A-

Web-Based Wireless MPEG -1 Audio Layer 3 (MP3) Player

EECS 129B
Professor Klefstad

Kevin Fung #
Mason Chang
Mark Hervey

Abstract: This project will focus on the creation, design, and implementation of a web-based wireless mp3 player. That is, we will design an mp3 player that will accept what mp3 commands a user selects through a web interface and execute them.

Overview

What: For this senior design project, our group has chosen a web-based wireless mp3 player in which we will be able to control an mp3 player through a website. Specifically, we used a first generation Ipod Shuffle.

Why: We used this device due to a low cost, and availability of information regarding relevant project information online. W e would like to port this device to other switching related devices and believe that this is a small step towards an innovative technology; being able to control lights and objects, throughout the household, with a simple click of the mouse button.

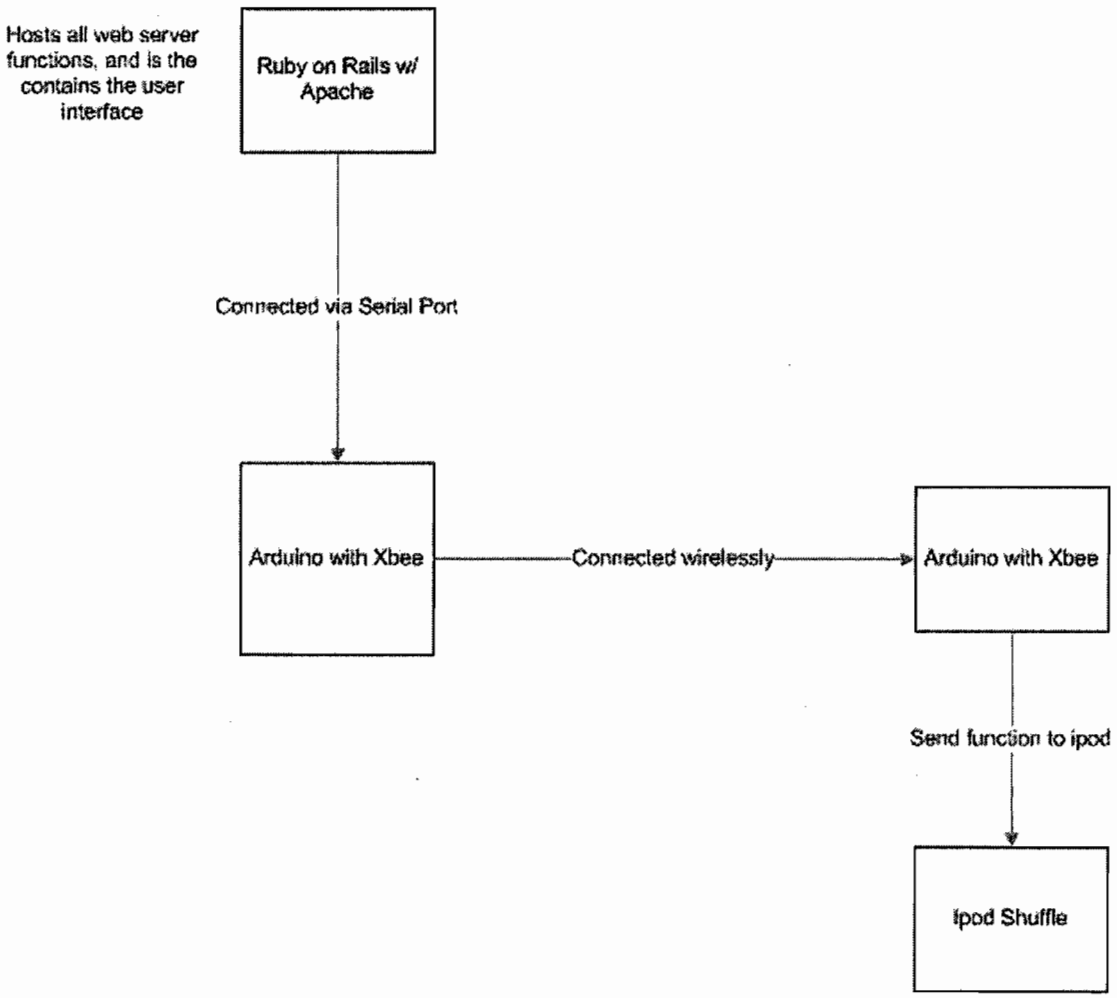
System Features: Our group will build the wireless transmitter with an Arduino board. This board will connect to a web server which is hosting Apache and the Ruby on Rails framework. Arduino supports interaction through the serial port with the ruby programming language. A user will then select one of the five functions that come on an iPod shuffle. The website will communicate directly with the Arduino, and send the signal of the function wirelessly to another Arduino / Xbee combination. Finally, another Arduino with an Xbee will be decode which signal was sent, and load that appropriate function on the iPod shuffle.

Purpose:

The reason that our group chose to do this project is that wireless technology is becoming more practical in our everyday lives. Being able to do things wirelessly gives us the freedom to do multiple tasks without as much effort. Therefore, as computer engineers, our group wanted to embrace new and upcoming technology and see how this technology works first hand. We chose to make a wireless web-based mp3 player not only because of our love of having music at the palm of our hand, but also our desire to make our everyday lives easier when we are sitting at our computer. Doing such a project would enable us to hook our mp3 player into perhaps better stereo speakers than our desktop machine's, and be able to control what songs to

play without moving from our computer. This project will be mainly build for our own personal use as well as our interest in wireless technology, as well as those who are very into building their own gadgets. It can also be applied as a device for disc jockeys to make what they do easier. Finally, this project is just a novel idea to test out and see how well it works.

System level block diagram:

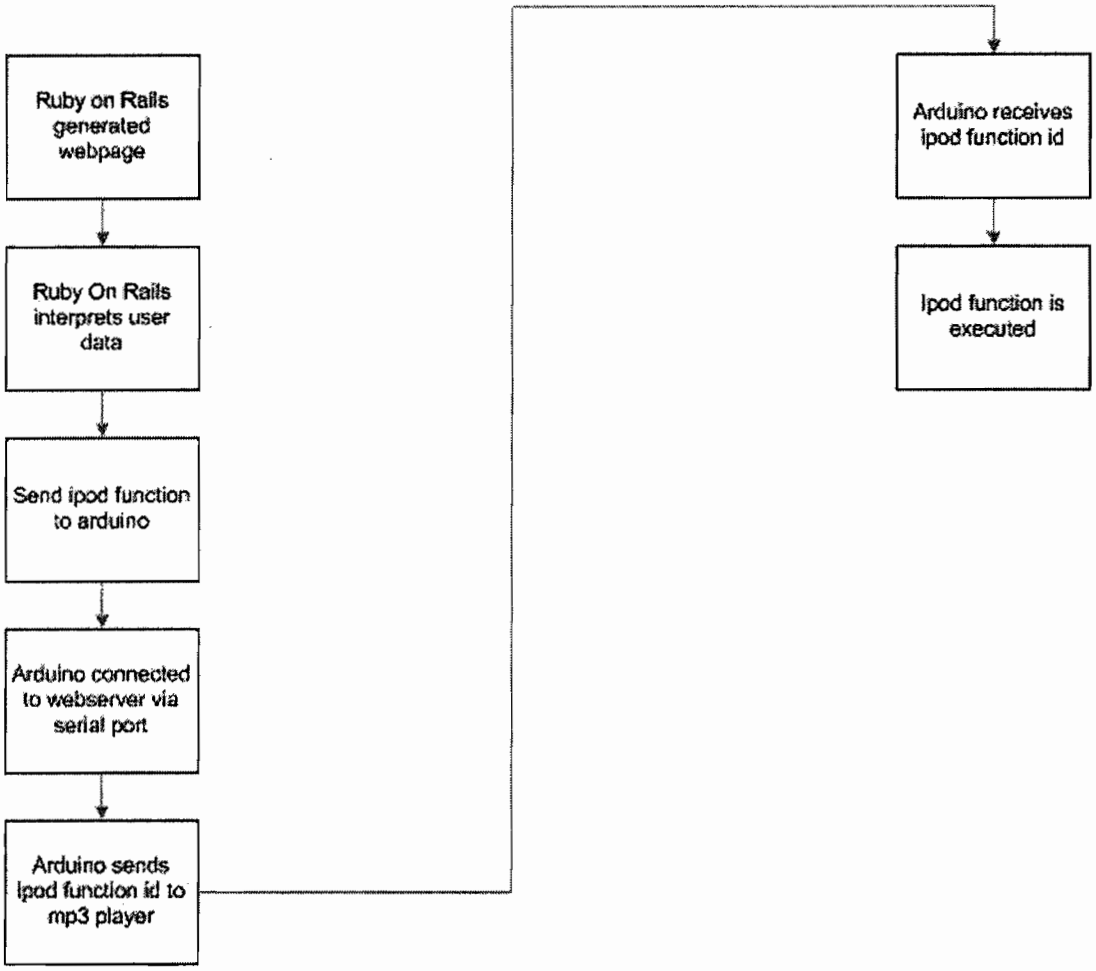


Software Description:

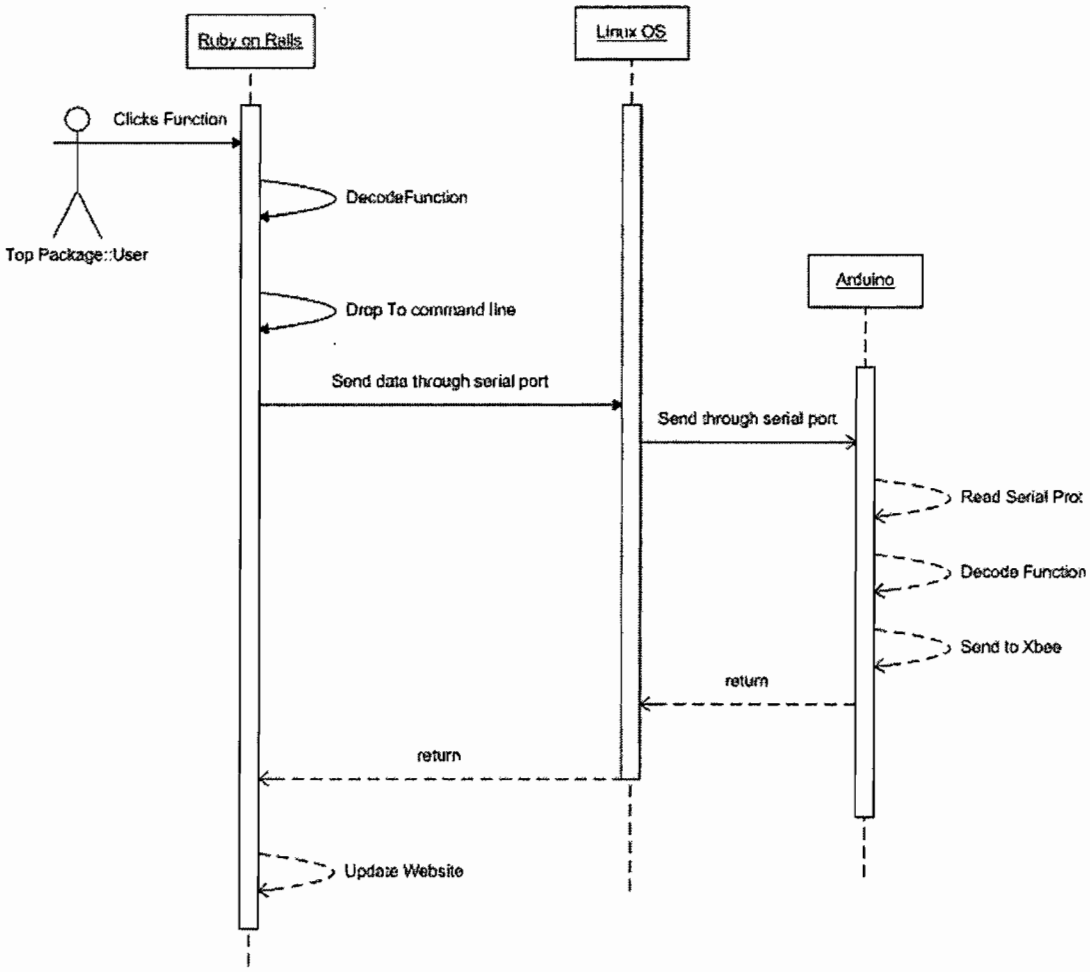
All the software to connect to and from the computer, was written in ruby. This was mostly due to the fact that we would be able to learn a new language, and new platform as a whole with this project. The website is built with Ruby on Rails, and CSS technologies. As it is a simple web page, the use of ruby was limited in its functions. The language used to program the arduinos was a modified C, the one supplied with the arduino hardware. The connection between the ruby on rails website, and the arduino was through the computer's USB port. In order for ruby to communicate with the serial port, we had to use a package called ruby-serialport[1]. Finally, the need to drop to the command line, and execute a separate ruby script, thereby instantiating two instances of ruby, was required due to some incompatibilities with the ruby on rails framework and the ruby-serialport library. Our conclusion for these incompatibilities is that either a) The ruby on rails framework runs in a sandbox environment, and will not allow direct access to hardware. B) Ruby on rails alters the ruby language to the extent that some of the functions required by ruby-serialport act differently than expected.

[1] http://rub yforge.org/projects/ruby-serialport/

Software flow chart:

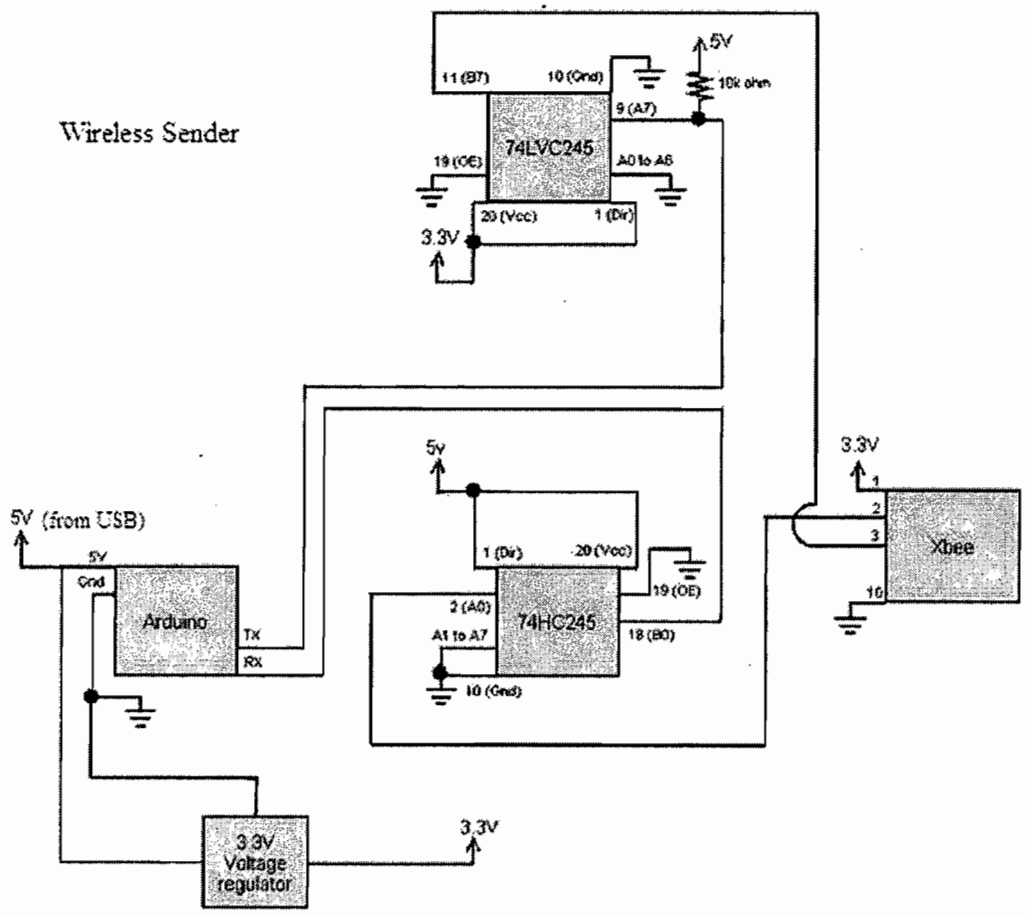


Detailed sequence:

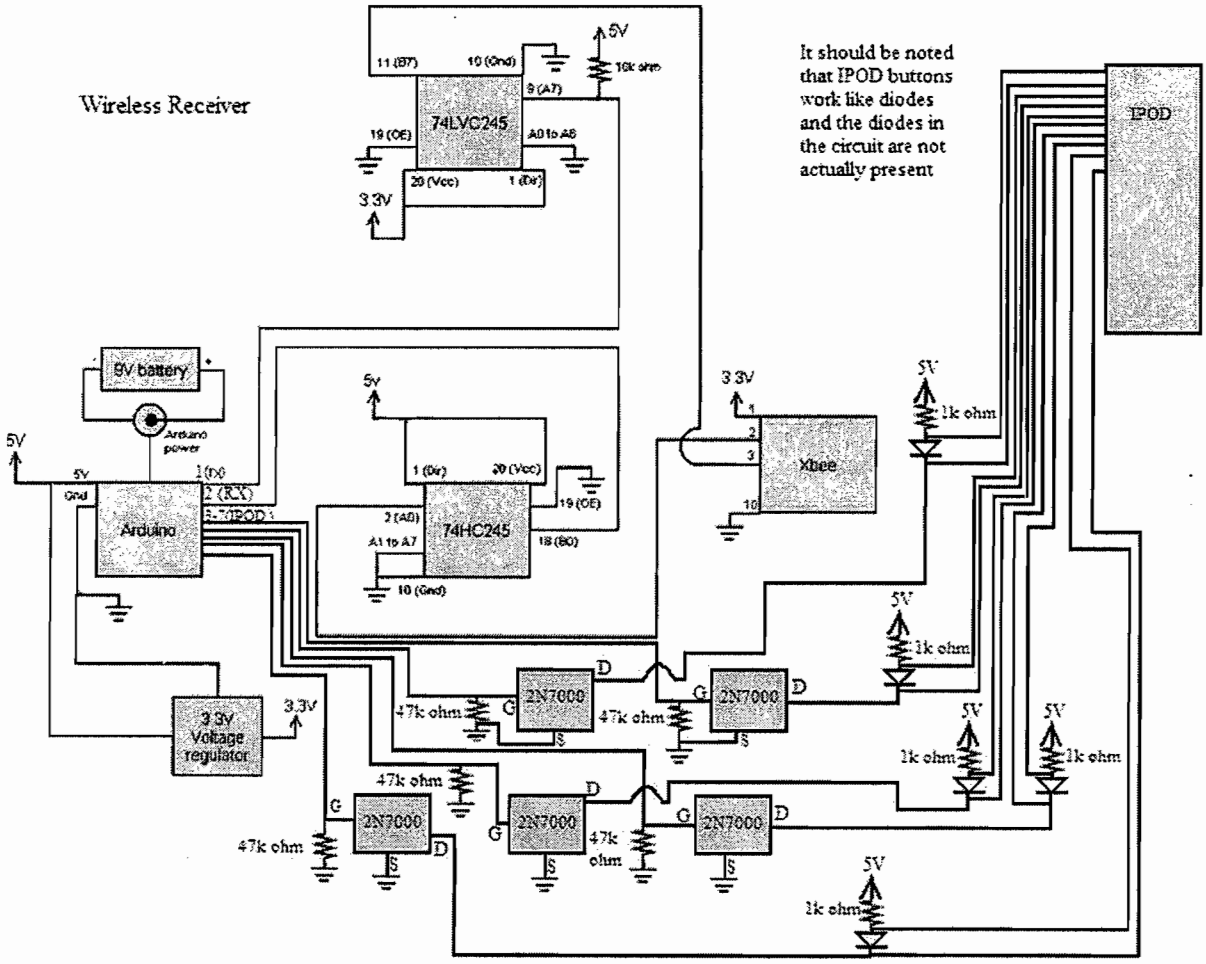


Circuit Level Diagrams:

The following is the diagram of the sender:

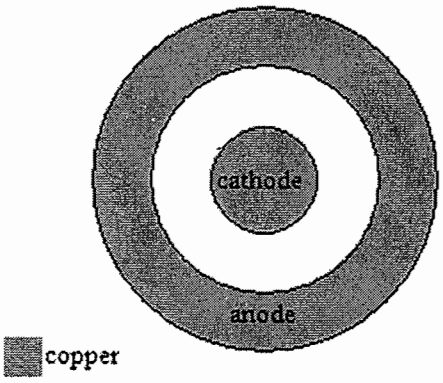


The following is the diagram for the receiver:



The ipod connection is shown in a picture below and should be wired as the diodes:

Each IPOD shuffle button looks as modeled below:



System Test Plan:

This system has five main functions which directly correlate to the play/pause, back, forward, volume up, volume down functions. The interface on the ipod is mimicked on the interface on the website. Thus the five black box test cases are:

1) Push play/pause on the website, and ensure that the ipod plays/pauses;
2) Push the forward button on the website, and ensure that the ipod skips to the next song.
3) Push the back button on the website, and ensure that the ipod skips to the previous song.
4) Push the volume up button on the website, and ensure that the ipod's volume increases.
5) Push the volume down button on the website and ensure that the ipod's volume decreases.

For white box testing, there are three main components. First is the ruby on rails website, up until the point of sending the correct signal to the serial port. Second is the arduino that is connected to the webserver via the serial port. We can test that the arduino sends the correct signal when reading from the serial port. Finally, the last part is the arduino connected to the ipod, and ensuring that the arduino is reading the correct signal, and outputting to the correct port for a given signal.

To test the website, we can have an instance of the serial port monitoring program, "minicom" listening on the given serial port. Then, when the user pushes a button, we should see the value being sent to the serial port. The five test cases are the same as the website. Instead of ensuring the correct ipod function, we have to ensure the correct signal is sent through the serial port.

To test the arduino connected to the webserver, we can load a minicom again, but this time connect the serial port to the arduino. Then, on the hardware side, we have to connect LEDs to a specific port. In the software, we have debugging information to output a certain LED for a given input signal. We can type the expected signals into minicom, and ensure the LED lights up for each signal. This again has the same five test cases.

Finally, for the arduino connected to the ipod, the test cases are exactly the same as the previous step. With this test plan, we can validate the three major components, and the integration points between the three components.

Cost Analysis:

Qty	Name	Amount
1	iPod Shuffle (1st Generation)	$50
2	XB24-AWI-001-ND MODULE ZIGBEE 1MW W/WIRE ANT	$46.40
2	Arduino USB board	$63.90
2	568-1423-5-ND IC TXRX OCTAL BUS 3TATE 20DIP	$1.16
2	296-8503-5-ND IC OCT BUS XCVR TRI-ST 20-DIP	$1.20
2	497-1491-5-ND IC LDO REGULATOR +3.3V TO-220	$1.54
2	194271 breadboard	$31.90
5	2N7000 N-Channel FET (TO-92)	$1.50
Total:		**$197.6**

Summary:

The result of our project is undoubtedly a success. Each of the functions that we wanted to implement on the web page works correctly on the iPod. When we press a button on the website that we created, the iPod will perform the function that we wanted. Our goal was to turn a fully functioning iPod Shuffle and implement it wirelessly as well as have it controlled by a website. However, our completed circuit is slightly different from the original circuit. The circuit that we had planned contained an integrated circuit (IC) part called an opto-isolator, which was supposed to act as a switch and closed the circuit when voltage was applied. However, this particular IC did not work as intended. Therefore, we went with another alternative, which was a simple MOSFET switch. In this process, we found that the iPod button acts as a diode. When you press the button on an iPod, a metal contact is depressed and closes the circuit. The voltage flows from the inner contact to the outer contact, which signals the iPod that a button was pressed. Even with this minor change, the iPod is working with the Arduino board just as we wanted. Thus, we feel that our project accomplished our stated goals.