

PiCar

A Wi-Fi Controlled Car Using a Raspberry Pi Created By: Alexander Wittman Instructor: David Tarnoff Class: CSCI 4677-901

Table of Contents

[System Design: 2](#_Toc58439916)

[Electrical Diagram: 3](#_Toc58439917)

[Parts Required: 4](#_Toc58439918)

[GPIO Pins Used: 5](#_Toc58439919)

[Node Packages Installed: 6](#_Toc58439920)

[MQTT Topic Details: 7](#_Toc58439921)

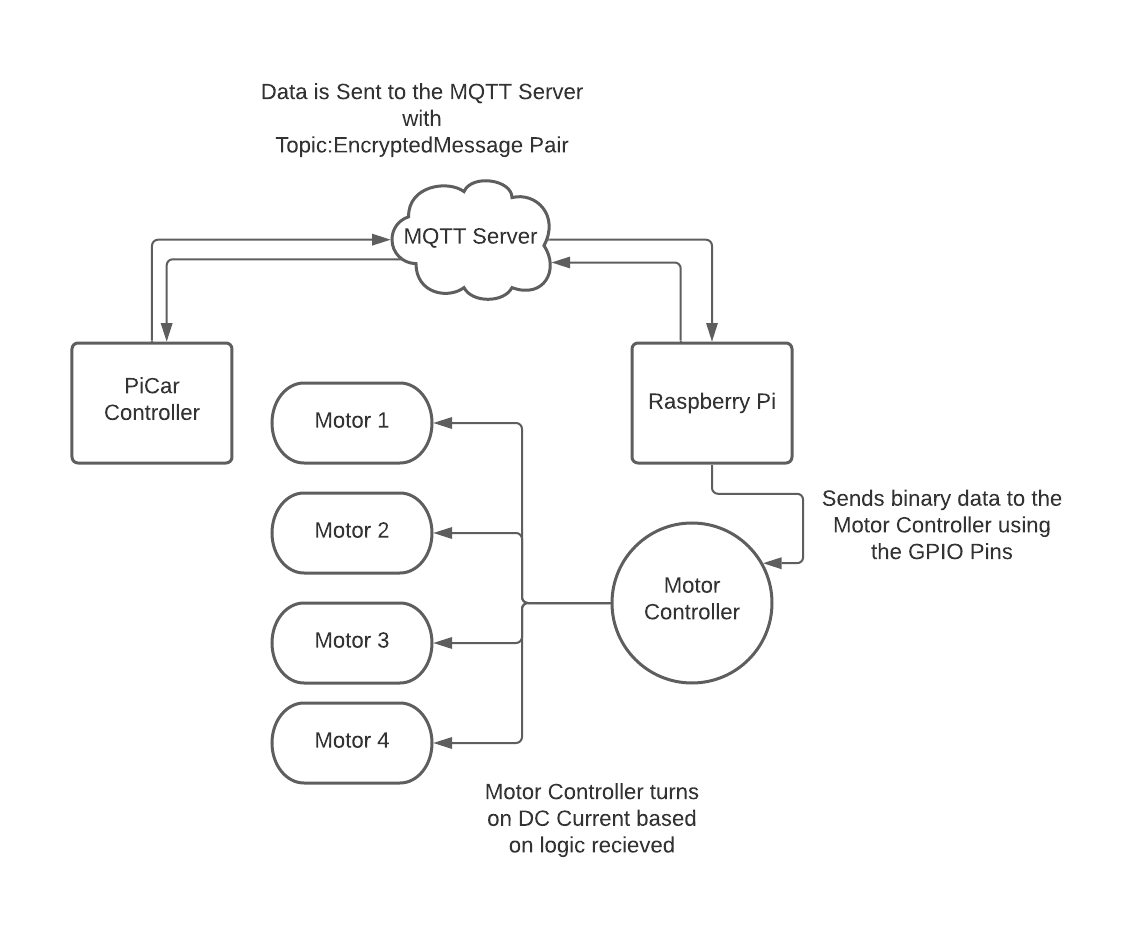
[Encryption Details: 7](#_Toc58439922)

[PiCar NodeJS Diagram: 8](#_Toc58439923)

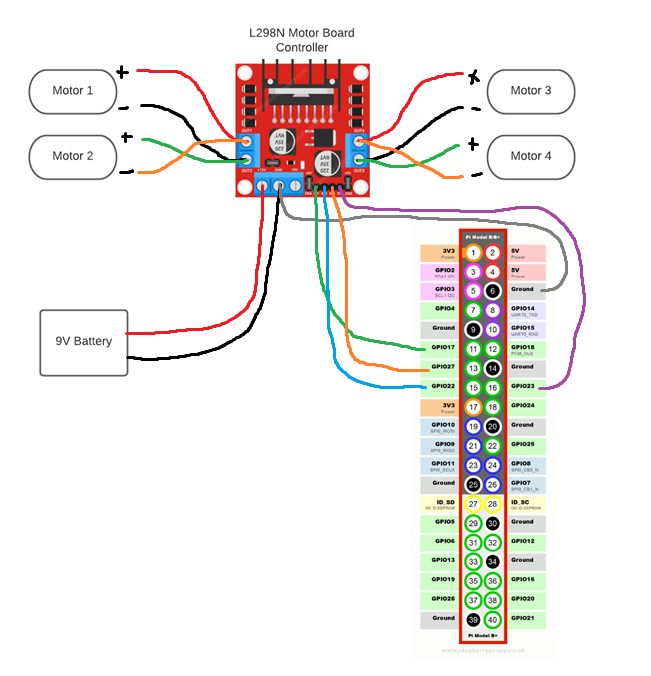
[PiCarController NodeJS Diagram: 9](#_Toc58439924)

[Resources: 10](#_Toc58439925)

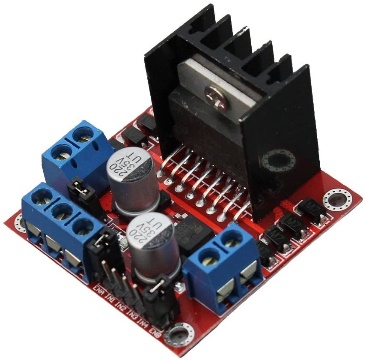
# System Design:



# Electrical Diagram:



# Parts Required:

*  Raspberry Pi Model 3B+ (Used as the brain of the car)
* L298N Motor Controller Board (Gives power to the motors)
* Frame, DC Motors, and Wheels Kit (Gives the car a body to work with)
* Female to Female Wires (Can also use Male to Female Wires if a breadboard is available)
* 9V DC Power Box that hold a 9V Battery (Powers the motor controller)
* Power Bank (Used to power the raspberry pi, there are power banks that allow for 12v and 5v output allowing for the 9V power box to be removed)

# GPIO Pins Used:

* **GPIO 17:** Controls the right wheels to go in reverse. This is paired with the left wheels to go in reverse for the PiCar to go backwards.
* **GPIO 22:** Controls the right wheel to go forwards. This is paired with the left wheels to go forwards for the PiCar to go straight or can be uses by itself to turn the PiCar left.
* **GPIO 23:** Controls the left wheels to go in reverse. This is paired with the right wheels to go in reverse for the PiCar to go backwards.
* **GPIO 27:** Controls the left wheel to go forwards. This is paired with the right wheels to go forwards for the PiCar to go straight or can be uses by itself to turn the PiCar right.

# Node Packages Installed:

* **MQTT:** MQTT allows the program to communicate to a dedicated server by passing topics and messages. MQTT runs as Subscriber/Publisher server so that anyone can listen to certain topics and receive messages on those topics.

**\*The PiCar and the PiCarController both require this node package to function\***

Installation:

1. Go to file location of the code in command line that will use MQTT
2. Enter in the command line “npm install mqtt”

Website: https://www.npmjs.com/package/mqtt

* **RSA-Node:** RSA-Node allows for encryption with a public and private key. Keys can be generated randomly by giving them a certain number of bits for the key to generate, or they can be imported if there is already a key available.

**\*The PiCar and the PiCarController both require this node package to function\***

Installation:

1. Go to file location of the code in command line that will use RSA-Node
2. Enter in the command line “npm install rsa-node”

Website: <https://www.npmjs.com/package/node-rsa>

* **PiGPIO:** PiGPIO allows for the use of the GPIO pins on the Raspberry Pi. This node package allows us to fully utilize the pins to send a signal for the motor controller. PiGPIO also allows the program to enter a specific signal level from (0-255). With this the motor controller can make the wheels spin faster or slower.

**\*The PiCar requires this node package to function\***

Installation:

1. Go to file location of the code in command line that will use RSA-Node
2. Enter in the command line “npm install pigpio”

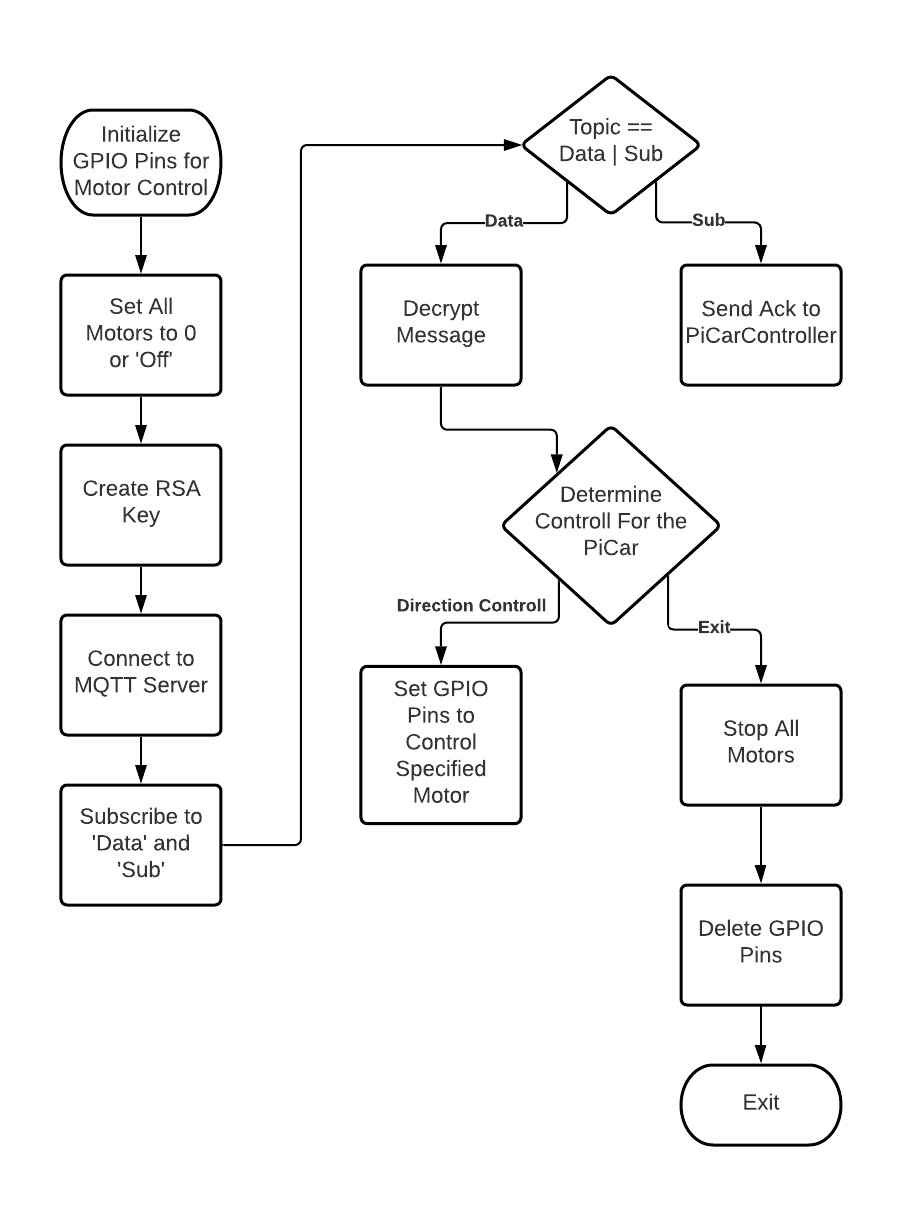
# MQTT Topic Details:

* **SUB:** Sub is used by the PiCarController to send a message to the PiCar to determine if it is connected to the broker. If the PiCar is connected, it will then send an ‘ACK’.
* **ACK:** Ack is used by PiCar to tell the PiCarController is online. Once the PiCarController receives the message, it will allow for data to be sent to the PiCar on where to go.
* **DATA:** Data is sent by the PiCarController to tell the PiCar on where to go. All messages in data are encrypted and the PiCar decrypts the data upon receiving it.

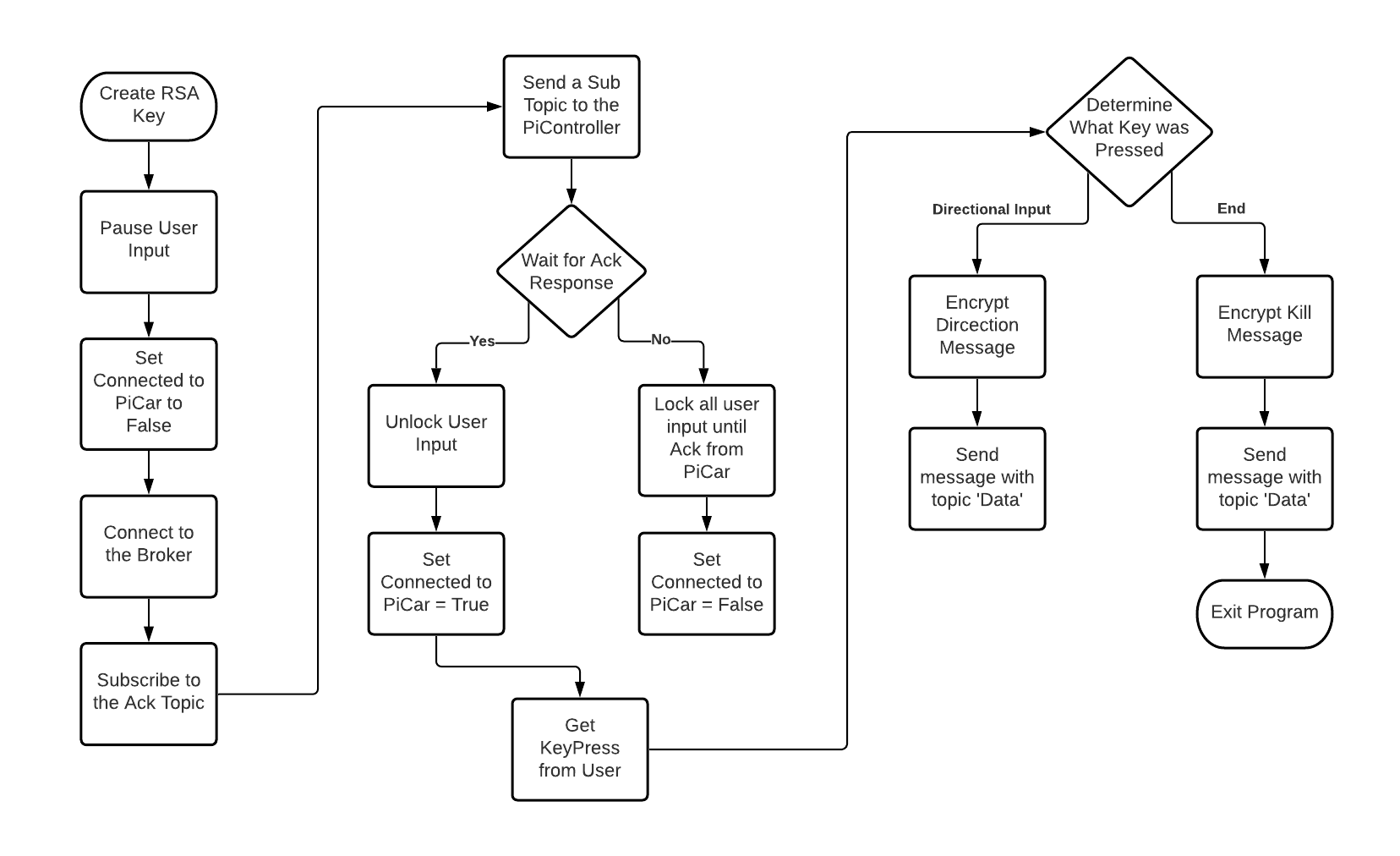
# Encryption Details:

* The encryption used for the PiCar and PiCarController is an RSA algorithm that uses a shared 1024 bit key that has a public and a private key. The PiCarController sends information to the PiCar by encrypting it using the public key version. The PiCar then decrypts it using the private key version. The same goes when the PiCar sends data back to the PiCarController, in which it uses the public key version to send data, and the PiCarController uses the private key to decrypt any information received.
* With MQTT, the topics are left unencrypted, but the messages that are sent with the topic are encrypted. Therefore anyone can subscribe to the topic, but are required to have the private key in order to decrypt the message.

# PiCar NodeJS Diagram:



# PiCarController NodeJS Diagram:



# Resources:

* <https://stackoverflow.com/questions/59185152/raspberry-pi-node-js-gpio-control>
* <https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/>
* MQTT Lab Notes
* Servo Lab Notes
* Encryption Lab Notes