University of Illinois at
Urbana-Champaign

Door Access Tracker

Patrick Connelly (prc2), Ben Wasicki (wasicki2),

and John Scholl (johnts2)

ECE 445 Design Document Check
Team #41
TA: Chi Zhang
February 25, 2020

Table of Contents

1. Introduction

1.1 Objective:

Many areas of day-to-day life involve the opening and closing of a door. We believe that better information on the state of a door can improve one's quality of life. For example, one could monitor a door as a security measure, such as a front door, a liquor closet, or a medicine cabinet. Alternatively, some doors may also have a tendency of getting stuck open. In this case, knowing that the door was not closed properly may be good information to have. In addition, knowing when the mailbox has been accessed could be time saving, especially for someone who has mobility problems because they would not need to check the mailbox unnecessarily.

Our proposed solution is the Door Access Tracker. This tracker would consist of a sensor to detect the state of a door, a microcontroller, a wifi card, a cloud server, and an android app. This would be a portable device that would be adhered to a door. The primary functionality involves the user getting an update on their phone via an application when the state of the door is changed. In order to make this product more versatile, we would allow for different configurations on when to send notifications. For example, a consumer may want to know the instant a medicine cabinet or liquor cabinet is opened; however, they may only care about a door's state if it were to be left open for a specific amount of time before being closed.

1.2 Background:

There are many situations in which the monitoring of a door or cabinet may be useful. From a security perspective, knowing when an area is accessed could be extremely useful information, especially for knowing when something has been tampered with. From a convenience perspective, putting this device on something such as a mailbox would let someone know when they should go to check for mail. Finally, from an energy-savings perspective, this product could let a person know when a door or window is left ajar, leading to heat loss in the winter and air conditioning loss in the summer. A key issue that needs to be addressed as well is that a user may want a different notification or set of notifications for different situations. For example, they may want to know immediately when a door state is changed, they may only want to know when a door is opened, or they may only want to know if a door is left open for a certain amount of time.

There are some products on the market that attempt to achieve the same functionality as our project. Our design would not only be cheaper than available products, but it would also have additional functionality. The application we propose would be more configurable than currently available alternatives; giving the user the option to select when, why, and how they are notified. Our solution is also stand-alone and does not require any subscriptions or any hub device.

1.3 Visual Aid:

1.4 High Level Requirements:

The following are the three most important qualities our project must exhibit in order to be successful:

- The door sensor sends a different signal based on the state of the door (i.e., open or closed).
- The system controller takes the signal from the door status sensor and updates the back-end server with the door's state.
- The back-end server can send the Android application an update based on the information it receives from the system controller and the current configuration set.

2. Design

2.1 Block Diagram

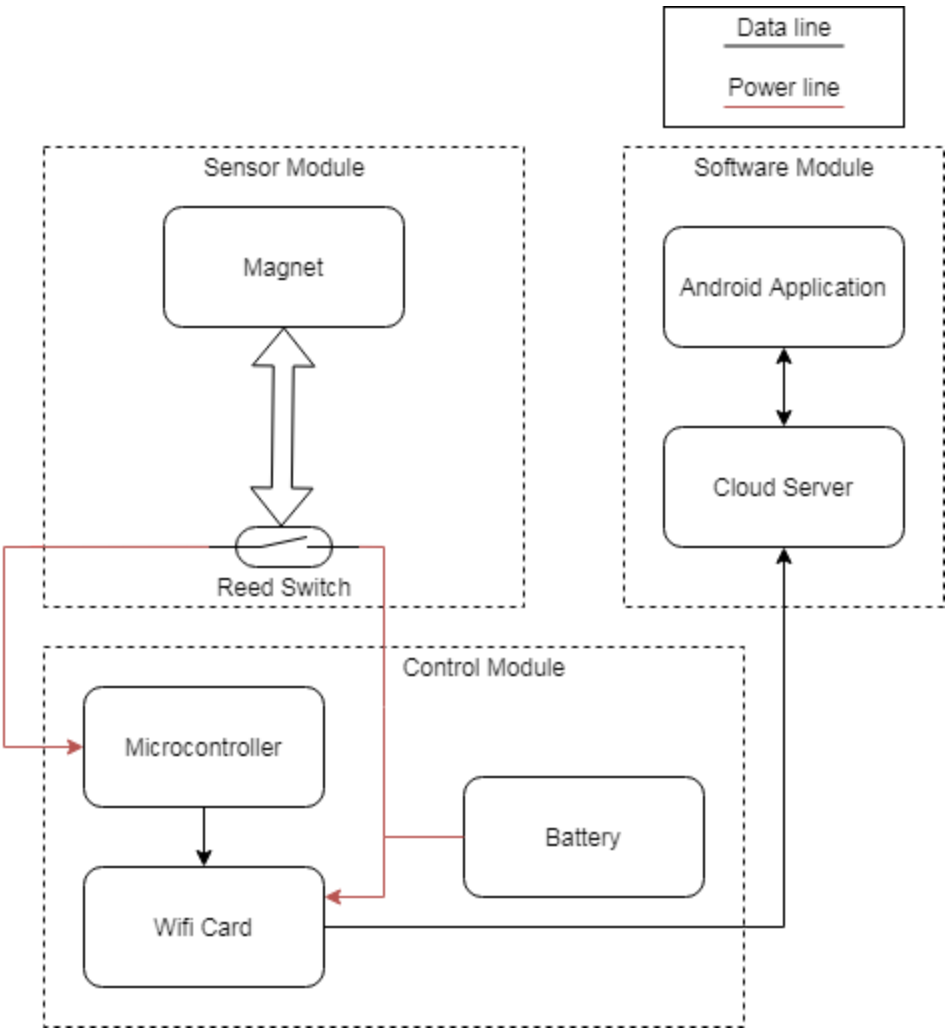
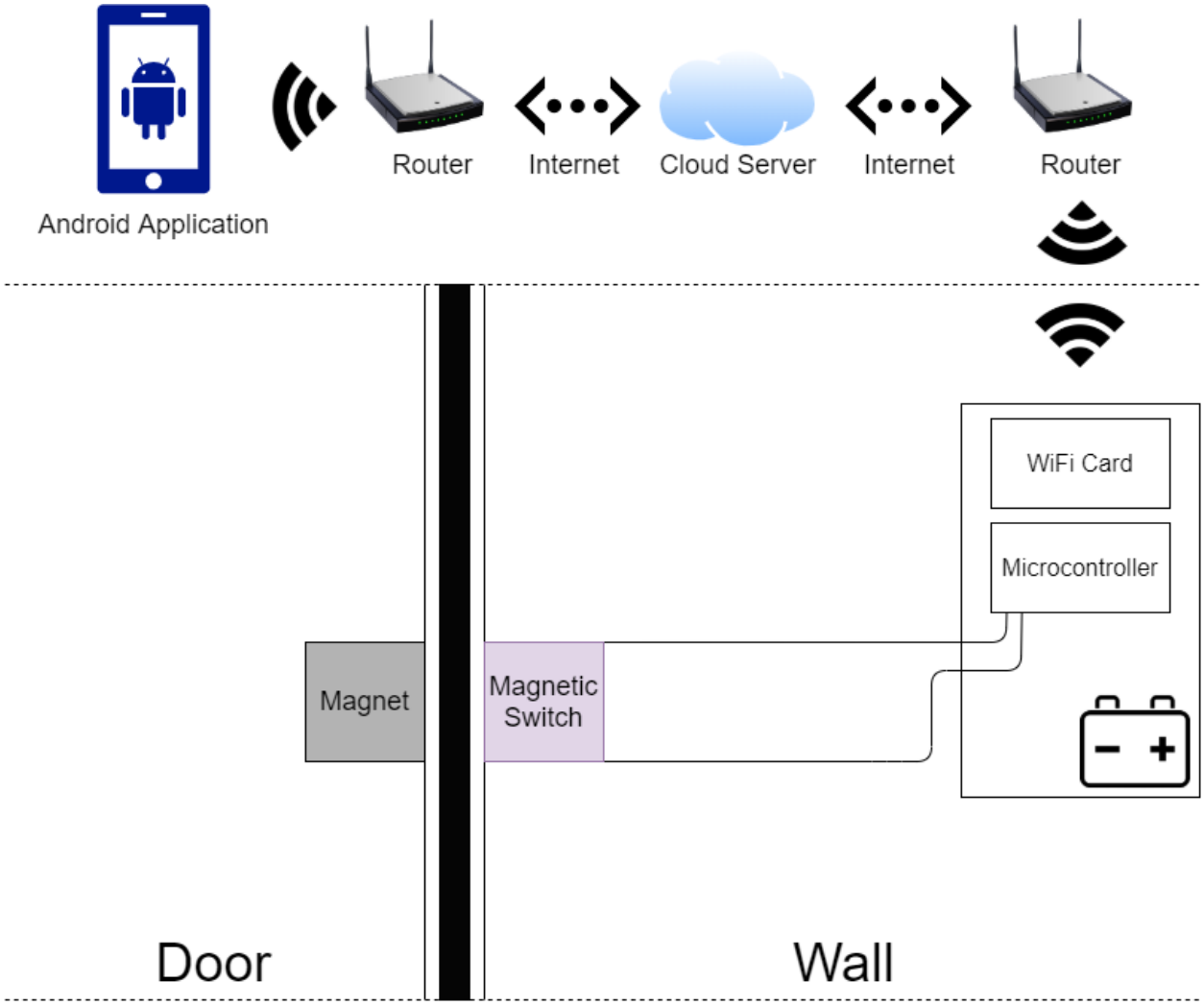


Fig. 2 Block Diagram

2.2 Physical Design

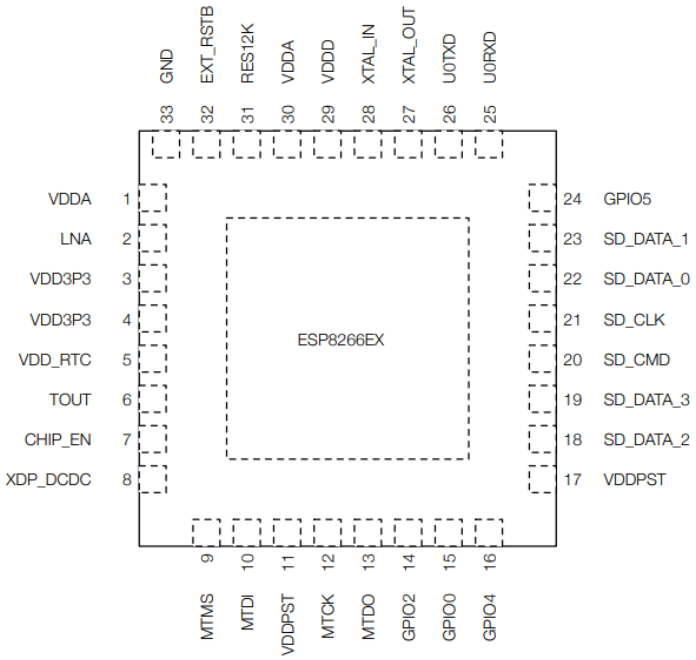


2.3 Subsystems

2.3.1 Magnetic Sensor

2.3.2 WiFi Chip and Microcontroller

The WiFi chip we intend to use for our Control Module will be a variant of the ESP8266. This will simplify our design, as the onboard microcontroller has extra resources to process our sensor input, which removes the need for a separate microcontroller.


GND 33
EXT_RSTB 32
RES12K 31
VDDA 30
VDDD 29
XTAL_IN 28
XTAL_OUT 27
U0TXD 26
U0RXD 25
VDDA 1
LNA 2
VDD3P3 3
VDD3P3 4
VDD_RTC 5
TOUT 6
CHIP_EN 7
XDP_DCDC 8
24 GPIO5
23 SD_DATA_1
22 SD_DATA_0
21 SD_CLK
20 SD_CMD
19 SD_DATA_3
18 SD_DATA_2
17 VDDPST
ESP8266EX
9 MTMS
10 MTDI
11 VDDPST
12 MTCK
13 MTDO
14 GPIO2
15 GPIO0
16 GPIO4

2.3.3 Battery

2.3.4 Cloud Server

- Description:
  The cloud server contributes to the second and third high level requirement. It contains most of the computations and memory that will be needed. As such, it will take requests from the user via the android application and from the control module, be able to set configurations, store history, and send updates to the android application based on the set configurations. This will be run as a pod in a kubernetes cluster with a service and ingress to connect it to the outside. The benefits of running this on a cluster are persistent volumes, easy scalability, and self-healing.
- Requirements and Verifications Table

Requirements	Verification
1. Server is able to receive state change from at least 1 control module 2. Server will be able to complete registration of a specific application with a specific serial number. 3. Server will be able to send updates to at least 1 application after a state change is received from a control-module. 4. Server will be able to use configurations specified by an application to deliver updates to said application as set by configurations. 5. Server will keep a history of door state changes with corresponding timestamps based on the number specified by the application and will send state changes to an application by request.	1. A. Do ***Step 1***. B. Do ***Step 2***. C. Start the test pod with the *--test1=true* argument specified in the pod yaml file. D. Check the logs of the server Pod to see if they have the following output: *“<Timestamp> : test1-serial-number : open”* *“<Timestamp> : test1-serial-number : closed”* 2. A. Do ***Step 1***. B. Do ***Step 2***. C. Start the test pod with the *--test2=true* argument specified in the pod yaml file. D. Check the logs of the server Pod to see if they have the following output: *“<Timestamp> : test2-serial-number : <IP>”* 3. A. Do ***Step 1***. B. Do ***Step 2***. C. Start the test pod with the *--test3=true* argument specified in the pod yaml file. D. Check the logs of the test Pod to see if they have the following output: *“<Timestamp> : test3-serial-number : open”* *“<Timestamp> : test3-serial-number : closed”*

	4. A. Do ***Step 1***. B. Do ***Step 2***. C. Start the test pod with the *--test4=true* argument specified in the pod yaml file. D. Wait 10 seconds then check the logs of the test Pod to see if they have the following output: *"Timestamp1 : test4-serial-number : open"* *"Timestamp2 : test4-serial-number : closed"* E. Ensure *Timestamp2* is approximately 5 seconds after *Timestamp1*. 5. A. Do ***Step 1***. B. Do ***Step 2***. C. Start the test pod with the *--test5=true* argument specified in the pod yaml file. D. Check the logs of the server Pod to see if they have the following output: *"---HISTORY BEGIN---"* *"<Timestamp> : test5-serial-number : open"* *"<Timestamp> : test5-serial-number : closed"* *"<Timestamp> : test5-serial-number : open"* *"<Timestamp> : test5-serial-number : closed"* *"---HISTORY END---"*

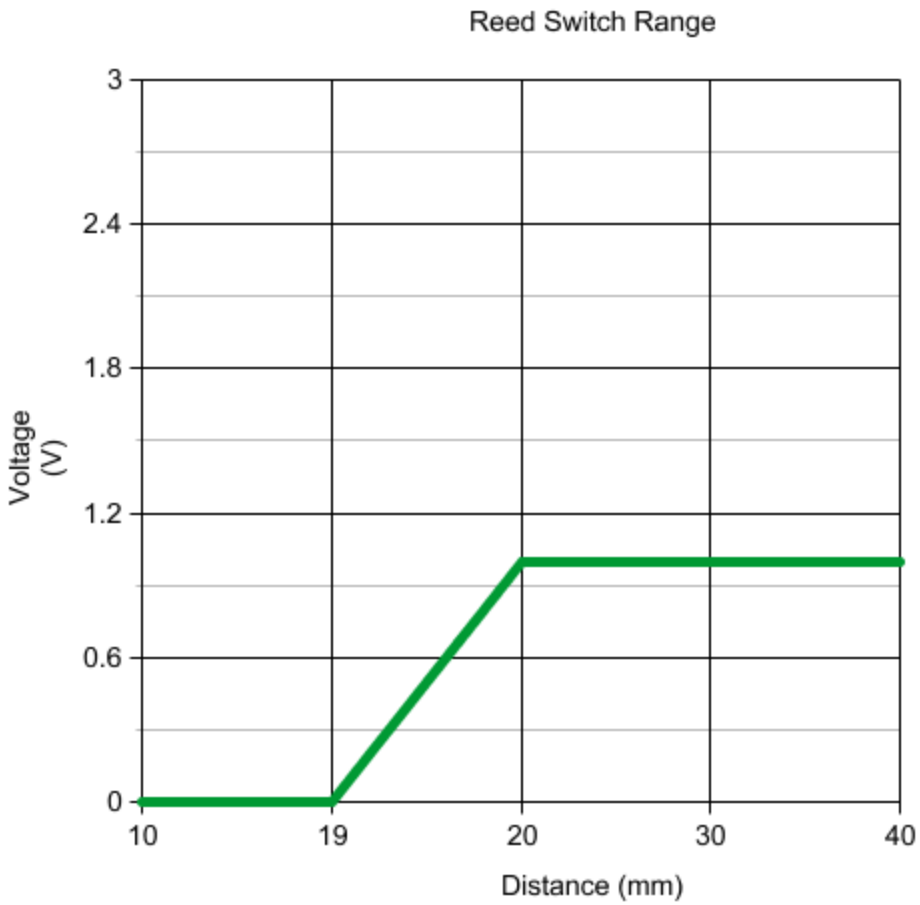
*The Following are steps that must be repeated for each of the above tests. They are presented here to avoid repetition:

- ***Step 1***: Ensure that the Kubernetes cluster is running with both the server Pod's Ingress and Service, and the test program Pod's Ingress and Service configured correctly.
- ***Step 2***: Start the Pod using the *verbose=true* argument specified in the pod yaml file.

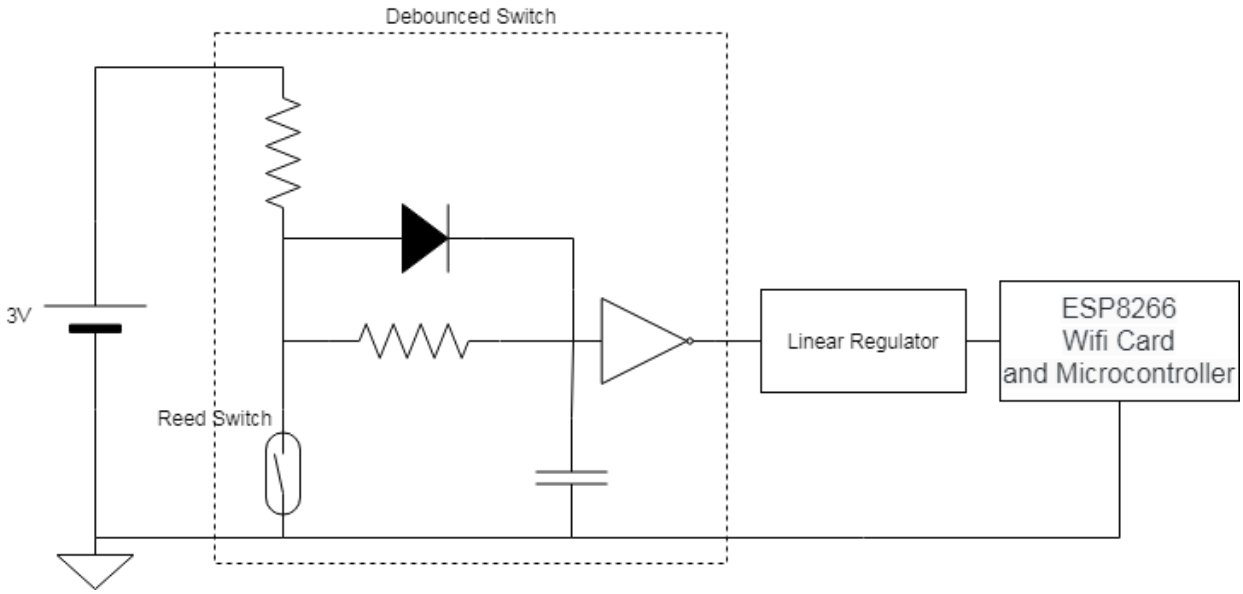
2.3.5 Android Application

2.4 Tolerance Analysis

2.4.1 Plots



2.4.2 Circuit Schematics



2.4.3 Calculations

- When the door is closed, the WiFi chip is the only power draw, consuming about 20μA in its “deep-sleep” mode. An ENERGIZER CR2032 3V battery has a typical capacity of 235mAh. Thus, with no closing or opening of the door, our device should last approximately:

$$\frac{235\ mAh}{0.02\ mAh} = 11750\ h \approx 490\ days \approx 1.3\ yrs$$

3. Cost and Schedule

3.1 Cost Analysis

3.2 Schedule

4. Discussion of Ethics and Safety

As the developers of this project, we believe it is important that we produce a safe, reliable, and efficient product to our user. We commit ourselves to holding a high degree of professional conduct in accordance with both the IEEE and ACM Code of Ethics. We will avoid ethical breaches by following all device specifications, working in our respective areas of competence, and clearly stating proper operating procedure (ACM 2.6). At the same time, we acknowledge that our device could be misused; therefore, we will take all necessary precautions to prevent any harmful modes of operation.

In accordance with the ACM Code of Ethics, this project will pose no risk to the user or community under standard operations. Given that our project monitors when a door is opened and closed, it could pose a safety risk to the user if the data is compromised. We will ensure that all wireless protocols are followed, and communications will be secure. The data gathered by our sensor will be the sole property of the intended user of the device (ACM 2.9). All software will follow accepted community standards.

Following the IEEE Code of Ethics, we have decided it is important to make our design as energy efficient as possible to minimize waste. As designers, it is our responsibility to limit the environmental impact of our device. We have implemented a circuit break when the door is closed to ensure power is only consumed when necessary. This will limit the amount of waste associated with battery replacements.

In addition, we will ensure there is no exposed wiring or electrical components in our design to minimize the risk of electrical shock. Similarly we will ensure all components are operating within their respective operating regions to reduce the risk of a short or fire hazard.

5. Citations

[1] “IEEE Code of Ethics,” *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 12-Feb-2020].

[2] “ACM Code of Ethics and Professional Conduct,” *Association for Computing Machinery*. [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed: 12-Feb-2020].