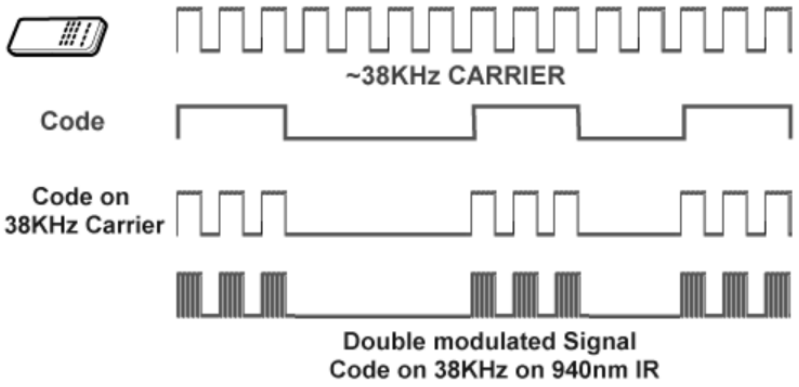
Group 11 - Project Proposal
Infrared Remote Control
Collin Finnan, Kara Garvey, Elizabeth Gonzalez, Adrienne Niewiadomski, Nick Osborne
EE 41430 - Senior Design
11/07/22

Introduction:

One need not think hard to find issues with their current remote control set up. Oftentimes there are too many remotes for different things, the remotes do not have intuitive hot-keys, the connection is too slow or doesn't have sufficient range, etc. By investigating the fundamental properties guiding a remote control, and building on these properties to create an advanced system, we will provide users with the ability to customize their own remote via an application and / or website that is functional for them specifically.

To begin, we need a brief overview of how a TV remote works. The remote is essentially a transmitter, sending infrared signals to a receiver on the TV. The infrared light is generated via an LED (or more correctly, an IRED since we cannot see the light) with a wavelength that corresponds to that required by the receiver (typically ~940 nm). The LED blinks on and off rapidly, creating binary data that the TV receives and interprets. Because infrared noise exists everywhere (sunlight, body heat, etc.), the IR signal must be modulated to a specific frequency set by the receiver, so that noise is filtered and the signal is amplified. This modulation will be in the range of 32-40 kHz typically (source 1).

Figure 1. Basics of Remote Control Signal



But how do we generate these signals? We can blink the LED on and off to make nice square waves like this, but how do we know which "code" to make? The answer is unsatisfying, but essentially it depends on the TV being used. For example, many IR receivers use a standardized system like NEC, while others like Samsung TVs have decided to make their own protocol. Luckily, these protocols are quite similar, and both transmitting and receiving are supported within the Arduino architecture (source 2).

We've now introduced the general concepts behind how a TV remote works. We have not, however, gotten into any of the nuances we plan to add to make this primitive model superior.

Source 1: https://www.laser.com/dhouston/ir-rf_fundamentals.html
Source 2: http://techiesms.blogspot.com/2016/01/ir-protocol-decoding-and-transmiting.html#:~:text=NEC%20protocol%20is%20most%20commonly,mark%20and%204.5%20ms%20space.

Problem Descriptions:

The problems plaguing today's TV remotes have only been accelerated by the advent of smart TVs, where new user freedom has overwhelmed the aged model for controlling the system. The problem description is best seen as a list that will correspond to a secondary solution list in the next section. These are high level complaints that most people find themselves saying when operating a remote.

1. Missing remote –
    - There is nothing more frustrating than finishing all your work, sitting down on a comfortable couch, placing your food and drinks on the coffee table in front of you, only to discover the remote is nowhere to be found. Addressing this problem is paramount.
2. Effective range –
    - Although less prevalent in modern remotes, there is always the problem of having your buttons actually transmit. When the battery is low or the angle is just wrong, it can be frustrating to get the TV to do what you want it to.
    - Maybe you want to pause something while outside the room because the task is taking longer than expected. This is currently not possible.
3. Battery lifetime –
    - Transitioning from the previous point, one could also find themselves upset with the battery lifetime of remotes. AAs are used for packaging purposes, but they get used up fairly quickly.
    - There is also no built-in indicator for battery life, so it always comes as a complete surprise.
4. Hot-keys –
    - Today's remotes often come with hot-keys. Want Netflix to pull up right away? Click the Netflix button. But what if you want HBO? Or YoutubeTV? Sure there are ways of programming the remote to pin these to other, unmarked hot-keys, but wouldn't it be nice if I could customize the layout of the remote myself?
    - What if I want a simplistic remote for grandma, which has 4 buttons only, whereas I would like more options with several? Without hiring a universal remote guy to come in, this is impossible.
5. Access to control –
    - There is typically a single TV remote for a whole room. Want to put something on? "Pass the remote!". But tossing the remote across the room could damage the devices inside, and then you're completely out of luck. Fundamentally, you could

have as many remotes as you wanted, so long as they weren't being used at the same time. Why limit yourself to having one person in control?

6. External Devices –
    - Universal remotes were the fad of the early 2000s, but with new devices cropping up every month, the remote you just paid hundreds of dollars to have set up would become antiquated in less than a year.
    - But the idea behind them is still solid. Why have a remote for your speaker and for your TV? Why not put it all in one place? Is there a way to do this that is Easily adjustable for new devices?

Proposed Solution:

The proposed solutions all revolve around a single idea: a virtual TV remote, controlled by the user via a website. Immediately, people will be hesitant to this solution. Why complicate everything with an app? But the beauty of TV remotes is in their simplicity: they are just transmitters. If you have someone over that doesn't want to go to a website or download an app, just hand them the remote! It will still work. In this solution, a PCB with a microcontroller and transmitter circuit will be mounted near the receiver. The microcontroller will have a codebase of various TV commands that generate signals corresponding to the TV's protocol. A front end for users will be built upon these lower-level operations. The issues provided in the previous section will be addressed with this solution as follows:

1. Missing remote –
    - The transmitter itself will be mounted, and the remote becomes your phone. No one ever loses their phone.
    - The physical remote that comes with the TV may get lost, but the set up we are providing only fails when you misplace your phone or move the PCB.
2. Effective range –
    - Once again, the transmitter is mounted, so it will be perpetually in front of the receiver (but with enough space open so that the actual TV default remote can still be used).
    - But if I leave the room, my phone is still on the WiFi, meaning I can send signals to the TV from anywhere in the house! Go upstairs and forget to turn off the TV? That's okay! Just refresh the website and press the power button.
    - The packaging for this device will have some sort of easy-clip that can attach to a standard, thin TV along the bottom. The only issue will be helping users identify where the TV receiver is for installation.
3. Battery lifetime –
    - This solution can go a couple ways, so we believe our PCB should give the user both options.
    - First, the device could simply plug in. It does not have to be mobile, like an actual remote, and because the TV is itself near an outlet, so too this device could be externally powered, removing the battery problem.

    - But this introduces a cable which may be unattractive to the user, so as a second option, we provide the following: a larger battery space. The setup is not limited by size like a handheld remote, so we could put several additional batteries in parallel, resulting in a longer lifetime.
    - An LED could be added on the PCB to indicate low battery if the second option were selected. This would address the lack of power indication in modern remotes. Alternatively, a power indicator on the user-interface could also be presented. This second solution requires less power consumption and seems more intuitive.
4. Hot-keys –
    - This is where the beauty of the idea comes in. Because there is not a physical remote in this design, the layout is completely up to the user.
    - Some sort of prompt could be provided which asks the user what features they want on their remote. Move “arrow” keys, on / off buttons, and volume would be default, but other buttons would be dependent on input from users. Only use your TV for Netflix? Make a remote with less than 10 buttons and have netflix take up half the board. Need everything on your remote? Have 200 buttons.
    - A GUI could even be devised that allows users to click and drag where they want the buttons to be, so not only the hot-keys were custom but the layout itself.
    - Clicking a button on the app / site would send an HTTP signal via WiFi to the microcontroller. The microcontroller converts this signal to the given protocol and actuates it via a GPIO pin to the LED subsystem.
5. Access to Control –
    - Because it all occurs via WiFi, multiple users can send signals using different devices, eliminating the need to pass the remote or find it when it is lost.
    - To avoid issues with multiple users sending, there could be a ranked queue based on who joins / connects with the CPU first, and it is reset every time the TV turns on and off.
    - Some sort of methodology would need to be implemented that “remembers” users so that home-users do not need to reconnect / go through a long process each time they boot up their TV.
6. External Devices –
    - The icing on top is that because the TV is now controlled via a general purpose microcontroller, we have the freedom to extend the functionality of our application / website beyond IR protocols.
    - If we have a speaker that connects via bluetooth, we could add a feature to the program that has a bluetooth button. All it would require is additional libraries for whatever device types we want the user to be able to connect with.

Demonstrated Features:

This section is best tackled chronologically, since progress in this project will be iterative. The first few points are features that will be established first and are mandatory for the rest of the endeavor. The final few points are end-user features that, while still required, do not have

anything built upon them. The number of features is limited, but each one is multifaceted and requires a lot of heavy lifting.

1. Functional communication with various TVs from the microcontroller –
    - At the heart of this project are the protocols and commands being used to physically change something on the TV screen. The final product must be able to move a cursor left if the left button is pressed. Communication with the TV is a fundamental feature that all other features build on.
    - The project wouldn't be much good if this product only worked with one TV type. Implementing this solution for various products will be paramount in its real applicability. Primary research has shown that the protocols are similar, so hopefully this will not be an impossibly difficult task.
2. Connection between user-phone and microcontroller –
    - Once it is established that the microcontroller can generate signals and "talk" with the TV, we need to transition this communication to the phone level.
    - Using what we learned about WiFi earlier in the semester, we can create a simple interface and verify that the phone is able to communicate with the microcontroller, and subsequently the TV.
3. Multiple users –
    - Once the prior two features have been established, we can get into the nuance of the device. For example, we want anyone in the room to be able to control the TV with their own phone.
    - Having multiple people access the same webpage / application space becomes slightly hairy. The GET requests can become overlapped, and who is "in control" of the remote will be up for question.
    - Creating a hierarchy of user-control and organizing multiple users in a seamless and transparent way will be important so that this feature shines.
4. Effective and attractive user interface –
    - A project like this, that exists to correct user-based problems, needs a clean polish to it. Although this feature is not exactly "demonstratable", a quality UI is necessary.
    - Sampling from users outside the group will be used to correct issues with the UI, and external research on complaints about modern remotes will influence how the UI works.
    - Hot-keys are an especially important piece of this interface; users should be able to add and subsequently use any hot-key to take them anywhere on the smart TV. Adding and using these buttons must be easy and an attractive enough feature that it's more desirable than using the actual remote.
5. At least 1 external device –
    - To make the remote "universal", there should be at least one device that can be controlled with the same interface that is not the TV. This could be a bluetooth speaker, a light system, etc.
    - This is the "icing on top" feature, but it is still needed as a proof of concept for a more full universal implementation of this small-scale project.

Available Technologies:

Needed features and their description, possible device implementation, price, etc. are listed below:

1. PCB
    - Typically costs $50.
    - Designed and laid out in Eagle (free)
2. Microcontroller
    - The processor may need multiple cores - one to process communication with the TV and one to run the front end application.
    - Additionally, we will need bluetooth for the external device and WiFi for phone communication.
    - The ESP32-WROVER-E has on board bluetooth transmitter and receiver as well as WiFi. This handles the majority of our communication issues.
    - It also has plenty of GPIOs that can handle the needed outputs to the LED, and two cores to handle both the front end and back end of the project.
    - It is only $3.60
3. IRED
    - Need one that is in the wavelength of interest
    - Need one that can have a decent amount of current run through it - the more current, the stronger our output signal.
    - Digikey part number when these are applied: 475-SFH4727ATR-ND
    - This part takes up to 2 A of current and has a wavelength of 950 nm.
    - Incredibly cheap to buy LEDs.
4. Power Transistors
    - The GPIOs likely will need to power the gates of a MOSFET, not actually drive the IRED since they have a current limit.
    - The transistor needs a decent bandwidth, since the single must be modulated around 40 kHz
    - Digikey part number: 3141-G1002LTR-ND
    - It is 50 cents, so price is not a problem.

Overall, this is more of a code-heavy project. The hardware is affordable, and should take too long to implement.

Engineering Content:

1. PCB Design –
    - There is still a significant hardware portion to this project. A PCB will need to be designed, and components will need to be selected very carefully according to the constraints of the design.

    - For example, we have the wavelength of the LED, the needed power for the transmission, the implementation of MOSFETs on the board, ESP32 placement, etc.
    - There are not going to be many hardware devices, but the ones needed are specific and need careful consideration.
    - This step is going to require a lot of research into how actual remote controls implement their circuit boards - what IREDs they use, which transistors they use, what kind of passives are needed, how the microcontrollers interact, etc.
    - It is deceptively simple, yet crucially important.
2. Communication protocols –
    - This step is going to also require lots of outside research.
    - We required ourselves to interact with multiple types of TVs, meaning we need to know the protocols of all these receivers.
    - Not only that, but we will need to know what to ping on the device address wise and what data to send in order to get to higher level functionality like “open netflix”.
    - The goal of this portion of the project is to learn about different TV protocols and how to navigate their OS, and implement higher level libraries for the microcontroller to use so the UI programmers can call things like “open_netflix()”. It should abstract the complications of interacting with the TVs.
    - This is obviously going to be not only a ton of research but lots of coding and learning about both hardware and software - it will take time and many engineers.
3. UI –
    - The project is predicated on having a better experience than a traditional remote, meaning the UI needs to basically be flawless.
    - High level coding needs to be implemented. A web page / application should be designed so that non-tech users can easily connect devices and operate the TV.
    - Having built a few UIs, this will take a decent amount of time, and is certainly enough engineering to justify it as a project.

Conclusion:

The modern TV deserves a modern remote. Creating a fully customizable experience via a smartphone will eliminate several of the problems currently existing in the remote control space. By having the transmitter fixed and communicating via personal devices, traditional issues like lost remotes, low battery, accessibility, and general interfacing will vanish. To implement will require intelligent PCB design using heavily researched components, thorough library building for effective communication between microcontroller and TV, and flawless high level design for a near perfect UI. The end result will be a completely redesigned remote control that has aspects needed for the progression of television watching.
