SMART PARKING SYSTEM

THOMPSON RIVERS UNIVERSITY

EPHY 2990

Cem Doganay and Jerry Kuo

April 16th, 2019

THOMPSON RIVERS UNIVERSITY

1. Overview

1.1 Introduction

North American vehicles have been slowly adding advanced safety systems as the technology started becoming readily accessible and affordable. Devices such as small cameras and discrete proximity sensors are now integrated into a majority of cars. We will continue to see better safety systems installed in cars, especially with the research and development going into these systems being motivated by the push to fully autonomous cars. Many modern cars (manufactured after 2014) come equipped with some sort of safety system. These include back-up cameras, blind spot obstacle warning, parking assist, lane drift detection, and more. But, while modern cars include these safety features, the older vehicles on the road may be less safe to operate because they did not come with these features from factory.

1.2 Purpose of project

The main purpose of this project is to research the feasibility of an affordable, easy to use parking obstacle detection and warning system. This obstacle detection and warning system is intended to be used when users are attempting to park.This system will not distract the user, but will be provide the necessary information at a glance. This ensures the user is fully attentive to the operation of their vehicle at all times, with the added enhancement of knowing about hazards they may not see while parking. Below is a list of expanded purposes.

1.3 Objectives

1.3.1 Identifying how the system will function technically. This includes hardware and code.

1.3.2 Verification of a working system through various tests explained in the testing section.

1.3.3 Checking the financial achievability to verify if consumers can reasonably use said system.

1.4 Purpose of document

This document aims to present a detailed breakdown of the Smart Parking System. The designs, features, methods of intended use, and operational constraints will be detailed. The scope, timeline, performance requirements and determining their success through testing will be included as well.

1.5 Scope

This system will be parking system meant to be mounted on vehicles without having to damage the car. This statement refers to damages such as drilling holes in bumpers, having adhesives damage paint, etc. The system aims to warn the user of dynamic (pedestrians, animals, moving objects) and static (curbs, poles, non moving objects) obstacles around their vehicles via screens that display the distance an object is away from the vehicle. Specifically, this system will be made up of multiple units each containing a microcontroller, sensors, and displays. The target audience is vehicle operators with older cars, motorcycles, and other vehicles who may not be able to upgrade (financially unable, sentimental attachment to older car, etc). Keeping the cost as low as possible will be an important aspect of this project, however, the main focus will primarily be on the technical feasibility. The following lists contain our original scope, which has been altered throughout the project. The changes will also be detailed below. In the lists, the functional requirements refer to

core features this project would not function without. The non functional list details features that will be included, but are not completely necessary for functionality. They are as follows:

1.5.1 Functional requirements:

1.5.1.1 Detect obstacles within a 2-meter ranger.
1.5.1.2 Display warning and obstacle distance on LCD screen.
1.5.1.3 Flash warning lights indicating which corner the critical obstacle is in.
1.5.1.4 Sound speaker to warn the user.

1.5.2 Non-functional requirements:

1.5.2.1 Create an interface that allow warnings to be displayed on the phone.
1.5.2.2 Utilize wireless sensors.
1.5.2.3 Weather-proof outside sensor enclosures.

1.5.3 Changes to scope

From the Core features we removed the warning lights and sounds. When we were doing some testing we realized the lights and sounds proved very distracting and did not align with our goal providing a non-distracting experience for the user. We instead added four 7-segment displays that display the most critical distance at each of the four corners. The bigger LCD displays the distance each sensor detecting should the user require more information than just the critical distances. The wireless functionality was also moved into functional features because we could not find a way to use wires without damaging the vehicle. Since we have multiple slave units, going wireless reduces the installation difficulty by eliminating the need to run long wires to each corner of the car. Taking all that into consideration, wireless data transfer makes the most sense. The non-functional features list underwent a lot of other changes as well. We did not include an interface for warning display on mobile devices. Mixing phone usage with driving is again more distracting than helpful, and is illegal. These reasons saw that requirement removed in very early stages of project design. Weather-proof enclosures were also scrapped. We did not have the time towards the end as troubleshooting took more time than anticipated. The local 3D printer also malfunctioned so we could not create even prototype enclosures. The non-functional features also saw switches and a battery level detection system added to the list. For maximum convenience to the user, we wired in physical On/Off switches and each Slave and Master unit. The user can choose to shut off the system with the flip of a switch instead of opening each individual enclosure and unplugging the batteries. The Master unit also has a Driving/Parking mode, meant to be toggled as their names suggests. This feature will be further discussed in the Design section. The battery level monitor was also included for additional convenience for the user. Once low battery levels are detected by any unit, a warning message will ask the user to replace the low batteries.

1.6 Definitions

User	Refers to person(s) operating vehicle of which the parking system is mounted to.
Obstacle	Objects that would damage vehicle if an impact between them were to occur.
Master Unit	Unit of which the LCD display panel and four 7-segment displays are wired to an Arduino Mega, used to display all data. Mounted inside vehicle.
Slave Unit	Externally mounting units containing an Arduino Nano, ultrasonic sensors, and a NRF chip sending sensor readings to the Master unit.
PR	Performance requirements.
NER	Non essential performance requirements.
Critical Distance	The distance that is deemed the closest to one Slave unit, measured between the 3 sensors on the unit.

2. Performance Requirements

2.1 Requirement

Detect obstacles within sensor range.

2.1.1 Details

The sensor on each sensor unit will scan for obstacles at set intervals. They will collect data representing whether they have detected any obstacles directly in front of each sensor.

2.1.2 Reason

The system needs external inputs to determine whether there are obstacles close enough to pose as hazards to the driver and vehicle. The sensors are necessary for detecting input.

2.1.3 Success Criteria:

Each sensor needs to be able to measure an obstacle to distances between 5 cm to 200 cm. Verify sensor functionality through the displays on the Master unit, or by

plugging a Slave unit into a PC and verifying distances through the serial monitor accessed from inside an Arduino IDE.

2.2 Requirement

Wireless communication between the Sensor and Master units.

2.2.1 Details

Data collected by the sensor will be sent wirelessly to the Master unit for further processing. Low battery warnings (see **3.2**) will also be sent to the Master unit wirelessly. Communication will be done using NRF technology. Each Sensor unit will send a unique address to the Master. The Master unit utilizes this unique I.D. to determine the location of the Sensor unit on the car.

2.2.2 Reason

An optimal way to transmit data without drilling holes for running wires is to transmit wirelessly (see **2.4.**). Additionally, the Sensor units are required to be mounted outside of the vehicle. As each vehicle is of different dimensions, for the sake of universally fitting on multiple types of vehicles, wireless technology eliminates potential wiring hassles. End user will not have to worry about varying length of wires to fit system to their automotive.

2.2.3 Success Criteria

Ensure there are no wires connected between Slave unit and the Master unit. Additionally, criteria **2.1.3** needs to be fully met for **2.2.3** to be deemed fully functional.

2.3 Requirement

Display critical obstacle distances on 7-segment displays and all distances on the LCD module.

2.3.1 Details

Four 7-segment displays mounted to the Master unit will display the critical distance an obstacle is at with respect to the sensor location on the vehicle. The LCD will display every distance input from each of the 12 sensors, including the critical.

2.3.2 Reason

The user will need to know how far away their vehicle is from obstacles to judge when they should park. Displaying the distance quantitatively will quickly provide the driver with the information. The displays shows the distances at a glance, allowing the user to focus on parking only having to briefly look at the screens to accurately obtain necessary information.

2.3.3 Success Criteria

Turn on all four Slave units and verify each 7-segment display is displaying a number. Check that the switch on the Master unit is set to "Parking Mode" and that

twelve distance values are being displayed on the LCD screen. Refer to **2.1.3** and ensure the distances being displayed are within the acceptable range.

2.4 Requirement

Mount the system (i.e. each unit) to vehicle without damaging anything.

2.4.1 Details

User will not be required to make damaging modifications to their vehicles when using the system. The Master unit will attach in the interior with a mount. The Sensor units will mount externally using a mount system as well. No holes or drilling will be required.

2.4.2 Reason

Being able to use this parking system without damaging the vehicle is a big incentive to using the system. Furthermore, the wireless aspect (see **2.2**) allows for non-damaging installation and usage. As wireless communication is a core feature, it naturally leads to integrating the system to the vehicle without damage.

2.4.3 Success Criteria

Verify that no damage has occured to vehicle exterior and interior post installation. Check all systems equipped on the vehicle from factory are in working order after the installation as well.
Note: Drilling holes is unnecessary for installation.

2.5 Requirement

Provide a separate enclosure for each individual unit.

2.5.1 Details

Each sensor unit will be secured to and contained inside a closed enclosure.

2.5.2 Reason

The enclosures will shield the electronics from the elements and from agents that could cause them or the circuits to malfunction. Dirt particles, salt, rocks, etc are some examples of aforementioned agents.

2.5.3 Success Criteria

Ensure each unit is firmly secured inside enclosure by checking that nothing is freely moving around inside each enclosure. Also verify all switches on all units are accessible, that nothing is blocking the sensors on the Slave units, and that all displays on the Master unit are visible.

3. Non Essential Requirements

3.1 Requirement

Each Arduino Nano will function on a real-time operating basis when the switch on the Master unit is set to "Parking Mode".

3.1.1 Details

The sensors will be taking measurements on a rotating cycle. They will take measurements and send them to the Master unit for a set amount of time, then be put into a low power mode until it is their turn to take measurements again. The Master unit will output a “stay awake” signal to the Sensor units every time the Master unit receives data from them. If the Sensor units do not receive this signal, they will go into a low power state until they receive a “wake up” signal and will not take measurements. If the Master unit gets turned off, as in no power going to it, when the user goes to turn it on, it will automatically send a signal to the Sensor units to wake them up. When the switch on the Master unit is set to "Driving mode", the sensors will stop taking measurements.

3.1.2 Reason The whole system relies on multiple batteries. Without the real-time operating system, the batteries will be drained too quickly and the user will have to constantly change them. Since each sensor does not need to be on and taking measurements all the time, keeping them off when they are not in use will dramatically reduce power consumption.

3.2 Requirement

Include a switch that toggles between "Driving" and "Parking" mode

3.2.1 Details

Building off of **3.1**, this switch allows the system to know whether the user wants to be seeing data from the sensors or not. If not (i.e. Driving mode), the Smart Parking System will automatically shut off the sensors on the Slave units to save the maximum amount of power possible.

3.2.2 Reason

Refer to section **3.1.2**.

3.2.3 Success Criteria

Toggle the mode switch on the Master unit to "Driving" mode. All four 7-segment displays should turn off. All distance readings on the LCD screen should be replaced by battery statuses.

3.3 Requirement

Warn user when low battery

3.3.1 Details

Voltage will be measured across the Master unit and each Sensor unit. Once the measured voltage drops below a factory set threshold, different processes will happen based on which unit is detecting the low voltage. On the Master unit, the microcomputer will output the low battery warning onto the LCD screen. When a Sensor unit detects low voltage, it will wirelessly send a signal to the Master unit, telling it to display a warning.

3.3.2 Reason

Warning the user when power is low is an important part of ensuring the system is functioning as intended. When battery gets low, the sensors may start not detecting obstacles accurately. If the battery dies completely in any unit, the user will lose all operations from it. The user needs to be aware of power levels to prevent above mentioned scenarios from hindering system functionality.

3.3.3 Success Criteria

Due to voltage detection limitation on the Arduino, this requirement is success criteria is more complicated to test. To test the battery monitoring system, connect a voltage source between 6 to 7 volts to the V_{In} pin of any Arduino. Connect this source in parallel to the voltage divider attached to the included battery packs and ensure the output of the divider (the branch where the 2 resistors are connected) is plugged into pin A_0. The LCD screen should now display a low battery warning message and ask the user to replace the batteries on the unit detected with low batteries.

3.4 Requirement

Turn system On/Off manually, operated by user.

3.4.1 Details

Multiple power switches will be included with the system. This allows the user to shut off the device when they do not require the system warnings.

3.4.2 Reason

When the user is not trying to park, they may not want the sensors to stop detecting obstacles as to not get constant warnings. Allowing the user to manually turn the parking system On/Off enables them to choose when they want warnings. This feature also allows the user to save power, therefore reducing the frequency of battery changes.

3.4.3 Success Criteria

On the Slave unit, turn flip the switch to "Off" position and verify the green external LED is not on. User could also look at the Arduino Nano and check that the onboard LEDs are off as well. When the switch is in the "On" position, all LEDs should be lit. On the Master unit, flip the power switch to "Off" position and the 7-segment as well as the LCD screens should be off.

4. Hardware Design

4.1 Bill of Materials

Equipment	Quantity
Arduino Mega	One

Arduino Nano	Four
Ultrasonic Sensor (HC-SR04)	Twelve
NRF Wireless Transceiver (NRF24L01+)	Five
LCD Screen (HD44780)	One
7-segment Display	Four
Switch (Single Pole, Single Throw)	Five
Resistor (1 MΩ and 860 kΩ)	Five of each
Resistor (10 kΩ)	One
LED (green)	Five
Breadboard (ALLCA-BB-106)	Two
Breadboard (LS00018)	Four
Jumper Wires	Multiple

4.2 Arduino Mega

4.2.1 Requirement

We need a Master unit that can be mounted to the dash of a car and can process all the data being picked up by our sensors and manipulate them in certain ways.

4.2.2 Alternatives

We considered other controllers such as Raspberry Pi, Embedded Controllers, LaunchPad, Beaglebone, etc.

4.2.3 Explanation

Since the master unit is dash-mountable, the controller had to be relatively small and portable. This criteria automatically made tablets and devices like that unviable. As for the remaining competing microcontrollers of similar size, we decided based on a couple of hardware criteria. Our project needs the data to be processed quickly so the users can get accurate information, so we did not choose something like LaunchPad with it's lower specs compared to the others. On the other hand, we are not running a huge amount of processes simultaneously, so the Raspberry Pi's specifications seemed overkill, and it's more expensive than a Mega. Additionally,

we have a lot of peripherals that need to interface with this main controller, so we chose the Mega for its plethora of available pins.

4.3 Arduino Nano

4.3.1 Requirement

Our Sensor units needs to be a small unit for external mounting on vehicles. It also needs a device that can execute processes that tell the ultrasonics to start collecting data and the transceivers to send data to the “master” unit for further processing.

4.3.2 Alternatives

Again, we considered other controllers and even pursued the possibility of meeting the requirement without an external microcontroller.

4.3.3 Explanation

Due to us wanting to keep the Sensor unit enclosures as small as possible, we needed to choose a small microcontroller. The Nano has the appropriate dimensions for fitting into small enclosures. Plus the Sensors do not have a lot of components that need to be attached, so we do not need a unit like the Mega with it’s 60 pins. Another reason for choosing Arduino here is because we went with an Arduino for our Master unit, so we figured we would keep the same type of microcontroller. That way we could potentially use the same libraries and even sub-routines in the code.

4.4 Ultrasonic Sensors (HC-SR04)

4.4.1 Requirement

Our system not only needs to detect obstacles in front of it but it needs to know how far away obstacles are too.

4.4.2 Alternatives

Other distance detecting sensors were considered. We looked into Infrared, Li-Dar, Laser, Optical Light sensors, and integrating a camera with Raspberry Pi and Open Source Computer Vision (library that can perform human recognition among other things) and even proximity sensors.

4.4.3 Explanation

The top reason we chose the Ultrasonics is because they are way more cost efficient than the other solutions. For example, the cheapest Li-Dar we found was ~$55, and more precise models went for upwards of $2000. In comparison, the ultrasonics we bought were ~$3 per sensor. The proximity sensors we found did not have enough range, so they were not chosen either. Another reason for picking theses sensors are their good reviews on Arduino forums. Other people have successfully used them for essentially detecting obstacles in various environments such as in their homes, outside by a doorbell, etc. Since obstacle sensing has been done using

these exact sensors, we knew it was possible, and we would have pre-existing support and resources instead of starting from scratch.

4.5 Wireless Transceivers (NRF24L01)

4.5.1 Requirement

Send data from the Sensor unit to the Master unit for further processing. Data has to be sent without damaging the vehicle, so we cannot use wires.

4.5.2 Alternatives

We researched Bluetooth and NFC technologies.

4.5.3 Explanation

We eliminated NFC very early on because it can only read/write data to devices known as “tags”. The “tags” themselves do not interface with Arduino so using them in our project would not make sense. The other solution, Bluetooth, could potentially work with our Smart Parking System. However, Bluetooth is a lot more complicated to code and we estimated it would take too much time to learn. It is also harder to use it for multiple units because Bluetooth can only connect one pair of transceivers at a time. For example, when Sensor unit A has finished sending data, it has to disconnect and Sensor unit B has to automatically connect before it can feed data. The automatically connecting portion is unreliable, and if it fails, the user would be missing a lot of crucial data (i.e. an entire Slave unit would be offline). While both Bluetooth and NRF only support one stream of data communication at a time, the NRF allows simultaneous connections with other transceivers, which eliminates unreliability in that area and thereby provides a massive advantage over Bluetooth.

4.6 LCD Module

4.6.1 Requirement

The Smart Parking System needs to display some crucial information to the end user. First, the distance each sensor is detecting need to be shown. Second, warnings for the user (i.e. a unit has low battery) need to be displayed as well.

4.6.2 Alternatives

Other warning methods were taken into consideration. Audio warnings and LED lights specifically were some methods we thought to use to warn the user.

4.6.3 Explanation

The LCD displaying text is the clearest way for a user to receive the type of warnings we want to send. While the alternative methods would work, we felt they would be too ambiguous and extremely distracting. For example, hearing a sound will not tell the user how much battery is left and would break their concentration on the road. Moving on to LED’s, using only lights would be inconvenient because it again does not show a percentage of battery life left. A bunch of LED’s would also occupy a lot

of pins on the Arduino, which is not ideal in case we need to use them for other functions that are more important (i.e. add more distance showing displays). The LCD screen offers the flexibility of displaying different warnings and other text information all on one screen, so we ultimately chose it over the alternatives.

4.7 7-Segment Digital Display (4 Pin)

4.7.1 Requirement

An interface that can display distance the car is away from an obstacle.

4.7.2 Alternatives

Refer to **4.6.2**.

4.7.3 Explanation

Similar to the explanation above, using the alternatives would not be precise enough for our purposes. The LED’s only would not be accurate enough for the user to know the exact distance between the vehicle and any given obstacle. The audio warning could not show an exact distance either, and it may irritate the driver, thereby proving more distracting than helpful. We chose our 4-pin display because it contains a TM1637 chip which automatically converts a digital signal to analog. This eliminated the need for us to send signals to the proper pins on a 10-pin segment display, which simplifies our code overall

5. Software Design

5.1 Design Preferences

This section will include design choices both slave and master units. In this project NRF24l01+ wireless devices runs on the same channel. The main units open a reading pipe and each slave sends their data through the same pipe. This method is a workaround that we came up with and it not might be ideal due to the possibility of unwanted interference. The first intention was to create a wireless network between units. However, lack of technical knowledge and time limitations made us take a different path. With a wireless network, units would be able to both send and receive data simultaneously. That would allow slave units to communicate in a synchronized pattern so there wouldn't be any interference between the devices as mentioned before. However, Arduino executes 16 million instructions per second and that makes interference a low possibility. Also, each slave unit delays the code randomly in a 0-500ms time interval. This addition even decreases the possibility of continuous interference. Another advantage of the wireless network would be the ability to turn off ultrasonic sensors during the driving mode in order to increase battery life. However, ultrasonic sensors draw only 10µA per loop cycle. This value is very low compared to 15mA current draw of NRF24L01+. Setting up a network would enable

both sending and receiving signal functionality of the NRF24L01+ and that would draw more current compared to neglectable 10μA. In conclusion, not creating a wireless network did not harm our project.

5.2 Slave Unit

Each slave unit starts with calling necessary functions and declaration of pins as well as variables such as an array that can store up to 7 variables. Channel address has also been created at the beginning. In the setup function, each pin matches with correct I/O sequence. Functions have been called in order to start NRF24L01+ and to create a writing pipe. Also, Arduino sets NRF24L01+ into low power mode with limiting its range to 25 meters. Loop function starts with an analog read of battery current in 0-1023 range and a line of calculation take place in order to find the actual current. According to the values that have been provided by our electrical engineer, an “if, else” statement decides whether the battery is healthy or low. This information saved into the array. Then functions for finding each distance from ultrasonic sensors take place. An algorithm finds the smallest value and saves into the array as the most critical distance. Also unit ID, location of the most critical sensor and all distances have been saved into the array. All values that has been saved into array displayed on the serial monitor with their meanings. Finally, this array sent by the NRF24L01+.

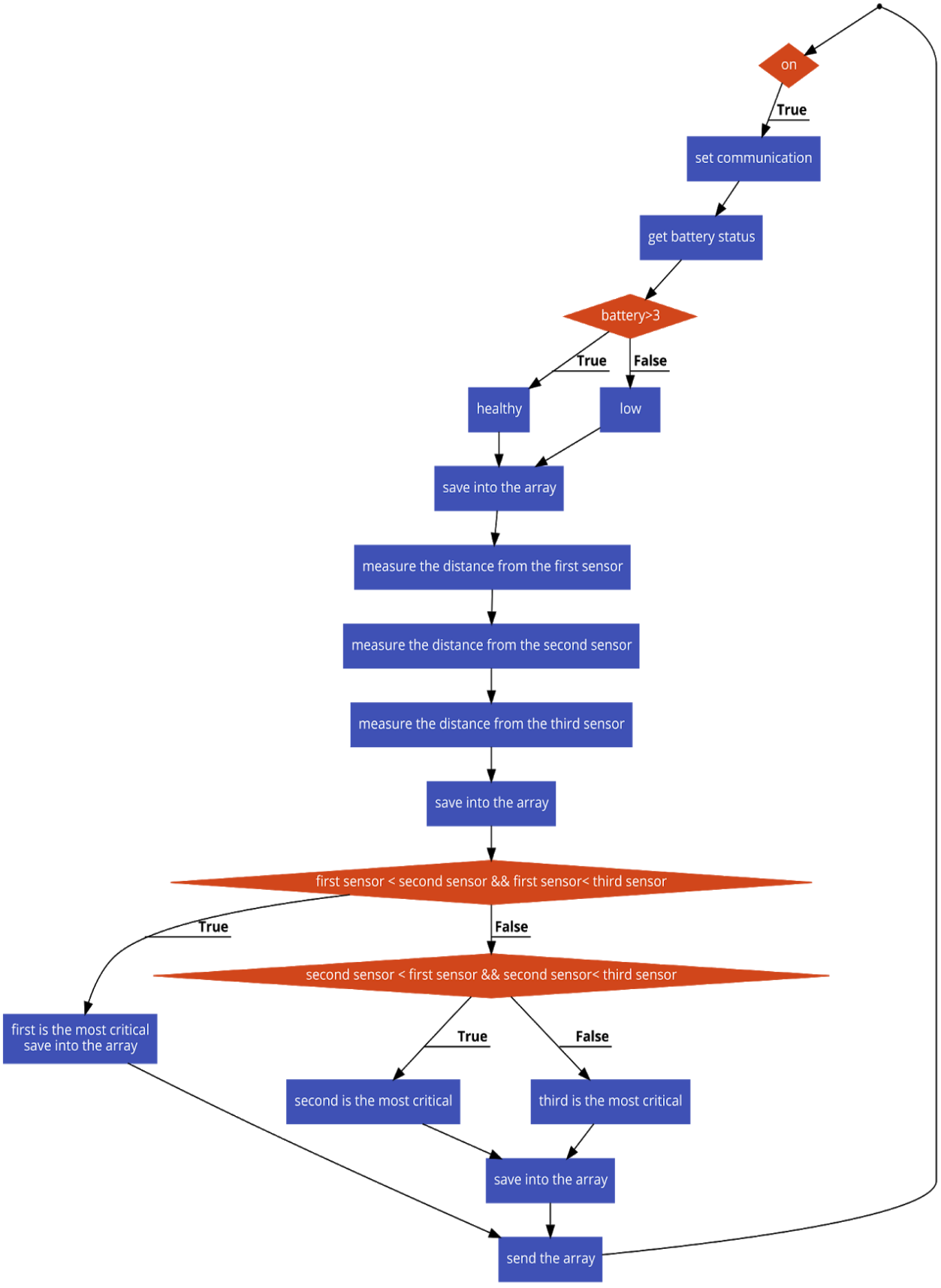


Fig1. Flow diagram for the slave unit.

5.3 Master Unit

The software will start by declaring NRF protocols for wireless communication. This stage includes pin locations for devices, declaring integers for the signal duration and id numbers of the master as well as all sensor units. Also, integers that will be needed during the main loop will be declared as well. Setup function will be similar to sensor units, such as matching pin locations for each device and calling required NRF functions. The main unit will also include 2 main loops named parking mode and driving mode. This selection will be made by the user. The two options will be provided on a physical switch that will be placed on the main unit. After selection of the status mode, the correct function will be executed.

5.3.1 Parking Mode

When parking mode is selected, the main unit will check if there is an incoming signal from sensor units. If there is a signal, master unit will get the array that includes critical distance and other distance informations. After tthis process, location information of the most critical sensor will be displayed on the corresponding 7 segment display. Also, all 3 distance information will be displayed on the lcd under their location information. Whole cycle of displaying 12 different sensor information will take approximately 0.5 seconds

5.3.2 Driving mode

Driving mode will be created to display battery status to the user. When parking mode is selected, master unit will look for signal. If there is a signal, battery information will be displayed on the LCD display. Then the system will check if the driving mode has been switched back to parking mode.

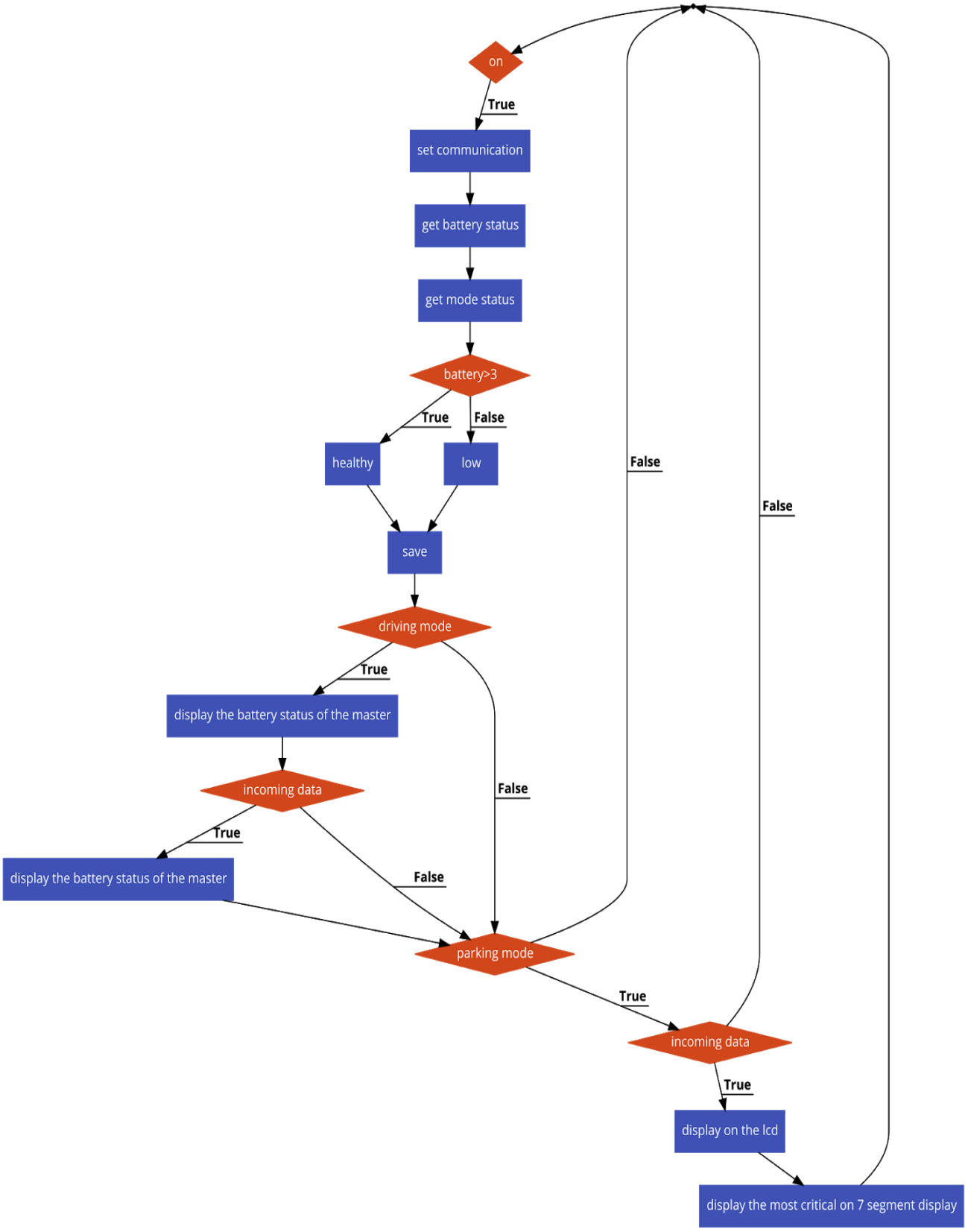


Fig2. Flow diagram for master unit

6. Project Build

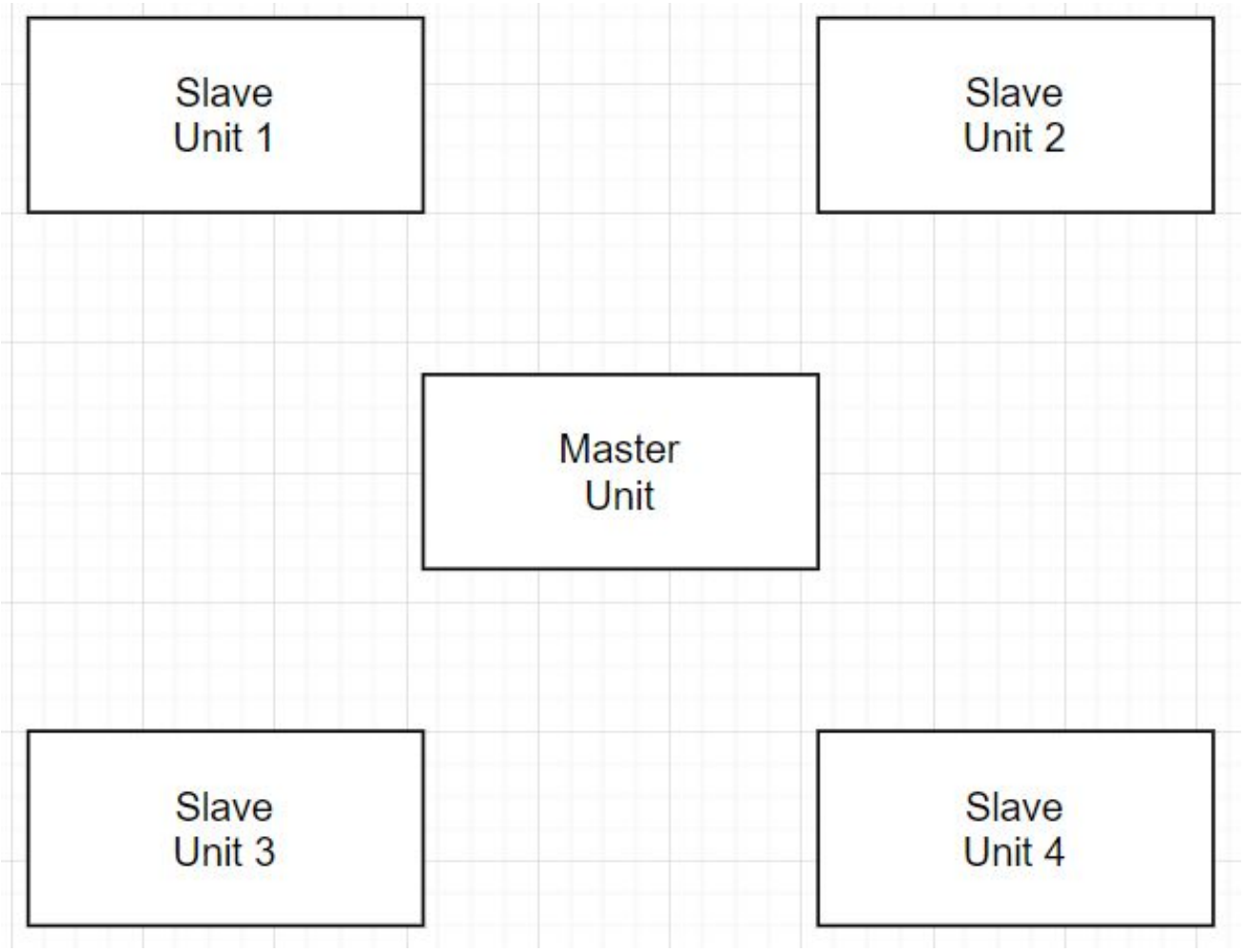


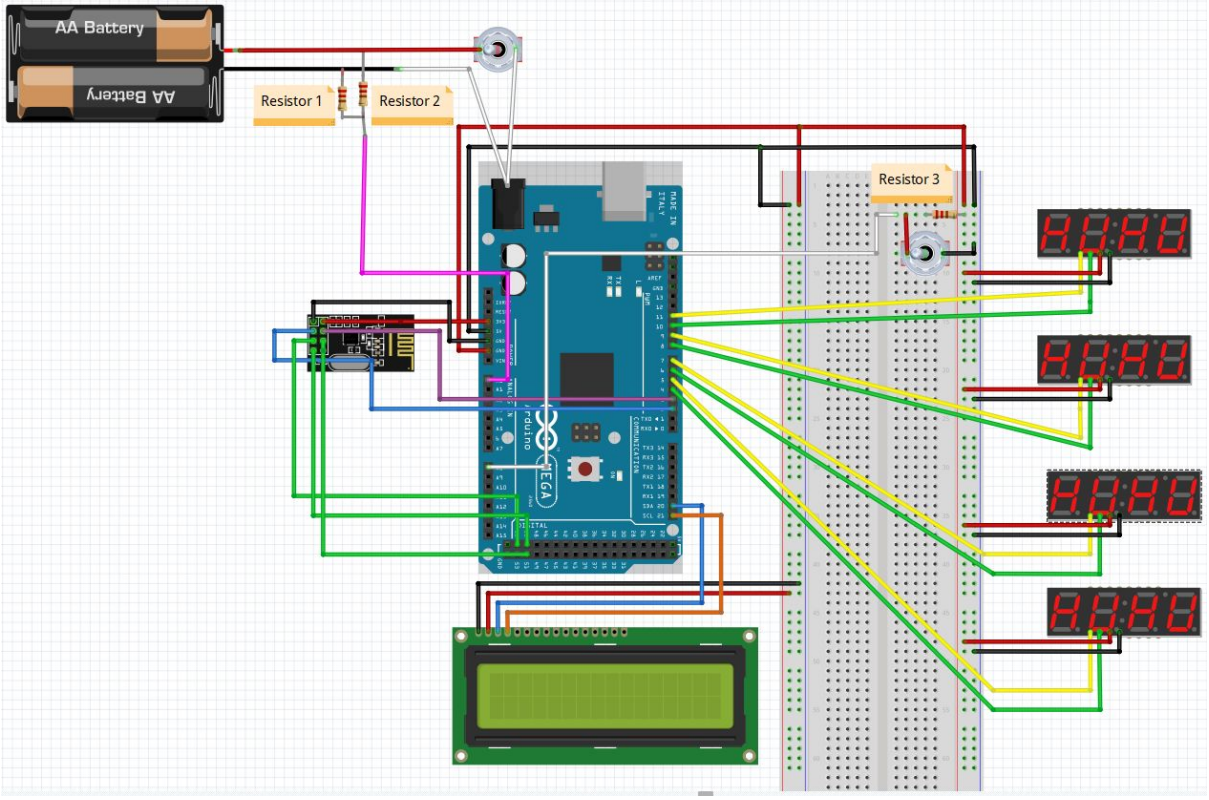
Fig 3. Simple block diagram of system

The Smart Parking System is made up of one Master unit linked wirelessly to four Slave units. The Master unit is intended to sit inside the user's automobile while each Slave unit is to be mounted to each corner of the vehicle. For transportation vehicles that may not need four corners monitored (motorcycle, bicycle, etc) having only two Slaves monitoring distances is a possible configuration. For easy Slave unit identification, unit 1 and unit NorthWest (NW) are used interchangeably, unit 2 with

unit NorthEast (NE), unit 3 with unit SouthWest (SW), and unit 4 with unit SouthEast (SE) respectively. Below are more detailed explanations of each subsystem, their wiring diagrams, and the codes for the Master unit and each Slave unit. Additionally, in the wiring schematics for each unit, red wires always represent lines with power running to them, while **black wires** are always ground wires. The rest of the connections are colour coded to the wiring diagram in their respective tables.

6.1 Master Unit

6.1.1 Wiring Diagram



6.1.2 NRF Chip

The NRF chip is powered by the Arduino Mega's 3.3V output, and is connected directly into the Mega so it can process and output the data it receives to the various displays. It's wiring schematic is as follows.

VCC	3.3 Volts
GND	GND
CSN	PWM D8

CE	PWM D7
SCK	D52
MOSI	D51
MISO	D50

6.1.3 LCD

Our LCD includes a built-in driver chip that enables us to operate the screen using only 4 pins**.** The back of the LCD also has a contrast adjustment knob attached that allows users to easily adjust the screen to their viewing preferences.

VCC	5 Volts
GND	GND
SDA	C20
SCL	C21

6.1.4 7-segment Display

Much like the LCD, each 7-segment display includes a built in digital to analog converter that allows operation using 4 pins. The brightness of each individual screen is adjustable, but it has to be done in the code, making it less accessible for users. Screen 1 refers to the topmost screen, and screen 4 the bottommost screen.

VCC	5 Volts
GND	GND
CLK	Screen 1: D11 2: D9 3: D7 4: D5
DIO	Screen 1: D10 2: D8 3: D6 4: D5

6.1.5 Mode Switch

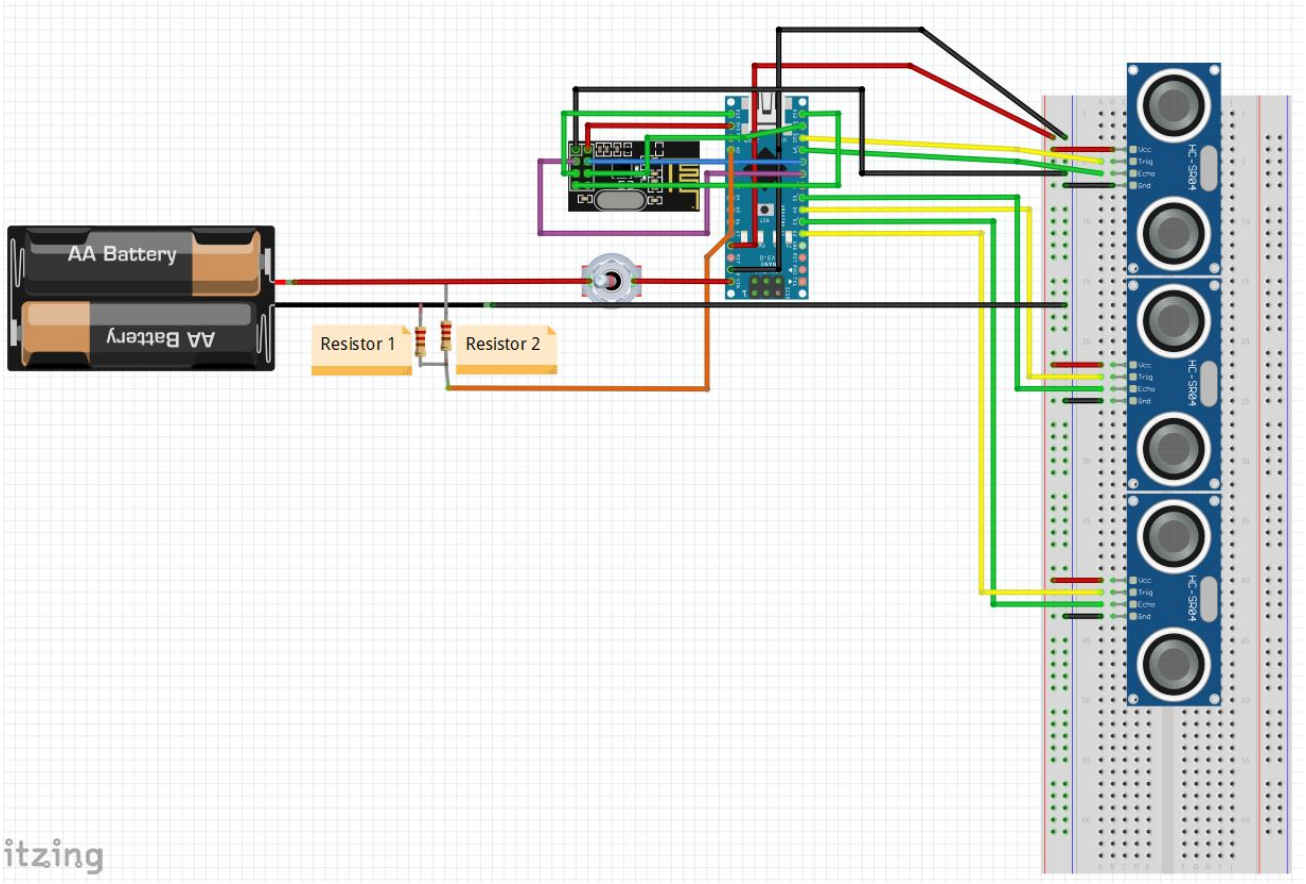
The mode switch uses a pull up resistor to send either 5V or 0V back into the Arduino. The details of how this pull-up system is wired together is as follows. First, 5V is fed into a 10 kΩ resistor. The other end of the resistor goes into pin A_8 of the Arduino Mega. Connected in parallel to this end is a wire feeding into one end of the switch. The other end of said switch is grounded. If the Arduino senses 5V being inputted into A_8, it will be in "driving" mode. 0V sees the Arduino going into "parking" mode.

6.1.6 Power System

The Arduino Mega is powered by a battery pack running 12V (8 AA batteries in series) into the battle connector. The Arduino then uses its internal voltage regulator to step down to 5V. The system is set up as follows. First, the wire off the positive terminal of the battery pack is connected to one end of the switch. The other end is connected to the positive side of the battle connector, while the wire off the negative terminal is connected to negative on the battle connector. In parallel on the positive terminal is a 1 MΩ resistor connected in series to an 860 kΩ resistor, which is then connected to the negative terminal wire of the battery. At the side where both resistors are connected to each other, an additional wire is connected and feeds into pin A_0. Because Arduinos can only have input voltage up to 5V, the resistors are there to act as a low current draw voltage divider. Full battery (12V) equals 5V going into pin A_0. When the Arduino detects 3V or less (7.2 or less volts at the battery pack) it will send a low battery warning to the user. The reason 7.2V was chosen is because Arduinos can run from unregulated voltages ranging between 6 to 20V. 7.2 is low enough that the user should know it's almost time to swap out the batteries, but gives the user enough time to still have a functioning system before the batteries die completely.

6.2 Slave Units (4 Identical units)

6.2.1 Wiring Diagram



6.2.2 NRF Chip

Similar to the NRF chip installed on the Mega (refer to 5.1.2) but with slightly changed wiring to suit the Arduino Nano. The wiring is as follows.

VCC	3.3 Volts
GND	GND
CSN	D8
CE	D7
SCK	D13
MOSI	D11
MISO	D12

6.2.3 Ultrasonic Sensor

Each sensor runs off of 5V provided by the Arduino Nano. The trig pin is set for 10 microseconds to high, which sends out 8 cycles of soundwaves, hit an obstacle, and bounce back to be collected by the Echo pin. The Echo pin will then output the time the soundwave travelled in microseconds, which is directly proportional to the distance travelled. In order to calculate this distance, the travel time needs to be multiplied by 0.034 cm/microsecond (the speed of sound) then divided by 2 (to account for the soundwave travelling forward and bouncing back, essentially doubling the distance we're looking for). Sensor 1 refers to the topmost sensor.

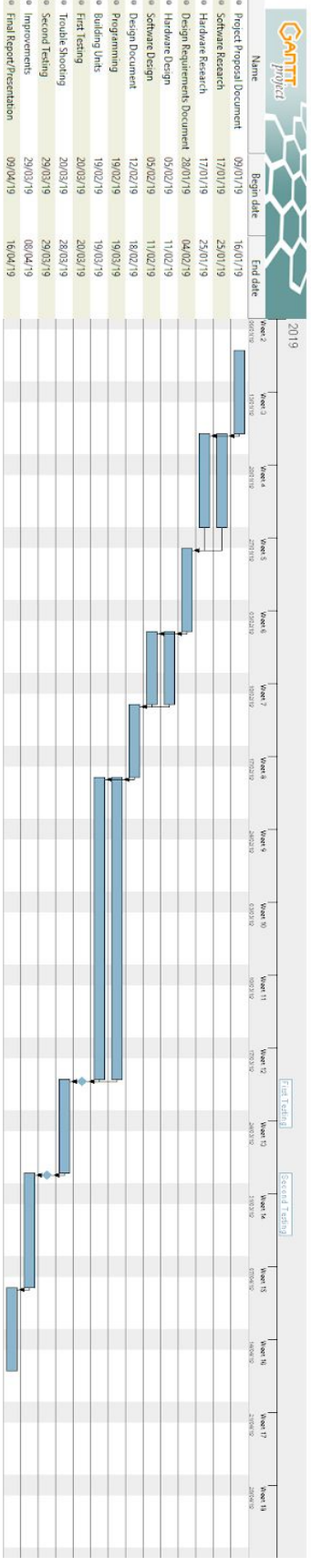
VCC	5 Volts
GND	GND
TRIG	Sensor 1: 10 2: 4 3: 2
ECHO	Sensor 1: 9 2: 5 3: 3

6.2.4 Power System

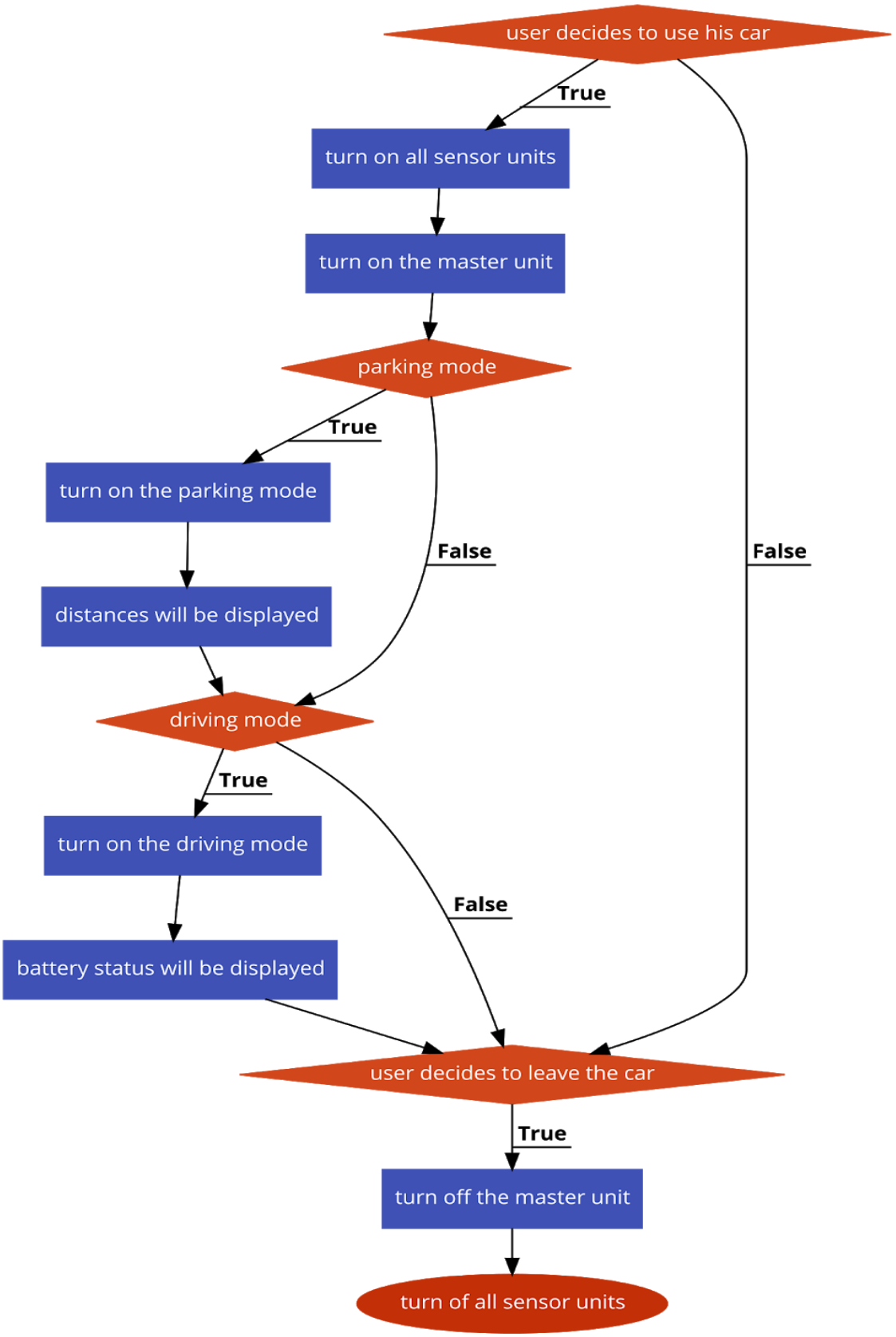
Identical to the power system used for the Mega (refer to **5.1.6**) except the wire off the positive terminal goes into V_{In} and the negative wire goes to ground because the Arduino Nano does not have a battle connector and the USB port does not take inputs higher than 5V.

7. Appendix

7.1 Project Gantt Chart



7.2 Flowchart for product operation



7.3 General functionality testing (post functionality addition)