Project report on

Electronic pong

Manzir Zaman, Scott Holt, Steven Blount, Jeff Keurian

ECE 511 Microprocessors-Fall 2014

TABLE OF CONTENTS

ABSTRACT

For this project, we were to create a product which would entertain the user. The group decided on adaption of a classic college game of beer pong and how to customize the game features for our project need; hence the electronic pong. In selecting our project goals the team wanted to create a project which would be challenging and to leverage the capabilities of MSP430 while at the same time being fun and entertaining to use. With four team members, it was important to derive four high level features which would comprise the project such that each person had a decent amount of responsibility. In the end, all the desired features for electronic pong were fully functional featuring an LCD display, LEDs, Buzzer, and Photo sensor.

1.1 Motivation

The motivation of the project was to create an electronic version of the classic pong game in order to digitally keep track of the score, and to further enhance the enjoyment of the game by providing music and colorful lights when balls are sunk and certain threshold scores are achieved

1.2 Solution

One of the key goals for the electronic pong project was to maximize the resources available in the MSP430. To do this, features were added which would utilize each available peripheral on the microcontroller. The game features an LCD display LEDs, Buzzer, and Photo sensor. To sense an object in a cup, in this case a ping pong ball, a photosensor is used to detect its presence, after which the sensor sends an input value to the MSP430. The MSP430 then signals the LCD and the buzzer to perform their particular tasks like signaling the LCD and the buzzer to change the score and play music, respectively.

1.3 Block Diagram

The components are implemented according to the block diagram below.

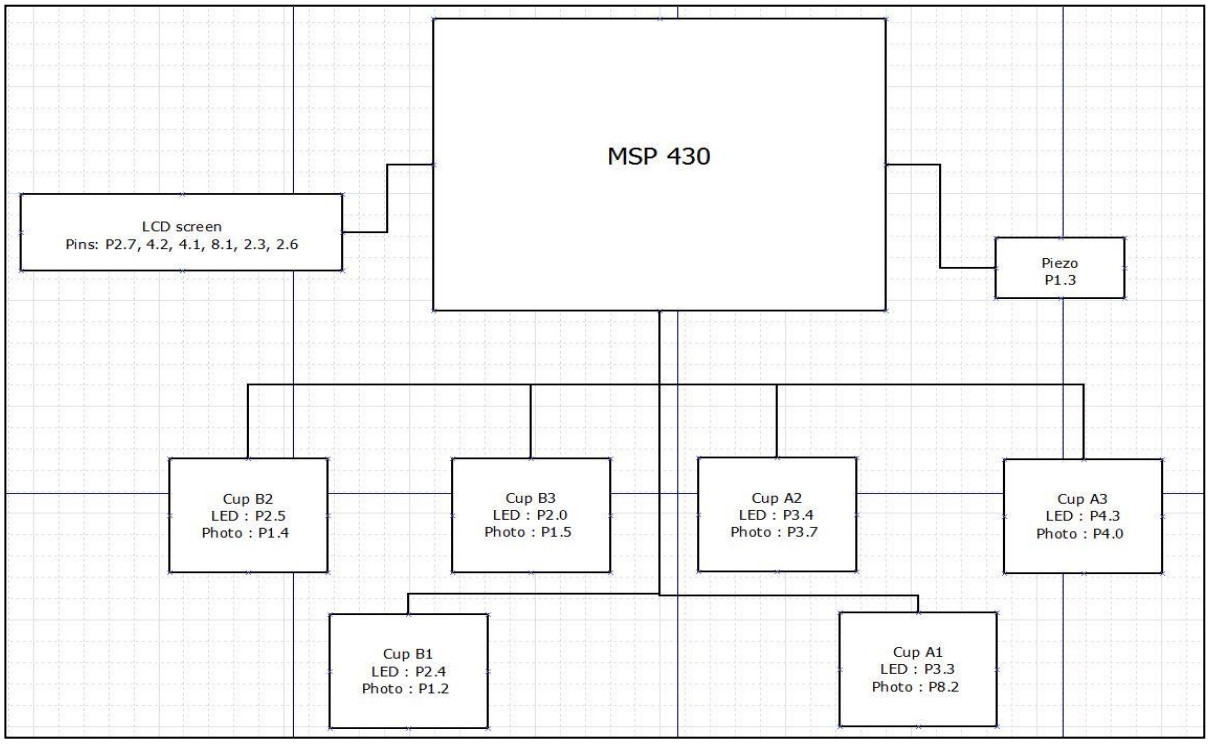


Figure 1: Block Diagram of Electronic pong

1.4 System Overview

This project consists of an electronic version of the classic “pong” game. The game is played by having opposing team members attempt to sink a ping pong ball into a number of cups aligned on a game table, as shown below:

Figure 2: Game Setup overview

When a ball is sunk into a cup, an LED is turned on, and the score is updated on an LCD. When the final desired score is reached, the LED congratulates the victorious team members, and one of a selection of songs is played.

An overview of the device is given in the figure below. As shown, a circuit board connects the five major electronic components of the device together. These are 1) LCD display 2) photosensor 3) LED’s 4) MSP430 5) buzzer. Each of these components will now be discussed in greater detail.

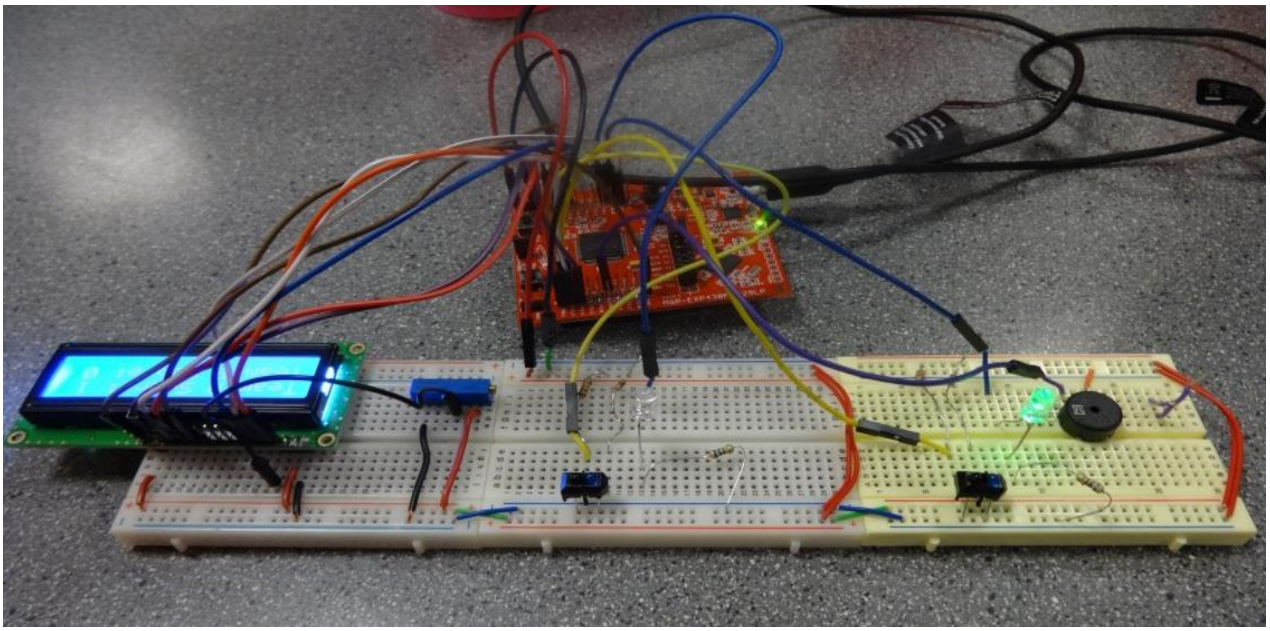


Figure 3: Devices connected to MSP 430

2. Components

2.1 MSP 430

The MSP430 is the microprocessor which contains the code that coordinates the functioning of the rest of the components. It basically operates in this project as a comparator by maintaining a constant voltage of 5V by looping through a cycle, until that cycle is disturbed by the event of having the ball trigger the photo sensor to create an input source. This input source is then compared to the 5V source cycling through the loop, and the difference indicates the presence of the sunken ball, causing the MSP430 to carry out further events, as described earlier.

2.2 LCD display

The LCD used is shown below:

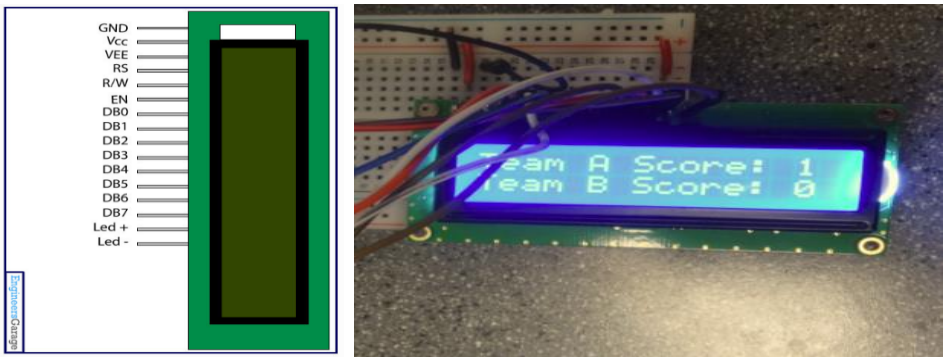


Figure 4: LCD display

The LCD has 16 pins available for use. It requires several connections to power and ground (for the actual characters and also for the backlight). The LCD can operate in either four or eight bit mode. Four bit mode was used for this project in order to use less IO pins. The pins used on the MSP430 are four GPIO pins (pins 4.1, 8.1, 2.3 ,2.6), the register pin (pin 2.7) and the enable pin (pin 4.2). An LCD library was found online and was used to drive the LCD. The configuration which drives the LCD is presented in software configuration.

2.3 LED's

As shown at right, two LEDs of 12,000 mcd brightness, with a 3.4V forward drop, are used to further indicate the presence/absence of a sunk ball. They are connected to the MSP430 via the dedicated pin.

The configuration used to trigger their illumination is discussed in software configuration.

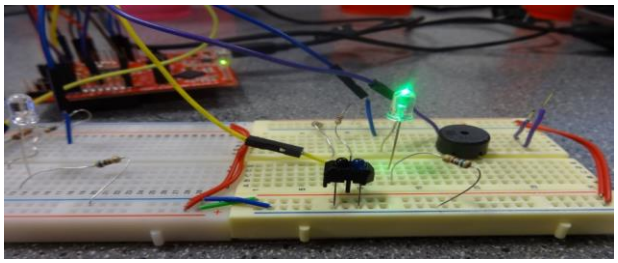


Figure 5: Single LED connected to the circuitry

2.4 Photo sensor

Six photosensors, one of which is illustrated in the figure above as the foremost member, are used to detect the presence of a ball. Their output is connected to dedicated pin numbers on the MSP430, and their operation, as described above with respect to that device, is to detect a reflected image of the ball off of the cup and send a voltage signal of 3V to the MSP430, which subsequently uses this value in comparison to a 5V benchmark to indicate the presence of a sunk ball.

Figure 6: Single photosensor

2.5 Piezo Buzzer

Piezo Buzzer Part Number: PKM13EPYH4002-B0 the cylindrical member on the right in the figure above is used to play music when a threshold score is reached, as described above. The code used to drive this device is provided in software configuration.

Figure 7: Piezo Buzzer

3. Software Configuration

The software for this project was responsible for handling the overall game logic, and interacting with the hardware components. To do this, the IDE Energia was used to update the firmware of the MSP430. Energia is a C++ wrapper around the standard C language used to program the device, and comes with very useful libraries. The Energia compiler is smart enough to optimize out the parts of the libraries not needed in the design.

3.1 HW-SW interaction

One primary task of the software is to interact with the hardware components. The 4 major hardware components are the LCD screen, piezo buzzer, LED's, and photo sensors. The LEDs and photo sensors were set up as simple general IO, with LED's being defined as output pins, and the photo sensors as input pins. The LCD screen requires 6 GIO pins for reset, enable, and 4 data pins. The piezo buzzer also requires one general IO pin, configured as an output. Finally, the two push buttons on the MSP430 are used as control, and configured as inputs

3.2 Energia Library calls

- Energia standard library:
- pinMode(int pin, int mode) - configures a pin as either INPUT, OUTPUT, or INPUT_PULLUP.
- digitalWrite(int pin, int value) - writes a specific value to the specified pin number
- digitalRead(int pin) - reads and returns the value on the input pin.
- delay(int ms) - delays the processor for the given number of milliseconds
- tone(int pin, int freq, int delay) - configures the Timer A with the appropriate PW for creating a tone used by the buzzer
- noTone(int delay) - clears the Timer A registers to silence the buzzer
- LiquidCrystal.h:
- setCursor(int col, int row) - moves the cursor to the specified row/column
- print(char* string) - prints the character array starting at the cursor position
- begin(int cols, int row) - initializes the lcd screen

3.3 Game logic

The game logic was trivial to write after all the hardware was configured correctly. Each team has a score, and whenever a cup is made, that team's score is incremented by one, and the buzzer plays a small beep. If a team is able to get a score of three, they win. However, if a team makes a cup that has already been made, the point doesn't count! To add some flair, each team can select a theme song so if they win, their theme will be played along with flashing lights.

A hard mode was also created, but never finished. In hard mode, if a user makes a cup that has already been made, a point is deducted from the team score, and the LED in that cup comes back on! This caused some issues with the original mode, and was ultimately scrapped for the final report.

4. Hardware Configuration

The hardware for the project was configured so that users could stand side-by-side while playing. This allows the game to be used in smaller spaces than a normal pong game could be played. The components were attached to the baseboard in a manner that allowed concealment of all electrical wiring to decrease the chance of failure.

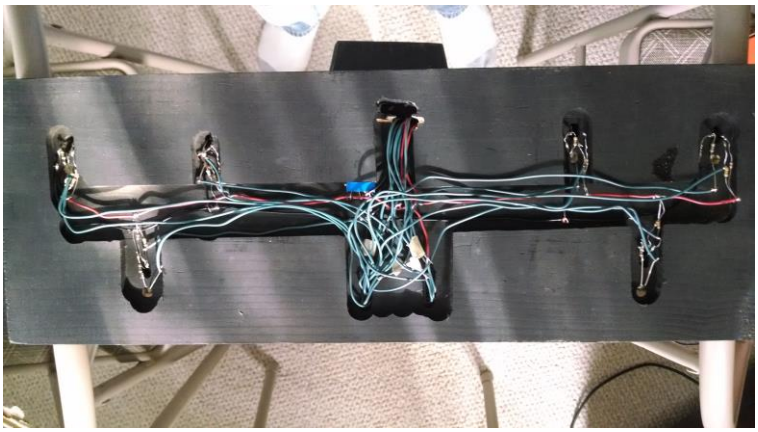


Figure 8: Wiring under the baseboard

Six cups in total (three per team) were used. Each cup had the bottom removed to allow the photo sensor and LED to be placed below. Small magnets were glued inside each cup and into the baseboard for mounting purposes. This feature also allowed for the removal of the cups for storage. For details about individual parts and their interfacing, parts list.

5. Results

We were able to successfully integrate the components to produce the desired result. The hardware we purchased, with the exception of some bad leads, was reliable and interfaced well with the programs we wrote. As the literature describes trouble with the sensitivity of the photo detectors, we had a few. This could be due the fact that our cups were not filled with fluid, the position of the ping pong ball inside the cup as well as the make and color of the ball. The software was also reliable and performed its function well, especially given the fact that it was written by multiple team members.

5.1 Lessons Learned

One thing that was learned was that it is important to order a few extra components, especially when they are low cost, in case one of them gets broken or lost. In the case of low cost items, like magnets, it is worth the extra money, and important to check the estimated shipping dates. It was also learned that it is a good idea to break the project into smaller components and test them individually. It was also learned that it is a good idea to experiment with different software packages when creating the code, as some perform better than others. Finally, a good literature review can be very helpful.

5.2 Achievements

The device worked as planned, and there were fewer problems encountered than we expected relative to other people who have built similar devices, as is documented in the literature. By having a prototype which worked at least adequately before the deadline approached, we were able to improve on its capability without worrying about it being non-functional near the end of the semester.

5.4 Conclusion

The flexibility of the MSP430, as well as the variety of software packages used to develop the code, were helpful in the development of the project. Its proper functioning can be attributed to these aspects, as well as the facts that a functioning prototype was developed reasonably early on, the project was broken into subparts, and we had a good working group.

6. Appendix

6.1 Work Breakdown

Jeff Keurian

Jeff was primarily responsible for the Piezo Buzzer and the integration of coding for each components, final testing and circuit breakdown.

Steven Blount

Steven was primarily responsible for the Photo sensor functionality. He also assisted with the capturing of accurate ping pong ball signal and helped integrate the various software components.

Manzir Zaman

Manzir was primarily responsible for the LED portion of the project, which encompassed both the hardware and software portions. He ensured the LED functionality, adaptation of the incoming signals and appropriate output for the desired scenarios on the circuit.

Scott Holt

Scott was primarily responsible for LCD module of the project that includes both hardware and software. He also helped with the final testing of the circuit and coding. He designed and fabricated the baseboard for the project.

6.2 Parts List

Component	Model	Quantity	Manufacturer
MSP 430 Launch Pad	MSP430G2553	1	Texas Instruments
Piezo Buzzer	PKM13EPYH4002-B0	1	Sparkfun
LCD Display	LCD-0033	1	NKC Electronics
LEDs	T1 ¾ 5mm, 12,000mcd brightness 3.4V forward drop	6	Sparkfun
Photo Sensor	TCRT 5000L	6	Sparkfun
Resistors	100 Ohm, 10K Ohm	6	Sparkfun
Power Supply	Portable Phone Charger	1	General Vendor
Breadboard	Generic	3	Sparkfun
Wood Plank	10 in. x 1 in. x 21 in	2	Hardware Store
Cups	Generic	6	Home Pantry

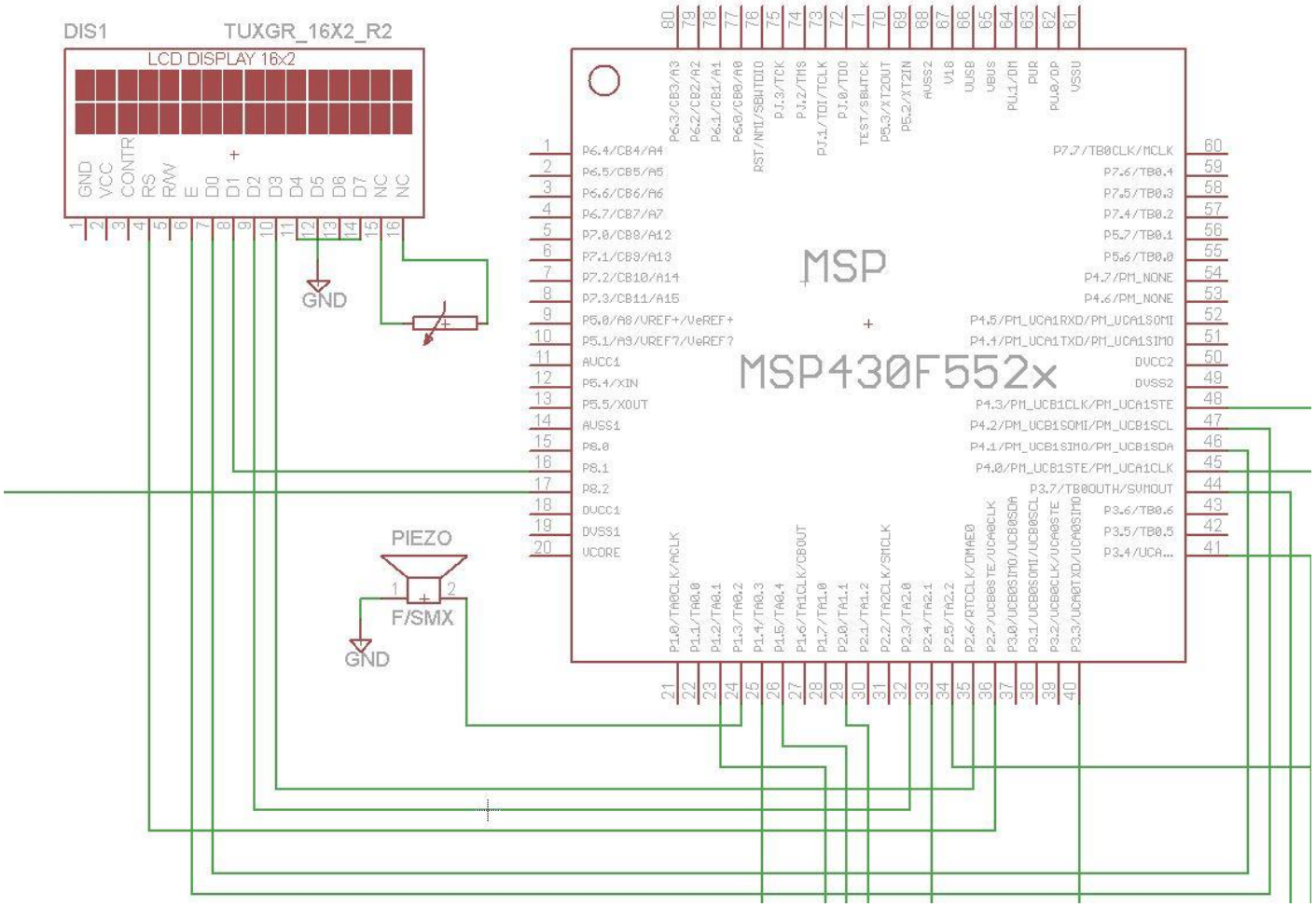
6.3 Open Source Code

The tone values for the Piezo Buzzer was from the following website

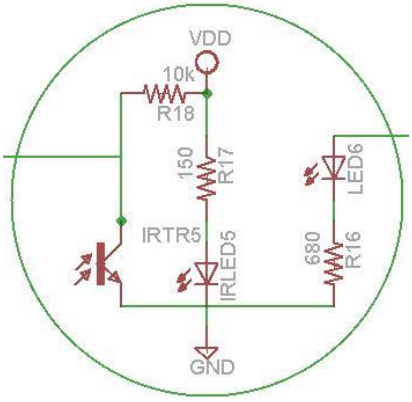
http://blog.wikifotos.org/wp-content/uploads/2011/02/rttt.zip

6.4 Schematic

MSP430



A Single Cup



Full Schematic