EASE: Notifying Feminine Hygiene Pad

Alexander Nguyen, Matthew Poole, David Garzon, Dillon Sardarsingh

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — This senior design project explores the creation of a smart feminine hygiene product that provides real-time usage notifications. The device incorporates sensor technology and electronic components to detect changes and alert users discreetly. Research focused on material selection, circuit design, and power optimization to ensure reliability and safety. Prototyping involved iterative testing to refine functionality and improve user experience. Key challenges included sensor accuracy, power efficiency, and device miniaturization. The project demonstrates the potential for innovative solutions in personal care, paving the way for future advancements in smart hygiene technology.**

***Index Terms* — Smart feminine hygiene, sensor technology, real-time notifications, circuit design, power optimization, prototyping, user experience**

I. Introduction

In recent years, advancements in wearable and smart technology have paved the way for innovative solutions in personal care and hygiene. Despite significant progress in health technology, feminine hygiene products have seen relatively little integration of smart features that enhance user convenience and discretion. This project aims to bridge that gap by developing a notifying feminine hygiene product that discreetly alerts users when a change is needed.

Our design integrates electronic components, including sensors, to monitor product usage and provide real-time notifications. The goal is to improve comfort, reduce uncertainty, and enhance hygiene management for users. Key design considerations include material selection, circuit design, power efficiency, and ensuring user safety. Throughout the development process, our team faced challenges such as sensor accuracy, wireless communication reliability, and device miniaturization. This paper details the research, design, prototyping, and testing of our device. It discusses the technical challenges encountered and the solutions implemented, providing insight into the feasibility of integrating smart technology into feminine hygiene products. Our findings demonstrate the potential for future advancements in this field, paving the way for improved personal care solutions.

II. Goals

The goals the team is hoping to achieve are broken down into basic, advanced and stretch goals. The basic goals the team is hoping to achieve are comfortable design, reliable capacity sensing, and a reliable notification/alert system. The user should be able to use and rely on the device to provide them with an accurate notification and prevent leakage without sacrificing comfortability. The advanced goals are ones that further improve the quality of the basic goals, here the team is looking to improve battery life, app compatibility, and a reusable design. These advanced goals build upon the basic goals and are designed to improve the quality of the product.

III. Engineering Specifications

The engineering specifications for the notifying feminine hygiene product focus on key design parameters to ensure functionality, safety, and usability. The device incorporates a compact electronic system with sensors to detect usage and trigger real-time notifications. Power management is optimized for low energy consumption, utilizing a small battery to maintain efficiency. The materials selected prioritize user comfort, safety, and compatibility with hygiene standards. Wireless communication is integrated for discreet alerts. The design also emphasizes durability and reliability, ensuring consistent performance throughout the product's lifecycle. These specifications establish the foundation for a functional and user-friendly smart hygiene solution.

In order to judge accuracy of the system, the device initially should have a volume accuracy of around ± 2 mL. However, the team is aiming to make the device as accurate as possible, since feminine hygiene products typically have a volume of 5 mL for regular sized products. A volume accuracy of 2 mL could lead to leakage as it is close to the total capacity of these products. In its current state, the team is aiming to have a volume accuracy of ± 0.5 mL. After reaching 75% and 95% capacity respectively, the device should send a notification to the user that the product needs to be replaced.

IV. System Components

In order to design a system that meets the engineering specifications, several individual components must be interfaced to create an operational final product. A multitude of various technologies and manufacturer models were researched for each component. This section provides an overview of the components selected and used within our design.

A. Microcontroller Unit

The core of the design must utilize a microcontroller unit to operate in any capacity. For the prototyped version of the notifying feminine hygiene pad, an ESP32-WROOM-32 module was selected as it already includes an antenna and crystal oscillators required for our design. Additionally, the modules footprint is large enough to troubleshoot as necessary in the design process. Future design iterations should utilize a TI CC2640R2L as it better adheres to the physical and economical constraints of the project in terms of minimizing both size and cost.

B. Capacitive Sensor

Due to the notification being heavily reliant on feminine hygiene capacity, a capacitive sensor was designed with a flex PCB. This type of sensor can provide some form of volumetric measurement while still being easily implemented into a feminine hygiene pad for realistic application unlike other sensors which would be too large or restrictive. Changes in capacitance of the flex PCB can be detected and once a certain capacitance threshold is met that is equivalent to the 75% and 95% capacity of the feminine pad, a notification can be sent properly.

C. Capacitance to Digital Converter

A large limitation to using a capacitive sensor is its lack of accuracy which can be extremely problematic as real application of the product should be expected to detect extremely low liquid volumes (up to 5 ml) with very high accuracy for notification performance. A capacitance to digital converter was implemented to combat this issue. The AD7746 from Analog Devices was selected for its accuracy, 24-bit resolution, temperature resilience, and availability for designing relative to other components.

D. Solid State Sensors

Although capacity is a strong indicator of leakage potential within a feminine hygiene pad, it is not a guarantee. Leakage could occur laterally along the pad, even if the pad itself has not reached full capacity. Because of this issue, solid state sensors were designed and implemented. These sensors line the perimeter of the flex PCB and are designed concentrically to trip if shorted. This effectively acts as a safeguard in the case that leakage is occurring despite capacity not being reached.

E. UART to USB Bridge

Implementing a method of communication between the MCU and peripheral devices for the purposes of debugging and flashing software is necessary. A UART to USB bridge IC was chosen due to its simplicity and ease of implementation. The arduino IDE works seamlessly with COM ports and the CP2102, a very popular UART to USB bridge IC is recommended by Espressif making it our first choice. We did not consider using any other method of communication with the MCU due to the massive advantage compatibility with USB has in terms of ease of use. In the event that we want to shrink the footprint of our board we would still most likely keep our current setup and instead plan on removing the USB A connector to save space on the board and keep its physical size down.

F. Battery

After initial research, a CR2032 battery was chosen to power the device due to its small size and voltage output, however after initial testing and integration, it was found that the CR2032 battery could not provide sufficient current to power the microcontroller and its peripherals. In order to keep this device user friendly and still be able to provide the 3V required by the voltage regulator with enough current, the team decided to use a 9V battery. After testing, this battery proved to be successful in powering the device. The only drawback is its size relative to the rest of the design.

G. Voltage Regulator

A boost converter is necessary to provide a steady voltage input as the batteries discharge and to boost the 3V output of the 9V battery to 3.3V, which is required by the microcontroller. The team initially decided to go with the LTC1682-3.3 by Linear Technologies because of its size, switching frequency, and ease of implementation. However, due to its inability to provide sufficient output current in the altered design, the voltage regulator used was changed to the NCP1117 with fixed 3.3V output to accommodate the new load requirement. The voltage regulator comes in an SOT-223 package size, configured with 10 uF input and output bypass capacitors.

V. System Concept

The system concept for the notifying feminine hygiene product is designed to enhance user convenience through real-time monitoring and discreet notifications. The device

integrates sensor technology to detect changes in usage and transmits alerts through a compact electronic system. A key focus is ensuring power efficiency, enabling the device to operate with minimal energy consumption while maintaining reliability. The system is designed with user comfort and safety in mind, utilizing materials that adhere to hygiene standards. Additionally, wireless communication is incorporated for seamless notifications, making the product a practical, discreet, and user-friendly solution for improved hygiene management.

A. System Block Diagram

The figure below corresponds to the hardware block diagram for the project. This diagram presents a high-level visual of the major components of our electronic device and how they are expected to integrate and interact with each other, providing a system overview. Additionally, the figure represents a simplified version to clearly indicate the relationships and flow of how the system will operate to generate the desired outcome. Aspects of the diagram are color coded to reflect work distributions and responsibilities as indicated in the corresponding legend.

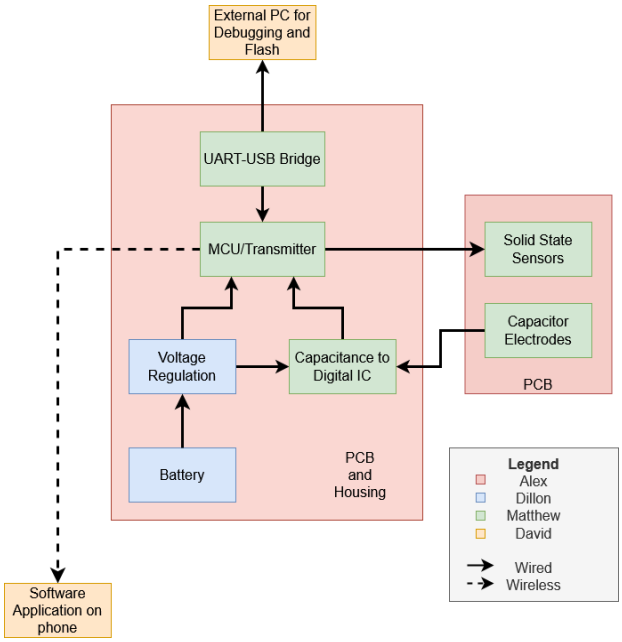

Fig 1. Hardware Block Diagram

VI. Hardware Design

The hardware design consists of two PCBs, one fr4 board that contains the MCU and most subsystems, and a flex PCB electrode pad which houses the capacitive sensor electrode and the solid state sensors. The fr4 board handles all transmission and capacitance to digital conversion while the flex PCB is kept separate to reduce the size of the insert within the pad. The two PCBs would be connected via a 12 pin fpc ribbon cable and this would allow the main PCB to sit outside the hygiene pad and avoid interfering with measurements. This setup was chosen to increase our chances of project success as the current design will serve as a proof of concept rather than a marketable product.

A. Electronic Schematic Design

The FR4 board which we will refer to as the main PCB consists of a 3.3 V boost regulator, capacitance to digital converter circuit, UART to USB bridge IC and the ESP32. The main PCB only utilizes two layers with a common ground plane occupying most of the second layer. Most of the main PCB schematic follows common ESP32 dev board layouts including the UART to USB auto bootloader circuit and the enable and reset buttons.

The electrode schematic is much simpler due to the lack of ICs and components. The bulk of the electrode pad is the capacitive electrode and the solid state electrodes which are only copper pours and traces. A few resistors are used to limit current flow on the solid state electrodes and the fpc connector and ribbon cable used to connect it to the main PCB. Additional peripherals such as oscillators and JTAG debugging ICs were not included because we deemed them unnecessary for our project.

An original plan for the project involved a different MCU, the TI CC2640 which we chose due to its small package and RF capabilities. Our goal was to create a very condensed main PCB that would ultimately also sit within the hygiene pad alongside the electrode pad. This was done to prioritize comfort and a discrete form factor, placing PCB size above all other metrics. We inevitably changed this for our project and moved to the ESP32 platform because it better suited our needs in terms of wireless communication and had the benefit of being a much easier platform to develop on and design. Once the concept is proven to be possible the idea of transitioning back to a condensed main pcb is a good option and would solve a lot of the real world drawbacks our current design imposes.

B. Main PCB Layout Design

The main PCB layout follows common circuit design practices. The second layer is used as little as possible for traces in order to preserve ground continuity. All subsystems use the same common ground even though the capacitive to digital converter and its traces would benefit from its own ground plane and isolation from the digital

ground. This was done to keep costs down as a single ground plane proved to be sufficient during testing.

Most traces around the board are limited to 20 mils but will often shrink down to 12 mils for smaller pitch ICs such as the UART to USB bridge, ad7746 capacitance to digital converter, and the 12 pin fpc connector. Bypass capacitors are used for each IC to ensure stable input voltages. In order to ensure correct functionality this iteration of the design uses a spread out layout, opting for more space in between circuit subsystems for better readability and debugging. For a design that is intended to be marketed the main PCB would need to be considerably downsized in order to incorporate it discreetly and comfortably inside of a pad.

C. Electrode PCB Layout Design

The electrode PCB layout is where its hardware design is most challenging. Originally we wanted to employ a solid state matrix surrounding the capacitor electrodes but this required multiple diodes which restricted the size of our capacitor electrode and hindered the comfortability of the pad once inserted. Instead the design was changed to use concentric ring traces which would have their solder mask removed allowing for shorts to occur and be detected by the MCU as shown in Figure 2 and 3. This uses much fewer components and allows the capacitor electrodes to be much larger and complex. With this we changed the capacitor electrode to a interdigitated fringing field electrode design. This design is often used in capacitive sensing circuits and provides the benefit of a larger field depth and stronger field which gives the capacitor much higher precision when in the presence of different dielectrics. The biggest drawback of this design is its biggest strength and that is the capacitance of this electrode is measured to roughly 444 pF which requires us to use an additional range extension circuit in order to correctly measure it's full capacitance range and offset the base capacitance range of the AD7746. Due to the unique manufacturing requirements of the flex PCB there are multiple spots around the solid state rings where the solder mask needs to stay intact. Removing the solder mask in this area is harmful to the FPCs structural integrity and is not advised by our manufacturer.

D. Capacitance to Digital Converter Extension Circuit

The designed capacitive sensor has a capacitance of 445 pF which lies outside the operating range of the AD7746. The application note AN-1585 from Analog Devices outlines a method to extend the capacitive input range of the AD7745 and AD7746 capacitance-to-digital converters (CDCs) by utilizing an external circuit in conjunction with the on-chip digital-to-capacitance converter (CAPDAC)[1].

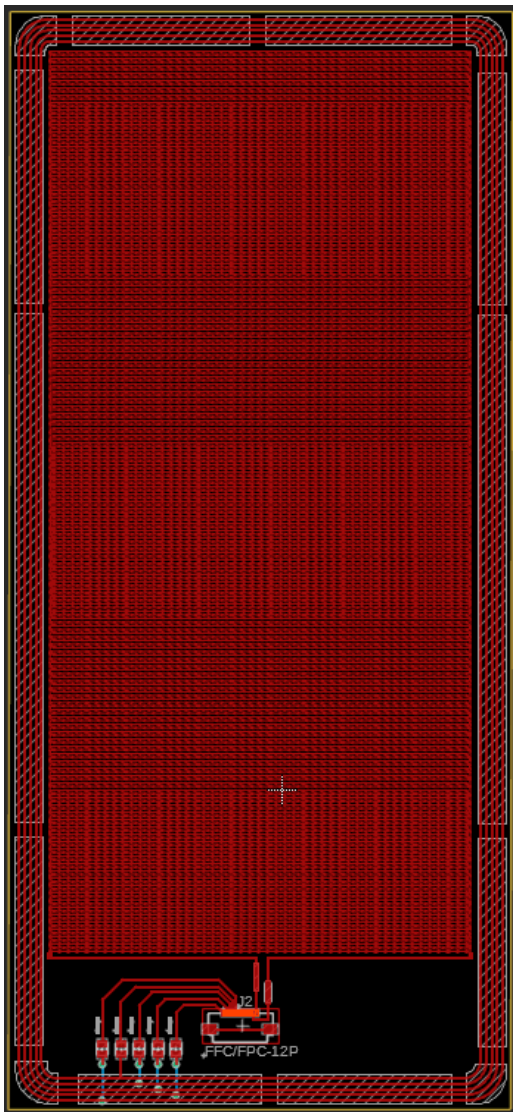

Fig.2. Electrode Pad full view

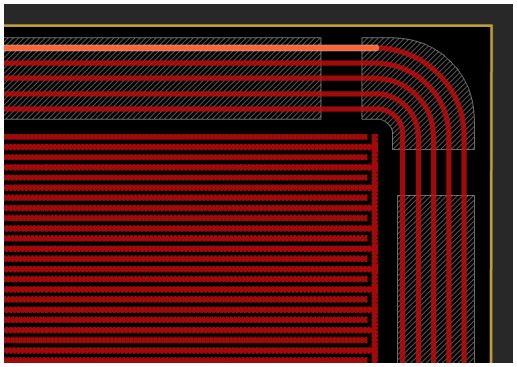

Fig.3. Closeup of the Capacitor Electrode and Solid State Rings

This approach enables accurate measurement of larger capacitances while minimizing errors and optimizing performance. The AD7745 and AD7746 CDCs measure capacitance using a switched capacitor technique, where the charge (Q) is proportional to the product of voltage (V) and capacitance (C), expressed as Q = V × C [1]. The conversion result represents the ratio between the input capacitance (C_SENS) and the internal reference capacitance (C_REF), with fixed excitation voltages (EXCA and EXCB) and internal reference voltage (V_REF) [1].

To accommodate larger input capacitances, the excitation voltage is attenuated by a factor (F), allowing the measurable capacitance to increase correspondingly. This is achieved by configuring EXCB as the inverse of EXCA and connecting resistors R1 and R2 in a specific arrangement [1]. The range extension factor (F) is determined by the ratio of the differential excitation voltage (V_EXC(A–B)) to the attenuated excitation voltage (V_EXCS) at the input of an operational amplifier (op-amp) such as the AD8515 [1]. The op-amp serves as a low impedance source, ensuring the sensing capacitance is fully charged during sampling [1]. The configuration of the external extension circuit is provided in Figure 4.

The AD7745/AD7746 feature CAPDACs capable of compensating for sensor bulk capacitance. With a full-scale value ranging from 17 pF (minimum) to 21 pF (typical), the CAPDAC can offset significant capacitance variations [1]. However, due to potential variations in on-chip capacitances across different devices, calibration is necessary [1]. The factory calibration factor (F_GAIN_CAL) is stored in the Cap Gain Calibration register and can be used to adjust measurements, ensuring accuracy despite inherent component variations [1].

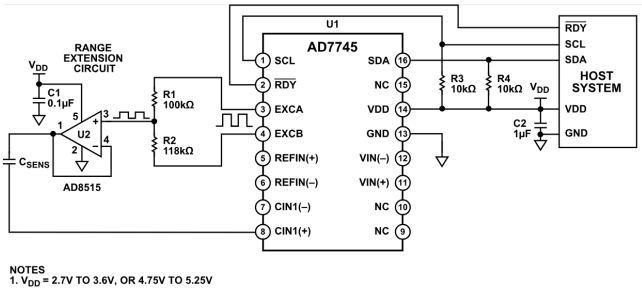

Fig. 4. External Range Extension Circuit for the AD7746

Due to uncertainty as to how far the range extension circuit operates beyond 200 pF, the designed 445 pF flex PCB capacitive sensor was placed in series with two parallel 330 pF and 36 pF capacitors. This effectively reduced the measured capacitance to 200 pF. R2 and R2 were selected to be 100 kΩ and 120 kkΩ to similarly maintain the sample provided giving an F value of 11. After applying these configurations to the PCB, parasitic capacitance was present, negatively impacting functionality. Thus, an alternative solution was designed.

E. 555 Timer Frequency Circuit

Rather than directly converting capacitance to a digital value, a 555 Timer Circuit can be designed in such a way that changes in the capacitance of the electrode pad can alter the frequency of the square wave generated. This frequency can then be outputted to correlate with a specific threshold value. Utilizing the relationship below, as fluid is applied to the capacitive sensor (increasing its capacitance), the frequency of the square wave should decrease from its base frequency of 88 kHz, when utilizing resistor values of 10 kΩ.

$$f = \frac{1}{0.639\ (Ra+2Rb)\ C} \qquad (1)$$

This circuit was constructed on a breakout board and then directly wired to the main PCB designed previously. By displaying the serial output data of the recorded frequency, repeated testing approximated the 75% capacity and 95% capacity to equate to a frequency of 56 kHz and 50 kHz respectively at full absorption. These frequency thresholds were then coded to act as the trigger for the notifications.

VII. Software Design

The primary software goal of the EASE Notifying Feminine Hygiene Pad project is to facilitate real-time data collection, transmission, and deliver a push notification to the user's mobile device, informing them when their hygiene product has reached a critical threshold and needs a replacement. By integrating advanced Bluetooth Low Energy (BLE) communication protocols, the system enables secure and reliable data transmission from the device to the user's mobile application. This section defines the design, its implementation, and how components will be tested. Our software design consists of four main components: MCU configuration code, sensor code, BLE related code (such as configuration and transmission), and mobile application code.

A. Basic Goals

The system's basic software goals include:

1) *Seamless communication*: The mobile application must be able to receive data from the transmitter

reliably and without significant delay (more than a few seconds). This will ensure the user will receive a notification when their pad is at or near capacity.

2) *User-friendly notification system*: The mobile application must notify the user when the hygiene product is near full capacity or when it has reached the predefined capacity threshold. The notification should be clear, easy to understand, and actionable.
3) *Compatibility*: The mobile application software must be capable of functioning across Android and iOS devices to ensure accessibility for a wide range of users.
4) *Accurate monitoring*: The software should process sensor data and evaluate it to assess if the feminine hygiene product has reached a predefined capacity limit. This evaluation must be performed continuously or at set intervals.
5) *Low power consumption*: Both the transmitter and the mobile application should be optimized for low power consumption to prolong battery life, particularly during passive monitoring.

B. Implementation

For our software implementation we had to decide our development environment and how we would structure our software. We can split our environment into two major components: embedded development and mobile development.

Our embedded development depends on our MCU as we want our development environment to be as integrated with it as possible. Our chosen MCU is the ESP32, which narrowed our development environment selection to a few choices: Arduino IDE, Visual Studio Code with PlatformIO Extension, Espressif IDE, and Eclipse with ESP-IDF Plugin. We chose to stick with Arduino IDE as our code complexity is quite small and Arduino IDE has minimal setup complexity and board support. Its best use case is for quick prototyping, which in our use case is what we are looking for. Because we are using Arduino IDE, we chose to use Arduino Sketch as our coding language. Arduino Sketch is Arduino's version of C++ which allows us to code in the IDE's native language. This boosts our ease of development as community samples and guides are typically written in this language.

Our mobile development environment ideally allows for cross-platform development for Android and iOS devices. Though we could code natively for each system, having one centralized code that works for both platforms helps us keep our code clean and consistent. Keeping that in mind, we eliminated Swift for iOS and Kotlin for Android as our main choices for mobile development frameworks and decided to choose between Flutter and React Native. We ultimately chose Flutter as our mobile development framework because it works for Android, iOS, Web, and Desktop applications. Flutter also has a near-native performance and is best suited for high performance. Another beauty of Flutter is that it utilizes "widgets" for its UI framework, which is platform consistent, unlike CSS which may be different between devices. Flutter uses Dart as its primary language and our chosen IDE is Visual Studio Code as its extensive libraries and community support allow for industry level development.

Our embedded development includes the MCU code, the Sensor code, and the BLE code. Our mobile development will be separate as it utilizes its own framework, but it will ultimately communicate and transmit data to and from the embedded development code.

C. MCU Code

Our MCU code focuses on MCU configuration. This includes the set up for the ESP32 and the setup for the digital timer. We used to use the AD7745 in which case our MCU code focused on setting up the ESP32 for its use along with setting up the proper registers for the AD7745. We need to list register values in our code and ensure all configuration is properly set up to run as intended. We are able to test this by re-reading our values within the code and verifying set values have changed from their defaulted positions. Certain values change from board to board, so general functions are being made in order to help facilitate swapping our hardware components in case a backup is needed.

D. Sensor Code

Our sensor code focuses on the reading of frequency data from the digital timer and converting that raw value to a calculated and measured value that will help us determine if the product has reached a capacity threshold. Once a value over a predefined limit has been reached, the code will set off a boolean variable in order to activate the BLE code to transmit for the push notification. The sensor code polls over a delay to take continuous measurements in order to ensure a push notification can be sent as soon as the threshold is met to avoid further delay.

The frequency data is read by reading from the digital timer using the FreqCountESP.h library. The library includes a method that allows us to directly get a frequency value, typically in the 70,000 to 90,000 range. As the value gets affected, the frequency will lower to a value like 55,000. Thresholds were set based on repeated tests using these values, ultimately choosing the most consistent range for the 75% threshold and for 95% threshold.

The sensor code is tested with repeated measurements and runs to ensure data consistency with different levels of liquids to meet different thresholds.

Our previous sensor code for the AD7745 used to read the RDY bit in order to determine if data was updated and the capacitance was obtained from the CAP DATA register. This capacitance data was then offsetted, and smoothed for a more consistent value. Values were offsetted by taking the first 20 data readings and averaging out to get a consistent starting position that we subtracted from our raw data. We then smoothed this by taking 10 samples and the average of those 10 samples became the data we used for threshold testing. The frequency timer is much more consistent and does not need this smoothing or offsetting.

E. BLE Code

The Bluetooth Low Energy (BLE) code consists of the configuration of BLE for the ESP32 and the handling of transmission over BLE from the ESP32 to the mobile device. The BLE code consists of initializing a device, creating a BLE server and service, adding characteristics to read and write, and a descriptor. The service will then be started and advertising the BLE connection could then begin. The code will wait for a mobile device to pair with the BLE connection before proceeding with the rest of the code. During this paused time, the sensor will perform its offset calculations in order to save time. Once a client connection is made, the code will continue, frequency data will be read and once a threshold is met with the smoothed value BLE will then transmit the data.

In order to transmit data via BLE, characteristics were made during initialization and the configuration of the BLE service. These characteristics can be modified and this will be sent over the BLE service to the connected mobile device given the correct Service UUID and Characteristic UUID. Once the characteristic is changed, the mobile device will be notified of the updated value and the mobile application code can then perform the necessary actions.

If the BLE connection is disconnected, then the BLE service will advertise and wait for another stable connection. Only one connection can occur at once and we have safety guards put in place to prevent multi connection. We do this in order to ensure only one mobile device is connected to our PCB at once as we do not want another user to possibly receive someone else's notification.

In order to test the BLE code we simply connect real mobile devices to our MCU and see if data is successfully transmitted while paired. We test disconnect and reconnect functionality. BLE is used because iOS devices struggle with Bluetooth connection but work better with Bluetooth Low Energy connections.

F. Mobile Application Code

The mobile application packages are bundled by the Flutter framework and allow for specialization between devices. We however opted for platform consistency and used one file for both devices with no extra modifications on a platform-level. We use the default packaging for Flutter and focus our coding on the main.dart file on the main library. The code file consists of three main components working together: BLE transmission connection and handling, push notification handling, and the GUI.

The BLE component on the mobile application code focuses on scanning for the BLE service advertised by the embedded code, connecting to that service, discovering services of that connection such as its characteristics and subscribing to its notifications, and handling those subscriptions and characteristics to then perform actions for the push notification. Once the ESP32 is connected to the mobile device over BLE, the characteristic communicated telling the mobile application that a threshold has been met can be read and subscribed to. If that subscription is activated in the mobile application, the mobile application then knows to check for a keyword and start the push notification sequence.

The push notification sequence is handled separately through Flutter and the mobile device directly. No computation is made on the mobile development side.

The UI is handled via widgets and is kept simple in order to encourage focus on the push notification which occurs while in and while out of the application. This is tested with live simulations.

The mobile application code can be tested via simulation and via live demo with a variety of Android and iOS devices. We mainly test on real, live devices as it mimics a real-life scenario best. In order to test the mobile application we've created "mock" code to run the embedded side while the board is still being developed and fully integrated code to test a fully integrated system. Our main focus is that the fully integrated system will run smoothly, timely, and without any bumps, obstacles or major issues.

VIII. System Fabrication

Realistic implementation of the design requires specific adherence to physical constraints without negatively impacting the performance of the device. This includes being compact, flexible, and durable while also being resilient to temperature, dust exposure, and fluid exposure.

This section presents how the flex PCB sensor and main PCB are expected to be integrated for realistic use.

A. Sensor Integration

Since the designed flex PCB integrates both the capacitive sensor and solid state sensor safety, considerations for both were considered for implementation. The capacitive sensor requires a dielectric separating the sensor itself and the fluid it is expecting to measure, while also being close enough to detect the small fluid changes. Solder mask from the flex PCB acts as a suitable dielectric that effectively protects the sensor from direct fluid contact. Alternatively, the solid state sensors require direct contact with the fluid to effectively short. Because of this, they are fully exposed on the flex PCB without any solder mask.

Standard feminine hygiene pads can be manually separated into its three primary individual layers: the top sheet, absorbent core, and back sheet. Since the absorbent core holds the majority of the fluid, the capacitive sensor must be placed directly under this layer. Thus, the flex PCB sensor is placed between the absorbent core and the back sheet. The flex PCB was designed so that the concentric solid state sensor rings lie just outside the width of the absorbent core. Thus, if lateral leakage were to occur, the solid state sensors maintain capability of detecting it. Integrating the combined sensor design in this way allows for both sensors to function optimally.

B. Housing Unit

Due to limitations in regard to the prototyped version of this project, the main PCB holding all essential ICs must be placed externally. Although much of the utilized ICs are extremely small, components such as the ESP32-WROOM in the prototype along with the battery holder are too large to be constrained within the feminine hygiene pad itself. Additionally, the main PCB is constructed out of standard FR-4 material, thus making it extremely rigid and awkward.

To maintain visual aesthetics, a housing unit is designed to hold the main PCB. It is designed and 3D printed in such a way that it does not add unnecessary bulk to the PCB itself. Holes are placed to account for all necessary connections, to include the battery holder wiring, ribbon cable connection, and USB-A connector. Additionally, the design allows for the battery holder to be fully mounted on the 3D housing unit. Structuring the main PCB in this fashion allows for the user to easily replace the 9V battery as necessary. while also maintaining visual aesthetics and discreetness for realistic application of the device.

C. Electronic Protection

To maintain electronic protection, MG Chemicals - 422C-55 MLCA 422C Silicone Conformal Coating are applied to both PCBs. For the flex PCB, open exposure of the ribbon connector, resistors, and test traces can negatively impact performance if directly exposed to bodily fluid and physical contact in real application. Silicone conformal coating provides protection while still maintaining the physical flexion capabilities of the PCB. Additionally, the coating provides further protection from user cleaning of the flex PCB for repeated use cases. For the main PCB, coating components, ICs, test points, and connections with conformal coating can add additional protection from dust or fluid exposure that may occur from accidental user misuse.

Acknowledgement

The authors wish to acknowledge the assistance and support of Dr. Lei Wei and Dr. Arthur Weeks for providing guidance and education in the design of this project.

Additionally, the authors would like to acknowledge Gabriela Mercado, the owner of the project, for providing requirements and expectations that the project should be expected to fulfill.

Furthermore, the authors further acknowledge the Review Committee composed of Professor Mark Maddox, Professor Elizabeth Coln, and Professor Saleem Sahawneh for their feedback, input, and assessment throughout this project.

Finally, the authors would like to acknowledge JLC PCB for their sponsorship and funding of significant portions of this project during the design and prototyping phases.

References

[1] [1]“AN-1585: Extending the Capacitive Input Range of the AD7745/AD7746 Capacitance-to-Digital Converter,” *Analog.com*, 2025. https://www.analog.com/en/resources/app-notes/an-1585.html (accessed Mar. 28, 2025).