Electrical Overview

Year: 2023 Semester: Fall Team: 8 Project: Smart Sensing System
Creation Date: September 12, 2023 Last Modified: September 16, 2023
Author: Gabriel Wang Email: wang4340@purdue.edu

Assignment Evaluation: See Rubric on Brightspace Assignment

1.0 Electrical Overview

The smart seat design system is comprised of 3 rather low-power sensors, and an ESP32 microcontroller for the control unit. The control unit will transmit Each of the sensors will have their data communicated to the ESP32 through a specific peripheral (peripherals will be explained in *3.0 Interface Considerations*). The sensor-micro environment will be powered by a lithium-ion rechargeable battery, with a developed voltage-regulation system.

1. The primary sensor is a force sensitive resistor, which will collect human force contact data while being read in by the microcontroller. The data will initially be sent as an analog reading, that will be converted to a digital output with the microcontroller.
2. A secondary sensor is a human presence sensor, which will collect radar signals to detect motion, and send the signal data via a specified peripheral to the microcontroller.

2.0 Electrical Considerations

Operating Frequency

The operating frequency heavily depends on how long the control unit is designed to be active. However, each part of the control unit can have their own independent active time. The PIR sensor is most likely going to have a lower run rate than the force resistive sensor, which will constantly detect if there is a force being applied to it. However, the FSR can allow for a constant reading due to it's output only being a digital analog signal. The ESP32 microcontroller has a CPU operating frequency that is adjustable from 80MHz to 240MHz. The FSR does not run on an operating frequency, but rather they are analog sensors that change their electrical resistance in response to a force. The PIR sensor's operating frequency refers to the frequency at which it emits and receives microwave signals for motion detection. It doesn't directly relate to the communication frequency used by the ESP32 microcontroller. The ESP32 uses the 2.4GHz band for its wireless communication protocols like Wi-Fi and Bluetooth. This frequency range is used for data transmission between the ESP32 and other devices or networks. These parts can coexist since the PIR sensor's operating frequency is primarily for detecting motion and presence, while the ESP32 uses the 2.4GHz band for wireless communication with other devices or networks.

Operating Voltage

All of our components can be ran adequately with a 3.3-5V input voltage supply. The ESP32 WROOM has an input supply voltage of ~3.6V[4]. Depending on the battery we eventually select (as discussed in *Power Supply*), we would need a combination of buck-down, boost regulator, voltage converters, and voltage translators

- Buck Down - To accommodate with the ESP32 requirements, we may need to incorporate buck-down regulators, to ensure the voltage regulation is stepped down enough to safely operate the microcontroller
- Boost Regulator - In case where the battery voltage falls below the acceptable range, step-up regulators can be applied to both sensors. This also helps the sensors receive adequate supply voltage even when the battery voltage decreases over time.
- Voltage Converters - Converters help adjust voltage levels to make it more stable and efficient between the regulators and sensors.
- Voltage Translators - Ensures data signals transferred between sensors and the ESP32 are properly converted and compatible.

Power Supply

We are considering the use of a 7V, 2200mAh Li-ion battery as the input voltage source. Since Li-ion batteries are susceptible to low voltage errors, typically occurring within the last 30% of State of Charge (SOC), and most of our components operate within the 3.3-5V range, this choice will provide more stable operating hours without significant concern about potential malfunctions. The 7V battery will be connected to two individual Buck Down Voltage Regulators, efficiently supplying power to each of the two sensors. A second consideration we can choose from is a 3.4V, 3000mAh Li-ion battery. This alternative battery option offers an mild voltage level that can be compatible with both the ESP32 Wroom microcontroller and the pool of sensors without the need for extensive voltage regulation. Its capacity of 3000mAh allows for extended operational time, making it a suitable option if we want to value battery longevity. If higher than expected supply voltages constrain this battery choice, we can devise a system of buck boost converters and regulators to raise it's supply voltage. The microcontroller already features a built-in 3.3V voltage regulator for self-management. These voltage regulators fall into two distinctive types: 3.3V and 5V outputs, offering subsystem flexibility as each component has different requirements. Both types allow for easy on/off control of each component, are tailored for battery-powered applications, exhibit high power efficiency with low power dissipation, and are equipped with voltage and current protection features. Lastly, our web server that receives and processes all the sensor data will most likely exist in a laptop/monitor that is plugged into a wall outlet.

1. FSR Sensor Requirements: There are no specific voltage and current requirements for this sensor since it is comprised of resistance strips.

2. ESP32 Microcontroller Requirements: Voltage and Current: 3.3-5V at 90-240mA. Worst Case Power Consumption: Approximately 1W.
3. Human Presence Sensor Requirements: Voltage and Current: 5V at 70mA. Worst Case Power Consumption: 0.35W.

4. Total Power Consumption: FSR (0.2W for added safety) + ESP (1W) + Human Presence (0.35W) = 1.55W.

5. Tolerance: Minimum Power: 10% of the total Power Consumption Maximum Power: 10% of the total Power Consumption.

3.0 Interface Considerations

1. UART (Universal Asynchronous Receiver/Transmitter) is a crucial serial communication protocol used for connecting electronic devices and facilitating data exchange. It's commonly employed to link microcontrollers with various peripherals like sensors and GPS modules. In UART communication, we talk about baud rates, which can range from standard values like 9600 to much higher speeds, depending on the hardware and requirements[1]. The ESP32WROOM has a 3 UART controllers that share a similar set of registers that allow for simplified programming. Some UART pins that can be used are GPIO 2, 4, 5, 18 which can route the necessary signals. The PIR sensor will speak UART to the microcontroller in our system

2. SPI (Serial Peripheral Interface) is another serial communication protocol that helps microcontrollers communicate swiftly with peripherals like sensors, displays, and memory modules. SPI is known for its high-speed data transfer and the ability to send and receive data simultaneously. The actual SPI data rates depend on the hardware and user settings, often reaching speeds in the tens of megahertz or more[2]. This will be more or less a “back-up” peripheral that we can use for extra sensors or if we run into problems with our original interfaces.

3. I2C (Inter-Integrated Circuit) is a synchronous serial communication protocol used for connecting multiple devices to a central microcontroller. These devices include sensors, EEPROMs (Electrically Erasable Programmable Read-Only Memory), and various low-speed peripherals. I2C offers different operational speeds, such as Standard Mode at 100 kilobits per second (kbps), Fast Mode at 400 kbps, and High-Speed Mode reaching up to 3.4 megabits per second (Mbps)[3]. The achievable data rate varies based on the hardware components and bus characteristics, like capacitance. This will be another “back-up” peripheral that we can use for extra sensors or if we run into problems with our original interfaces.

4. ADC (Analog to Digital Converter) is a peripheral that converts an analog reading to a digital, numerical value (12 bit readings). It can be programmed for analog signals at various rates and frequencies, making it widely used for plenty of IoT emmbedded applications. Additionally, it supports features like DMA, interrupts, and callbacks for data processing. The ESP32 integrates two 12-bit SAR ADC’s, allowing for 18 measurement channels[4]. The ADC peripheral will be the primary communication protocol for the FSR.

4.0 Sources Cited:

[1] Understanding UART. Rohde & Schwarz. [Online]. Available: https://www.rohde-schwarz.com/us/products/test-and-measurement/essentials-test-equipment/digital-oscilloscopes/understanding-uart_254524.html#:~:text=UART%20stands%20for%20universal%20asynchronous,and%20receive%20in%20both%20directions. Accessed: September 13, 2023.

[2] SparkFun Electronics. "Serial Peripheral Interface (SPI)." SparkFun Electronics Tutorials. [Online]. Available: https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all. Accessed: September 13, 2023.

[3] SparkFun Electronics. "I2C (Inter-Integrated Circuit)." SparkFun Electronics Tutorials. [Online]. Available: https://learn.sparkfun.com/tutorials/i2c/all. Accessed: September 13, 2023.

[4] ESP32 series - espressif systems, https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf (Accessed Sep. 16, 2023).

Appendix 1: System Block Diagram