Switch Wizard

Senior Design Laboratory - Project Proposal

Team #4
Logan Henderson, Aditya Gupta, Gina Li

TA: Pusong Li

October 4, 2024

Table of Contents

1. Introduction

1.1 Problem Statement

Pinball machines are complex, multifaceted devices, needing electrical and mechanical skills to troubleshoot, but we want to simplify the process. These machines primarily operate through switches that tell the machine what to show on the screen, solenoids to fire, etc. The issue is, troubleshooting these switches can be extremely challenging, because these machines typically have about 60-100 switches that all need to function properly. Pinball machines made in 1977 and later have rudimentary switch diagnostic features, but those produced before 1977 are designated as electromechanical, meaning they rely solely on physical relays and switches, without any software. This makes electromechanical pinball machines particularly difficult to troubleshoot.

1.2 Solution

Our solution to this problem is the Switch Wizard, a device capable of simultaneously monitoring multiple switches and transmitting data over Wifi to a nearby device. The device will be able to determine if a switch contact has been made and send a timestamp to the user's laptop. This product will be able to test switches while the game is in full operation, with live 6 VAC being used to power the pinball machine's logic. The score motors, which are cammed switched driven by a motor, are extremely difficult to diagnose and this product would provide a much simpler solution. We simply cannot overstate how much easier troubleshooting would become with this tool. If the product proves successful, we could genuinely see it having a valuable place in the industry. Gavin of Gavin's Game Service, a master technician in the Chicagoland area, has already shown interest in the product.

1.3 Visual Aid

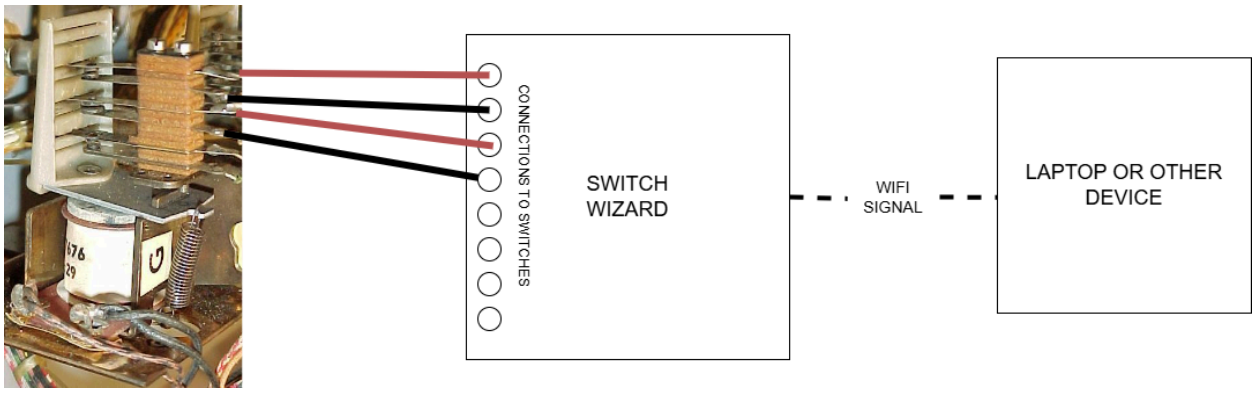

Figure 1: Visual Representation of the Switch Wizard

1.4 High Level Requirements

For this project to be considered successful, we have set some high-level criteria that the Switch Wizard needs to meet. These are:

1. The Switch Wizard must detect switch closures longer than 3ms, which is the estimated time for a successful switch closure.
2. The device will transmit and successfully display the switch closure data over wifi in less than 10 seconds.
3. 5 switches must be able to be connected to the system and a closure of one switch can be detected without affecting other switches that are also connected.

2. Design

2.1 Block Diagram

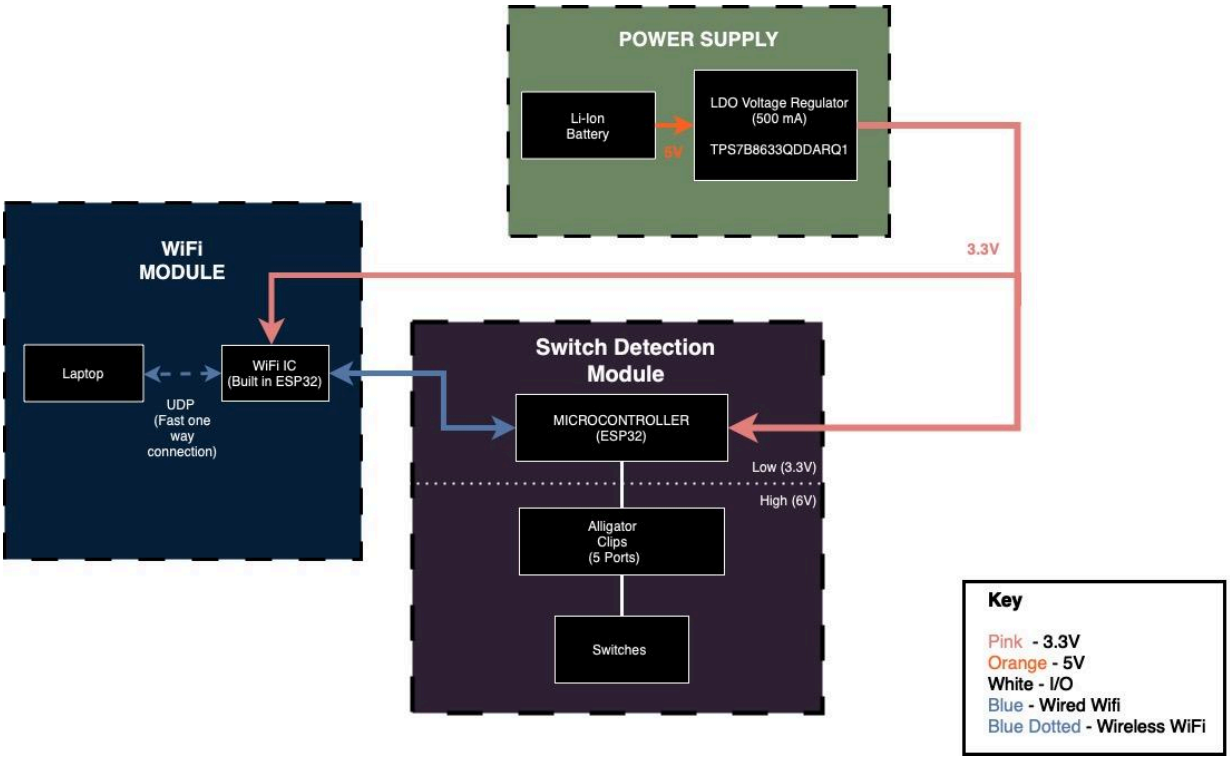

Figure 2: Block Diagram of the Switch Wizard's Subsystems

The power supply system consists of a 5V source regulated to provide stable voltage to the ESP32 and associated components. The switch detection subsystem utilizes alligator clips to connect the ESP32 to various switches, enabling it to send periodic pulses through these switches. Each switch's response is monitored by detecting whether the pulse successfully traverses to the other side, detecting closed or open switch state. Finally, the WiFi module utilizes the ESP32 Wi-Fi to transmit real-time data about switch status back to a laptop for analysis. These subsystems form a unit that enables efficient monitoring and control of multiple switches remotely.

2.2 Subsystem: Pulse Generation

The approach to achieving pulse generation with precise control over frequency, duty cycle, and timing between pulses for multiple switches involves the capabilities of the ESP32, particularly its hardware timers and GPIO control. To implement this, a time-division multiplexing method will be employed, where the ESP32 will generate pulses sequentially for each switch, ensuring minimal delay between pulses. The goal is to achieve a consistent 0.1 ms (100 μs) delay between pulses without overlap, allowing for clean time-division among switches. This will be accomplished using the ESP32's hardware timers, which offer microsecond-level precision, ensuring that the timing between pulse transitions is highly accurate and not affected by external factors or system overhead. The ESP32's GPIO pins will be directly controlled to ensure fast and efficient switching, avoiding the slower high-level functions to minimize delay. The pulse characteristics, including frequency and duty cycle, will be adjustable through software, allowing dynamic reconfiguration based on system requirements. By using timer interrupts, the system will maintain responsiveness and ensure that pulse generation continues smoothly even if other processes are running in the background. The pulses will be designed to remain stable under various load conditions, ensuring that no variation occurs in frequency or timing due to changes in the external circuit. Once the foundational logic is coded, extensive testing will be carried out using an oscilloscope to verify that the generated pulses maintain the correct timing, frequency, and duty cycle. This testing will also confirm that the 0.1 ms delay is consistently achieved and that no overlapping or interference between pulses occurs. Through this combination of time-division multiplexing, precise timing control, and careful signal monitoring, the system will efficiently generate distinct pulses for multiple switches, meeting the specified requirements. The actual coding to implement this solution will be completed as part of the development phase.

Requirements	**Verification**
Must be able to generate distinct pulses with controlled frequency and duty cycle for each switch.	Set up the ESP32 to generate a 500Hz pulse train with a 50% duty cycle. Use an oscilloscope to verify the frequency and duty cycle are accurate.

Must be able to generate pulses sequentially for each switch with a delay of 0.1 ms between pulses.	Set up a hardware timer on the ESP32 to toggle between GPIO pins with 100 µs delay between transitions. Measure the delay between pulses using an oscilloscope and verify it is 0.1 ms (100 µs).
Must generate stable pulses for each switch regardless of load on the circuit.	Apply different loads (e.g., 0 to 100mA) and verify with an oscilloscope that pulse characteristics (frequency, duty cycle, and timing) remain stable within ±5%.
Must ensure no overlapping of pulses between switches (clean time-division).	Connect the outputs for all switches to an oscilloscope. Verify that pulses for each switch are clean and non-overlapping during time-division sequencing.
Must support 5 switches and allow for independent detection of closures.	Connect 4 switches to the ESP32, simulate closures, and verify that pulse generation for each switch is distinct and follows the configured timing without interference.

2.3 Subsystem: Pulse Detection

To effectively detect pulse signals generated by the ESP32, we are monitoring the state of a GPIO pin connected to the output side of the switch. First, the GPIO pin should be configured as an input. The program can initiate a loop that checks the state of the pin at intervals 10 µs to check the pin state. This keeps the loop responsive without overwhelming the processor. When the pin detects a LOW signal, indicating that a pulse has passed through, the program enters a while loop where it continues to check the input state. A counter is incremented for each 0.1 ms interval that the signal remains LOW. If the counter reaches 30 within a total of 3 ms, this confirms that the switch is closed. If the GPIO pin reads HIGH before the counter reaches 30, the loop will exit, indicating that the switch is not continuously closed. This approach ensures accurate detection of switch closures while effectively mitigating false positives from switch bounce by requiring sustained signal presence. By integrating this detection logic, the ESP32 can reliably monitor switch states and respond appropriately to the dynamic conditions of the pinball machine.

Requirements	**Verification**
● Must be able to detect all distinct sequential pulse signals with 0.1 ms delay in between pulses	Constant polling of 10 µs ensures that every pulse is detected. Generate a certain number of pulses, and use a counter to determine

	whether all pulses were detected.
• Must be able to detect switch closure for at least 3 ms • Implement a debouncing mechanism for an unobstructed signal	Monitor the time between when the first pulse is detected and when switch closure is detected. Verify timing is 3ms or above.
• The delay between each iteration of the loop can contribute to the overall time it takes to detect and process the signal. Detect the 30 pulses over a 3ms period with less than a 0.5 ms delay	Reduce delay by optimizing code for the while loop. Verify timing when the first pulse is detected and when closure is detected. Verify timing is between 3ms and 3.5 ms.

2.4 Subsystem: Power Supply

The power subsystem will use a 3.7V 2500mAh LiPo battery with integrated Battery Management System (BMS) and an LD1117 Low Dropout (LDO) regulator. The LiPo battery was chosen for its compact size and high energy density. With a nominal voltage of 3.7V, this battery will provide the necessary power to run the system for approximately 5 hours at a 500mA draw, ensuring that the ESP32 and other components receive consistent power. The integrated BMS ensures that the battery is protected from overcharging and over-discharging, extending the battery's lifespan and preventing potential damage to the system. The LD1117 LDO regulator will step down the battery's output to a stable 3.3V, which is necessary for powering the ESP32, 5 LEDs, and the switch detection circuits. With a maximum output of 800mA, the LD1117 will handle the peak current demands of the ESP32 (which can draw up to 500mA during WiFi operations) along with the additional 100mA required for the LEDs and switching circuits. Capacitors will be added to both the input and output of the LDO to stabilize the voltage and ensure smooth operation, especially during periods of high current draw. This combination of a high-capacity LiPo battery, integrated BMS, and LD1117 LDO regulator will provide a reliable, protected, and stable power supply.

Requirements	**Verification**
Must provide a stable output of 3.3V to power the ESP32 and other components.	Connect the power supply output to a multimeter and ensure the output remains at 3.3V ±5% during various operating conditions.
The power supply must maintain voltage stability under varying battery conditions	Discharge the lithium-ion battery to 3.0V and charge it to 4.2V. Verify with a multimeter

(input voltage ranging from 3.0V to 4.2V).	that the LDO regulator maintains a stable output of 3.3V during both conditions and transitions.
Must be compatible with ESP32's peak power demands (up to 500 mA during WiFi transmission and pulse generation) without causing system instability.	Run the ESP32 in its most power-intensive mode (WiFi active and pulse generation) while monitoring the current draw and voltage stability using an oscilloscope to verify no voltage drops occur.
Must cut off power or send a signal when the battery voltage drops too low to prevent damage to the battery (e.g., below 3.0V).	Simulate a low battery condition (input voltage below 3.0V) and verify that the system shuts down or triggers a low-power mode, protecting the battery and connected components.

2.5 Subsystem: WiFi Module

The WiFi Module is built into the functionality of the ESP32 allowing us to communicate with outside devices such as a laptop using User Datagram Protocol. After the switch state has been detected by the Switch Detection Module, the WiFi module is responsible for transmitting the data. UDP is the most efficient way to transfer the signal data because it sends UDP packets containing the switch states (open or closed) to a predefined IP address, allowing for immediate response times. This is especially advantageous for real-time systems.

Requirements	**Verification**
● The Switch Detection subsystem must be able to detect pulses, as read from the esp32 microcontroller and transmit it to the laptop. ● Configure the WiFi module to transmit UDP packets to the predefined IP address and set up a listening server to receive packets.	● Verify the WiFi module connects to the network by checking the connection status in the serial monitor. Check if there exists a connection to the correct SSID.
● The WiFi Module must handle dropped packets, a rare occurrence. Packet loss should ideally be less than 1%.	● Each packet sent out will be identified with a sequence number that is incremented per packet. Checking for each sequence number will identify

	the packets that were dropped. Ensure dropped packets are consistently less than 1% of total packets sent out.
● 1 ms - 10 ms timeframe to send a UDP packet containing switch state from esp32 to laptop.	● Modify esp32 Arduino IDE to include timing of when packets were sent and received to confirm the time frame is between 1ms-10ms.

2.6 Tolerance Analysis:

In the design of a pinball machine using the esp32 microcontroller, a key risk is the precise polling of the pulses. When continuously polling, we use a small interval of 1µs to check the pin state. This keeps the loop responsive. When the processor is continuously executing a tight loop to poll the pin state, it consumes a significant amount of CPU cycles.

Number of detections = 3ms / 1µs = 3000

Number of detections (considering 0.1 ms delay between pulses) = 0.1 ms/ 1µs = 100

With a tight bound on polling, we are able to detect pulses while considering delay of generated pulses. However, it consumes a large amount of CPU cycles. We can troubleshoot this by using 2 esp32 microcontrollers, one for generating pulse signals and another for detecting them.

3. Ethics and Safety

3.1 Ethics

There are some potential ethical considerations that may arise during the development and use of the Switch Wizard. One such concern during development is ensuring strong collaboration within our group. Poor communication, unfair workload distribution, and/or a lack of participation would negatively impact our project. We are committed to maintaining a professional and ethical working environment by ensuring each member in our group contributes fairly and meaningfully to the project. Another major ethical issue pertains to the possibility of referencing existing designs for any part of the Switch Wizard that are on the market. We will ensure that any such references are done with ethics in mind and make sure to properly cite such references.

We will uphold the IEEE Code of Ethics by adhering to all of their principles including but not limited to: upholding the highest standards of integrity, showcasing responsible behavior, treating all persons fairly and with respect, and avoiding injuring others or their property. We will avoid any potential ethical issues through many ways. To ensure strong and fair collaboration among the team, we will communicate regularly through a group chat. This will help us be reliable and allow us to hold each other accountable. Consistent communication will also ensure that all of the work we do is original and of high quality. If we have any questions or concerns about anything with regards to ethics or safety, we will make sure to contact our TA or any other members of the course staff. All of this will ensure that the IEEE Code of Ethics is always at the forefront of our minds as we progress through this project and course.

3.2 Safety

With regards to safety, we have made sure to identify several potential risks we may encounter. These are mostly related to the power supply and the integration of the pinball machine's live components. The Switch Wizard relies on a lithium-ion battery to power the ESP32, which presents typical risks associated with these types of batteries such as overheating and the potential for fire. To help mitigate these risks, we will make sure to read and thoroughly understand the safe battery usage document on the ECE 445 course website. The integration of the pinball machine's live components also poses a possible risk due to potential electrical hazards. We will always make sure to follow standard lab safety protocols from our lab safety training and ensure that all wiring and soldering are conducted with care. Throughout the development of the Switch Wizard, we will comply with all relevant safety regulations and standards.