Raspberry Pi 3B+

Austin Comeaux, Jonathan Eanes, Jeremy Perando, and Christian Pross

# Introduction

The Raspberry Pi 3B+ is an inexpensive SBC computer capable of running a Linux operating system on just a gigabyte of memory. This is not a landmark achievement for computers. What is significant about the Raspberry Pi is that it runs the OS with as big of an SD as you can dream of, multiple USB and GPIO ports, wireless, and ethernet access all in an extremely compact form factor and for just thirty-five dollars plus shipping. The computer is not considered a microcontroller, as it is more advanced than Arduinos and similar boards. However, Raspberry Pis can act both as a microcontroller and as an interface for multiple other microcontrollers to interact over an internet connection. Raspberry Pis can do many different things, but their relatively low cost compared to a normal computer makes them especially suitable for applications where they can be set up and forgotten about. Due to their low power draw, standardized design, and robust Linux operating system, they are fantastic for operations that require a cheap system to be online around the clock. Because of this versatility and reliability, it will do very well as an IR signal decoder and web server in our project.

# Literature Review

Since the release of the original Raspberry PI, there have been 4 generations of newer PI devices, with the current device being the Raspberry PI 5. There have also been several minor revisions to some versions, such as the Raspberry PI 3 B and the Raspberry PI 3 B+. Until the PI 5, the base model of each PI generation was priced at roughly $35, making the Raspberry Pi line one of the cheapest and most accessible entry points for people looking to learn more about computers or make their own IoT devices. Because of this, the Raspberry PI line has sold over 57 million units and become a widely known device in tech spheres.

One project using a Raspberry Pi with an infrared receiver and remote is a remote-controlled Raspberry Pi media center from Adafruit [3]. This project uses the same parts our project will be using, a Raspberry Pi, IR receiver, and IR remote, and uses a package called LIRC (Linux Infrared Remote Control) installed on the Rasbmc OS. The hardware for this project is simple, consisting only of wiring the infrared receiver directly to the GPIO pins on the PI. There is also very little code used in this project since most of the configuration for the remote is done through LIRC. The only code included is a configuration file that maps the infrared signals from the remote to key names so the system can interpret them. The final product of this project is a media center that can play downloaded videos and music and be controlled with the IR remote.

A more complex project using a Raspberry Pi as a universal remote is shown on the website of Diligent, a subsidiary of National Instruments [2]. This project uses an infrared receiver to parse controller inputs and then uses an infrared light to transmit those signals on command. The electronics are more complex because this project involves an IR receiver and transmitter wired separately, and the IR transmitter must be tuned to the correct frequency. Like the Adafruit project, this one uses LIRC, but in this project, LIRC is used to parse signals from any regular TV remote, and then save them in the PI for later use. This allows the PI to train with multiple remotes, and then control various devices with the saved infrared information. While there is no real code for this project either since it is primarily generated through LIRC, the project does have a LabVIEW interface that provides a simpler configuration option for users not familiar with Linux.

A project that includes more code rather than fully relying on the LIRC GUI is available on the Digikey website [1]. This project is similar to the last one in its wiring since it also uses an IR receiver and transmitter, but this one provides more details on how the code is implemented. Like the others, this project makes use of LIRC, but this project shows how LIRC can be used in Python to trigger actions in the code. The code for this project uses sockets to connect to LIRC and then listens for signals from it. When a signal is received, the Python code parses it and can perform an action, in this project, it listens for the power button and then ends the program. The code for sending out an infrared signal is much simpler; just one call to the *irsend* function provided by LIRC containing the remote name and button that is being sent.

All these projects demonstrate different ways infrared receivers and transmitters can be used with a Raspberry Pi. They all make use of the LIRC library, which is an easily available package on the Raspberry Pi OS, and give a good idea of how a project using these technologies could be set up. Most of the reviewed projects did little to implement a custom function using the infrared transmitter, so our project will do more than these already available projects by implementing code to handle different button inputs from the remote.

# Memory

The raspberry pi that we chose for this project, the Pi 3B+, uses 1 GB of LPDDR2 SDRAM. What that means is fully explained in the following section. SDRAM stands for synchronous dynamic RAM, which is dynamic RAM that syncs its clock cycle with the CPU’s clock. Dynamic RAM is well-suited for a cheap computer platform as it is cheaper than the alternative, which is static RAM. Dynamic RAM is also more stable than static RAM because it is not so easily influenced by beta radiation. [7]

## SDRAM

The raspberry pi’s use of SDRAM is not an accident; SDRAM is the predominant RAM type for modern standalone computers. The fact that cheap, high-capacity memory was not always able to sync with the CPU clock restricted computers to the more expensive and lower-capacity SRAM. However, in 1993, Samsung was the first to implement SDRAM which then completely replaced standard DRAM in consumer electronics by 2000. SDRAM had a few key advantages that resulted from it being synchronized with the CPU.

The synchronization of SDRAM with the CPU enables manufacturers to place more than one cell of SDRAM on the same board with the CPU which allows for scalable architectures in memory design and likely resulted in the eventual development of the DIMM slot, or Dual In-Line Memory Module. Due to size limitations though, the Pi 3B+ only uses a single 1GB cell. Also, every new iteration of SDRAM technology has resulted in memory that is more energy efficient and enables faster data transmission. [7]

## DDR Generations

As SDRAM developed, its first new iteration was called DDR, which stands for Double Data Rate. DDR's major improvement over standard SDRAM is the ability to send data to the CPU twice during a single clock cycle: once on the rising edge, and once again on the falling edge. This proceeded from SDR, which only sent data on one of those (manufacturer’s choice). Ever since 2000, DDR has been iterated upon to improve the data rate, transfer rate, total capacity per CPU, and power draw. Starting at 266-400MT/s, 2.1-3.2GB/s, 2.5-2.6V, and a maximum capacity of 1GB, DDR1 was not very powerful. The Raspberry Pi 3B+ has a data rate of 533/800MT/s, a transfer rate of 4.2-6.4GB/s, a power draw of 1.8V, and a maximum capacity of 2GB per module. Finally, the “LP” code before DDR2 represents the fact that the memory that the Pi 3B+ uses is low power. [9]

## Non-Volatile Memory

For our Raspberry Pi, we chose a 32GB SD card, which will be more than enough to boot the default Raspbian operating system, download necessary applications and languages, and store data files such as images and webpages.

Operating System

Raspberry Pi computers do not have a specific operating system that users must install. There is a recommended operating system, Raspberry Pi OS, but users can choose Linux distributions, Windows 10 and 11, and many others. If an operating system can run on Arm processors, and the model Raspberry Pi meets the minimum RAM requirement of the operating system, there’s a good chance it will run.

Raspberry Pi OS is the first choice for most users because it can run on every model of Raspberry Pi and is specifically designed for the hardware [10]. There are currently three different versions compatible with every Raspberry Pi model: Raspberry Pi OS with desktop, Raspberry Pi OS with desktop and recommended software, and Raspberry Pi OS lite. The first two versions, with and without recommended software, are very similar. The only difference is that the former comes bundled with programs that users might find useful, such as LibreOffice and a web browser, and the latter does not. The third version, Raspberry Pi OS lite, does not have a desktop environment and uses the terminal to interact with the computer. It’s designed for projects where users won’t be interacting with the Raspberry Pi like a traditional computer, such as a being a server. All three of these versions are 32-bit, but 64-bit editions exist of each and are only compatible with certain Raspberry Pi models [10]. Regardless of version, all can be installed using a tool called the Raspberry Pi Imager. Using another computer running the Raspberry Pi Imager, an operating system can be installed on a micro-SD card and inserted into the Raspberry Pi. With that, Raspberry OS has been installed.

Windows 10 and 11 can technically run on Raspberry Pi computers, but it’s not officially supported by Microsoft or Raspberry Pi [13]. Doing so relies on Windows versions that are designed for Arm-based laptops and tablets, which are not designed for Raspberry Pi’s and thus not guaranteed to work properly [11]. As such, there is no official way to install Windows on a Raspberry Pi, and it’s mostly done by hobbyists who want to push the limits of the computer.

Given that Raspberry Pi OS is itself a derivative of Debian, it makes sense that other Linux distributions are supported as well. Ubuntu is one of the most popular, having an edition specifically designed for Raspberry Pi computers [12]. There are three main versions on the Ubuntu for Raspberry Pi website: Ubuntu Desktop, Ubuntu Server, and Ubuntu Core. The desktop and server versions are designed for what their names imply, but Ubuntu core is for embedded devices that connect to the internet and need to be highly secure [12]. Basically, it’s not for users who want a desktop experience, but for people who need an operating system that can run reliably without constant user intervention or maintenance. All versions of Ubuntu for Raspberry Pi can be installed using the same steps as installing Raspberry Pi OS with the Raspberry Pi imager [12].

For our project we will use Raspberry Pi OS, without recommended software. A desktop environment will be convenient when building our project in case we must do any programming on the Raspberry Pi. Using Windows 10 or 11 isn’t much of a possibility, given that we’d have to pay for a license to use Windows on Arm, and that it’s not officially supported on Raspberry Pi. Ubuntu is a viable choice if we decide to forgo Raspberry Pi OS, but as of right now Ubuntu doesn’t offer anything we need that Raspberry Pi OS doesn’t also offer.

# Communication

Each model of the Raspberry Pi features a set of GPIO pins positioned along the top edge of the board, allowing for the connection and communication with various electronic components [4]. These GPIO pins serve as a vital interface, enabling the Raspberry Pi to switch components on and off or receive data from sensors and switches [8]. Additionally, the Universal Asynchronous Receiver/Transmitter (UART) plays a crucial role in serial communication, facilitating the transmission of data between devices [6]. Operating asynchronously, UART requires no clock signal and ensures data integrity by sequentially transmitting individual bits and then reassembling them into complete bytes at the receiving end. All Raspberry Pi models come equipped with GPIO pins and UART interfaces, enabling seamless interaction with a wide range of peripherals and devices.

Bluetooth technology operates in a master-slave mode, forming Pico-nets comprising master and slave devices, allowing for point-to-point and broadcast communication. The HC-05 Bluetooth module, chosen for its availability and cost-effectiveness, operates at +3.3v DC/50 mA and offers data transmission rates ranging from 9600 to 115200 bps. Similarly, Wi-Fi connectivity further enhances Raspberry Pi's communication capabilities, enabling wireless interaction with networks and devices. The Ethernet port serves as the Raspberry Pi's primary gateway for communication with other devices and the Internet, providing seamless connectivity via auto-sensing ports and USB Ethernet adapters.

Evaluation metrics for Wireless Sensor Networks (WSN) encompass communication rate, power consumption, and range [5]. Optimal node placement is crucial to establish interconnected networks, with communication rate influencing node performance. Higher communication rates result in reduced transmission time and lower network power consumption, albeit often accompanied by increased radio power consumption. Ethernet and Wi-Fi, along with Bluetooth and UART, contribute to Raspberry Pi's diverse communication capabilities, empowering users to create a myriad of projects spanning from embedded systems to IoT solutions and educational endeavors.

# Conclusion

Although the Raspberry Pi is not the most powerful computer, it is well suited to the needs of our infrared receiver project. The base version of the Pi comes with only 1 GB of RAM, but it is designed to work around this limitation. The Raspberry Pi OS is highly optimized to run well on all versions of the Pi, and the array of wireless communication features makes it very convenient for IoT projects. There are multiple examples of Raspberry Pi projects using infrared receivers and transmitters, so we will be able to expand on what we learned from those when implementing our project. Overall, the Raspberry Pi will make a good device for both decoding infrared signals and hosting a web server to display data.

# References

[1] “How to send and receive IR signals with a Raspberry Pi,” DigiKey, <https://www.digikey.com/en/maker/tutorials/2021/how-to-send-and-receive-ir-signals-with-a-raspberry-pi> (accessed Mar. 13, 2024).

[2] A. Stanton, “Using a raspberry pi as a universal remote,” Digilent Blog, <https://digilent.com/blog/using-a-raspberry-pi-as-a-universal-remote/> (accessed Mar. 12, 2024).

[3] S. Monk, “Using an IR remote with a Raspberry Pi Media Center,” Adafruit Learning System, <https://learn.adafruit.com/using-an-ir-remote-with-a-raspberry-pi-media-center> (accessed Mar. 12, 2024).

[4] Raspberry pi documentation - raspberry pi os, <https://www.raspberrypi.com/documentation/computers/os.html> (accessed Mar. 13, 2024).

[5] (PDF) Raspberry Pi as a wireless sensor node: Performances and ..., <https://www.researchgate.net/publication/269291637_Raspberry_Pi_as_a_Wireless_Sensor_node_Performances_and_constraints> (accessed Mar. 13, 2024).

[6] Design and development of daughter board for USB- ..., <http://www.sooxma.com/docs/Abstracts/Design> and Development of Daughter Board for USB-UART Communication between Raspberry Pi and PC.pdf (accessed Mar. 13, 2024).

[7] M. Rouse, “What is synchronous DRAM (SDRAM)? - definition from Techopedia,” Techopedia, <https://www.techopedia.com/definition/2815/synchronous-dram-sdram> (accessed Mar. 13, 2024).

[8] A. Negi, “Raspberry pi 3 B+ pinout with GPIO functions, schematic and specs in detail,” eTechnophiles, <https://www.etechnophiles.com/raspberry-pi-3-b-pinout-with-gpio-functions-schematic-and-specs-in-detail/> (accessed Mar. 12, 2024).

[9] “Different types of RAM explained,” Crucial, <https://www.crucial.com/articles/about-memory/different-types-of-memory-explained> (accessed Mar. 12, 2024).

[10] Raspberry pi OS – raspberry pi, <https://www.raspberrypi.com/software/> (accessed Mar. 12, 2024).

[11] “Windows on arm documentation,” documentation | Microsoft Learn, <https://learn.microsoft.com/en-us/windows/arm/overview> (accessed Mar. 12, 2024).

[12] Install ubuntu on a raspberry pi | ubuntu, <https://ubuntu.com/download/raspberry-pi> (accessed Mar. 12, 2024).

[13] L. Pounder, “How to install Windows 11 on a raspberry pi 4 (updated),” Tom’s Hardware, <https://www.tomshardware.com/how-to/install-windows-11-raspberry-pi> (accessed Mar. 12, 2024).