Introduction to Embedded Systems

JTT Arcade Application Note

Tomas Uribe, Tanner Smith, Jake Fraser

Section 4

December 22, 2017

JTT Arcade Controller

Uribe, Smith, Fraser
Rowan University

December 22, 2017

1 Introduction

The JTT Arcade Controller is a versatile gaming device that caters to the user's own preferences. With various LED configurations the consumer is able to switch between different modes that best suits their mood or game. The JTT Arcade Controller also does not limit its range with a wired connection. Instead, it is Bluetooth compatible, allowing the user to play within a 50ft range of their PC. More importantly, the user is not restricted to just one gaming platform as the Bluetooth interface makes it simple to map individual buttons on the arcade controller to keys on the PC. Overall, the JTT Arcade Controller provides a user friendly, customizable product that offers more versatility than other competing products on the market.

2 Design Specs

2.1 Case

The first design specification that needed to be decided was what the desired appearance of the final product should be. A couple hours were spent on the internet searching through competing products and various home made projects. Through this research, it was agreed upon that the final design would have a matte black top with clear rings around each button as well as the entire board. The sides would be made out of finished wood and the bottom would be clear acrylic to produce an underglow lighting affect which let all the electronics be seen.

2.2 PCB

In order to control every aspect of the arcade controller, a printed circuit board was designed and built. The board needed to have an on board processor because so many pins were going to be used. The board needed to regulate power and transfer data

from the processor to each one of the components on the controller. The board had to have outputs for each of the subsystems on the board and even needed to house some of the components on board, such as the bluetooth chip. Furthermore, to save money, the PCB had to fall within a certain size requirement of 4 inches by 4 inches.

2.3 Inputs

The inputs used in this project included 12 buttons along with an 8 directional joystick, a directional pad, a touch pad, and an on/off switch. Since the controller is connected to the bluetooth chip as a keyboard, each button, as well as the DPAD and joystick are mapped to a keystroke on a keyboard. From here, the buttons are mapped to commands in the video game emulator on the computer. The bluetooth is set in combo mode so that the touchpad can act as a mouse and the buttons can act as a keyboard. This allows the user to control the whole computer just using this arcade controller.

2.4 Liquid Crystal Display (LCD)

In order to display information to the user, it was decided that the controller should contain a programmable LCD. Further more, it had to have an RGB backlight to make the LCD more aesthetically pleasing and allow it to be configured with the LEDs. Additionally, it was decided that the LCD had to have a resolution of at least 128X64 Dots so the display would be easy to read and understand.

2.5 Touchpad

In order to enter the arcade controller market successfully, it was decided that the JTT Arcade controller needed to have at least one aspect no other arcade controller had on the market. Through brainstorming, a touchpad was added to the specifications to provide a complete stand alone experience to the user where they did not need to touch their PC at all to use this controller. Capacitive sensing was favored over resistive sensing because it was only necessary to send data to the processor if an event was triggered by the user.

2.6 LEDs

One of the main selling features implemented is complete customization. The user should be able to light up each button, the top ring and bottom underglow to any color they please. To do this, addressable RGB LEDs were chosen to individually control each LED pixel throughout the controller.

2.7 Bluetooth

The final specification was how the controller would connect to a game. Most controllers on the market use a wired USB connection to transmit and receive input data

to any PC. To differentiate the JTT Arcade Controller, a bluetooth connection was chosen to attract customers who wanted a wireless connection. The bluetooth function would make physical movement easier and overall increase the customers experience.

2.8 Power

Since this controller had to be standalone, it needed some sort of battery to power it. This specification was decided after all individual parts were selected because the power requirements of each subsystem had not been decided. The selection of the power subsystem will be explained further in the design approach.

3 Design Approach

3.1 Case

One of the team members working on the JTT Arcade controller had previous woodworking experience with his father. To create an aesthetically pleasing controller, his father's resources and woodshop was used for the case fabrication.

3.2 PCB

In order to design the main printed circuit board, the design software, Diptrace, was used. Many components were designed as custom parts and breakout boards were also designed for many of the parts for testing. Once the designs were completed, they were sent out to PCB Way, where they were manufactured. After the boards were received, the components were soldered up using a standard pick and place machine located in the pick and place lab at Rowan University.

3.3 Inputs

To implement each of the inputs, a slot was cut out of the case to secure them in place. On the PCB, hardware debouncing circuitry was implemented. Once the circuitry was taken care of, a ribbon cable was used to connect each button to the PCB in a clean manner. For the touch pad, a hole was cut into the top side of the case so that the wires could be funneled through and connected to the PCB. The only support circuitry for this component was 4.7k Ohm I2C pull up resistors.

3.4 LCD Screen

To implement the LCD screen, a module which functioned through SPI communication was chosen. This allowed messages to be sent from the processor to the display, pixel by pixel. Text was able to be transmitted to the display as well as personalized designs. Additionally, the module contained an RGB back light which made it possible to customize the color of the screen as well.

3.5 Touchpad

To implement the touchpad feature, a capacitive sensing module which could detect gestures or basic movement was chosen. This module sent data to the processor using I2C communication. The controller would first initialize communication to the slave module. Once the module acknowledged the controller, the controller could then request different gesture or movement data from various registers within the module.

3.6 LEDs

The addressable RGB LEDs chosen communicated through a modified format of SPI. Each LED pixel had a data input, data output, clock input and clock output. Each pixel stored 4 bytes of information (brightness, red, green and blue) and then passed on the rest of the data it received. When daisy chained, these LEDs would be programmed in sections which made it possible to configure independent light modes for the over glow, buttons/joystick, and underglow.

3.7 Bluetooth

In order for complete versatility for any PC game emulator, the controller had to function with a keyboard interface, the input keys could then be mapped to any button on any emulator controller. To do this, a bluetooth module was chosen which could be configured as a human interface device (HID). Combo mode would be used to send key presses for the buttons and mouse movements for the touchpad.

3.8 Power

After all parts were chosen for each subsystem, power calculations were completed to choose the correct battery. All components ran off of 3.3V except the LEDs which ran off 5V. For current, all current draws were very negligible compared to the LEDs. The LEDs drew about 25mA per pixel. With 70 LEDs strung together, this would total 1.75A. To make sure all subsystems had enough power, 4 alkaline D batteries were placed in series to output 6V and between 1200 to 18000 mAh. A diode was used to drop the voltage to 5.3V for the LEDs and a 3.3V regulator regulated the input voltage to all other subsystems including the processor.

4 Results and Discussion

4.1 Case

Once each component of the controller was completed individually, they were installed into the case, and connected to the main PCB. Each component had a preplanned specified spot in the case, and each one was able to fit into its spot. Ribbon cables were used to clean up the wiring in the controller, and the inside of the controller could be seen through the clear acrylic on the bottom of the controller.

4.2 PCB

Once all of the printed circuit boards were completed and built, they were tested. Because the boards were designed to be modular, each component on the board could be tested individually. Once it was determined that a part on the board worked the way it was supposed to, the next part was tested. After each piece was tested, code was written to test all of the components at the same time. The design of the PCBs used in this project allowed the designers to test modularly and helped to create a working model at the end of the process.

4.3 Inputs

When the inputs were put in place, they were tested individually. Each button and joystick stroke was mapped to a keyboard ASCII value using the bluetooth chip. From there, each input was tested simply by typing on a computer. Once it was determined that each button was connected properly, the touchpad was connected. By programming the bluetooth chip to send out touchpad data after a certain amount of time, the touch pad could be tested. Once it was determined that the mouse could be controlled using the touch pad, it was installed into the case.

4.4 LCD Screen

The LCD proved to be simple to implement once SPI was properly functioning. To accomplish this task, the driver libraries, within Code Composer were used. These libraries came equipped with multiple communication protocols, among them being SPI. Overall, the libraries proved to be a very high level coding platform which made SPI communication very simple to understand. The main issues that were encountered with the libraries was that some of the default settings did not output the correct clock and MOSI signal and furthermore did not correlate to the functionality of the LCD. For example, in order to send commands and data properly the processor had to send out the MSB first for commands and the LSB first for data. This issue was resolved by using the pin "A0" on the LCD. The purpose of this pin was to identify when a command or data was being sent to the LCD. When this pin received a high signal it indicated that data was being sent, while a low signal meant a command was sent. By using pin "A0", SPI communication was reinitialized every time a command or data was sent to the LCD in order to get the correct bit sent first.

4.5 Touchpad

The touchpad proved to be a significant challenge to set up individually. The reason for this was I2C communication. Since I2C requires a specific protocol in order for the two device to interact, it was quite difficult to get the touchpad to acknowledge the MSP430FR5994 master. Extensive research had to be performed on not only I2C but the touchpad model as well in order to identify what signals had to be sent to each pin on the touchpad in order to conform to the I2C protocol. Ultimately, it was identified that the reason the touchpad would not "ACK" was because the Not Reset pin could

not be left floating. Once this pin was given a high signal, the touchpad was able to acknowledge the master's request and return the desired data.

4.6 LEDs

The implementation of the LEDs did not prove to be to difficult. Since the LCD screen subsystem was completed first, the same SPI software was used with different pins. Instead of finding already made software online, the addressable LED code was made from scratch to implement it specifically to this controller. Three 2 dimensional arrays were made to have the same number of rows as the number of pixels for button LEDs, ring LEDs and underglow LEDs. Each row in the array had a red, green and blue byte value. This is where the colors were stored. Any number of pixels can be written to at any color. There were three modes implemented that could be changed at anytime. The modes were solid, chase and shift. Solid displayed all colors normally, chase masked all colors except a certain length but incremented where it started the mask, and shift just kept incrementing which pixel color it sent first. Wiring the LEDs were the only problem because 70 LEDs had to be wired on the PCB with a specific design,

4.7 Bluetooth

Initially a Bluetooth Low Energy chip was selected to integrate into our project due to having some example code for it. After working on its implementation for a long time, the baud rate was set to 115.2k, the correct name was set up and devices could be connected to it. However, the devices could not talk to each other because there was no driver. Instead of figuring out how to write a driver, more research went into finding the correct bluetooth chip. Eventually a bluetooth chip with programmable HID interface was chosen to be integrated. This chip was set up using CoolTerm (a command window). Then the processor communicated with the chip over UART which then packaged the message in bluetooth protocol and sent out the desired message. With the change of bluetooth chip, the desired function was achieved and it was much easier to use.

4.8 Power

As previously explained, 4 alkaline D batteries were placed in series with a diode between the positive terminal and the board. This should have produced a 5.3V input to the circuit board but instead created a 5.7V input. The drop across the diode was 0.7V as expected but the batteries were outputting around 6.4V which wasn't expected. To fix the problem we placed a second high current diode in series to drop the input voltage to 5V. In future implementation, a 5V regulator should be used to maintain a constant reliable 5V input. The 3.3V regulator for the processor and other components worked perfectly and outputted a 3.3V constantly. The correct bulk and bypass resistors were used for all power inputs.

4.9 Final Integration

A significant constraint that affect the final integration of the controller was the design of the case. I was very difficult to connect all the subsystems to the processor in a neat and organized manner. Additionally, when it came to integrating the subsystems in code, modularity had to be stressed in order for debugging to be simpler and furthermore for the code to be more understandable. Ultimately, by creating functions for each subsystems it was easier to call the necessary methods whenever an event on the physical controller was triggered. The code developed was high level which made it possible to make simple changes if an error was identified.

5 Conclusion

The finished arcade controller was equipped with 5 different LED configurations. Each button was mapped to a specific letter on a keyboard, allowing the controller to be completely interfaced to gaming emulators on a PC. Future iterations of the controller will involve more research in order to identify more inexpensive subsystem parts in order to maximize profit if the controller was ever commercialized. As it stands now, the arcade controller has an estimated market value of $200-$350 based on other competing products. However, this product introduces a versatility that other product on the market do not have which is why it is considered to be a favorable product to commercialize.