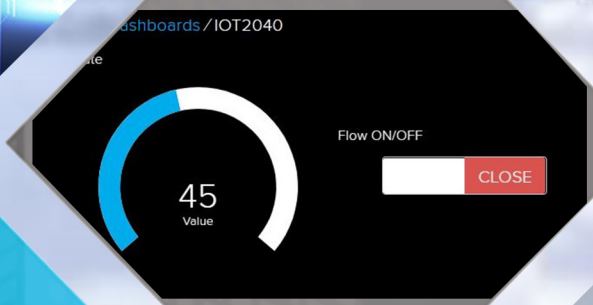
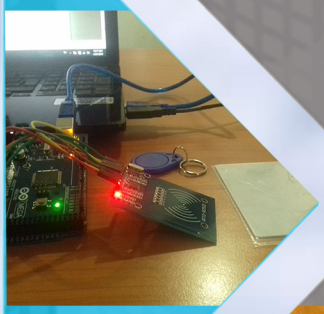
TLM.YASIR

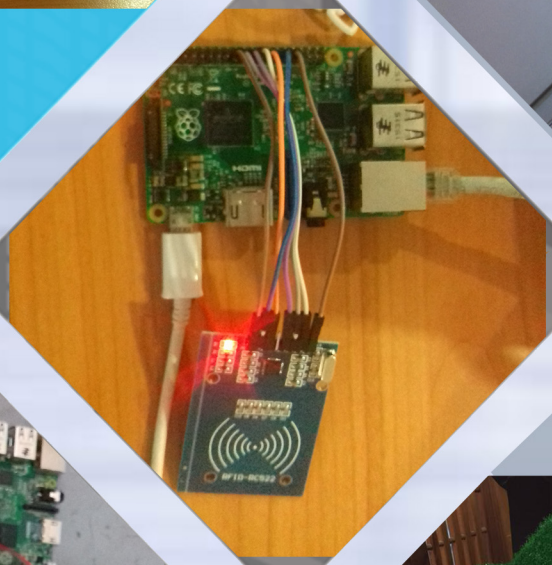
The Portfolio

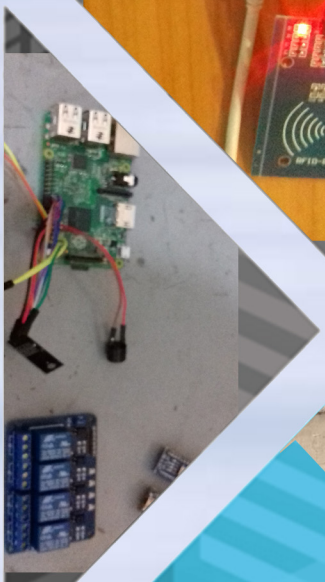
tlmyasirs@yahoo.com

+971-501430209









IoT for Home Automation

Figure 1 Home automation +iot photos

Designed a Home automation system with IoT. The design consist of a Raspberry Pi 2 model B, 4 arduino nano boards,5 nRF24L01 wireless modules, sensors, relays, other essential electronic components and specially Adafruit IoT platform via io.adafruit.com.

I have used Raspberry Pi's SPI interface to connect nRF24L01 wireless transceiver module. And sensors and relays connected to arduino slaves are connected to Raspberry pi over 2.4GHz wireless communication with the help of nRF24L01.

And this Raspberry Pi is connected to Adafruit IoT platform. So the end devices are online as expressed in the below figure 2.

And Adafruit io have the dashboard to watch real time and historical data from sensors and control outputs of relay devices. And Adafruit is able to connect to external web application as twitter and IFTTT.

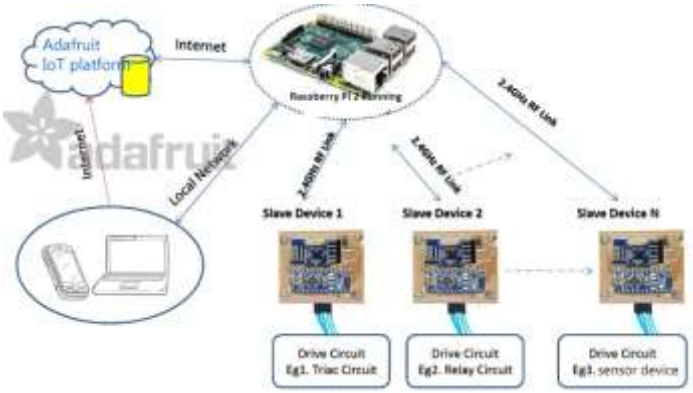


Figure 2 Raspberry Pi Hosting clients to IoT

A machine to machine (M2M) protocol MQTT is used for communication.

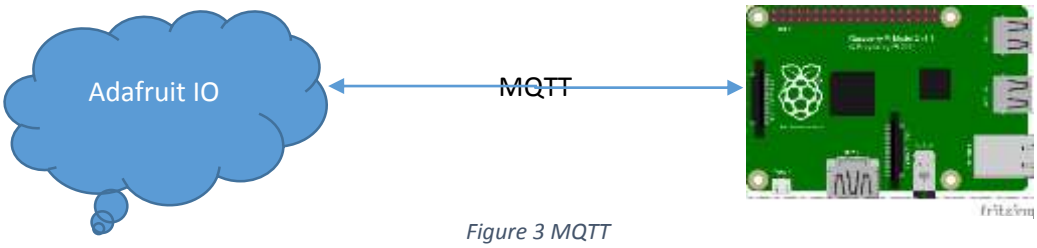


Figure 3 MQTT

IFTTT

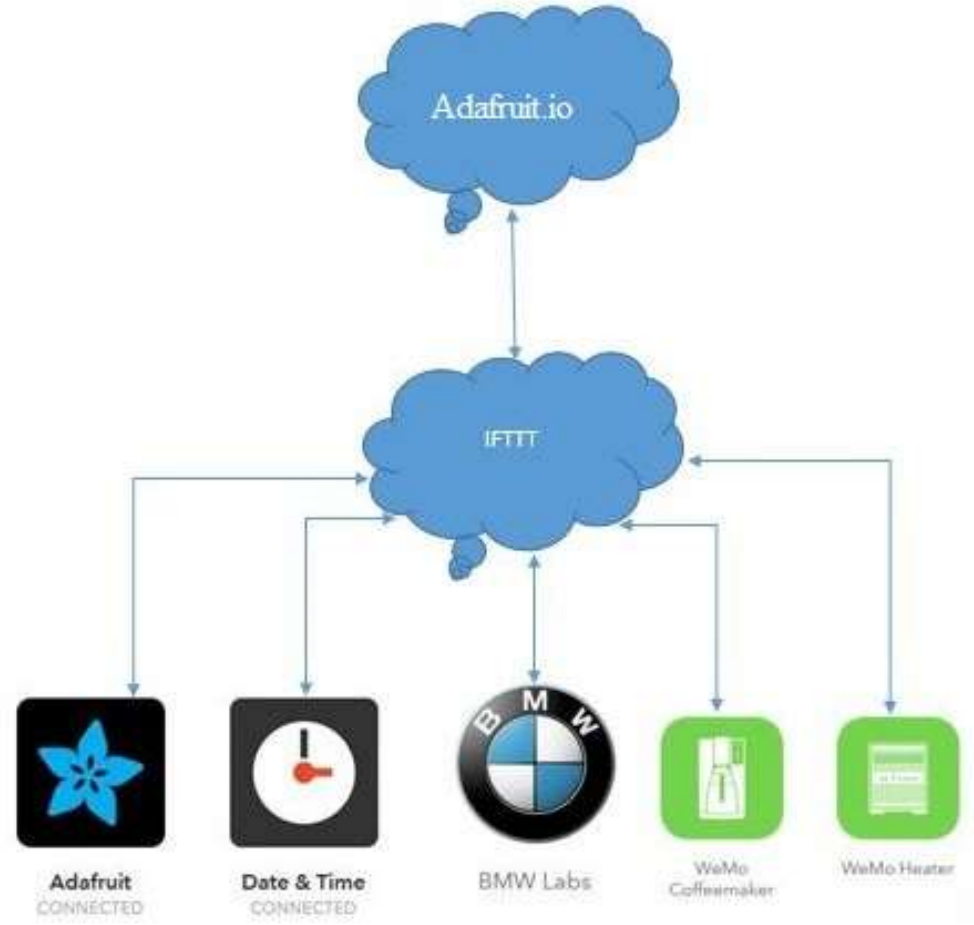


Figure 4 IFTTT

IFTTT is a free web-based service to create chains of simple conditional statements, called applets. An applet is triggered by changes that occur within other web services such as Gmail, Facebook, Telegram, Instagram, or Pinterest.

Using MQTT client library I wrote a python code to receive the sensor data and feed it to IoT platform. I have used a MQTT client application MQTT.fx to do the testing. The data successfully reached IoT

Figure 5 data flow from sensor to IoT platform

To control the slave I have used C++ to handle wireless module nRF24L01, to feed the data to Adafruit IO I have used python code. The below figure shows the architecture.

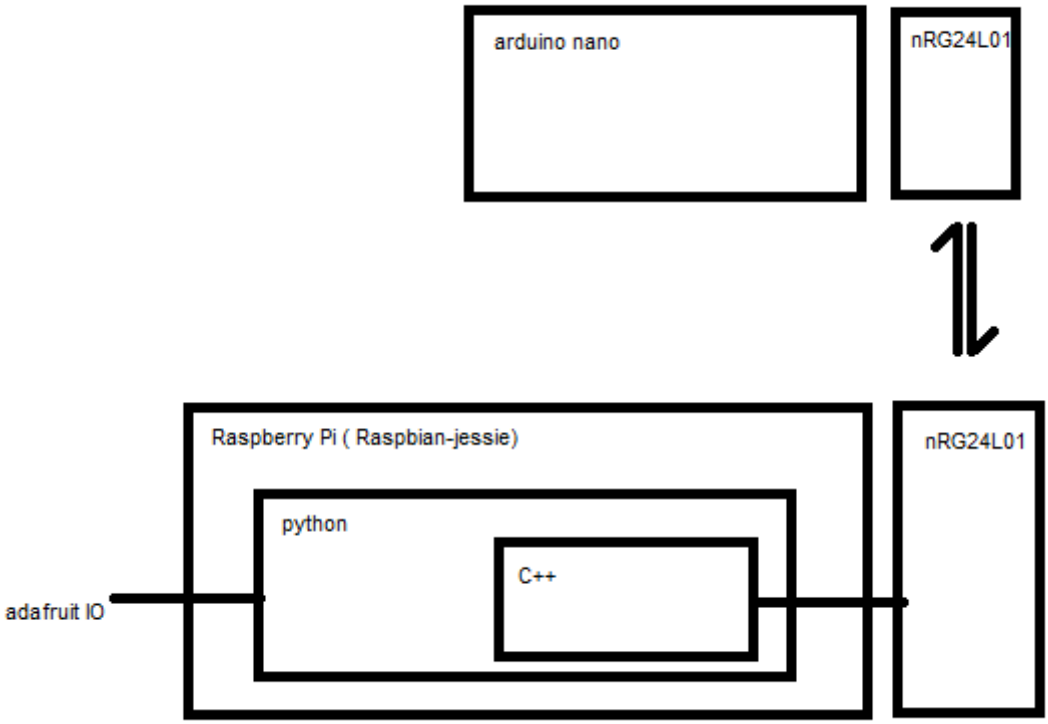


Figure 6 architecture of relay operation communication

Industrial internet of things (IIoT)

Figure 7 IIOT network example

The application of the IoT to the manufacturing industry is called the IIoT. The IIoT will revolutionize manufacturing by enabling the acquisition and accessibility of far greater amounts of data, at far greater speeds, and far more efficiently than before. A number of innovative companies have started to implement the IIoT by leveraging intelligent, connected devices in their factories.

Global economy on IIoT

- 46% of global economy that can benefit from the Industrial Internet
- 100% Industrial Internet potential impact on energy production
- 44% Industrial Internet potential impact on global energy consumption[1]

Industry

In order to make better business decisions, the IIoT offers companies the ability to:

- Aggregate data from existing sources.
- Create additional data sources in a cost effective way.
- Gain visibility into new data.
- Identify patterns.
- Derive insight through analytics.

SIMATIC IOT2040 from the Siemens

Figure 8 IOT2040

This device is made to be the mediator which connects the non-smart edge devices to the internet. This product IOT2040 stands out of by having special features like Arduino sketch support and pin out. Which means you can run arduino sketch on this device. And this runs yocto Linux which means you can run you python, C, C++ or java code in this machine and control industrial devices using Ethernet or Modbus through a PLC or even directly the device.

With acquired knowledge, discussions and decisions made over several weeks; I started to design the prototype as a proof of concept. The design is to connect field devices all the way to IOT platform/cloud service through the PLC and through IOT2040. So the local control and automation is done by the PLC and remote data acquiring, archiving, analysis, visualization and supervisory control can be done at IOT platform.

This design also enables the engineers to implement this concept on an existing conventional automation system.

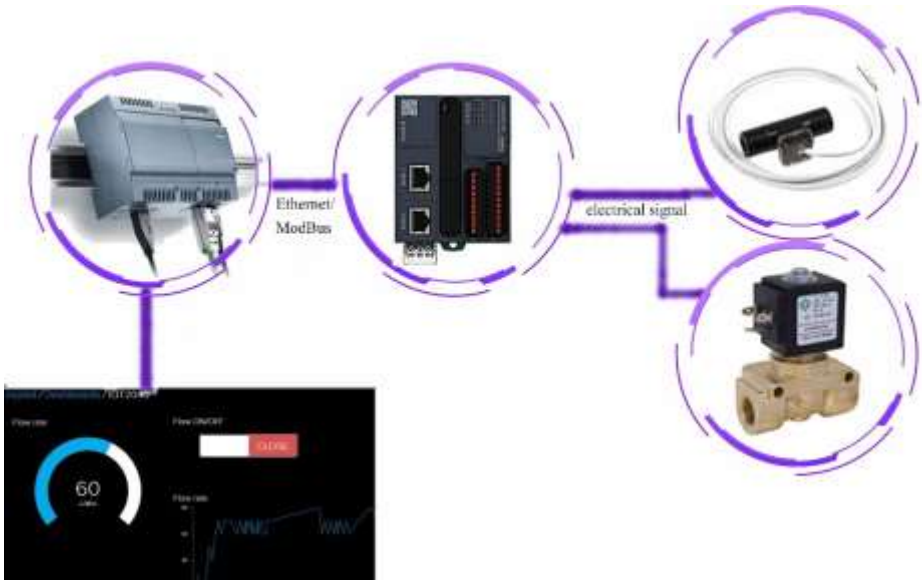


Figure 9 Design for PLC

This diagram shows communication from a remote device to iot2040 over the cloud

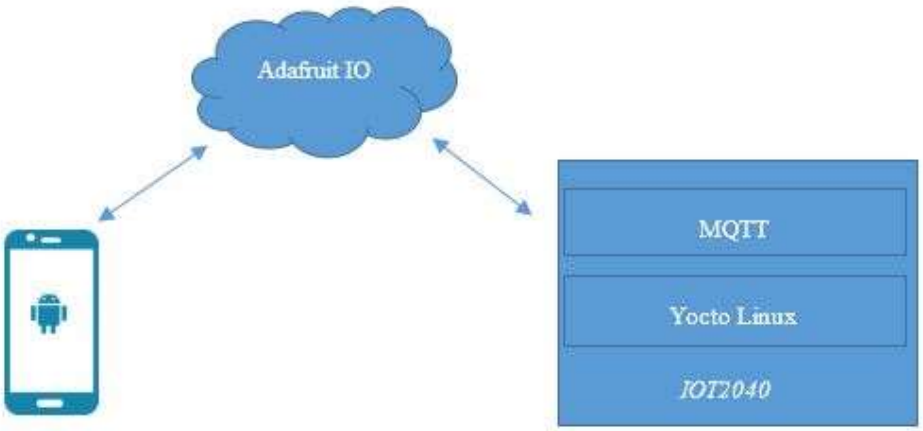


Figure 10 IoT platform

When we consider the Adafruit IO, we don't wave more flexibility on scalability but the price is fixed. And its learning system is very effective and simple. I have studied the leaning materials from Adafruit IO and built a Dash board in Adafruit IO and programmed the IOT2040 device accordingly.

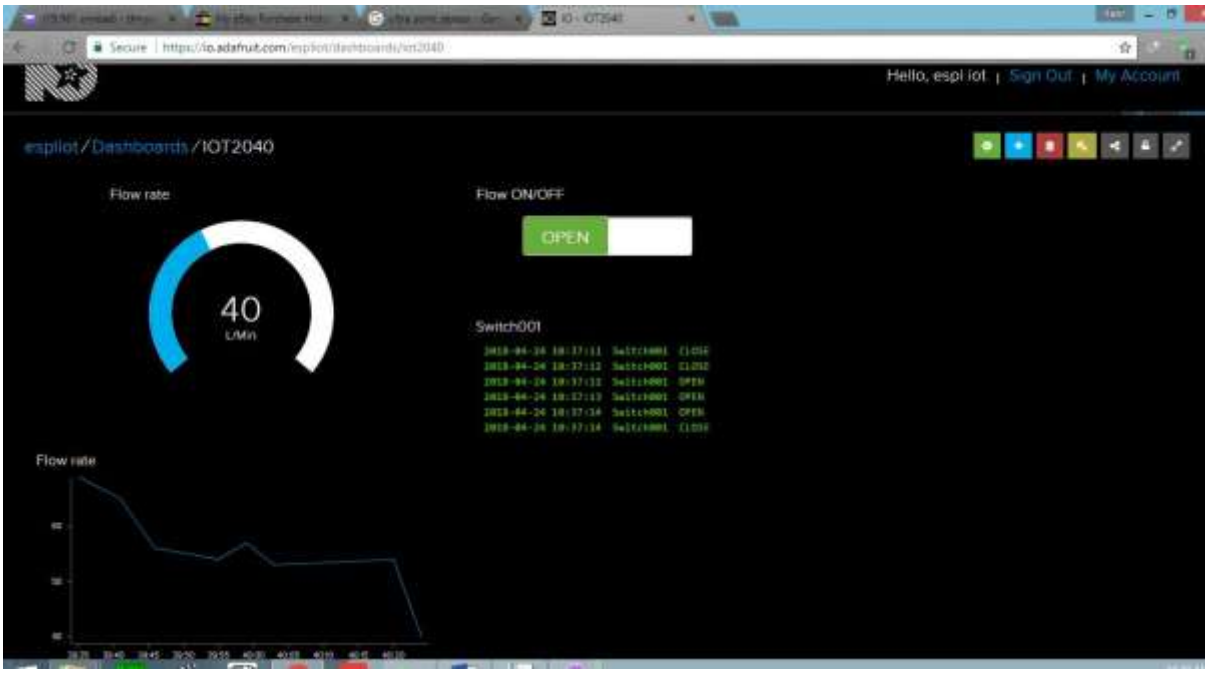


Figure 11 Adafruit IO dashboard for IOT2040

Car park management system

I have deigned and prototyped a car park management system which features High frequency RFID Reader/Writer module, proximity sensors and a Raspberry pi computer. RFID readers are used to recognize the vehicle and proximity sensors are used to detect the occupancy of each and every slot. A graphical guide to available slot is provided based on available data. And the Data is available online through an IoT platform.

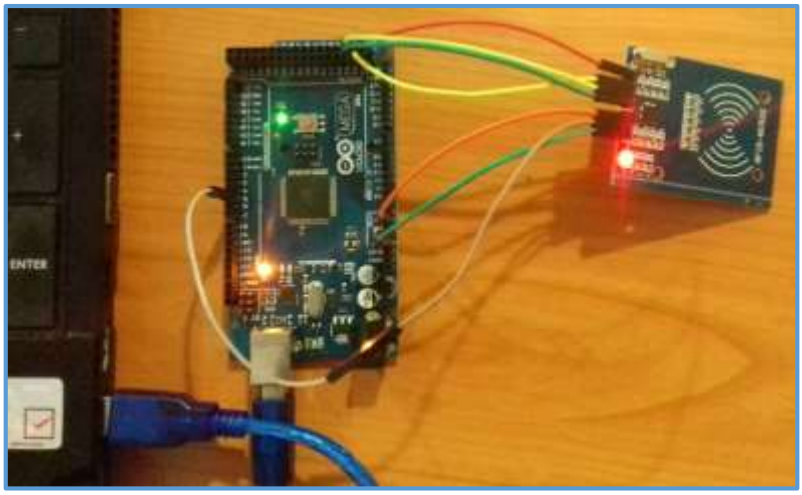


Figure 12 RFID module connected to laptop via Arduino

Parking is the way a property first welcome its visitor, putting a first belief for the rest of the visit. By and large we confront part of issues in stopping our vehicle, similar to where to park, who will deal with our vehicles and imagine a scenario where somebody stoles my vehicle. Parking system encourages everybody to make the parking region a more secure place for its visitors.

It is a private parking facility, so the main theme of this project is access control to the parking area and also for each specific slots for selected slots and parking space availability/occupancy monitoring.

13.56MHz High frequency RFID

Figure 13 RFID reader (Left) Smart card (middle) RFID Tag (Right)

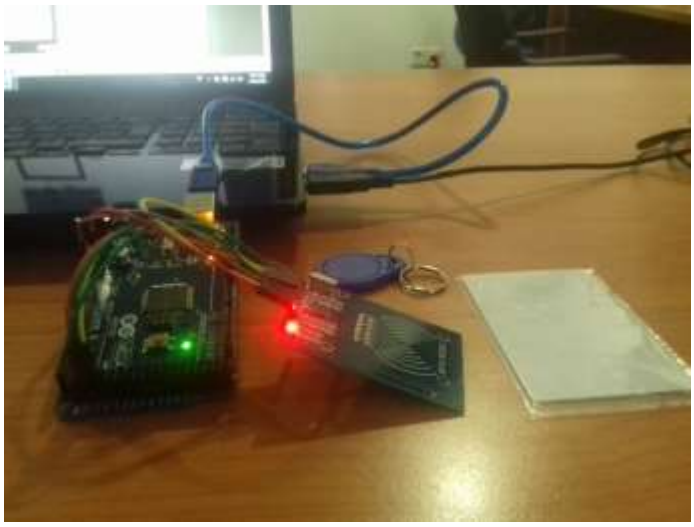


Figure 14 RFID > Arduino > Laptop (Arduino IDE)

Using this setup at the entrance/exit of the parking arena to identify the vehicle. The driver will show the Smart card or a RFID tag to a RFID reader to get the access to parking arena and get the information of space availability and its location.

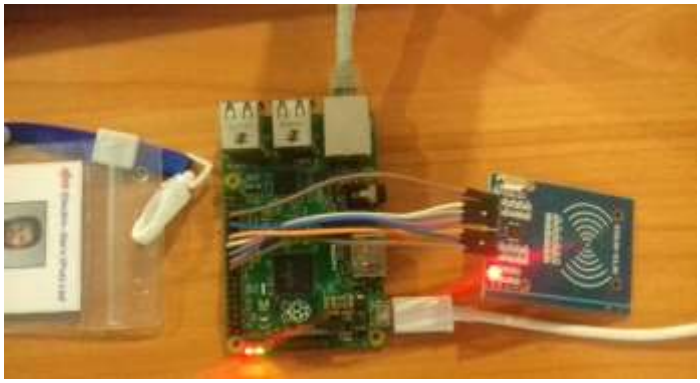


Figure 15 Raspberry Pi > RFID Reader

Centralizing is a better approach to connect things together and work as a group. So RFID and the occupancy sensors will be connected to Raspberry Pi. With this setup; every time a car comes, we scan the Smart card, recognize, show available slot/slots and record which slot is taken by that car/vehicle. So we can have the data which slots are occupied and by who.

For this purpose I have used Raspberry Pi 2 Model B. even though the latest version is Raspberry Pi 3 Model B+, Pi 2 Model B is already available in the inventory and the specifications are enough so I just have used it.

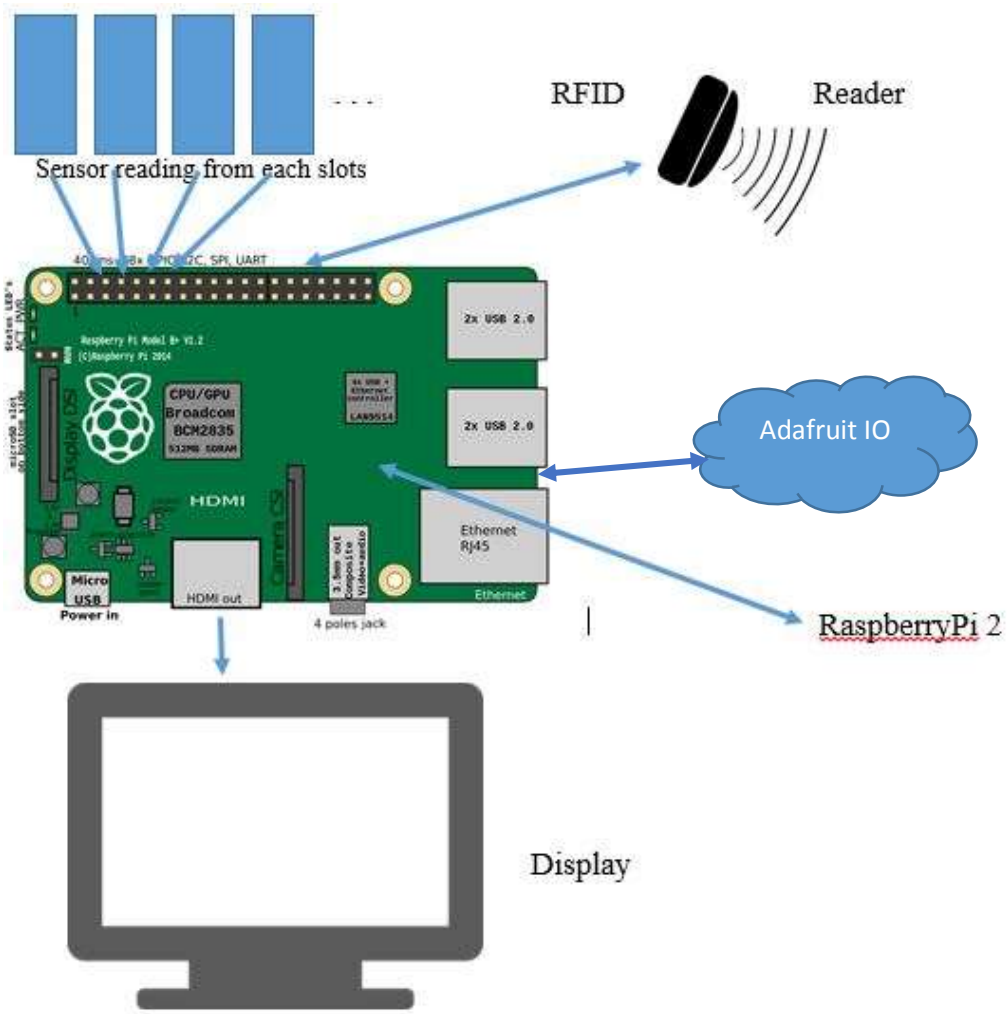


Figure 16 Overall connection diagram

A successful reading of the RFID card from Arduino IDE serial monitor is shown in the figure 41 below, the information of “Manager” and “Employee2” are printer as Name while I have already written to the RFID card.

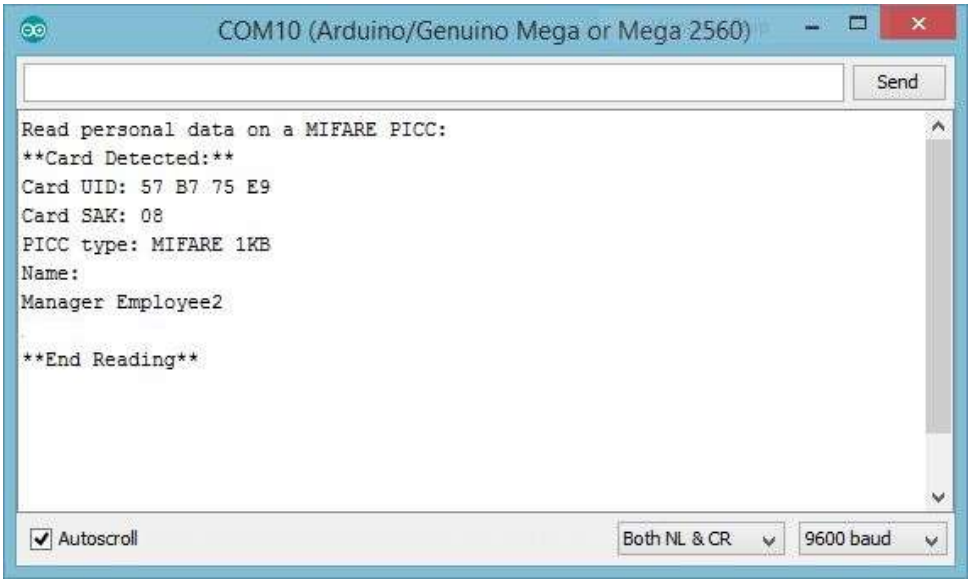


Figure 17 RFID Card read success

Graphical output in Display

Figure 18 Path to slot

Adafruit io Dashboard visualization of the data

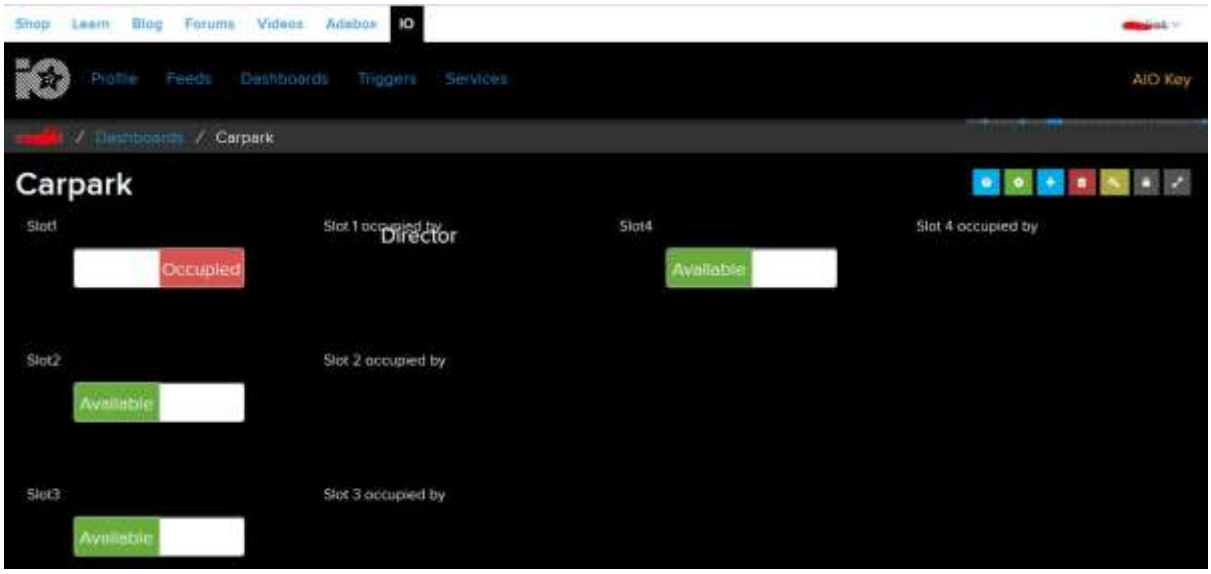


Figure 19 web-view I

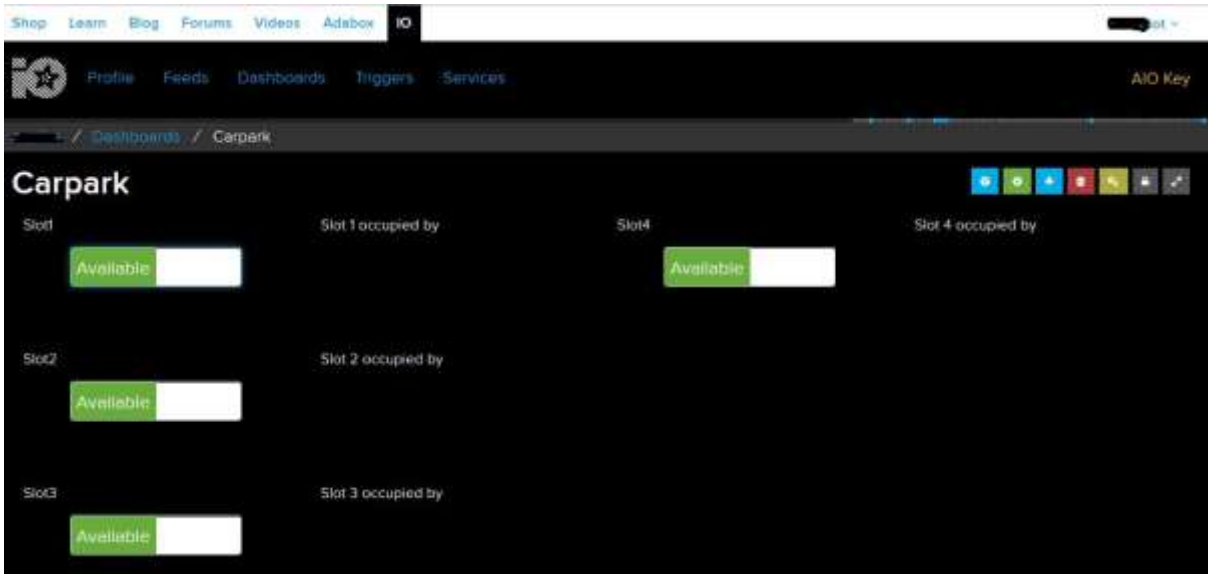


Figure 20 web-view II

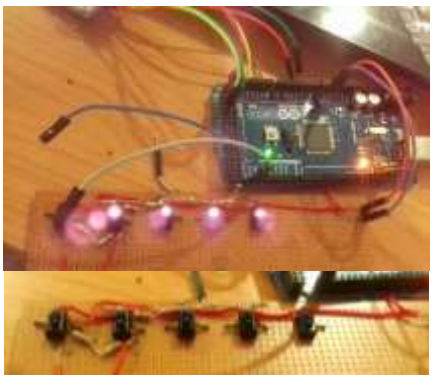


Figure 21 IR-Sensor arrangement POC for slot detection

Real-time Unmanned Car Park Ticketing System

Figure 22 Prototype

When it comes to ticketing systems of parking system it will be either manned or token based both are slow and less efficient which is not helpful for the busy world. Even though there are systems with ALPR, it seems to be slow on recognizing the license plate of a vehicle. So we came with a solution which recognizes the vehicle almost instantly and automatically, calculate the fee and provides a printed ticket.

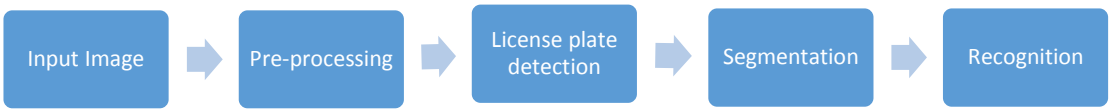


Figure 23 ALPR Steps

We have used K-NN (K-nearest neighbor) algorithm. The k-nearest neighbor algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression:

In k-NN classification, the output is a class membership. An object is classified by a majority match of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor. In k-NN regression, the output is the property value for the object. This value is the average of the values of its k-nearest neighbors. Figure 4 below shows K-NN Classification with showing K-value assigned.

As the project is an embedded system we have to choose a single board computer to handle the required processing task with speed. I chose Orange Pi plus2 board considering the processing power of 1.6GHz (Quad-core), 2GB RAM and 16GB EMMC. And then a proper OS is needed to utilize the hardware resource and run our ticketing system. So I looked at available operating system for orange pi and chose Armbian Ubuntu; built for Orange Pi plus2. When choosing OS I checked whether the OS supports OpenCV3-python. Because I am unable to compile OpenCV with python in some Operating systems.

With perfectly running OS, OpenCV3 library is compiled and installed. After that I have modified the python script of algorithm to best fit the OS. Up to this part it was done to two orange Pis' one for entrance system and one for exit system.

And for entrance system I have mounted a remote directory of other Orange Pi (exit system) on local Pi which makes the storages of both Pi's virtually one and it is executed with ssh protocol so the data is accessed over secure channel.

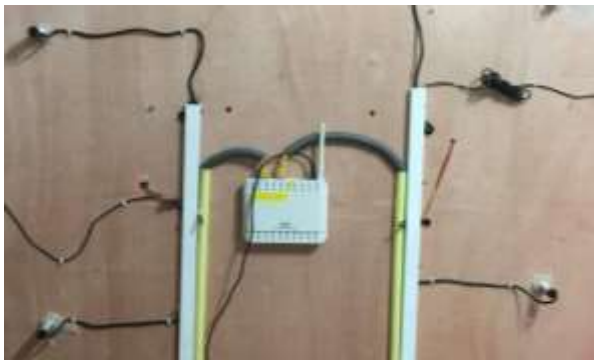


Figure 24 router mount under the platform

And wrote scripts to read sensors, activate camera, save a snap on local directory and call License plate recognition system and it will run the plate and save the time in a text file with a file named of the license plate number and save it in the remote directory (mounted) and open the barrier.

The scripting includes reading from and writing to GPIO pins to handle sensors outputs and barrier opening/closing.

The process loop is visualized below.

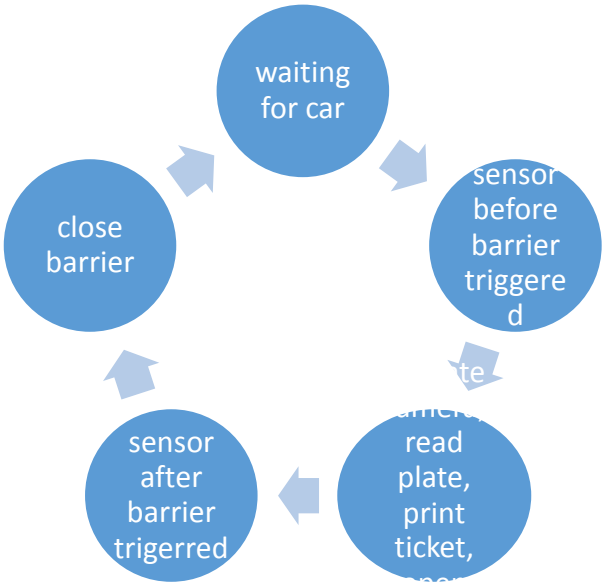


Figure 25 Process flow at entrance

And at the exit system main algorithm is scripted to read the plate and look for the text file named as the plate number and calculate the duration as well as the ticket price and print the ticket, deletes text file which holds the time of the vehicle entrance with license plate number as filename and open barrier whenever it is called. A script will be running waiting for the sensor output before the barrier like the entrance system and whenever there is a car it wall call the main python script of algorithms and when car hits the sensor after the barrier it will close the barrier.

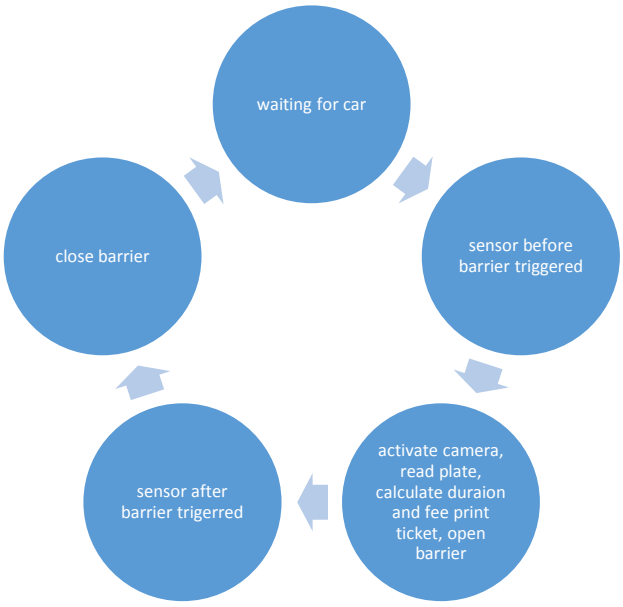


Figure 26 Process flow at exit

Figure 27 the progress

Result

Data

Condition	Total plates	Detected correctly
Daylight	50	46
Lowlight	50	31

Accuracy

Accuracy is calculated for two condition by collecting data in two conditions

Accuracy in Daylight is 92%

And in Lowlight it is 64%

POC Video: https://youtu.be/NyBRi3nz8C0