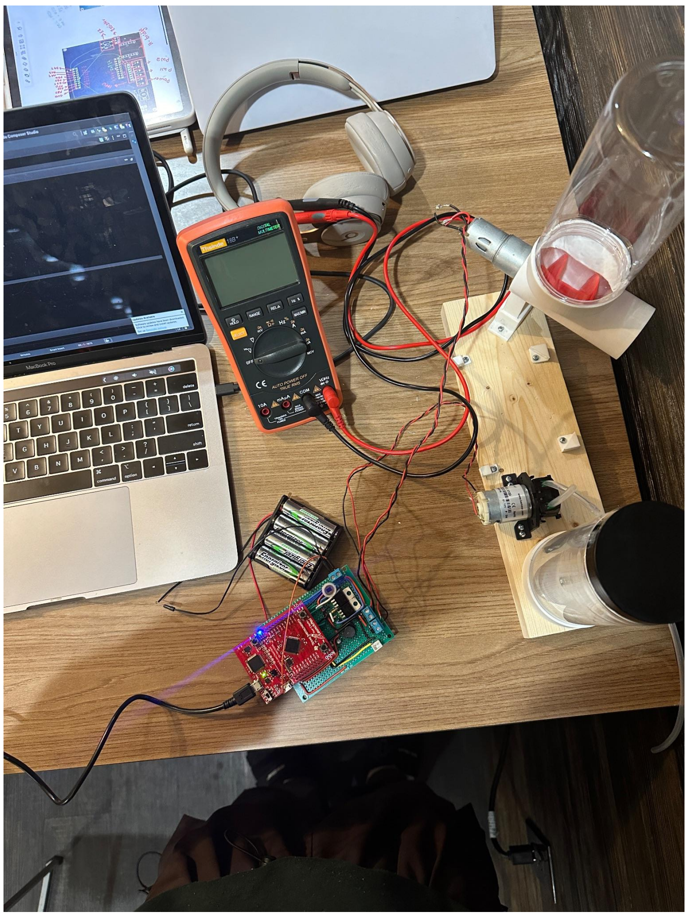
Lab Report: Embedded Systems for Weekend Pet Feeder

Abdulmalik Ajisegiri

December 5, 2023



Contents

1 Introduction 3

2 Hardware Architecture 3

3 Software Capabilities 3

4 Code Overview 3

4.1 Main 3

4.2 User Command Handling 4

4.3 Sensing & Dispensing 4

5 Detailed Code Analysis 4

5.1 initHw() 4

5.2 initUart0() 4

5.3 initHm() 4

5.4 initEeprom() 4

5.5 parseFields() 4

5.6 isCommand() 5

5.7 PWM Initialization (initpwm()) 5

5.8 Main Function (main()) 5

5.9 RTC Initialization and Interrupts 5

5.10 Hibernation Module Initialization 5

5.11 EEPROM Functions 5

5.12 System Initialization 5

5.13 Fill Mode, Alert, and Schedule Commands 5

6 Observations 5

7 Design and Implementation 6

7.1 Feeder Structure and Components 6

7.2 Water Delivery System 6

7.3 Food Dispensing Mechanism 6

7.4 Electronics Integration 6

7.5 Safety Considerations 6

7.6 User Interface and Control 6

7.7 Power Management 6

7.8 Conclusion on Design 6

8 Enhancements 7

9 Conclusion 7

1 Introduction

This document presents an embedded systems project showcasing an automated weekend pet feeder. The system incorporates features such as water level sensing, scheduled food/water dispensing, user configuration, and low resource alerts, providing a reliable solution for pet owners during weekends.

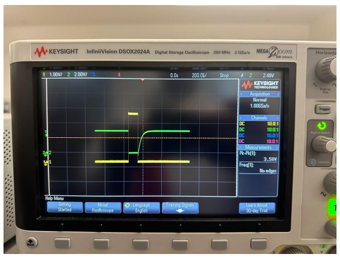
2 Hardware Architecture

The hardware comprises commercial off-the-shelf components and 3D printed parts, including a wa-ter bottle, food container, pet dish, delivery tube, auger, base, capacitive sensor, motors, MOSFETs, microcontroller, and peripherals (e.g., PIR, transducer, EEPROM). Safety measures, such as food-safe containers and sealed prints, are implemented. The choice of materials and integrated sensors allows for precise control of food and water resources.

3 Software Capabilities

The software demonstrates sophisticated features:

- Water Level Detection: Utilizes a capacitive sensor circuit interfaced to an analog comparator and timer modules for real-time water volume measurement.



- Scheduled Feeding: The TM4C123 MCU allows pre-programming and storing feed amounts and times in EEPROM to dispense food per a defined schedule.
- User Interface: A UART peripheral enables user configuration via commands for setting the current time, programming feeds, calibrating water volume, toggling motion-activated refills, and low resource alerts.
- Input Validation: Ensures user-supplied data is parsed into fields for validity before execution.
- Persistent Data: Feed schedules are saved in internal EEPROM to retain the programmed schedule.
- Precision Control: PWM and MOSFET controlled motors dispense measured amounts of food and water.

The code exhibits modular organization, hardware abstraction, and expert handling of the MCU's peripherals. Custom commands interface with components to enact system behaviors like scheduling feeds, sensing water, and alerting users.

4 Code Overview

4.1 Main

The main routine handles:

- Hardware initialization
- UART, PWM, EEPROM, and peripheral setup
- Parsing user commands from the UART RX buffer
- Calling appropriate handlers for each command

The main while loop polls the UART, ready to receive and parse user commands.

4.2 User Command Handling

Key user-facing functionality is exposed through UART commands. Custom handlers process and execute each command.

Example Feed Scheduling Command Flow:

1. User enters event details over UART
2. Input fields are parsed and validated
3. Extracted feed data gets written to the next available EEPROM slot
4. UART message confirms the written schedule

Similar flows handle time setting, water level alerts, reviewing schedules, etc. Invalid commands print error messages.

4.3 Sensing & Dispensing

MCU peripherals like ADC, PWM, Timers, and GPIO allow interfacing with attached hardware to enact system functions:

- ADC comparator triggers on capacitive sensor to measure water level
- PWM signals control motors for precise dispensing
- Timers enable timed events and sensor sampling

5 Detailed Code Analysis

5.1 initHw()

The initHw() function initializes various hardware components, including GPIO, analog comparator, PWM, and others. It ensures the proper configuration of the microcontroller peripherals required for the pet feeder system.

5.2 initUart0()

The initUart0() routine configures the UART0 peripheral for serial communication. It sets registers such as UART clock, baud rate, and data length, facilitating communication with external devices or a user interface.

5.3 initHm()

The initHm() function initializes the Hibernation Module, utilizing the real-time clock (RTC) in calendar mode for scheduling events and timer interrupts. This module enables efficient power management for scheduled tasks.

5.4 initEeprom()

The initEeprom() function initializes the EEPROM controller, ensuring reliable read/write operations for non-volatile storage of schedule data. It is a crucial part of the system for persistently storing user-defined feeding schedules.

5.5 parseFields()

The parseFields() function plays a key role in handling user input. It parses raw ASCII user input into discrete data fields for further processing. It identifies alphanumeric and numeric fields, allowing the system to extract command arguments accurately.

5.6 isCommand()

The isCommand() function checks if the parsed input corresponds to an expected command. It validates that the required minimum arguments are provided before executing the command. This function enables the addition of new UART commands with ease.

5.7 PWM Initialization (initpwm())

The initpwm() function initializes the PWM module, configuring GPIO pins and PWM parameters for controlling the pump and auger. This is critical for precise dispensing and control of feeding mechanisms.

5.8 Main Function (main())

The main() function orchestrates the overall functionality of the pet feeder system. It initializes hard-ware components, sets up UART communication, and enters an infinite loop where user commands are processed. The system responds to commands related to time scheduling, water level sensing, PWM control, and user interface through UART.

5.9 RTC Initialization and Interrupts

The code includes functions (enableTimerMode(), timer(), WideTimer1Isr(), and comparator0Isr()) related to RTC initialization and handling interrupts for time measurement and PWM control.

5.10 Hibernation Module Initialization

The initHm() function initializes the Hibernation Module for scheduling events and timer interrupts. It demonstrates efficient power management in the system.

5.11 EEPROM Functions

The functions (initEeprom(), writeEeprom(), and readEeprom()) manage EEPROM operations, cru-cial for storing and retrieving persistent schedule data.

5.12 System Initialization

The initHw() function ensures the initialization of various hardware components, providing a robust foundation for the proper functioning of the pet feeder system.

5.13 Fill Mode, Alert, and Schedule Commands

The main loop in the main() function processes user commands related to setting the time, scheduling events, displaying the schedule, deleting events, setting the fill mode, and managing low-water alerts. This demonstrates the user interface and control aspects of the pet feeder system.

Custom UART library displays volume measurements

. The parser separates user volume commands to activate pumps appropriately.

6 Observations

The system architecture is well-considered from a technical and user perspective:

- Careful coding practices aid readability, troubleshooting, and extensibility.
- Full utilization of internal resources illustrates strong embedded systems abilities.
- User configurability to tailor operation to pets' needs increases real-world viability.

With well-encapsulated functionality and robust features, the system forms an ideal baseline model to evolve capabilities like feed customization and automated self-correction.

7 Design and Implementation

7.1 Feeder Structure and Components

The feeder's physical structure is built on a 12" x 3 1⁄2" x 3⁄4" whitewood base, providing stability and durability. The 3D-printed components, including the feeder body (kibble.stl), feeder auger (auger.stl), and feeder bottom mount (stand mount.stl), form the core of the feeding mechanism. These parts are designed to securely hold the 16oz plastic jars for water and kibble, ensuring a safe and hygienic environment for pet consumption.

7.2 Water Delivery System

The water delivery system employs a DC gearhead motor connected to a peristaltic pump. This config-uration allows precise control over water dispensing. The 300mm x 5mm OD x 3mm ID silicone tubing serves as the conduit for water delivery. Barbed couplers are used to connect the tubing to the water bottle and the filler bracket, creating a sealed and reliable water supply system.

7.3 Food Dispensing Mechanism

The feeder utilizes a 3D-printed auger, driven by a motor, to dispense predetermined amounts of kibble. The auger's design ensures efficient and controlled dispensing, preventing overfeeding or clogging. A jar holder and nozzle bracket securely position the kibble container and control the direction of food dispensing.

7.4 Electronics Integration

The electronics integration involves interfacing various sensors and actuators with the TM4C123GXL microcontroller board. Copper strips are used for the capacitive water level sensor, ensuring accurate and reliable water level detection. The passive IR sensor (PIR) detects the presence of a pet, triggering actions such as water freshening.

7.5 Safety Considerations

Given that the feeder involves food and water, safety considerations are paramount. The 3D-printed parts are designed with food-safe materials, and precautions are taken to seal the PLA material to prevent bacterial growth. Attention is also given to prevent any potential choking hazards for pets.

7.6 User Interface and Control

The user interacts with the feeder through UART commands, facilitated by the TM4C123GXL board. The feeder supports commands for setting the time, scheduling feeding events, displaying the schedule, deleting events, entering fill mode, and managing low-water alerts. The UART interface provides a straightforward and accessible means for users to configure and monitor the pet feeder.

7.7 Power Management

The Hibernation Module (HM) is utilized for efficient power management. By integrating the real-time clock (RTC) in calendar mode, the system can schedule events and utilize timer interrupts while minimizing power consumption during idle periods. This ensures optimal use of power resources and extends the operational lifespan of the device.

7.8 Conclusion on Design

The design of the Weekend Pet Feeder integrates mechanical, electronic, and software components seam-lessly. The use of 3D-printed parts, reliable sensors, and a microcontroller with efficient power manage-ment highlights the comprehensive approach taken in creating a functional and user-friendly embedded system for pet care. The feeder not only meets its primary objectives but also incorporates safety features and a robust user interface, contributing to a successful proof of concept.

8 Enhancements

Future enhancements include:

- Enclosure safety and durability testing
- Automated monitoring and correction of food/water amounts
- Wireless connectivity and app integration
- Computer vision for food level detection

9 Conclusion

The code demonstrates efficient leveraging of the TM4C123GXL peripherals and custom UART rou-tines to build the key functionalities of a weekend pet feeder prototype. The integration of water/food distribution, timed scheduling, configurable alerts, and peripheral manipulation illustrates a compre-hensive, real-world solution. Further testing and refinements are recommended; however, the current implementation indicates a promising prototype with discernible embedded systems aptitude.