# RC-CAR PROJECT

## IMPLEMENTING IR SIGNALING

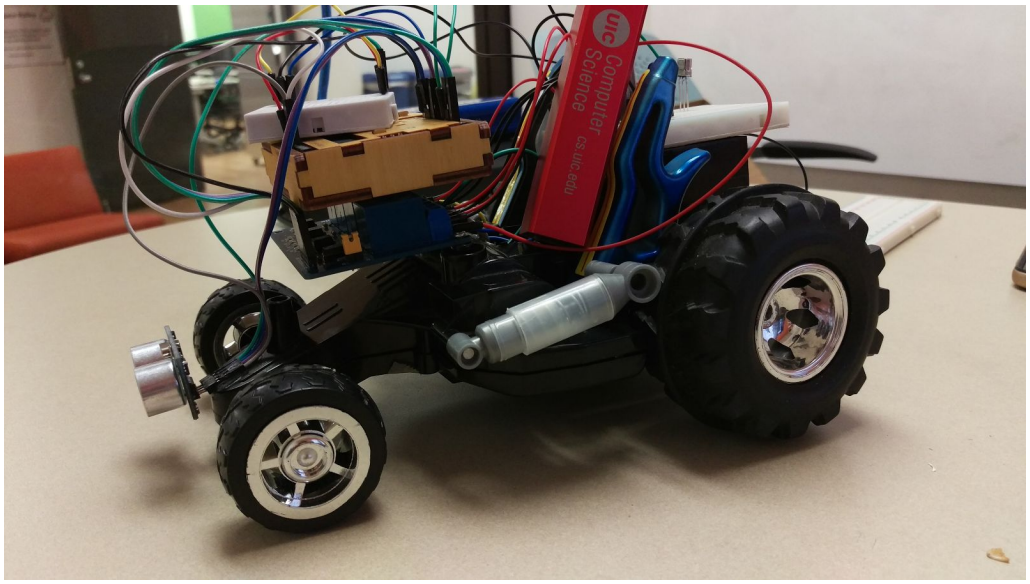
Basu, Siddharth & Kupiec Bart

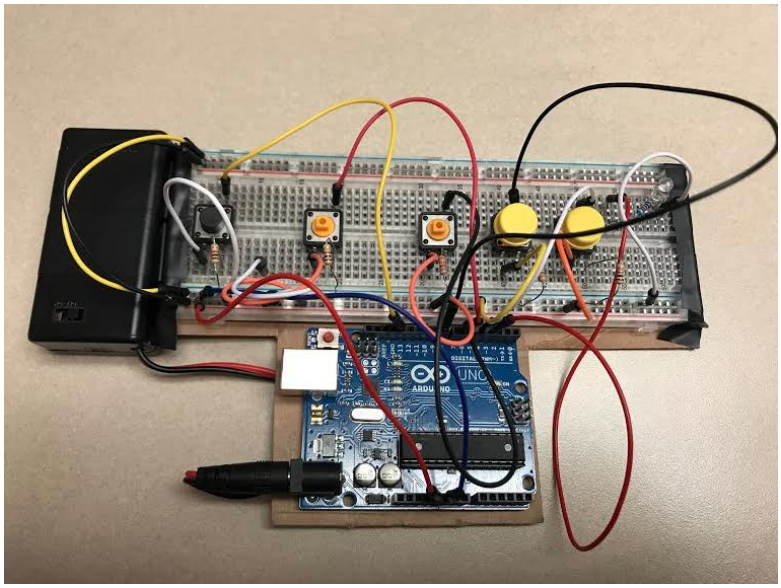
[CS362] | FINAL PROJECT WRITE-UP

## Table of contents

## Overview

The goal of our project was to take an existing RC car, remove any components that controlled the RC car and replace them with our own. By doing this we can make the car "smarter" because we can add our own sensors, and have finer control of how fast we want to go. Then we can also create an "off" button that stops the RC car when something happens , for example when it reaches a wall.



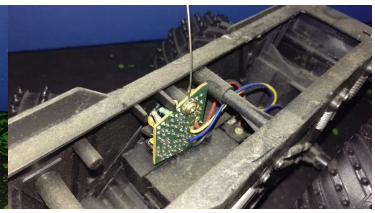
**Purpose**

The purpose of our project was to implement a unique type of RC car which uses IR signals to be transmitted from a remote wirelessly to the RC car. We wanted to make our own remote, and our own RC car.

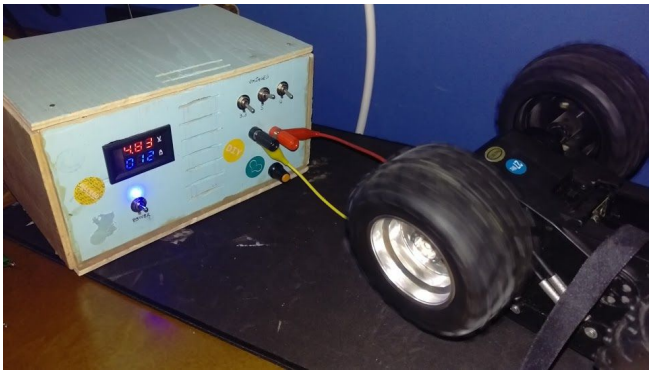
**The RC Car**

Starting out the RC car, we knew the best approach would be to go based off an existing platform. We didn't want to worry about wheels falling off or the frame breaking due to poor construction on our part. So the first phase of our project was to find a good framework for the RC car, and after asking around I was able to find 3 candidates ( one of them was in a very poor state so it wasn't even considered for this project. )

*The yellow pickup truck*

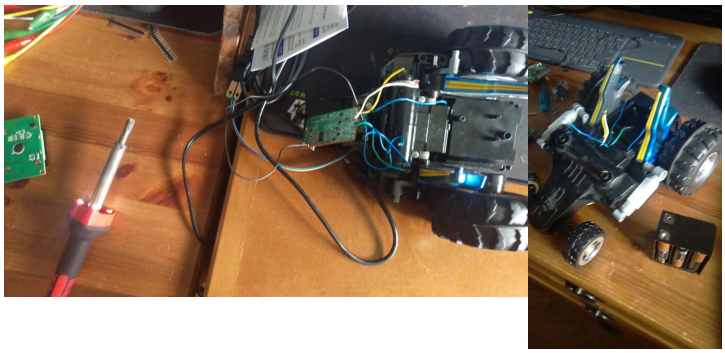


So the first I started working on was this yellow pickup truck. After getting it open I was able to see how it functioned. The one thing that came quickly apparent was that it only used one motor to spin two of the back wheels. The car was able to backwards and forwards on that one motor, but this was a huge drawback. The upside was that the tires were able to spin at a reasonably low voltage, and the simple design could offer benefits. At this point, I



was curious to see how the other car worked. So for now I put the yellow pickup aside, and focused on the other RC car.

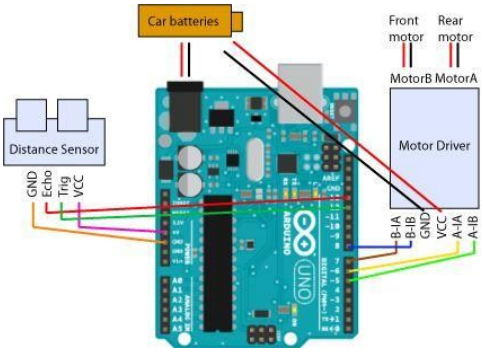
*The blue car*



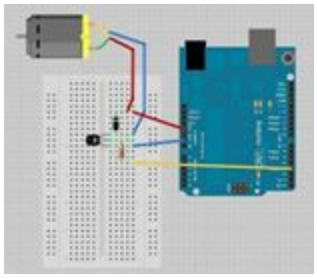
The other RC car was much better laid out, as inside there was clearly two motors. I proceeded to taking out the internals by desoldering them. So all that was left was the car with just the wires to the motors. Also the battery bank that came with it could easily be used to power the motors later down the line. Now that we had a good foundation to work on, we can focus on the other details of the car.

**Making the motors work**

Before we proceed to make the motors work, we wanted to see the best ways to make them work. First I found a guide on how to change speed via serial input, and make it into a analog output on the motor, which would be great since we would be able to control the speed. [ Motors 2 ]

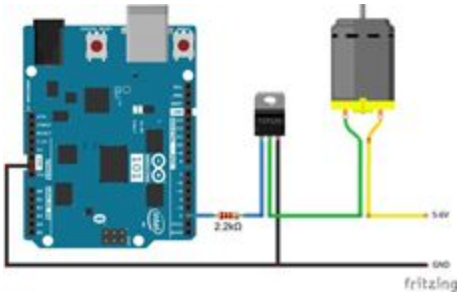



After looking around a bit online, I found a great resource about how to make my new transistor work. [ Motors 3 ] I used only the diagram from this website, as most of the code was only used as digital output and didn't use serial. What happened next? It worked perfectly! I could run on 12V from my power supply, and worked well even with the 5



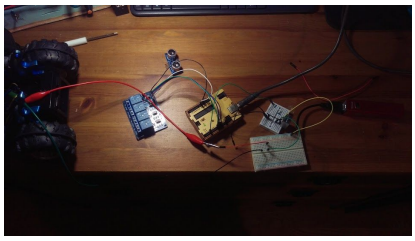
V. I think it might have been the high amps that caused this issue in the first place. So at this point in time, I can receive serial line input, and control a motor. This can be easily done for two motors, so the foundation for how to control the RC car has been made.

Since we want to have forward/backwards ability, and turning ability using two motors we must figure out how to do it. For us, the car can go in both directions, since the motors can both go forward or backwards. The car's motors are color coded. The car has two cables, green and blue, and depending on which ones I apply the voltage. Say I apply 5 volts to green and ground to blue, the wheels will spin forward. I was thinking of using 4 different transistors to control the speed for the motors, which would probably work really well, but since I only had this one, and since the way the remote will be designed, I decided to only keep one which will determine the speed, and then have 4 relays instead.

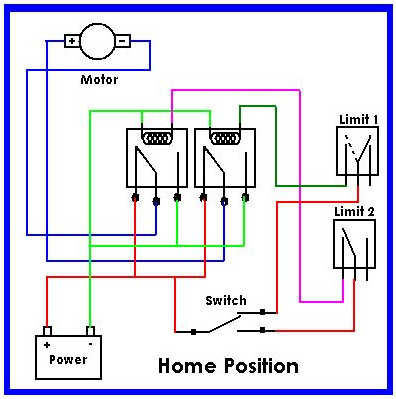



The transistor had a similar layout but now looked more like the picture on the right. Later on we got rid of the potentiometer to determine the speed and rather kept a single speed, since the potentiometer created problems for both the remote and the RC car it self.

## Relays



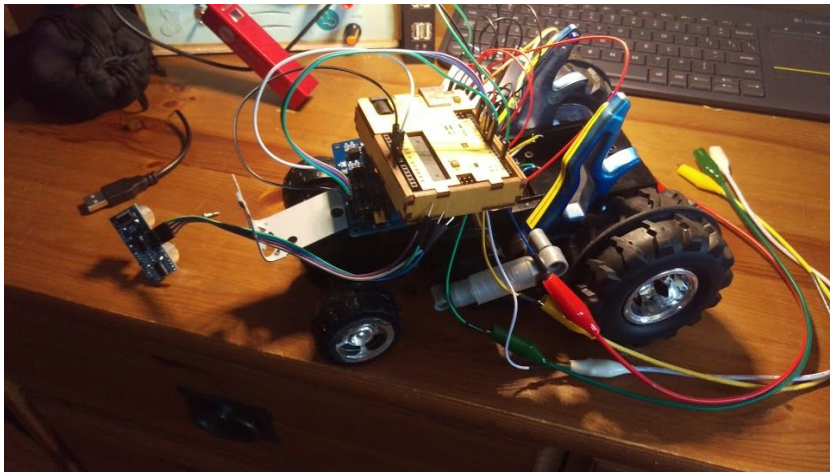
We made the decision to work with relays instead of the controllers that were in the example codes since relay modules are much cheaper, and are easily workable with, and made sure we don't have to concern ourselves with how much power is being delivered. So I made sure to buy a relay module of 4 ( since each wheel needs two ) and made sure that the rating for voltage/amps is higher than what we required, as I didn't want anything to blow up like it had earlier with the transistor. The way the relays work are by changing the polarity of the motor. You want to either have both off, or only one on at a time to prevent making a short circuit. In Arduino code we made sure that this wouldn't happen by having functions which made sure to turn off the relays that are not being used. The concept is shown best with the picture that we provided. [ link to picture ] Each cable of a motor would have it's own relay terminal, where the power from the batteries would go to every single first terminal of the relays. And then the ground would always be the last pin of the relays, this



made the overall product look very clean and it was very easy to tell which cable is what when looking at how they are connected.

This wiring caused a bit of issue, since the motor cables were so far, so we had to cut out a part of the RC car's plastic frame to make sure we can get them through.While doing this one of the cables was accidently cut, and other had a exposed part of it. So one was taped, while the other was removed and replaced ( the green cable, notice how different it is from the all the other original ). By uck the previous screw holes of the RC car lined up with the relay module, so it was very easy to screw in with the screws that were taken out earlier.

**Distance Sensor / Buzzer**



The sensor was really easy to hook up, the only problem with it later was the library that we tried to use with it was using the same interrupts as the code for the IR receiver [ Motor ] . We made sure the motion sensor would be as close to the front of the car as possible, to give us the best possible reading on where objects might be. After playing around and trying to make a bracket to hold in place I resulted to just using super glue and making the sensor stationary in the front, as I didn't superglue the actual sensor but rather the little cables that hold the pins inside of them ( which have a plastic casing).

The motion sensor will keep polling the distance of how far things are, and when that distance reaches less than 5 cm, a buzzer on the back of the Rc car will go off [ Buzzer ]. The buzzer emits a single tone, and keeps making a noise as long as the object is still in front of the

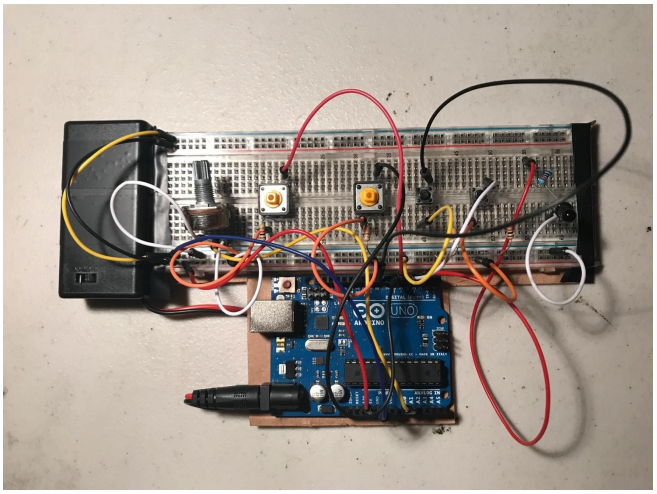
car. The car will not move at all while there is an object in the way, as all relays will be turned off completely, so nothing can be done. The hope was that the RC car would manually back up, but this proved to be much more harder to implement than initially thought. This is the reason we had abandoned that idea, and focused on other parts for the time being.

**Finishing the car**

When all the parts were put together, we made sure the car had enough power to run the motors. For this we just had used the battery pack that came with, which takes in 6 x AA batteries. This ran at 10 volts, and the perfect amount of amps to run the wheels, which meant that the motors would run at the speed that they were meant for. To power the arduino for the RC car we wanted to use a standard 9 volt battery, but for some reason using it caused the relays to not trigger properly. Instead we decided to use a small USB power bank ( one that I have received from UIC ) which gave enough power to make everything work properly. The car has it's own suspension, which makes controlling it a bit more difficult. But at the same time removing it makes it completely unusable.

We have experimented with the idea of having the tires be in a fixed place , by taping the blue "wings" of the car together and making them as stiff as possible, but this resulted in removing all the room we had in the back for the batteries and the components that resided there. So we decided to just let it be and give as much room to the components of the car as possible, since they are most of the bulk of the actual car. We took some measures to making things fit better: The relay board is screwed in directly into the car, making sure it won't go anywhere. Then the arduino itself is inside of a wooden encasing, to avoid any outside things that may

cause it problems, along with a strip of velcro that connects the top of the relay board to the bottom of the arduino.



**Making the remote**

Starting out the controller remote, we had used a breadboard layout to create a controller feel with buttons for different actions being sent to the RC-Car. We first wanted to utilize the potentiometer which would create ranges of speed and reverse functionality. Since the potentiometer range is from 0 to 1024. This creates an influx of large amounts of numbers to keep track of. To tackle this problem, we decided to take the values received by the potentiometer divided by 4. This reduced the scale of the potentiometer to 25% (0 to 256). Then further break down the scale to 5 sections in which it was broken into: reverse, stay idle, slow speed, medium speed, maximum speed. We also utilized a two button system in which one button would be used for left turn, which would shut off the motor for left side, and keep the motor running right. And also for the right button would have the right motor shut off and have the left motor running.

For the signaling, originally we had decided to use a network card, but discarded the idea for its complexity and the need for a network to function and the battery drainage wouldn't be the best idea for a small scale project. So we decided to go with IR signaling which is very cost effective, is more efficient than Bluetooth, and fit in the scope of this project. The idea is that the IR led will send hexadecimal signals to the RC car's receiver and depending on the the signal

received, the RC-Car will react to programing of the arduino connected to the car. The remote will play the role of emitter and the car will play the role of the receiver. [IR Signaling 3] So if the user presses right button and the remote sends a signal 0xFF6897, then the car should recognize this and look into its programing to see that this signal means to turn off right motor and the left motor to continue. The remote's power supply during testing was run off USB, however to keep it wireless free, we are using a 9V battery enclosure which is having the arduino regulating to 5V since that is the appropriate amount of power needed. [IR Signaling 1]

**How IR Signaling Works**

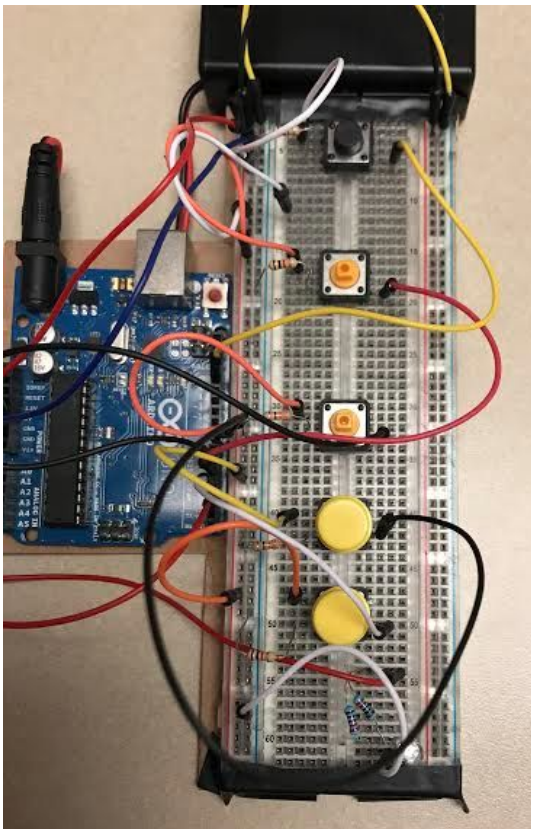
Emitter

```
while (digitalRead(leftTurn) == HIGH) //If the user pressed left button
{
  delay(100); //Slight Delay
  irsend.sendNEC(0xFF6897, 32); //Specific Hex decimal for receiver (Turn Left)
}
```

Receiver

```
else if(results.value == 0xFF6897) // left button
{
  left_backward();
  right_backward();
}
```

If the left button is being pressed, the IR led will send a signal of 0xFF6897 to the RC-Car. The car will recognize this signal through its programing and carry out the appropriate instructions to make the car do a left turn. [IR Signaling 2]



**Final Remote ( + instructions )**

For the final product of the remote, we had changed the schematics dramatically since we did not want to utilize the potentiometer anymore. So we implemented a 5-button system in which you can see to the right. The black button is a reset button to clear any instructions for the RC-Car so it can clear and reset the relays and motors. The two buttons in the middle are the Up and down, which allows the car to go in a straight line or reverse. The two Yellow buttons are for turning left or right. So when the left button pressed, it will shut off the motor on the left side, in which if kept held, the car will just go counter-clockwise and vise versa for the right button. For the IR led, the first two leds weren't sending signals so we scavenged an old stereo remote for its IR Led emitter. This solved our problem because it was sending proper signals and the receiver was actually receiving it, which we monitored through serial communication. [IR Signaling 4]


**Problems that we encountered**

-We had implemented the motion detector library and remote IR library in which they both share the same timer class which gives errors in conflict.

-The first IR LED was having problems sending hexadecimal signals to the RC-Car. However, when we use a normal LED it does show that the blinking does represent hexadecimal signals, but does not react with the IR LED.

-Implementing the speed from the potentiometer breaking it up in different ranges causes an influx in sending signals. So what speed will the car be when I press the button.

-The car that we ended up working with was not the original one we intended to work with, which slowed us down, so to remedy this we got a different RC car which met the criteria we wanted.

-One issue that we had to deal with was the resistor on the IR led. The IR led requires 100 ohm resistor, however the only size resistors that we had was 200 ohm and 1K ohm resistors. To fix this, we put two 200 ohm resistors in parallel which split the resistor to the led giving it the required resistance.

-One of the receivers stop working on us completely ( might have shorted out, or something, still uncertain ) which made it that we had to find another one, which we did by finding an old VCR with a remote controler. The receiver looked different, but it also had 3 pins and was working with the code we had, so thankfully no changes had to be made.

-After hooking up a transistor with a potentiometer, I could get a small motor to run, but ran into issues when trying to run the big motors that were in the blue car. It seems that 5v, and at the current I was supplying was not enough. I needed more power. So, I brought out the good old power supply to do more testing. I changed the input voltage from Arduino to the power supply, and same with the ground. Then I did the same thing and hoped more power would be given to the motor inside the RC car. And then the small transistor blew up in smoke. At least

nothing else was damaged. I learned my lesson, if you want to give more power, you need a better/bigger transistor. Lucky enough I horde many old parts electronics, so it didn't take me long to find one suitable for this task.

**Thoughts about expansion**

-The remote's efficiency definetly can be improved if we created sleep mode function in which if the remote has been idle for longer than 5 seconds, create an interruption in which the remote won't be waiting for a button to be pressed so saving it power and making the product more efficient. Same can be applied to the RC-Car's receiver. If the receiver does not get a signal for 30 seconds, we can put into sleep mode, so it is not constantly waiting for instructions.

-We could have also implemented acceleration in which the potentiometer would have played a great role in. Higher on the dail, the faster the speed. However for this project, we decided to keep the speed constant since we do not want to send conflicting signals which could possibly short out the RC-Car.

-The implementation of a second Ultrasonic Distance sensor to the rear view of the RC-Car. This would allow the car to stop when reversing from damaging itself. This would be great step toward autonomous driving if we put 4 or more sensors on the RC car since we can analyze the surrounding properly and allows self driving mode where the car would know how to react if obstacles or collisions are possible.

-Another possible implementation we could have done is create a camera drone. In which we would need signaling via WIFI since that would having the lowest latency and higher frequency. This would allow us to control the drone from the comfort of our computer.

**Other/similar works of interest**

We as a group are very eager about drone technology and how such technology is a great field for automation, military use, and simply commercial use to make lives easier. Drones can range from military type planes to a glamorized vacuum cleaner Roomba. This project showed us an insight of the possibilities of companies like Tesla working on self driving cars. Companies like Google that have their google map layout based on the pictures that their automated cars have taken.  This is an inspiring field that we hope to be a part of one day.

## References

Distnace Sensor - http://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/

http://www.instructables.com/id/Ultrasonic-Range-detector-using-Arduino-and-the-SR/

Remote - https://learn.sparkfun.com/tutorials/ir-control-kit-hookup-guide

Motors -

1.https://create.arduino.cc/projecthub/licensedGeek/controlling-a-dc-motor-from-an-arduino-101-board-f4954b

2.https://learn.adafruit.com/adafruit-arduino-lesson-13-dc-motors/parts

3.http://www.instructables.com/id/Controlling-AC-light-using-Arduino-with-relay-modu/

4.http://www.robmeyerproductions.com/tt4.gif

5.http://www.the12volt.com/installbay/forum_posts.asp?tid=114347

Buzzer - http://www.instructables.com/id/How-to-use-a-Buzzer-Arduino-Tutorial/

IR Signaling

1. http://garagelab.com/profiles/blogs/tutorial-arduino-ir-sender-and-receiver
2. https://learn.adafruit.com/using-an-infrared-library/sending-ir-codes
3. https://learn.adafruit.com/using-an-infrared-library/hardware-needed
4. https://learn.sparkfun.com/tutorials/ir-communication
5. http://www.instructables.com/id/The-Easiest-Way-to-Use-Any-IR-Remote-with-Ardiuno/