Client-Server Radar Security Circuit

Jason Rinhart, Trinh Huynh, Saurav Basnet
Department of Electrical and Computer Engineering
Wentworth Institute of Technology
rinehartj@wit.edu | huynht18@wit.edu | basnets@wit.edu

***Abstract*—This paper aims to aid in solving inherent problems with modern intrusion security systems. With the entry of microcontrollers in every angle of our lives, it may be surprising that home security is a sector that almost always has trade-offs when it comes to features and cost. The system established in this paper is a basic but expandable attempt at a cost-effective home security system, also functional in commercial and industrial environments. The designs prototyped in this paper demonstrate the implementation of a radar sensor to detect high accuracy movement within 36 feet. Then the notification of movement can be sent via phone notification, alarm buzzer, and screen display text. To mitigate false alarms in an outdoor setting, the design recognizes short and long movements in order to take appropriate actions. The control panel and remote sensor use Raspberry Pi and Particle Photon microcomputers respectively, which work together over Wi-Fi. With wireless ability, several remote sensors may be placed within range to enable a full home monitor system for an attractive cost.**

***Keywords*—home automation, Particle Photon, doppler effect, HFS-DC06, radar security.**

I. Introduction

Many people consider today to be the golden age for home security, yet all reasonably priced systems have various shortcomings. People are driven to purchase a system for its reliability, affordable cost, and ease of installation. Those without much time to spare may be compelled to purchase smart security devices from online retailers, often for a low price. While it is expected they perform to specification, many of them have a common flaw: if the remote server it relies on becomes offline, either for downtime or permanently, the device is effectively a brick. A further analysis into many Internet of Things (IoT) devices by Maya from CSU San Marcos revealed the intricate tie between commercially available IoT devices and their company servers they require for operation [1]. An ideal device would still have limited operability under conditions of no internet, a rare occurrence among low price IoT solutions.

As far as motion cameras, a sector within home security, many inexpensive options only function correctly under controlled environmental parameters. They may work to specification indoors; however, for outdoor use, their abilities to distinguish people from the environment quickly diminishes. This is a fundamental problem this paper aimed to solve. In other research, it is difficult to achieve complete accuracy for outdoor detections for a reasonable price. A proposed solution by Akter, Sima, Ullah, and Hossain [2] called for a Raspberry Pi and PIR sensor to aid in an automated doorbell. While useful for some locations, the system did not appear to be easily expandable, and PIR sensors were not as capable in daylight [3].

Another solution for security camera reliability is to run artificial intelligence (AI) on a video stream in real-time to scan for people, vehicles, animals, and other objects [4]. Because of the high computing power requirement this entails, it is only feasible for the average consumer if these calculations are run on a remote server by the camera manufacturer [4]. Products such as Reolink and Wyze cameras, which are designed for residential use, detect humans in this manner. However, this is a monthly, paid service and some have mixed reviews. Additionally, security cameras use high internet bandwidth, requiring a hard cable connection. Routing the cables can be frustrating and time-consuming, deterring most potential customers. The ideal intrusion detection circuit should still operate in times of internet (WAN) dropout, be easy to install, while maintaining a reasonable price to manufacture.

Customer may be driven to have a full security system installed in their home or business. This scenario is not ideal for several reasons, mainly cost, but also difficulty in installation. Generally, it costs over $1,000 for installation and up to $100 per month for automatically dialing to public safety. These systems have errors besides cost, possibly including vulnerabilities in the communication between components in the system. Frequencies can be jammed, many times without detection [5]. There are not many cost-effective devices which report when an endpoint drops offline. With everyone having a computer in their pocket, one would expect these systems to have smartphone notification compatibility.

This paper aims to lower the monetary cost to under $70 for an operational circuit, while addressing many of the above problems with both subscription services and standalone IoT devices. This system was expected to maintain affordability, reliability, modularity, and ease of interface.

II. Application

This design was modeled and prototyped spanning in two months. Multiple iterations of each phase were required for the functional ending prototype. The design consists of a single base station, or control panel, and remote sensors modules. Below is an outline of that process.

All renders were made using Fusion 360 (Autodesk, Inc) with the aid from 3D-models by “Robottronic”, “Scrtcwlvl”, Sobat Kupu, M A Motin, Enrique Alcala Guerrero, Pierre Gleizes, and Richard Barber. Their work was retrieved from GrabCad Library (grabcad.com).

A. Design Overview

The physical appearance was to be kept minimal while maintaining key functionality, including user input and output. Below, **Fig. 1** is a photograph of the front of the Control Panel product..

Fig. 1. Control Panel prototype, October 2021

From the beginning, this project was designed to be replicable. The 3D-printed shell has multiple holes, cutouts, and curves which fit the components while appearing elegant. Two buttons allow basic functions to be executed by the user, such as muting, disabling notifications, setting the threshold time for alarms, and other functions programmable in the Python code which is discussed in *Software Design*.

The LCD display acts as a status indication and settings menu. The backlight turns off automatically after several seconds of inactivity but illuminates once a sensor is active, or button is pressed. When movement is detected, the screen begins a timer and displays the sensor region, for example “Front Yard”. Lastly, after exceeding a threshold duration of movement, the buzzer sounds and an event is triggered to make calls to the internet, such as a phone notification. A variety of services could be set up for tethering to a phone, although ultimately this is up to the user.

The two buttons on the front panel control key settings. The left button, when pressed quickly, toggles the buzzer on/off. Holding the left button toggles phone notifications. The LED strip in the center flashes when the system is in alarm state.

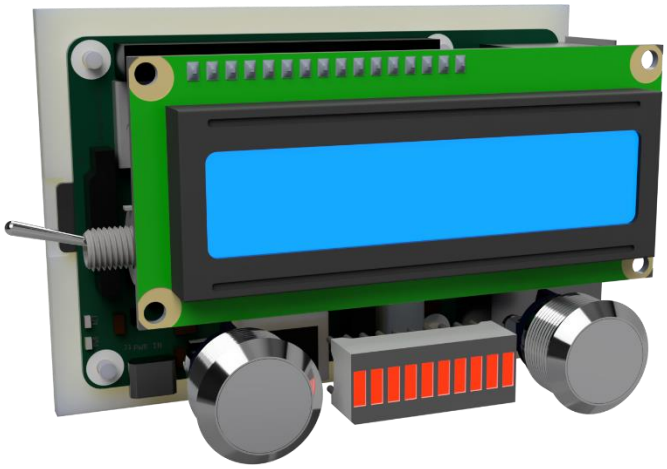

Fig. 2. Exact position of the Control Panel components with the front 3D printed cover hidden. Rendered using Fusion 360

The remote sensor contains a 5.8GHz HF-DC06 radar sensor, selected for its simple and reliable design. It utilizes the Doppler Effect to listen for varying frequency shifts to understand if the environment is changing. The sensor has three pins, one for voltage, ground, and a basic on/off signal pin. There is no interface besides an LED on this little box—it is meant to be mounted indoors or outdoors. It only needed a linear power supply and WiFi connection to operate.

B. Hardware Design

The Control Panel shown in **Fig.2** is built around a Raspberry Pi 3B+. Input/Output components include a 16x2 LCD panel, two 2mm push buttons, 10-segment LED cluster (generic), Piezo buzzer. The price breakdown for the project can be found in **Table 1** below. All components fit properly in the plastic enclosures. The switch, used as a master on/off, was positioned to the left side to keep clutter off the front panel. The Control Panel is to be mounted on a wall in an accessible location.

Fig. 3. Radar Sensor rendered using Autodesk Fusion 360

As for the radar sensor shown in **Fig.3** and **Fig.4**, four unique components (listed below) had been selected for their small size and function, along with a power supply.

1. Particle Photon
2. HFS-DC06 5.8GHz Radar Sensor
3. 2 pcs LM2596 DC-DC Buck Converters
4. 12V power supply

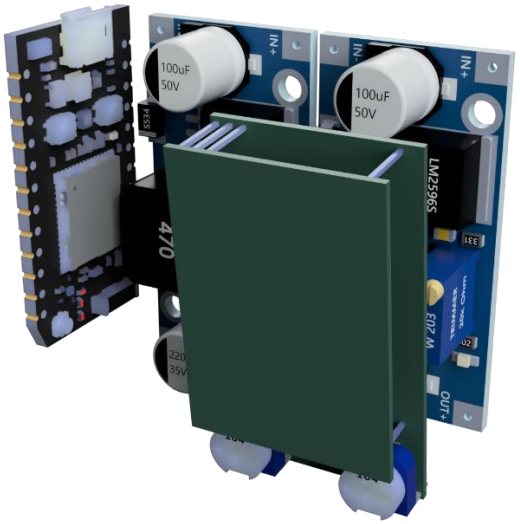

Fig. 4. Exact position of the Radar Sensor components with the entire 3D-printed shell hidden. Rendered using Fusion 360

Components in this design were selected for their cost (shown in **Table 1**), reliability, and ease of interface with each other. They were held to the shell (invisible) with plastic pegs and super glue. This was both to save cost and allow easy installation but will be improved in the future for serviceability. Overall, these designs are first-generation but fully functional in their purpose. Glue allowed time to be saved in the development process, with future prototypes to use metal screws.

TABLE I: COST OF COMPONENTS

Part No.	Part Name	Price
1	Raspberry Pi B 3+	$35.00
2	Particle Photon	$20.00
3	Radar Sensor HFS-DC06 5.8GHz	$1.20
4	LCD 16x2 i2C	$.150
5	5V and 12V Power supplies	$6.00
6	2pcs LM2596 DC-DC Buck Converters	$1.20
7	Resistor(4), button(2), 3-D box (2), LED(2), switch(1), buzzer(1), transistor(1), wires	$3.00
Total	Control Box	$41.10
	Sensor Box	$26.80
	Complete Package	**$67.90**

C. Software/Network Design

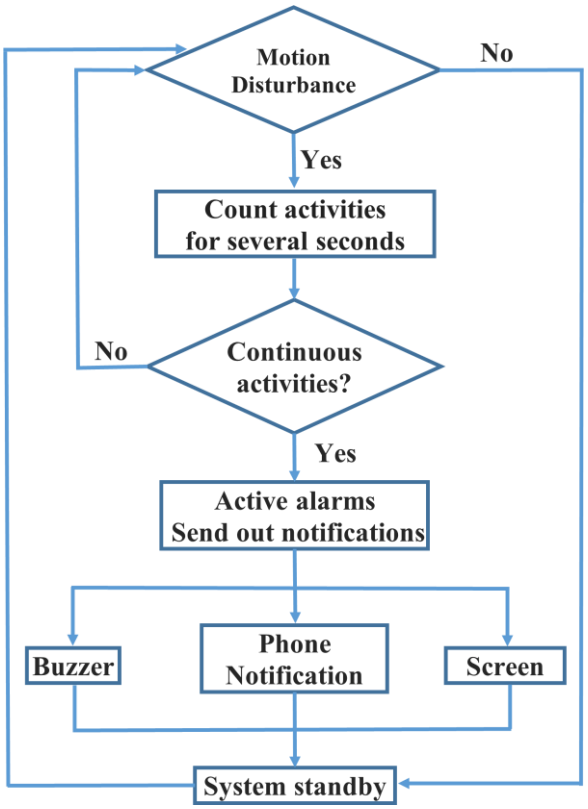

Fig. 5. Push notification algorithm

The Control Panel incorporated two fundamental functions of programming to operate: Transmission Control Protocol (TCP) and Multi-Threaded Python. TCP is a basic form of communication between IoT devices. A TCP Server was hosted from the Raspberry Pi, which accepted incoming packets from within the WiFi network. A binary '1' or '0' was sent to the server to describe motion, allowing the Control Panel to utilize several libraries to ultimately decide when to send an alarm.

Multi-threaded Python allowed all the functions of the control panel to operate on-time. Threading is the process of creating "living" objects that each execute code simultaneously. Threads work great for low-CPU power tasks, such as infinite loops with a "sleep" call. Threads allowed the program in this project to run many tasks at once, which was especially helpful in keeping time, making sound using the buzzer, and waiting for incoming TCP packets.

The Python 3 infrastructure, along with the threading, socketserver, time, RPi.GPIO, and Adafruit_CharLCD libraries allowed for the length of the Control Panel code to be only 250 lines. The programming work was mostly completed via pre-existing libraries, such as the interface between the Pi and the LCD display. Simply using `lcd.message("Hello")` would set the LCD display to this string of text. Many other functions were similar in that they condensed tens or hundreds of lines of code into a single line. If it were not for the code libraries accessible for free, the time of creation would have been much greater.

D. Electrical Circuit Design

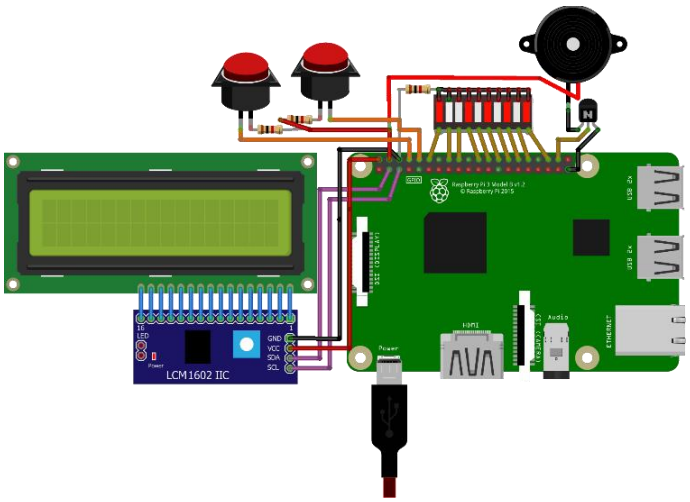

Fig. 6. Schematic of the Control Panel made using Fritzing

Fig. 7. The prototype Control Panel's inner wiring

The circuitry of the control panel consisted of the LCD, an i2c "backpack", two pushbuttons, a piezo buzzer, and the Raspberry Pi itself. All components shared a common ground, for the Pi to process incoming and outgoing signals. A piezo buzzer was selected for its simple integration with a single NPN transistor: It can either be on by powering the transistor base with 5 volts, or off otherwise. Unlike a classic speaker, no signal generator was needed. Components were soldered directly together; therefore, to disassemble the circuit means unsoldering each component. This ensured no wires will come loose when the unit undergoes movement.

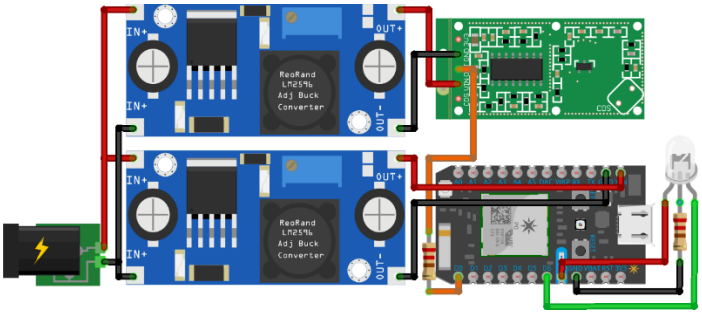

Fig. 8. Visual schematic of the Sensor Circuit made using Fritzing.

The sensor circuits were relatively basic and can be duplicated with ease for a multi-sensor setup. For this setup, two DC step-down modules were required to isolate voltage output to the Particle Photon microcontroller and the HFS-DC06 radar sensor. The sensor requires an extremely stable voltage; even deviations as little as ±0.05V can cause the sensor to report false motion. The entire circuit for the sensor module was powered using a linear power supply, in this example, 12 volts. Any power supply above 6 volts will work if the DC step-down modules are tuned correctly using their on-board potentiometers.

D. *Fabrication and Assembly*

Each component of the circuits required precise measurements to properly dimension and prototype the 3D model. After the measurements were taken, the components models were gathered online and assembled in Autodesk Fusion 360.

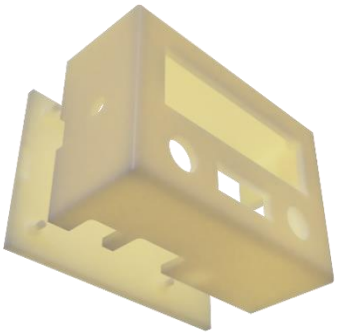

Fig. 9. Control Panel Shell

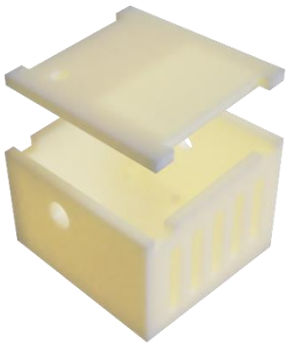

Fig. 10. Sensor Module Shell

After components were secured in the shell, each component had wires soldered to the appropriate pads. Not all pads were used on the microcontrollers, so some remain unused. At this phase, some basic code was developed, so next step was to test and improve it.

E. *Testing Procedures*

The system initially had several problems—many times with the TCP communication between the two. Constant trial-and-error rewarded a working Python script which appears to be stable for the short term, but longer testing intervals such as a whole month will be needed to establish the complete reliability of this circuit. Many of the various bugs in the code were due to the use of threading, where the timing of code execution is critical. Some code was being executed too early or too late, causing the functions to lose track of events. After this phase of testing, the final circuit worked extremely well in the short term. Longer tests will further be conducted.

III. Results

The completed, first-generation circuit composure appeared to perform correctly both indoors and outdoors. At first, the sensor was almost *too accurate* due to testing in a busy campus. After an adjustment to the sensitivity, the sensor detected movement and performed its functions, that is, changing the screen, beeping, and changing the LED color on time. While indoor usage is often a set-and-forget installation, further testing of outdoor usage proved that the system would need to be tuned for accuracy by adjusting the variable delay before alarms. Rain and wind are main setbacks in outdoor operation. Heavy rain deflected UHF electromagnetic waves, so the placement of an outdoor sensor should have two or more feet of distance from rainfall. The performs nominally for the goals initially set for both indoor and outdoor use. In the testing sessions, false alarms were not detected, and the sensor only read movement when a human or car was moving.

IV. Conclusions and Future Development

The prototype developed over the two-month period shows promise of further potential in that more sensors can be added, and the code can be refined. Even more abilities may be possible, such as integration with web cameras, or even other sensors such as temperature, as OTILIA et.al. added to their LCD display [6]. Furthermore, other variations of radar sensors can be tested for maximum range and reliability, such as B. Beszédes's research with three different radar sensors [7].

The envisioned deployment in a residential house would contain several sensors and cameras in order to notify the homeowner as soon as a positive detection is established. It is that positive detection that guided the research and knowledge pursued for this project: the reliability of positive detections was paramount in the project working as intended. The final result showed high potential in future improvement and deployment.

Acknowledgment

Staff from the WIT Technology Sandbox provided excellent 3D prints for the product. Adafruit Technologies has excellent documentation which were used to perform many of the tasks involved. Autodesk Fusion 360 gave a free one-year education licenses allowing this project to be visualized. Lastly, we acknowledge the many brilliant minds who developed the electronics used as a backbone for this project.

References

[1] M. Maya, "Internet of Things: A Deeper Dive in Your Privacy and Information," 2020, in press.

[2] S. Akter, R. A. Sima, M. S. Ullah, and S. A. Hossain, "Smart Security Surveillance using IoT," 2018, pp. 659–663, doi: 10.1109/ICRITO.2018.8748703, in press.

[3] Y. Bai, C. Cheng and Z. Xie, "Use of ultrasonic signal coding and PIR sensors to enhance the sensing reliability of an embedded surveillance system," 2013 IEEE International Systems Conference (SysCon), 2013, in press.

[4] A. A. Ahmed and M. Echi, "Hawk-Eye: An AI-Powered Threat Detector for Intelligent Surveillance Cameras," in IEEE Access, in press.

[5] Jover, R.P., Lackey, J. & Raghavan, A. "Enhancing the security of LTE networks against jamming attacks". EURASIP J. on Info. Security 2014, 7 (2014), in press.

[6] O.-S. Prelipceanu, M. M. Cazacu, I. A. Rosu, and M. Prelipceanu, "Wireless System For Monitoring And Control Of Environmental Parameters," MANSiD, 2018, in press.

[7] B. Beszédes, “Reliable Presence and Intrusion Detection with Collaborative Sensor Modules in Electronic Property Protection Systems,” 2019, in press.