IoT Based Traffic Management System

A Project report submitted in partial fulfilment of the requirements for the degree of B.E in Computer Science and Engineering.

By

G.VIJAYARAGAVAN(513221104318)

Under the supervision of

Professor & HOD

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Thirumalai Engineering college

IOT and the Maker Movement

The momentous adoption of the IoT system is the rise of the maker culture. The maker culture encourages hobbyists (and professionals alike) to create their own devices as well as tinker with existing ones to find solutions to solve their specific problems.

With the maker movement comes a host of DIY electronic platforms, such as Arduino and Raspberry Pi. Arduino (see **Figure 1**) allows you to connect to various external accessories (such as sensors) and create applications to use the data collected.

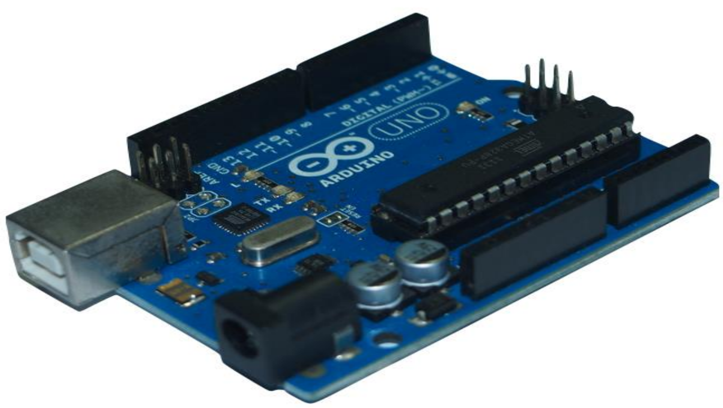

Figure 1: **The Arduino UNO board**

Another open-source hardware platform that has gotten very popular with hobbyists these days is Raspberry Pi. It's really a computer, by all definitions. Raspberry Pi is a low-cost, credit card-sized computer that connects to a computer monitor or TV using HDMI, and uses a standard keyboard and mouse. It can run a host of operating systems, such as Raspbian (Debian Linux), Android, Windows 10, IoT Core, etc.

- Raspberry Pi has gone through a few iterations and Table 1 shows the list of Raspberry models released over the years and their prices.

- Of the various models, Raspberry Pi 3 (see Figure 2) and Raspberry Pi Zero (see Figure 3) stand out.

Figure 2: **The Raspberry Pi 3**

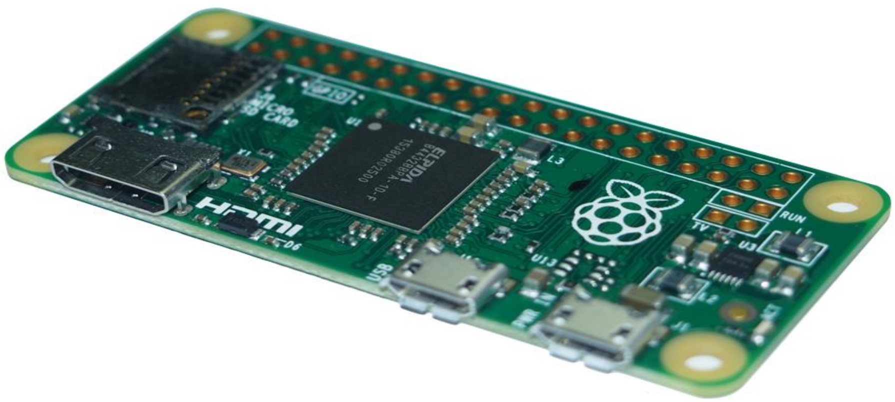

Figure 3: The Raspberry Pi Zero

Raspberry Pi 3 is the third generation of Raspberry Pi and it packs quite a formidable punch in its credit card-sized package. Most notably, in addition to the standard features of the Raspberry Pi (such as four USB 2.0 ports and built-in Ethernet), it has:

- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 802.11n Wireless LAN
- Bluetooth 4.1 Low Energy (BLE)

The powerful CPU coupled with Wireless LAN and Bluetooth 4.1 radio makes it an ideal candidate for IoT projects, because multiple sensors can be connected to it simultaneously. In addition, the Raspberry Pi has a 40-pin GPIO (General Purpose I/O) connector for interfacing with external sensors.

The Raspberry Pi Zero is the smallest Raspberry Pi ever made, and although it doesn't have a processor that's as powerful as the Pi 3, its small size is especially suited for embedded projects (such as wearables, etc.), where space is a premium.

Powering the Raspberry Pi

One of the most popular OSs used for the Raspberry Pi is the Raspbian Operating system. The Raspbian OS is based on the Debian OS, optimized for the Raspberry Pi hardware.

The easiest way to install the Raspbian OS for the Raspberry Pi is to download NOOBS

From https://www.raspberrypi.org/help/noobs-setup/. NOOBS stands for New Out Of Box Software.

The Raspbian OS boots off a micro-SD card and the entire operating system runs off the card. A typical Class 4 8GB micro-SD card is sufficient for most purposes, but you have the option to connect it to an external hard disk or flash drive for more storage.

Once the Raspbian OS is installed, you can proceed to log into it and see a full windowed system (see **Figure 4**). The default username is `pi` and the password is `raspberry`.

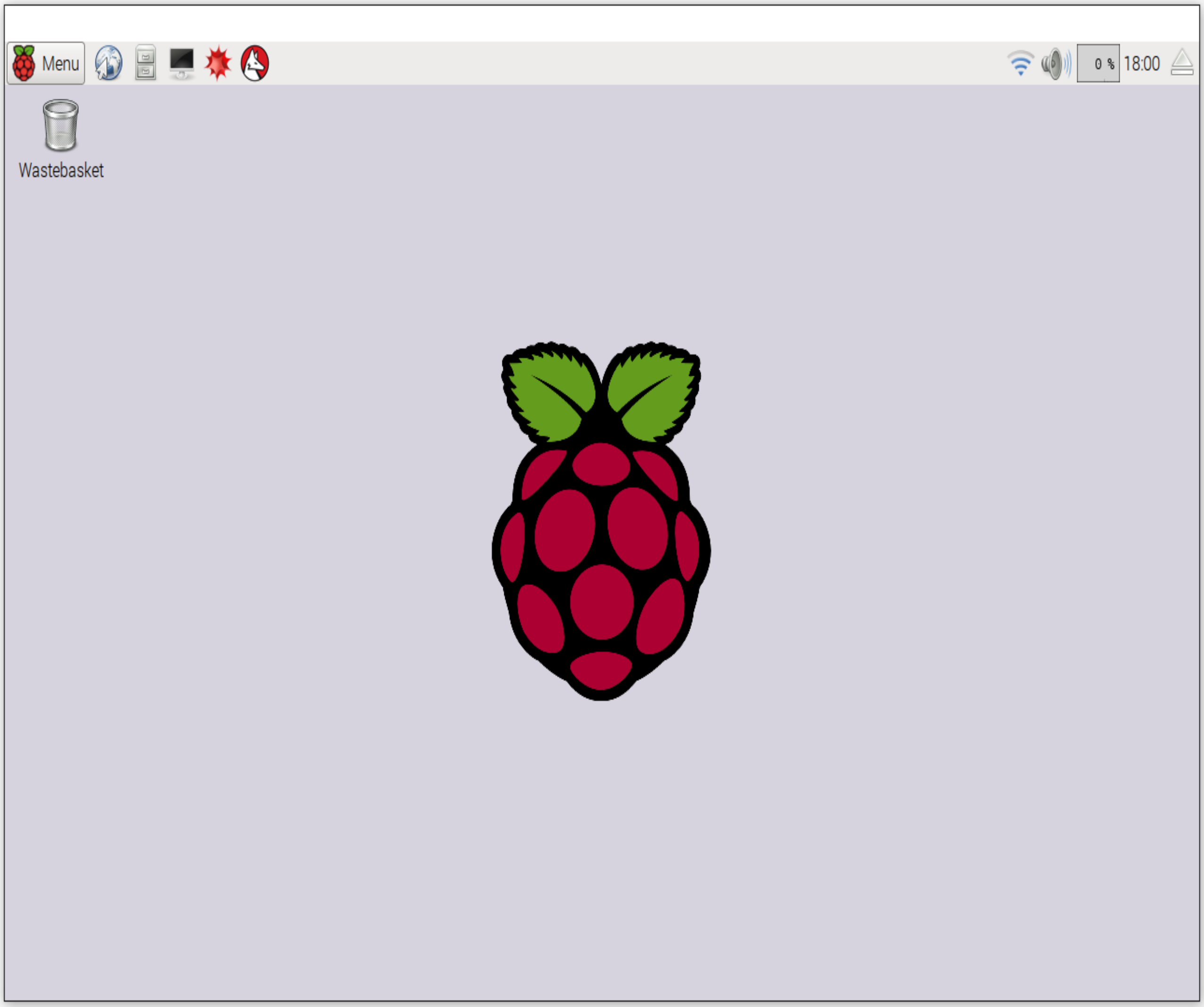

Figure 5: The Raspbian OS uses the LXDE (Lightweight X11 Desktop Environment)

Connecting to a Sensor to Detect Motion

To demonstrate how to use the **GPIO** to connect to an external sensor, we'll now use a **PIR** motion sensor to detect motion. For this, I used the **Parallax PIR Motion Sensor** (see **Figure 6**). The PIR Sensor detects motion by measuring changes in the infrared (heat) levels emitted by surrounding objects of up to three meters.

Figure 9: The Parallax PIR Motion Sensor

The Parallax Motion sensor has three pins :

- **GND:** The ground pin. Connect this pin to the GND on the GPIO.
- **VCC:** The voltage pin. Connect this pin to one of the 5V pins on the GPIO.
- **OUT:** The output pin. Connect this to one of the Input/Output pins on the GPIO.

When the PIR Motion sensor detects motion, it outputs a high signal on its output pin. You need to write an application to read the value of this output pin.

Bonding the Raspberry Pi and the Sensors: The Python Programming Language

Now that the Raspberry Pi is connected to the PIR Motion Sensor, it's time to write the code to make things work. In the Raspbian OS, Python is a first-class citizen, and the support for Python comes right out of the box. With its clean syntax and ease of learning, Python is a first choice for hobbyists and beginners to foray into the world of the Raspberry Pi. Coupled with the huge community support for Python, it's no wonder that it's the language of choice for developers.

Open a Terminal window in the Raspbian OS and create a text file by typing the following command:

```
$ nano motiondetection.py
```

The above command uses the NANO text editor and creates a file named `motiondetection.py`. Enter the statements as shown in **Listing 1**.

The yellow line is the OUTPUT and is connected to pin #4 on the GPIO. The black line is the GND and should be connected to GND on the GPIO.

Listing 1. Source code for using a PIR Motion Sensor

```
import RPi.GPIO as GPIO                                #1
import time                                            #2

pirsensor = 4                                          #3
GPIO.setmode(GPIO.BCM)                                 #4
GPIO.setup(pirsensor, GPIO.IN, GPIO.PUD_DOWN)          #5

previous_state = False                                 #6
current_state = False

while True:                                            #7
    time.sleep(0.1)                                    #8
    previous_state = current_state                     #9
    current_state = GPIO.input(pirsensor)              #10
    if current_state != previous_state:                #11
        if current_state:                              #12
            print("Motion Detected!")                  #13
```

When you are finished typing in the code, exit the NANO editor by pressing Ctrl-X and then to save it to the current directory. To run the Python script, type the following command in Terminal:

```
$ python motiondetection.py
```

Wave your hand in front of the PIR Motion Sensor. You should see the following output on Terminal:

```
Motion Detected!
```

Streamr.Network's applications in transportation management are vast and impactful. Let's explore some key areas where this platform brings tangible benefits:

1. *Traffic Management and Congestion Mitigation:* By aggregating and analyzing real-time traffic data, Streamr.Network empowers traffic management authorities to identify congestion hotspots, optimize traffic signal timings, and provide timely updates to drivers, reducing congestion and enhancing overall traffic flow.

2. *Intelligent Routing and Navigation Systems:* Leveraging real-time data on traffic conditions, road closures, and incidents, Streamr.Network enhances routing and navigation systems. Commuters can receive dynamic route recommendations based on current traffic conditions, resulting in shorter travel times, reduced fuel consumption, and a more efficient transportation network.

Transforming Transportation: Benefits and Advantages

The adoption of Streamr.Network in transportation management unlocks a multitude of benefits:

1. *Enhanced Operational Efficiency:* Real-time data enables proactive decision-making, reducing response times, optimizing resource allocation, and minimizing downtime. Streamr.Network's decentralized architecture ensures quick and secure data processing and dissemination, facilitating efficient transportation management.

2. *Improved Passenger Experience and Satisfaction:* Real-time data empowers passengers with accurate and timely transportation information. The network facilitates seamless communication between transport operators and passengers, enabling personalized updates, efficient trip planning, and ultimately enhancing passenger satisfaction.