Department of Computer Science
Technical University of Cluj-Napoca

Chef Detector Game

Design with Microprocessors

Laboratory project

Name: Mera Mihai Mircea
Group: 30434
Email: mera_mihai2001@yahoo.com

Teaching Assistant: Ana Rednic
anarednic@gmail.com

Contents

Chapter 1

Abstract

The project is implemented as a simple cooking game. The idea was born based on my personal passion regarding this hobby. At first, the aim of the project was a food detector, but the resources needed were beyond my possibilities.
The project starts from a simple kitchen scale that displays the weight of a measured product. Trying to orient the final product to something more sophisticated, but playful, I thought about 'teaching' the scale what ingredient it should expect. So, it came to mind to send the recipe to the scale, but without the user knowing the exact quantity for each ingredient. For a better representation, I chose to add LEDs and a buzzer that could communicate to the player the state of the game. Moreover, buttons with different purposes were added.

Chapter 2

Implementation

The whole implementation of the project is a puzzle, where the pieces are the known or learned possibilities for each component, along with external factors manipulation. In the next sections, implementation and problems regarding it will be analyzed thoroughly.
As mentioned, the project is implemented as a kitchen scale, but more intelligent. The player sends a recipe to the 'toy', that follows the format: each instruction on a separate line; instructions can be of three types:

- ADD instruction - it starts with one or two '@' symbols, depending on the name of the ingredient and contains: the name of the ingredient, the quantity and the measurement unit, the points received for adding it and an approximate time for preparing the ingredient
- MIX instruction - it starts with a '∼' and holds the approximate time and the points received
- COOK instruction - it starts with a '!' and tells the time for the food to be cooked

After receiving the recipe, the user presses the start button. The instructions are displayed one by one. At each step, the LED shows the loading of the ingredient. Along the way, the buzzer emits specific sounds, depending on the action that was done. (Special thanks to my friend that is a singer and pianist, Moldovan Victor, for composing the sounds for the project.) When the recipe is done, the score is computed and displayed on the LCD screen.

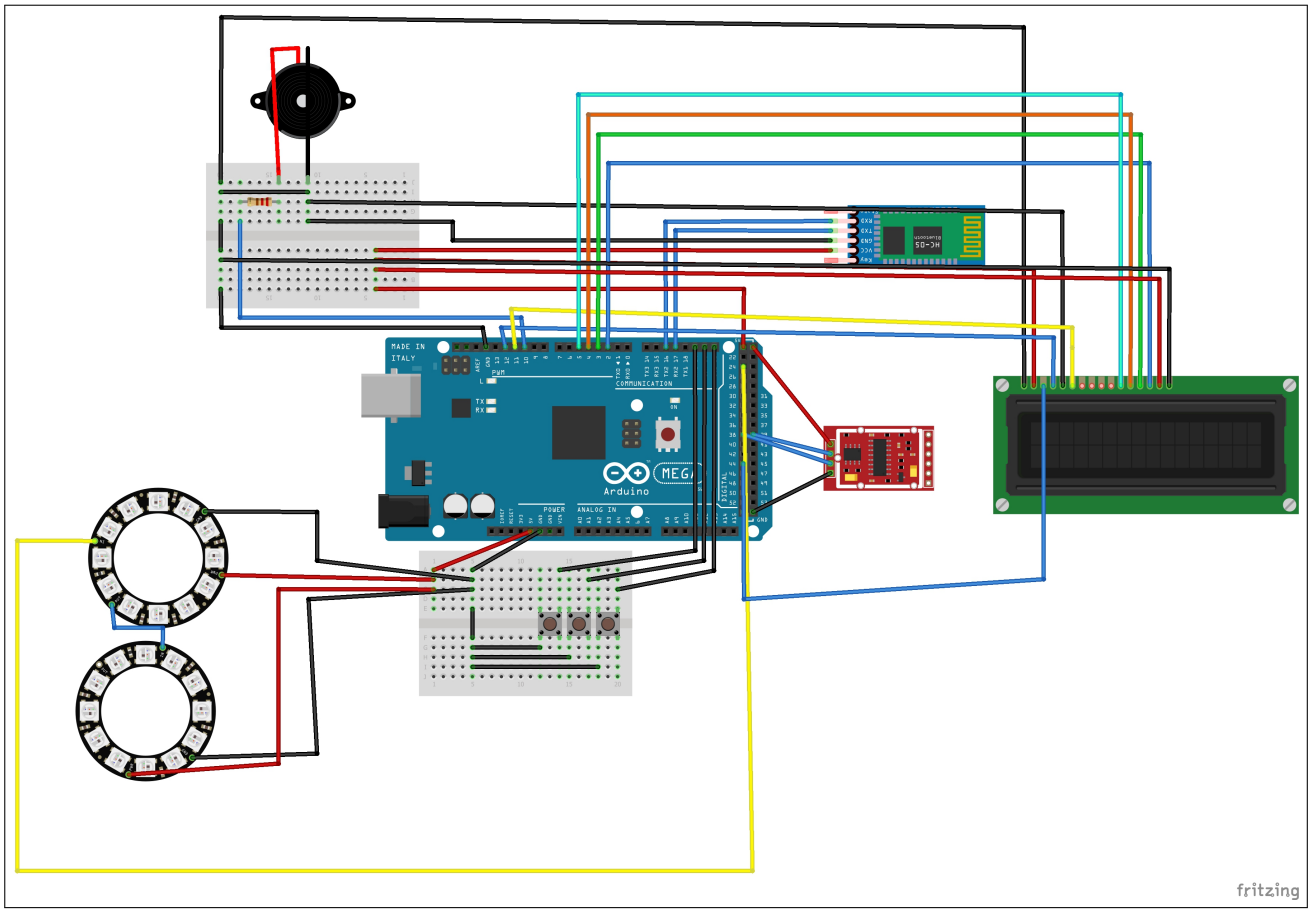
2.1 Components

The hardware part was easy to establish. Below, there is a list of components used:

- Arduino Mega 2560 board
- 5kg load cell - this type of load cell was opted for by approximating the needs of the project
- HX711 ADC module - for ensuring the connection between the load cell and the Arduino board
- HC-05 Bluetooth module - for sending the recipe to the board
- 2 RGB LED rings - these were used as loading rings, one for the current ingredient and one for the whole recipe
- LCD - for displaying the instructions

- 3 push buttons - the buttons are used for start - after sending the recipe the user presses a button so that the game begins; skip - if an ingredient can be omitted or the player does not want to add the whole quantity, it can be skipped; stop - the game can be stopped if the user does not want to play anymore
- passive buzzer - for a realistic representation of a game, the application produces some sounds at start, at the end, or when the user made a mistake
- 100Ω resistance - connected in series with the passive buzzer
- 2 small breadboards - especially for GND and VCC connections from the board to the components and for the buttons
- wires - female/female, male/female and male/male connection wires were used

The next diagram contains a schematic of the circuit (the 5k load cell is missing, because it was not available in the Fritzing library - it should be connected to the first four pins of the HX711 module).



2.2 Code

As it can be seen in the diagram, there are quite a few components, so in this section, I will go through each of them and briefly describe their way of working.

Load Cell&HX711 module. For using these components, a library, "HX711.h", had to be imported in the project. Using this library, the scale was declared by mentioning the

connection of the data and clock pin (38 and 39, respectively). Before use, the scale was calibrated using a ready-to-use program. Inspiration for implementing the scale was taken from https://www.instructables.com/How-to-Interface-With-5kg-Balance-Module-or-Load-C/. Calibration was a bit hard to do at first, and using the scale implied precaution, since the wires are sensitive.

RGB LED rings. The rings are represented as an array of LEDs and the coloring was done using the Adafruit Neopixel library. For imitating the loading action, I created a function that computes a percentage and applies color to the rings depending on the value of the percentage. The two rings are connected in series by using the output port of the first and input port of the second. They are connected to the digital pin 24.

Passive buzzer. The buzzer is connected to the digital pin 10, which is a PWM pin. This was essential for generating sounds, by using the tone() function. The songs were hard coded, along with the duration of the notes, the tempo and the size.

LCD screen. The LCD was one of the trickiest components, since it could not be connected directly into the board, due to the inconsistency in the order of the pins. It is also hard to connect, because of using many GND and VCC pins. It took some research time to find the correct schematic for the LCD I used, and also to understand the meaning of the pins. The contrast pin (V0) is connected to digital pin 44, this is used for adjusting the contrast. Only four of the data pins are used (transmission on four bits), D4-D7, wired to 2-5 digital pins. Other control pins, RS and EN, are wired to digital pins 11 and 12.

HC-05 Bluetooth module. Working with the Bluetooth module was harder than expected. The first issue was sending a whole file to the board. Initially, I wanted to create a small Python app to send the file line by line by creating a connection to the application. However, this could not be achieved, because of security protocols. Then, I discovered that files could be sent through the Serial Bluetooth Terminal application. Still, some problems appeared in synchronization, but were solved by setting the sending mode to 'Line' and the pause between lines to 400 ms. Processing the input was and remains the main issue of the program. The string manipulation methods could be deprecated since the IDE version is not the most recent or the text may be wrongly parsed, but a certain thing is the fact that the instruction could not be extracted accordingly from the file. A structure was defined for remembering the instructions. The RX and TX pins of the module are connected to the TX2 and RX2 pins of the board, since RX1 is used by a button for its interrupt support. For this reason, Serial2 is used for the Bluetooth connection.

Control buttons. There are three buttons that have the purposes that I already mentioned. These buttons are connected to the digital pins 19, 20 and 21, which support interrupts. The attached interrupts are the functions responsible for starting the game, skipping an instruction and stopping the game.

Chapter 3

Testing

Testing was done along the way at each step, so that I was sure that each component works properly. Separately, it was kind of easy to test the components. The buzzer works properly for the used range of frequencies (190-600 Hz). The function that colors the rings needed some testing that was time consuming. The computation was checked multiple times, and for that, the LEDs were lighted one by one, to observe if the effect is the one that I wanted.
The Bluetooth module was the hardest to test. The first problem occurred when establishing a connection with the computer. The initial plan proposed that the computer sends the recipe to the module, but this idea rapidly crumbled to the ground, because I was not able to maintain a connection. Then, I thought about giving up on the module, but this was an important part of the project (and a bit expensive, too), so I could not just leave it. Finally, the existence of the application hit me, and this solved part of my Bluetooth issue.
Another aspect that bothered me was testing the LCD. Because of poor wiring, I had a misinterpretation of the output. I thought that the LCD is broken, but after inverting and changing some wires, the LCD worked properly. Before this, it took some testing to decide on a proper brightness, using the V0 pin.

Chapter 4

Conclusions

The project does not work properly, seeing that the parsing operation cannot be done. Although, the components, along with their pieces of code, seem to work fine. Another impediment is the fact that the load cell wiring broke, so further improvement could be done only after repairing this wire. Some improvements that the project admits (after making it work, of course) could be:

- a more friendly interface - the actual project is built in a carton box
- a database for recipes and a program that could automatically transform recipes in the required format
- an auto-calibration button (tare) - pressing could trigger the calibration to 0 of the scale
- keeping track of high scores
- better wiring and breadboard connections