Final Project Proposal

By: **Caleb Probst** and **Leopold Tejkowski**

Spring 2023 ECE 381 Microcontrollers

Executive Summary

Our final project proposal is an idea for an automatic pet feeder. The reason for this is quite simple. Caleb's cat is quite fat and gets very angry when not fed on time. For proof of necessity, here is a picture of said cat on a doughnut box, shown by *Figure 1*.

Figure 1: Cat on a Doughnut Box

Now, you might say "He doesn't look that fat". Well, that was about 4 years ago. Here is a recent picture, not on a doughnut box, but fat. This is shown by *Figure 2*.

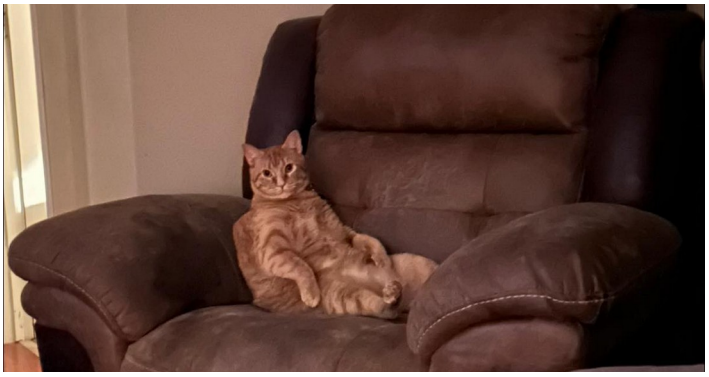

Figure 2: Fat Cat

Hardware Description

As far as hardware goes, the vast majority of hardware will be chosen by us, as well as the circuitry. Here is a general materials list of what we will need for the project, as well as the price along with it, and the link where we got it from.

Item	Qty.	Price/Unit	Link
PSoC4 4200M Microcontroller	1	$24.94	Digikey
Bipolar NEMA17 Stepper Motor	1	$0.00	Had Available
20x4 Hitachi I2C 2004 LCD	1	$10.99	Amazon
Push Button	1	$0.00	Had Available
Mechanical Encoder	1	$0.00	Had Available
L293DNE Bipolar Stepper Motor Driver	1	$0.00	Had Available
PCF8563 I2C Real-time Clock Unit	1	$7.29	Amazon
LED’s of Various Colors	4	$0.00	Had Available
12V to 5V Buck Converter	1	$14.99	Amazon
12V 6A Power Supply	1	$14.99	Amazon
P2N2222A Transistor	2	$0.00	Had Available
Various 3D Prints	7	$19.99	Cost of Filament
“Nacho Average Pet Feeder” PCB v 1.3.5	1	$17.14	JLCPCB
1x6 2.54mm Pitch Dupont Connector	1	$0.00	Had Available
1x4 2.54mm Pitch Dupont Connector	6	$0.00	Had Available
1x3 2.54mm Pitch Dupont Connector	1	$0.00	Had Available

1x2 2.54mm Pitch Dupont Connector	8	$0.00	Had Available
1x1 2.54mm Pitch Dupont Connector	2	$0.00	Had Available
100nF Non-Polarized Capacitor	3	$0.00	Had Available
1µF Non-Polarized Capacitor	1	$0.00	Had Available
10µF Polarized Capacitor	2	$0.00	Had Available
470Ω Resistor	6	$0.00	Had Available
1KΩ Resistor	4	$0.00	Had Available
Total Cost	**$110.33**		

The circuitry is designed by us in KiCad, and has been finished and ordered at the time of this proposal writing. The files for these can be found on GitHub through Brianna (or if you (Dr. York) would like access, please ask). As well as the general circuit diagram, a custom PCB was also designed and bought.

As far as the hardware and 3D models, all parts besides the auger, are designed by us. All parts are 3D printed and proper precautions (such as abrasion against food) have been taken into account.

As far as PSoC modules go, we will be using the I2C bus from the PSoC, as well as using the I2C LCD module that is built into PSoC Creator. This heavily simplifies all the register writing we have to do in order to get the LCD up and running. As well as the LCD being running on the I2C bus, the RTC (Real-time clock) unit will also be running on the I2C bus and will require modifying the proper registers in order to ensure that it is in an operational state every time it turns on. The RTC unit that we are using also has a battery so that it can save it's time when powered off, which was the reason for buying the module. The UART is not quite known what we will use it for, but will likely be for more terminal based use of editing times and amount, and for general debugging. We will also be using a control register for the motor, as we had done in Lab 4 with motors. As well as that, we are also using concepts from Lab 3, as the speaker and PWM signals will also be used. Below is the general theorized Top Design for the project, shown by *Figure 3*.

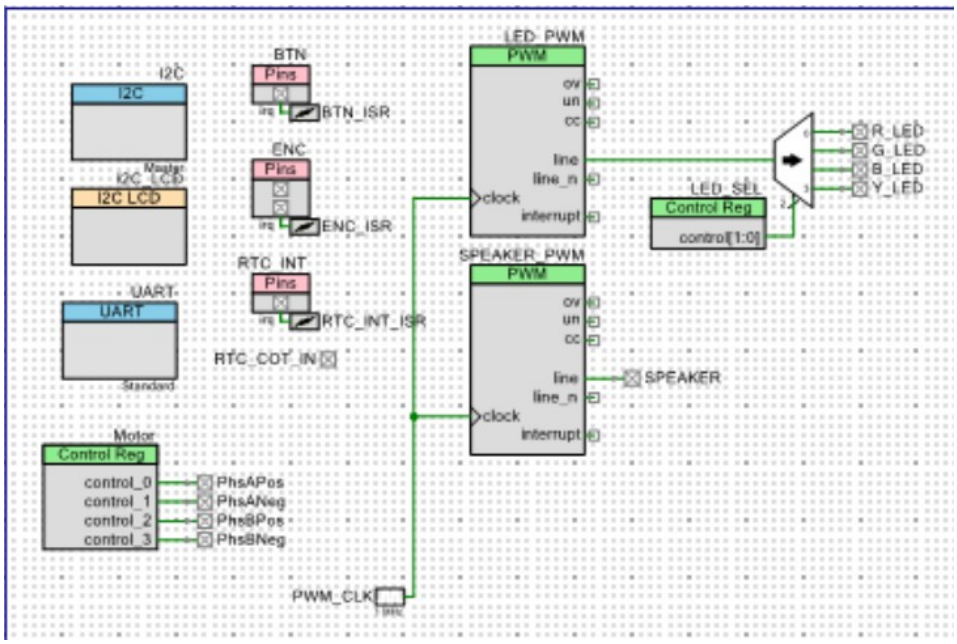

Figure 3: Theorized Top Design

Algorithm Outline

The code will have 4 main sections: standby, settings, settings options, and feed time.

Standby will simply have the LCD on, show the current time and date, along with the time and date of the next feed. This is so the user can see certain information that may be useful about the feeder itself. This also takes advantage of the RTC unit that we will be able to use without power to the PSoC.

The settings section will be once the user presses the button, and will use the encoder to select settings that will effect the pet feeder. These settings include setting the clock time (in the case of DST), setting feed times, setting amount of food per feed, and turning on or off the tone and lights.

Each one of these settings will have their own prompts and options. For the setting the clock time, you will be able to set the hour, minute, day, month, and year through various button and encoder turns. This will then be directly sent to the RTC unit, which will then go back to standby. For setting feed times, the user will be able to set the number of feeds, up to 10 (if user tries to go over 10, it will insult the cat by calling it fat). Once this is set, it will cycle through the number of feeds and set the time of day that that feed will occur. Then, this will be sent to the FRAM and RTC unit to set interrupt settings and have a backup of these settings. For setting the amount of food, it will use a horizontal bar slider and motor to show how much food is given for a certain amount of steps. This will then be confirmed and then sent to FRAM for backup of settings. For turning on and off the tone and lights, it will simply either turn on or off the lights or tone, and store to FRAM for backup.

The last setting is feed time. This is the main operation of the pet feeder, and will simply turn the motor a certain amount of steps, blink the LED’s and play a tone, then return to standby.

Overall, the code is very simple, and interacts with the hardware only on interrupts.

Timeline

For our timeline, this project started in February 9th, when we came up with the idea. From then to April, we worked on the 3D modeling, gathering parts, 3D printing, and designing a PCB for the project. So, from April 2nd, to the final day, here is our timeline.

April 2nd to April 8th (Week 12) – Start PSoC project, start displaying text on LCD, figure out RTC unit.

April 9th – April 15th (Week 13) – Start using RTC interrupts, use encoder and button to work on settings.

April 16th – April 22nd (Week 14) – Get motor running, assemble physical pieces together, get feed operation working.

April 23rd – April 29th (Week 15) – Get PWM's for LED's and Speaker working, put everything together.

April 30th – Final Day (Week 16) – Finalization, lab report, presentation.

Requirements

- ❒ Uses a 4x20 LCD properly
- ❒ Uses a set of interrupts for button, encoder, and RTC unit
- ❒ Gets time correctly
- ❒ Displays current time and date
- ❒ Displays time and date of next feed
- ❒ Settings:
  - ❒ Correctly sets clock time on RTC
  - ❒ Correctly sets feed time(s)
  - ❒ Correctly sets amount of food
  - ❒ Correctly turns on/off the LED's
  - ❒ Correctly turns on/off the speakers
- ❒ Uses stepper motor correctly
- ❒ Stores values in FRAM for backup in case of power off
- ❒ Is able to feed pet

Topics From Class Used

- ❒ GPIO – **Lab 01**
- ❒ Interrupts – **Lab 02**
- ❒ Encoders and Button**s – Lab 02**
- ❒ PWM – **Lab 03**
- ❒ Stepper Motors – **Lab 04**
- ❒ I2C – **Lab 05**