Design Report Components

(approved project-specific outline with lead instructor & client by April 7th)

# Letter of Transmittal (could be text of an email)

* Addressed to sponsor, interested parties
* Says “here’s an (interim, final) report on the XYZ project”
* Provide appropriate acknowledge for sponsor support

# Cover Page

Title, Authors, Contact Info, Logos

# Front Matter

## Table of Contents

Make it easy for people to find things

* For headings, look at the Report Body section below

## Executive Summary – ½ page

A short, powerful synopsis

* Intro sentence: what you are designing
* Needs -> Features of solution -> Benefits
* Pivotal technical and business merits
* Summary of quantitative test results

# Report Body

## Background – ½ page

* Describe sponsor motivation for the work
* Identify the need/opportunity associated with this project
* Summarize benefits to different stakeholders

## Problem Definition – 1 page

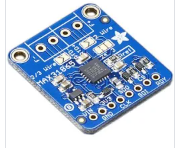
* Outline project goal(s) and deliverables
* Inventory specifications & constraints (preferably in table format w/units & target values)

## Project Plan – 1 page

* Describe team roles & responsibilities
* Discuss intended and actual Schedule (reference Gantt charts in Appendix)

**Alternate sensor/detection methods considered**

VL6180X – Adafruit time of flight sensor:



This sensor is similar in functionality to the Ultrasonic sensor that we ultimately used. A laser is used to ping and read, measure the time between the two, and calculate the distance to the object in front of it. The voltage is appealing for a 3v3 system, being only 2.7v to power the module.

Potential issues:

Cost: these sensors are roughly $14-23 dollars per unit, which significantly reduces our budget if they fail to work as we intended them to.

I2C interface:

Although there isn’t usually an issue with multiple modules using the I2C bus, in fact, this is the specialty of this medium, allowing for communication to dozens of addresses with only 2 bus wires, one to send and one to receive. However, when working in and out of deep sleep modes, the less time that is spent within an interrupt routine the better, as well as messing with global variables and having to implement semaphores and mutexes, to keep from freezing, corrupting or crashing the system. It seemed more reasonable to use a sensor with wiring independent of other modules in the system.

KY0030 microphone sensor:



This sensor can be used to detect the volume of surrounding noise. We figured that tuning it to recognize when a vehicle was in front of it, based on raised levels in the unit’s sound threshold, that we could have a cheap and accurate way of detection. The units are very cheap and require very little power to keep them running.

Issues:

Tyrel tested a couple of these sensors at his house early in the first semester, and quickly discovered that these sensors are very sensitive to the noises they trigger to. A snap of the fingers would set them off at the lowest sensitivity setting, making it impossible to use with any sort of accuracy in a noisy garage, looking for noisy vehicles.

LLC-161 light sensing resistor, paired with a 650-NM laser module



Another one of our considerations in vehicle sensing was a “broken light” sensor. The laser would be positioned at an angle, and project from the upper side of the stall, to the opposite and lower side. On the receiving end of the laser would be the light sensor, if the sensor detected light, there was no vehicle in the way, and thus the stall would be open, and if it no longer detected the light source, that meant that a vehicle was blocking it, and the stall was closed.

Issues:

Although being one of the cheapest sensor ideas we considered, the slightest bit of wind, dirt or tampering could easily send these units into false reads. Testing from home proved that these units are very picky to the angle at which the light is received; a few degrees off would “break” the light source and the system would detect a vehicle when only the laser had moved a tiny bit.

E0S2-1200TVL Micro FPV Camera:



Suggested to us by various faculty members, was the use of a low-power camera module to detect vehicles. This would work in much the same way as the Infrared sensor, and would compare frames to detect when there was a change, or a vehicle present.

Issues:

The main issue with using a camera, or at least the issue with our particular use, is the power usage. To run our Units on battery life for a year, there is no way a camera could be used for even a few minutes per day, let alone all the time, or even on interrupts. While this would be a great option in a “wired in” system with unlimited power resources, it was simply not feasible for our project.

**Concept Selection**

The sensors

**HC-SR501 PIR Motion Sensor 3v:**



This sensor is what we chose to “wake” the system. The Arduino board itself goes into a very deep sleep when there is no activity, and shuts down almost everything, waiting for an external interrupt to trigger and wake it up. This sensor has the ability to wake the Arduino in this way.

The module itself looks for heat signatures, and compares them to its previous reading. In this case, if the sensor sees no heat, it just waits for a few milliseconds and checks again. It does this at an extremely low rate of power consumption. When it finally detect that a heat signature has changed, or it “sees” heat, it sets its data pin to HIGH, and tells the Arduino to wake up.

**Factors in selection**

Power

This unit runs at 3v power as well as logic, and uses less than .1Ma when it is idle.

Price

When purchased in ordered of 5 or more, this unit is less than $1.50

Form factor and ease

Having only three pins, a power, ground and data line, this unit is extremely reliable and easy to work with. Also, its 3v version is much smaller and more linear shaped than the 5v counterparts, making it easier to place in a small enclosure.

Testing

***See testing results page in final report***

**HC-SR04 Ultrasonic Distance Sensor**



This sensor is used in conjunction with the PIR sensor above. It pings, or sends a very brief sound wave out from one side, and receives the rebounded ping on the other. The sound bounces off an object and based on the time it takes to come back, the sensor can determine the distance that the object is from itself. This sensor is used to redundantly check the presence of a vehicle after the PIR sensor has woken the system It performs 3 checks rapidly, and all three must be a reasonable distance reading, or the values are rejected and retested. This is done to avoid changing the LED to a false state based on false readings. The 2 – to – 1 vote ensures that erroneous reads are not accepted.

**Factors in selection**

Price

These units are one of the cheapest and most available sensors there are. One unit costs less than $2, and this price drops even more when purchased in small quantities.

Power

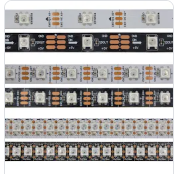
These units use less than 2mA statically, and they are on for less than a second whenever the system is woken from deep sleep. They work at both 5v and 3v power as well as logic.

Availability

These units are extremely easy to obtain, and can be purchased from a multitude of different vendors, making pricing and shipping much more easy and affordable.

**The LEDS**

WS2812B RGB LED Strip (One single LED)



For the LED indication, we used one single LED from these strips. The LED is used to signal to the commuter the state of the parking stall. As they are flat, they don’t emit light very wide, but rather pinpoint and harsh. A custom diffuser was made to enclose the LEDs and spread the light, making them visible in all lighting conditions.

**Factors in selection**

Price

Most Arduino hobbyists have a least one or two of these long LED strips laying around. The cost of one LED from them is virtually negligible. If we were outfitting multiple Units at once, a single $20 strip of these LEDs could cover over 100 units, at about 20 cents per unit, making it our cheapest component in the GSU

Ease of use

These LEDs are extremely easy to work with. I have personally written several routines and a couple of libraries for controlling them in personal projects. They are only one way in their communication. The microcontroller fires a status to them and moves on. For this reason, there are only 3 pins just like the PIR sensor, and the form factor is very small.

Visibility

These little LEDs are much brighter than the standard bulb-shaped ones that are found in many hobby kits for Arduino programming. The light is very strong and vibrant, and can be easily manipulated with various diffusing techniques.

**Other Modules**

The other modules and components used for the final unit design have not changed throughout this project, and are very standard, not subject to various operational traits and functionalities. They are simply listed below, as their properties and uses are standard for any use:

* 2500Mah Lipo battery
* Lipo charging board
* Latching power switch
* Real time clock
* LoRa antenna

## System Architecture – 2+ pages

**Design**

We wanted the design of this project to reflect the footprint of the power and sensors used within it. The enclosure is very compact, the sensors are tiny and use very little power. The mesh networking protocols are speedy, reliable and small as well. The simulation accompanies these other properties of the project by being hosted on a webpage that runs on a board not too unlike the one that is powering the units themselves.

The whole project, although implemented with full power, and a house-to-house communication with WiFi in between the LoRa transmissions, was designed to be deployed in a parking garage, and last as long as possible on a single battery charge. Moving forward, the programs running on the units can be changed to match the garage setting, as well as the networking and simulation. Only minor changes need to be executed to achieve its original and intended functionality.

**System Functionality**

The components and their basic system function can be found above in the Concept Selection area, and a more detailed explanation of system-wide communication and functionality can be found on our Github:

Below are the links to the various data flow diagrams of the full system architecture on each level, as well as a Fritzing diagram of all the components and how they are wired together:

Context Diagrams:

Level 0 - <https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%200.pdf>

Level 1- <https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%201.pdf>

Level 2- <https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%202.pdf>

Fritzing Diagram - <https://github.com/Jaxal83/parkingSensor/blob/master/Spring%202020/Hardware%20Design/Park-It-CDA%20fritzing.JPG>

**Novel Features**

Enclosure

The enclosure is the result of many changes to the project throughout the last two semesters. Every time a component changed; the enclosure changed. Zane designed the enclosure while learning Fusion360 modeling through Udemy.com, so each iteration was a litter more suitable and acceptable than the last. The earlier models were very rough, and did not have any inner cavities or standoffs to accommodate the various components. They were zip-tied and hot-glued to begin with and were ultimately measured and designed around in the final build.

Simulation

The graphical representation of the simulation is a very easy to read layout of a parking garage. Tyrel made the simulation small and concise so that the speed and availability of the webpage are consistent with the rest of the project. The entire webpage is hosted on a small microcontroller as well.

The overall setup of the project

The biggest “us” factor that is present in this project is the way the data is flowing for the demonstration. We had to pivot very hard due to the Covid-19 circumstances and were able to achieve house-to-house communication and a way to test our various portions of the project. This can be found in much greater detail in our Github.

**Satisfaction of requirements**

Sensing vehicles

As mentioned above, the two sensors in our units work in conjunction to accurately detect when a vehicle is present below. Erroneous results are thrown out, and the LED only changes when the stall status changes. A complete demo video of this can be found in the Expo presentation on Github

Mesh networking

The units are able to communicate with each other, as well as the gateway. Units that are too far from the gateway can send their status to the nearest neighboring unit, and so on until a unit close enough to the gateway is able to send the status of the stall.

Simulation

The simulation shows an accurate representation of the parking garage, as well as the parking stalls in Zane’s driveway. The gateway receives a LoRa transmission and is able to send it via WiFi to the Raspberry Pi that is feeding the data to the webpage. This happens very quickly and stall indication via the webpage can be easily and quickly portrayed. A demonstration of this can also be found in the Expo presentation section of our Github.

## Design Evaluation – 2 pages

* Analyze DFMEA against project specifications (1-2 page Excel document included in Appendix)

(define scoring rubrics, assess design risks, and summarize key ideas for remediation)

* Explain product testing procedures
* Sustainability assessment
* Provide results from product performance testing

## Future Work – 1 page

* Make recommendations for sponsor in project adoption/implementation
* Identify features that didn’t find their way into the current design
* Estimate scope, duration, and cost of the required next steps

# Appendices

Supporting documents to long or detailed for main body

* Calculations, drawings
* Large tables, figures
* Computer programs
* Vendor data sheets
* 1-page Project Schedule in Excel (as originally planned at start of project)
* 1-page Project Schedule in Excel (as executed at end of project)
* DFMEA worksheet
* Overview of folder/file organization on shared drive