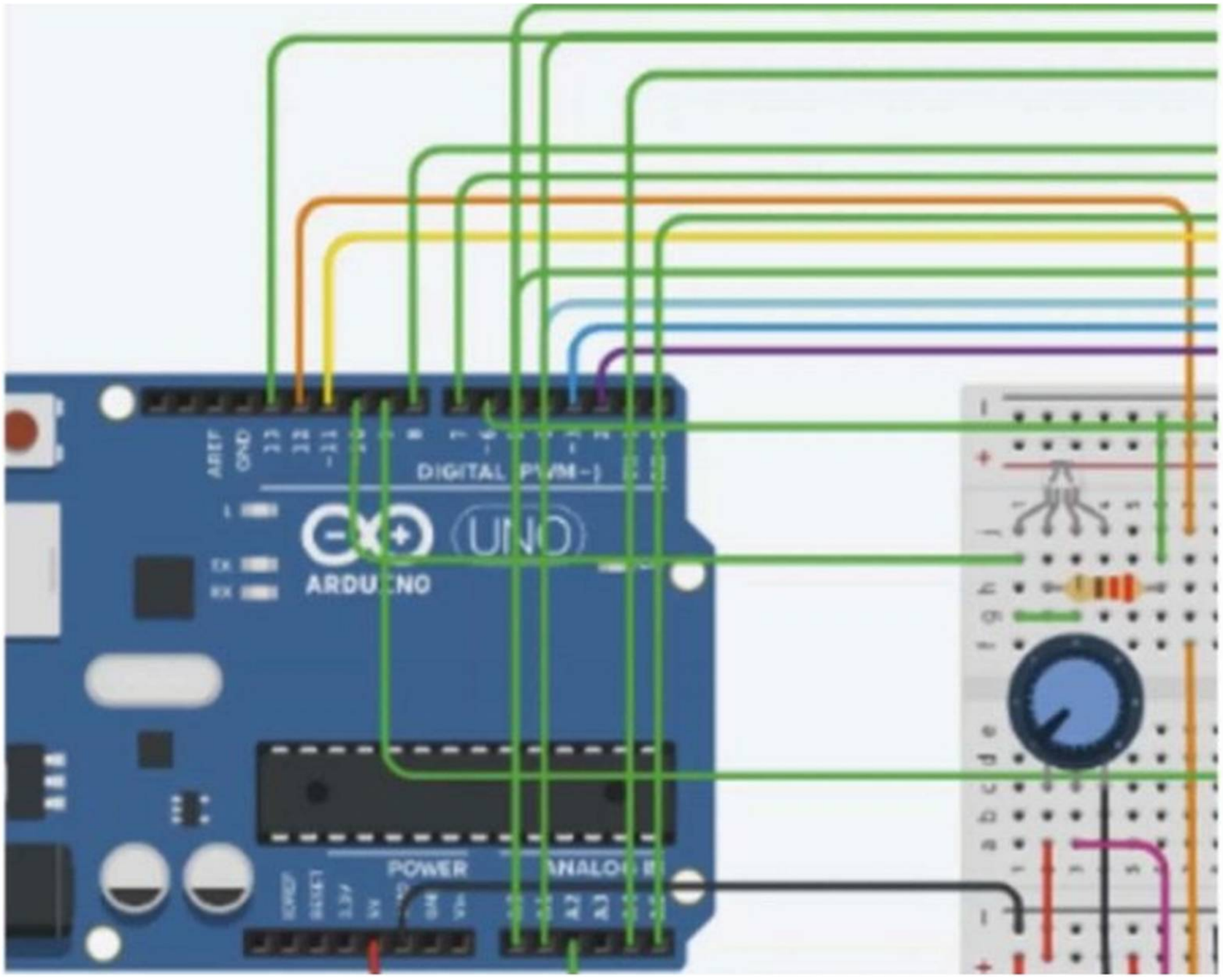
```
void setup()
// Here we enter the setup code to be executed only once.
{
 Serial.begin(9600); // start serial communication
 lcd.begin(16, 2); // Initialize LCD display: lcd.begin(columns, rows of display)
 delay(500); // wait 500 ms
 pinMode(gled,OUTPUT); // define green LED variable as output
 pinMode(rled,OUTPUT); // define red LED variable as output
 pinMode(buzzer,OUTPUT); // define buzzer as output
 lcd.setCursor(0, 0); // Set the position of the text on the display (column, row)
 lcd.println("ENTER THE CODE"); // Show text : "Enter the code" (display)
 lcd.setCursor(1 , 1); // Set the position of the text on the display (column, row)
 lcd.println("TO OPEN LOCK"); // Display text : "To open Lock" (Display)

// Now we need code for the servo motor
myservo.attach(11); // The servo motor is connected to pin 11
 myservo.write(5); // The servo motor should move to position (5°)
 delay(1500); // wait 1500 ms
```

One thousand five hundred milliseconds end of the code four void setup. Here we enter the main code to be executed repeatedly. Below is the code that verifies the entire user ID and password. The following message should be shown on the display. If the password was entered incorrectly, the position of the following text displayed. Said position of the following text display text waits three seconds. Allow input to start again. Said position of the following text displayed text said position of the following text display text. Wait point, five seconds. Start verification from scratch. If the check was correct, but the password was not or not yet, the following should be displayed. Said position of the following text display, text said position of the following text display text. If the attack was correct and the password check was also correct, the following should happen. Following is to code that will be executed if the passport and I.D. are entered correctly. Green added You should light up that position of following text display text said position of the following text displayed text wait five seconds. Said position of the following text display, text said position of the following text several motor opens 180 degree movement. One point five seconds. If the wrong input is made several times, the input should be locked for 30 seconds and threat led as well as the buzzer should be activated. The following is the code for this set position of the following text displayed text said position of the following text display text. Right, algae should light up and passa should be activated. Subsequently, wait 30 seconds and display text. Said position of the following text display, text display, number of seconds remaining. Display text switch off a red LED. Switch off buzzer said the position of the following text displayed. Said position of the following text, display, text and of the program code.

PROJECT 6 REMOTE CONTROLLED UNLOCKING MECHANISM

This project, we will control the mechanism for opening and closing a gate with an air remote control to be able to open the gate. The code should be entered on the remote control. For example, one six five eight eight six three in addition to arguing and that is in the buzzer, shall accompany the process. Furthermore, a temperature sensor should monitor the ambient temperature and issue an error message if the temperature is too high. Additionally, an already shall be activated if the photo sensor measures only little ambient light. Furthermore, we install a button switch for manual operation. Required components. One Arduino Uno two reports, one air infrared remote control, one air infrared receiver sensor, one LCD display, various jumper wires, six resistors, one potential meter, one busser one several motor two archipelagos one DC mute motor one l two nine 3D motor driver one LDR center for two resistor one temperature sensor one button switch. Connect all components as shown.



Before we start with the program coat, we have to install the required library for the air infrared remote control by arming your camera. The best way to do this is to use the library manager, search for the library in the I.D. and load it. You may have to do this for the display as well. Program code for the Arduino IEEE. First, we include the necessary libraries for the servo motor, the air remote control and the LCD display in

our program code. In the following, we first declared the variable for the air sensor by assigning the Connector eight to. We also need the following two expressions for the arsons or. Then we create, again, several objects so that we can also control the servo motor. The following code creates a variable for the LCD display with the numbers of the pin interfaces assigned to the LCD display. Below, would you find the function for opening? Tone at eight with 220 hertz for 100 milliseconds. The several motors should move to position zero degrees. Wait, 15 milliseconds. Declare a variable temp right temperature, well, you. As long as the temperature is above 250, 25 degrees since 10 million wrote 10 multiples equals one degree Celsius shall activate at PIN 13. Disable it, been a four. Said position of the following text showed text said position of the following text display text. Reached temperature value. Activate LTE at a four, deactivate it, pin 13.

```
// First we include the necessary libraries for the servo motor, the IR remote control and the LCD display in our program code.

#include <LiquidCrystal.h>

#include <IRremote.h>

#include <Servo.h>

// In the following we first declare the variable for the IR sensor (receiver) by assigning the connector pin A2.
int RECV_PIN = A2;

// We also need the following two expressions for the IR sensor (creates objects)

IRrecv irrecv(RECV_PIN);

decode_results results;

// Then we create again a "Servo object" so that we can also control the servo motor
Servo myservo;
// The following code creates a variable for the LCD display with the numbers of the pin interfaces assigned to the LCD display

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// Below we define the function for opening :

void door_open()

{

 tone(8, 440, 100); // tone at pin 8 with 220 Hz for 100 ms

 myservo.write(0); // The servo motor should move to position (0°)

 delay(15); // wait 15ms

 int temp = analogRead(A1); // declare variable "temp"; read temperature value
```

Activate PIN seven motor. Deactivate pin eight motor wait, three seconds. Set position of the following text display text deactivate PIN seven, enable PIN eight wait three seconds. Deactivate PIN seven, enable pin eight weight one hundred milliseconds. The several move to move into position, nine to decrease. Wait, 15 milliseconds and the following, would you find the function for the lady when there is too little light? Declare variable light red light. Well, you. Flight quantities over 500 then. Deactivate Penton said the position of the following text displayed text your flagged quantities under 500, then. Activate Penton. Said position of the following text display text. Here we enter the setup code to be executed only once. Initialize LCD display columns, rows of the display. Stop the eye or sensor with the following code. The survival motor is connected to the servo motor and should move in position 90 degrees. Wait, 15 milliseconds. Activate at four, deactivate LSD at PIN 13. Here we enter the main code to be executed repeatedly. If Terminal 85 gets current five volts because it is high, then.

```
void setup()
// Here we enter the setup code to be executed only once.
{
// Initialize LCD display: lcd.begin(columns, rows of the display)
 lcd.begin(16, 2);
// Start the IR sensor with the following code
 irrecv.enableIRIn();
// The servo motor is connected to pin 6
 myservo.attach(6);
// The servo motor should move in position 90° degrees
 myservo.write(90);
 delay(15); // wait 15 ms
 digitalWrite(A4,HIGH); // Activate LED (RGB leg) at pin A4
 digitalWrite(13,LOW); // Deactivate LED (RGB leg) at pin 13
}
void loop()
// Here we enter the main code to be executed repeatedly (in a loop)
{
out_door_light();
 if (digitalRead(A5)==HIGH) // If terminal A5 gets current (5V, because HIGH), then:
```

Set position of the following text showed text. Execute the open function set position of the following text. The following code says the following if a signal was given by the remote control or received via the air sensor. And if the signal corresponds to the following code one six five eight eight six three. Was an air signal received a match code? Said position of the following text showed text executing the open function. Received next, well, you. Said position of the following text showed text weight one hundred milliseconds end of code.