Term Report (ESET 369)

Kyle Rex
Department of ETID
Texas A&M University

Ethan Maresca
Department of ETID
Texas A&M University

***Abstract*—This report presents a comprehensive review of the lectures and laboratory activities completed in ESET 369 at Texas A&M University, emphasizing key topics in embedded systems, including microcontroller architecture, low-level programming, peripheral interfacing, signal acquisition, and system-level design. In addition, the report details the design and implementation of the Smart Plant Monitoring System, an embedded application utilizing the MSP430FR5994 LaunchPad to measure soil moisture and ambient temperature. The system integrates analog sensing, real-time data processing, LCD-based user feedback, and Bluetooth wireless communication within an energy-efficient architecture. Both theoretical concepts and practical applications are explored, demonstrating mastery of embedded system principles through a modular and scalable design.**

I. Introduction

Embedded systems play a crucial role in a variety of applications, ranging from consumer electronics to industrial automation. Understanding the architecture, programming, and integration of these systems is essential for designing reliable and efficient devices. ESET 369 provided an in-depth exploration of embedded systems, with a specific focus on microcontroller-based designs using the MSP430FR5994. The course combined theoretical lectures with practical laboratories to give students the knowledge and skills required for real-world embedded systems development. Key topics included microcontroller architecture, memory organization, GPIO control, assembly language programming, interfacing with sensors and external peripherals, real-time signal processing, and communication protocols. This report summarizes the main concepts learned in the course and demonstrates their application through the theoretical design of a Smart Plant Monitoring System. The system utilizes real-time sensor data to monitor plant health, incorporating both local and remote feedback mechanisms, thereby showcasing the integration of hardware and software in embedded systems.

II. Lectures/Labs

The lectures and laboratories in ESET 369 collectively provided a comprehensive foundation in embedded systems programming, system design, and hardware integration. Lectures initially emphasized the fundamentals of microcontroller architecture, memory organization, and the role of General-Purpose Input/Output (GPIO) in hardware control. These principles were directly applied in early labs where embedded C/C++ programming was used to manipulate GPIO pins on the MSP430FR5994 microcontroller, implementing LED blinking sequences and light patterns. Key concepts such as setting data direction registers, writing to output registers, and using software-based delays to create timing control were reinforced through hands-on activities. The importance of bitwise operations, masking, and shifting techniques for efficient register manipulation was also stressed, linking software techniques to hardware behavior. Timing and delay generation without relying on operating system support highlighted the real-time constraints typical in embedded systems.

As the course progressed, lectures introduced assembly language programming for embedded applications, providing insight into the processor's instruction set architecture, stack operations, and register usage. Labs accompanying these topics required implementing basic mathematical operations, loops, and conditional structures using ARM assembly, solidifying an understanding of how high-level logic translates to low-level operations. Particular emphasis was placed on optimizing code efficiency and managing control flow manually without the aid of compilers. The transition to assembly programming deepened comprehension of microcontroller inner workings, including the function of the program counter, stack pointer, and condition flags during operation.

Interfacing external peripherals with microcontrollers became a major focus in both lectures and labs. Concepts such as pull-up and pull-down resistors, input debouncing, and matrix scanning were introduced in class and practiced by integrating RGB LEDs, keypads, and joysticks into embedded designs. Students learned to handle multiple input sources and outputs simultaneously, applying techniques like polling and scanning to detect user interactions without introducing excessive latency. The lectures covered the design considerations for human-machine interfaces (HMI), addressing issues like responsiveness, electrical noise, and input prioritization. Labs reinforced these ideas by requiring systems that mapped keypad inputs to LED responses and joystick movements to analog voltage measurements, ensuring both hardware setup and software logic were functioning cohesively.

Real-time signal acquisition and processing were emphasized through detailed discussions on Analog-to-Digital Conversion (ADC) modules. Lectures explained sampling theory, resolution, reference voltages, and noise considerations when measuring analog signals in embedded systems. These topics were applied practically by using the MSP430's ADC module to capture sensor data, such as joystick positions and reflectance sensor outputs. Voltage readings were processed and displayed in real time using external LCDs, allowing students to visualize the relationship between physical phenomena and digital representation. Labs involving ADC integration demonstrated the importance of selecting appropriate sampling

rates and ensuring stable voltage references, reflecting the theoretical underpinnings presented in class.

Communication protocols and interfacing standards were another major lecture theme, particularly the use of Universal Asynchronous Receiver/Transmitter (UART), Serial Peripheral Interface (SPI), and Inter-Integrated Circuit (I²C) communication. Labs focusing on UART interactions demonstrated the basics of serial data transmission and reception, where text output and internal temperature readings were sent to a terminal via UART communication. These activities highlighted key concepts such as baud rate configuration, framing errors, and buffer management. Additional system designs included PWM-based control of servo motors, reinforcing lecture topics covering timer modules, duty cycle adjustments, and period calculations necessary for precision control in embedded applications. Understanding timers, interrupts, and their relationship to PWM generation provided a clearer picture of how real-time systems maintain responsiveness while managing multiple concurrent tasks.

Later stages of the course introduced the concept of modular programming and system abstraction, both critical for managing complexity in larger embedded designs. Labs transitioned to using the Raspberry Pi Pico microcontroller with MicroPython, applying higher-level programming skills to simulate and control systems involving sensors, motors, and interactive displays. Lectures discussed the benefits and challenges of using real-time operating systems (RTOS) versus bare-metal programming, concepts of task scheduling, and techniques for effective debugging and testing. Labs required methodical, step-by-step validation of hardware and software components, emphasizing modular code structure, abstraction of hardware layers, and documentation of system behavior for ease of maintenance and future expansion.

Throughout the semester, a strong emphasis was placed on debugging techniques, both in software (using logical analysis and test-driven development) and hardware (using oscilloscopes, logic analyzers, and careful observation of GPIO behavior). Concepts like race conditions, interrupt latency, and hardware timing constraints were explored in lectures and observed in practice during laboratory experiments. The integration of multiple hardware modules, synchronization of analog and digital signals, and proper management of user interfaces all mirrored real-world embedded system challenges, preparing students for more complex projects beyond the course.

Together, the lectures and labs provided a progressive and cohesive understanding of embedded systems, combining detailed theoretical knowledge with extensive practical application. Mastery of microcontroller programming, hardware interfacing, signal processing, real-time communication, and modular design principles emerged as central themes, equipping students with a strong foundation for further study or professional work in the field of embedded systems engineering.

III. Embedded System

The Smart Plant Monitoring System, that can be seen in Figure 1, is a fully integrated embedded solution based on the Texas Instruments MSP430FR5994 LaunchPad, specifically engineered to track key environmental parameters essential for

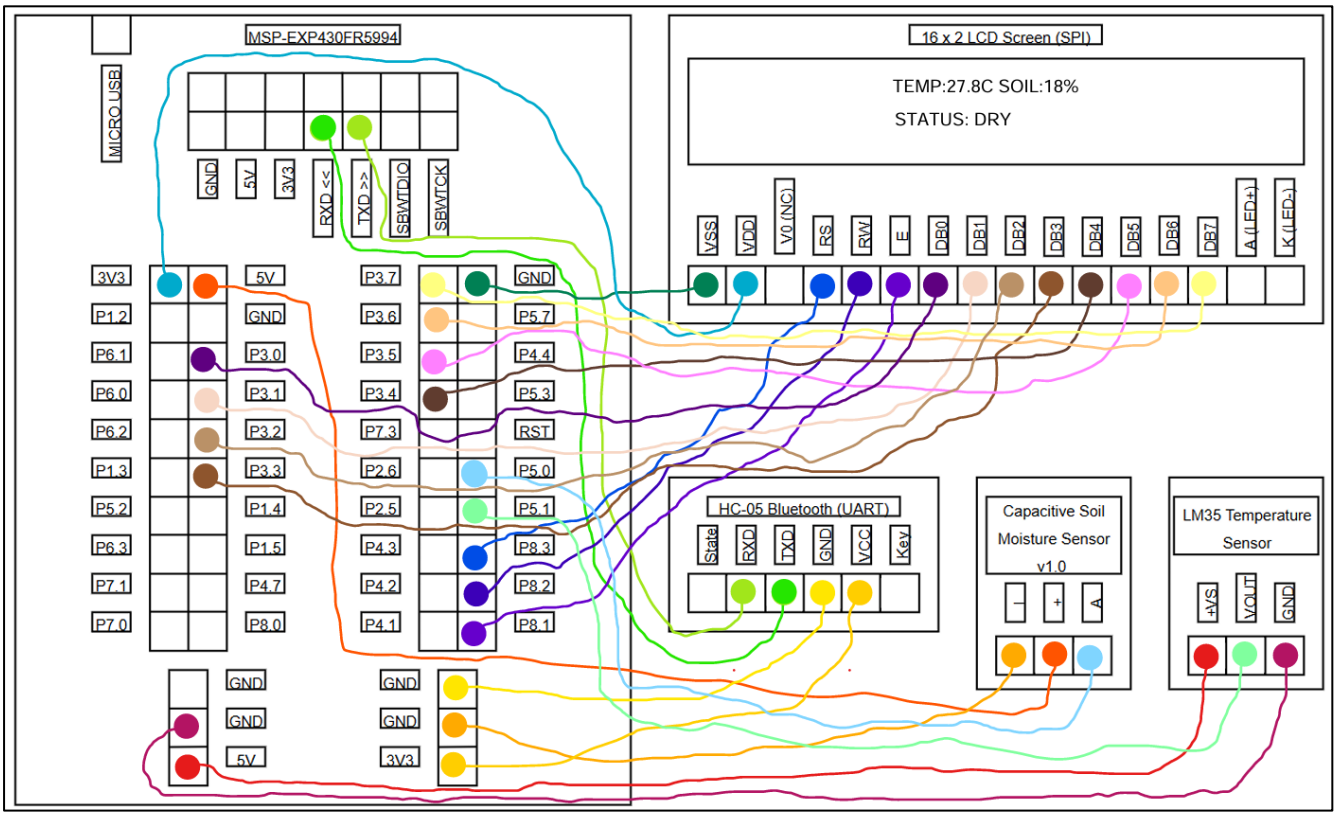

Fig. 1. System Block diagram

healthy plant growth: soil moisture content and ambient temperature. The core of the system leverages the MSP430's ultra-low-power FRAM technology, high-precision ADCs, and flexible peripheral configurations to provide continuous, accurate monitoring while consuming minimal energy. This platform is ideally suited for applications where battery life, wireless communication, and reliability are critical considerations. The physical system diagram can be seen in Figure 2.

At the sensor interface level, two analog input devices are deployed. The capacitive soil moisture sensor outputs a voltage inversely proportional to the dryness of the soil, and is physically connected to analog input channel A0 (pin P5.0) on the microcontroller. The LM35 analog temperature sensor provides an output voltage linearly proportional to temperature, typically 10 mV per degree Celsius, and is connected to analog channel A1 (pin P5.1). Both sensors are powered by the LaunchPad's 5V supply rail and grounded to the system common, ensuring consistent and stable readings. Proper attention is given to analog signal integrity; short wire runs and, if necessary, decoupling capacitors are employed to reduce noise susceptibility and improve ADC accuracy.

For local user feedback, a 16x2 character Liquid Crystal Display (LCD) module is used. The LCD is interfaced with port 3 and 8 GPIO pins of the MSP430, specifically mapped to maintain clear organization in code and to avoid conflict with other onboard peripherals. The Register Select (RS), RW, and Enable (E), lines, along with data lines D1 through D7, are individually connected to microcontroller output pins configured for digital output mode. Initialization routines include setting the LCD function mode, enabling two display lines, and issuing periodic display refresh commands to prevent screen burn-in.

Wireless communication functionality is provided through an HC-05 Bluetooth module configured in slave mode. The module's TXD and RXD lines are interfaced with the MSP430's eUSCI_A0 UART RX and TX lines, respectively, operating at a standard baud rate of 9600 bps with an 8N1 data format to ensure compatibility with conventional serial terminal applications. The Bluetooth link allows environmental data to be transmitted to smartphones, tablets, or PCs running generic serial terminal software, expanding monitoring capabilities beyond immediate physical proximity to the system.

ADC conversion timing is optimized to balance speed and energy consumption, and the system periodically samples the soil moisture and temperature sensors every few seconds to track gradual environmental changes without unnecessary energy expenditure. To translate raw ADC readings into usable measurements, the temperature sensor output is scaled, while the soil moisture sensor's ADC reading is mapped to a 0-100% moisture level using a calibrated dry/wet voltage range determined during system setup.

The control firmware running on the MSP430 includes a series of well-structured tasks: initialization of all peripherals, periodic sensor data acquisition, numerical conversion of sensor outputs, user interface updates via the LCD, and wireless packet transmission through Bluetooth. A simple state machine governs the system's main loop, ensuring that sensor sampling, display refreshing, and Bluetooth transmissions occur in a synchronized and efficient manner. The firmware is modular, with separate functions handling ADC reads, LCD updates, Bluetooth message formatting, and system status evaluation.

A key functional feature of the system is its decision-making algorithm for plant health evaluation. If the soil moisture percentage drops below a predefined threshold (e.g., 30%), the system flags a "DRY" warning, advising that the plant needs

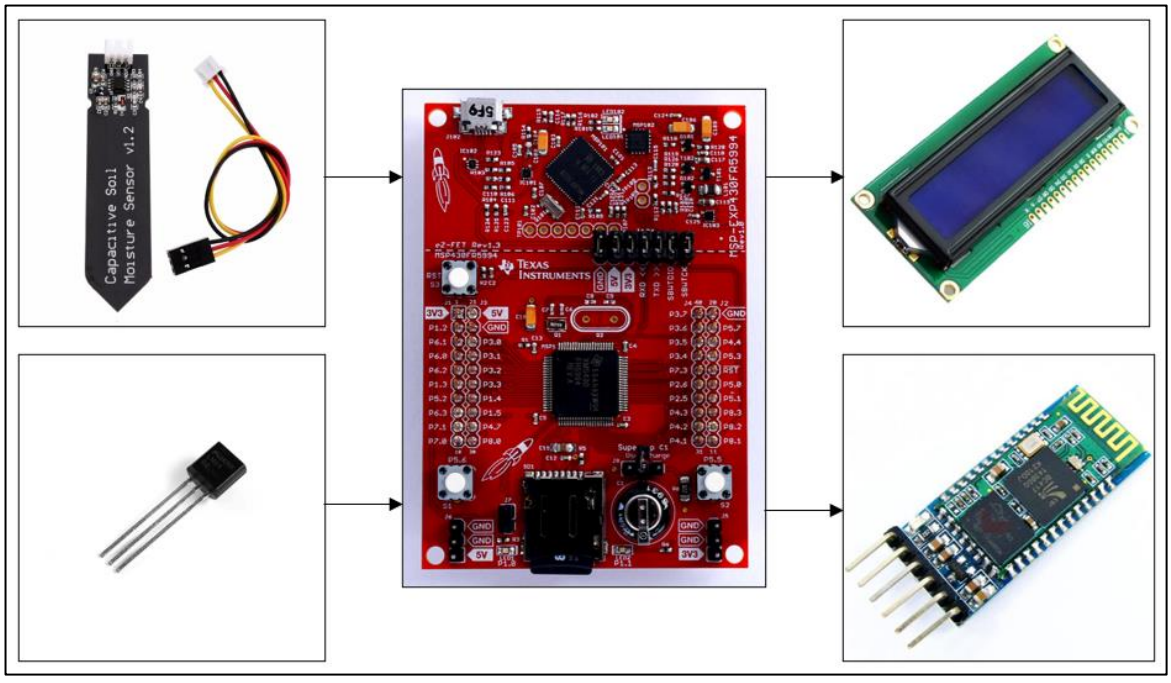

Fig. 2. Physical System Diagram

watering. If the soil moisture exceeds an upper threshold (e.g., 80%), it flags a "WET" condition, warning of potential overwatering risks. Otherwise, a "GOOD" status is displayed, indicating that the plant's hydration is within an optimal range. These evaluations are clearly presented on the LCD's second row and are simultaneously transmitted wirelessly, ensuring that users can react appropriately in a timely manner.

Power management considerations are integral to the design. The system is powered through the LaunchPad's micro-USB connection, which supplies 5V that is internally regulated to 3.3V for all components. The MSP430FR5994's low-power modes can be strategically employed between sensor readings to further reduce power consumption, making the system suitable for extended operation from battery packs if necessary. The system could be expanded in the future by integrating an energy harvesting module (such as a small solar panel) to support fully autonomous operation in outdoor environments.

Overall, the Smart Plant Monitoring System showcases the synergy between sensor interfacing, embedded processing, real-time user feedback, and wireless communication within a highly efficient, low-power architecture. Its clear modular structure, careful selection of components, and robust firmware make it a scalable platform, adaptable for more advanced agricultural automation tasks such as irrigation control, environmental logging, or integration into Internet of Things (IoT) ecosystems.

IV. Summary and Conclusion

This course offered a comprehensive introduction to embedded systems, providing students with both the theoretical foundations and practical skills required for designing and implementing embedded applications. Through the course's lectures and laboratory exercises, students developed proficiency in programming microcontrollers, interfacing with external sensors and peripherals, and implementing real-time systems. The Smart Plant Monitoring System example successfully demonstrates the theoretical utilization of a multitude of these skills by integrating analog sensors, displaying real-time data on an LCD, and transmitting data wirelessly via Bluetooth. The system's modular design and efficient power management further illustrated key concepts learned throughout the course. Overall, the course equipped students with a solid understanding of embedded system design principles, preparing them for more advanced studies or professional work in the field.

References

[1] Texas Instruments, *MSP430FR5994 Mixed-Signal Microcontroller*. Texas Instruments, 2016. [Online]. Available: https://www.ti.com/product/MSP430FR5994

[2] IEEE, *IEEE Editorial Style Manual*. IEEE, 2022. [Online]. Available: https://journals.ieeeauthorcenter.ieee.org/create-your-ieee-article/use-authoring-tools-and-templates/ieee-article-templates/

[3] D. E. Comer, *Essentials of Computer Architecture*, 2nd ed. Boca Raton, FL, USA: CRC Press, 2017

[4] M. A. Mazidi, R. D. McKinlay, and D. Causey, *MSP430 Microcontroller Basics*. Amsterdam, The Netherlands: Elsevier, 2008.

[5] B. Hur, *Learning Embedded Systems with MSP430 FRAM Microcontrollers*, 2nd ed., 2023.