Acoustical Analysis Unit

By

Kristine Cabrera

Kevin Chen

Joseph Shim

Project Proposal for ECE 445, Senior Design, Spring 2013

TA: Ryan Corey

February 06, 2013

Project No. 14

Contents

1 Introduction

1.1 Statement of Purpose

The work and research of Professor Swenson and Dr. White gave us the initial interest to pursue this acoustic analyzer project. The current device they are using for acoustical research is the size of a desktop computer and outdated by several years which makes the functionality limited and quite cumbersome to bring out into the field. We plan on miniaturizing the device and still replicate all the important functions of the original. Those working in this field will be able to connect up to three microphones and bring it easily outdoors when trying to analyze noise. It will be a lightweight version of the device they need and therefore there is a good demand if it works. Also, we will design it as to make the programming easily modifiable to add further functions and calculations as needed.

1.2 Objectives

1.2.1 Goals

- Create a device capable of being more portable to be used outdoors and that can be used with different types of microphones
- Use audio inputs from microphones to graph and collect acoustic intensity
- Have modularity in programming to allow additional data collection, functionality, calculations, etc. for end customer

1.2.2 Functions

- Inputs audio from microphones
- Distinguishes what noise/audio coming from each microphone
- Outputs acoustic intensity in the form of moving graphical bars
- Has various physical buttons for menus

1.2.3 Benefits

- Capable of being brought outside and is not too heavy or cumbersome to carry around
- Simple program that is capable to be easily modified to increase the amount of functions that will be shown on the screen

1.2.4 Features

- Device powered by battery or wall outlet
- Has a LCD display that contains at least a 128x64 resolution to clearly show data and menus
- Has three standard BNC connections that can fit different microphones or other devices with this type of connection

2 Design

2.1 Block Diagram

In order to split up our work efficiently we have seperated out project into several key components/modules. Each module can be independently tested and verified before attaching it to the other blocks. Figure 1 shows the general block diagram our project will take on. The arrows depict the flow of information or power from one component to another.

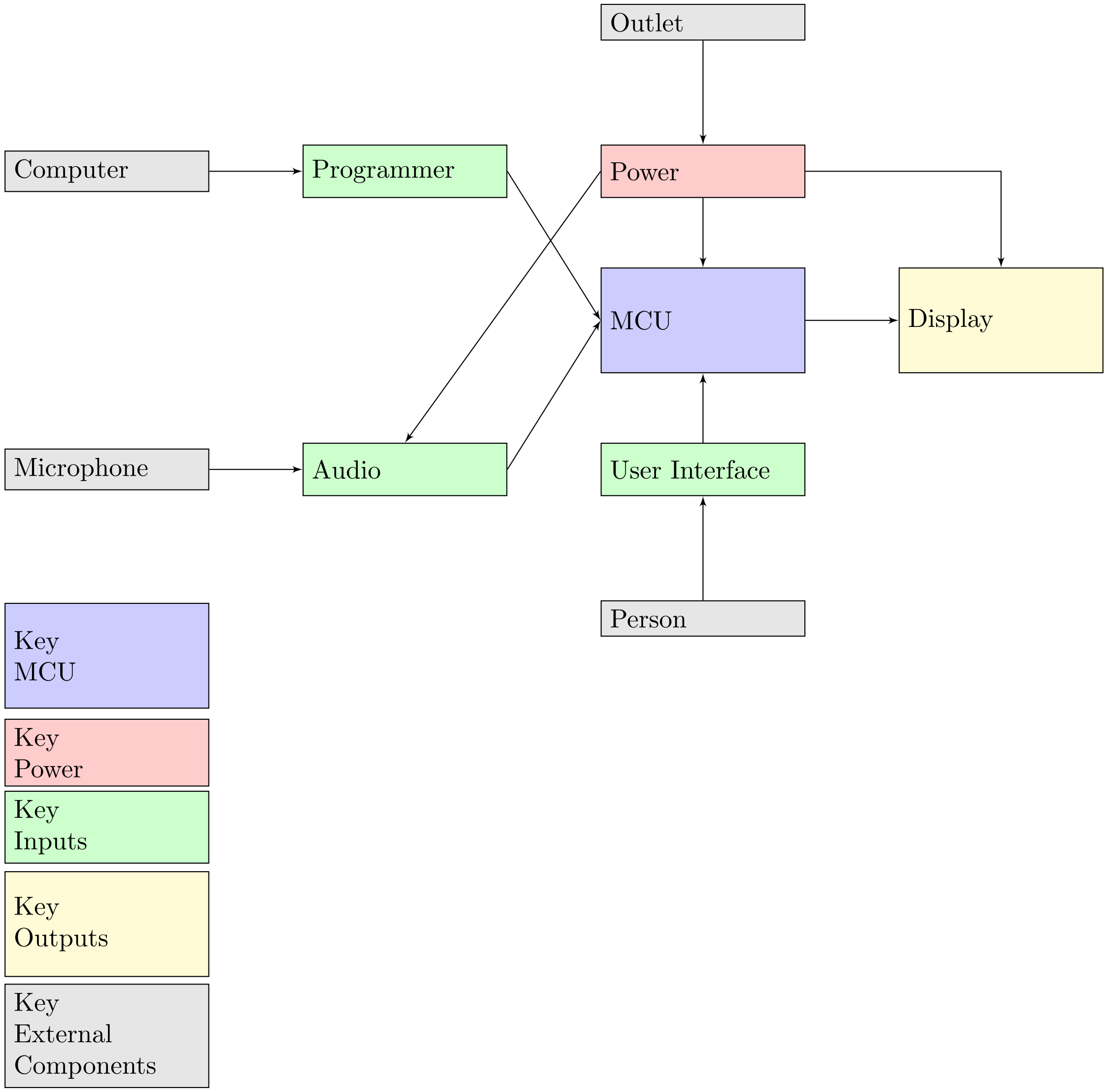

Figure 1

2.2 Block Diagram Details

2.2.1 Module Descriptions

Summary

The acoustic analyzer unit will consist of 3 main parts: power, MCU, and display. The overall system will take the audio from the microphones, process it, and show it to the user through the display. The user can make the system display the specific data that he/she wants and in the form he/she wants through manipulation of the user interface. If there are extra functions that the user needs that are not already preprogrammed, he/she can add them through the programmer.

Microcontroller

The MCU used is a Texas Instruments Stellaris LM4F120. This will be the primary processing center of the whole system. It will be connected to audio through BNC connections and some circuitry. The audio signal will then be processed by an onboard ADC. Once the signal is processed, the data may be shaped in any form wanted by the user and then shown on the display module. The LM4F120 was chosen as the MCU due to its onboard ADC units and DSP abilities.

Power Supply

The power supply will power the MCU and the display components of the system. It will output 3.3V (1V), 5V (1V), and potentially 12V (1V) depending on the needs of the analog circuitry. It will include connection to wall outlet and also to a portable battery which allows for both benchtop use and portability.

Programmer

The programmer is the interface between the MCU and the software functionality of the system. It will be designed so that the user may add his/her needed functions easily beyond the initial functions of the device to process the incoming audio signal. In the event that the chip we select include a programming module we will be using that, effectively outsourcing the work for the programmer.

Audio

The audio block consists of the analog circuitry that will take the audio signal from the microphones to the MCU. The circuitry will include any preamplification, filtering, and buffering that's needed and will provide a usable signal for the MCU's ADC as well as protecting the Microphone from any electrical damage and vice versa.

User Interface

The user interface will consist of pushbuttons around the display. The user can use the pushbuttons to navigate through functions and menus on the device much like how modern oscilloscopes do.

Display

The display will consist of 1-2 LCD screens each with at least a 128x64 resolution. It will be connected to the MCU which will process what is to be shown based on processed data from audio and also the users inputs through the user interface. The LCD screen will be controlled by a standardized parallel protocol.

2.3 Performance Requirements

1. *Programmer*

   The programmer should be able to communicate with the MCU to program the microproccessor and potentially allow for live debugging.

2. *Audio*

   This module should be able to take input from the microphone and send a usable signal to the MCU. Should prevent unwanted current/voltages from reaching other modules.

3. *Power*

   The overall voltage of the power supply ideally should be 3.3V, 5V, or 12V (but range could be 1V). The module should be able to provide a stable voltage that the other modules can use. In terms of efficiency the power supply should have an efficiency rating of over 50

4. *Microcontroller Unit*

   Since this is the main module that actually controls the device, this needs to recognize each input signal from audio (if it is noise or actual data, etc.) and needs to know how to distinguish this data. This module should also take this data and convert it into the acoustic intensity should similar as a histogram graph.

5. *User Interface*

   There should be ample room for all the buttons to sit on the actual device. The button layout should be easy to use and prevent multiple buttons from being pressed accidentally.

6. *Display*

   The display or displays (if we have two one for menus and one for actual data) need to have at least a 128x64 resolution to clearly show the typeset of the menus as well as well-defined data bars.

2.4 Testing Procedures

1. *Programmer*

   We will program the MCU with a couple of simple programs. If the MCU behaves as our program intended we sucessfully know that the Programmer can interact with the MCU. Programs will test the ADC input and basic I/O funcionality.

2. *Audio*

   For the audio portion we will take the microphone inputs, send it through the Audio Module and make measurements on an oscilliscope. If the oscilliscope output seems correct we know that our circuit works. For testing protection and proper buffering we will run a series of stress tests and use several lab equiptment to see if the voltages and current are within the tolerances of key components (Tolerances will be know via Datasheet of respective components).

3. *Power*

   To see if the power supply had met our ideal voltage outputs, we will attach a multimeter to the power module and take several measurements loaded and unloaded.

4. *Microcontroller Unit*

   We will check for valid connections from each of the various modules to the microcontroller. Afterwards, we will use a simple program known to be bug free with audio input and check for expected outcome. Further tests will include ability to take inputs from the User Interface as well as properly displaying text from the MCU.

5. *User Interface*

   To test this section of the device, we would go through various menus and functions by depressing the respective buttons. If the correct operation is occurring on the screen, then the buttons seem to be working with the microcontroller. We will also test if one button executes one operation or if more than one functions occurs.

6. *Display*

   We will use the analog contrast controller (a potentiometer) to see if the screen is responsive. If that works then we will go on and test the screen with a known working microcontroller LCD screen driver to see if the screen responds.

2.5 Tolerance Analysis

The most critical portion of this project is the method of converting all input signals into usable, distinguishable data to show on the display screen. Thus, the actual programming portion in tandem with the microcontroller unit will be the most work intense. The goal of this tolerance analysis is to verify that the program is correctly defining each audio input and has the correct calculations/algorithms to show acoustical data. Since we are also performing an FFT on the data, this needs a different calculation that needs verification. The original unit from the offices of Professor Swenson and Dr. White shows the data they want to be shown and analyzed, we will use this as a basis for our unit (since we are basically creating a miniature version of this). The acceptable tolerance will be a 10% error of the acoustic intensity shown from what is acceptable by Professor Swenson and Dr. White.

3 Cost and Schedule

3.1 Cost Analysis

Table 1: Labor

Member	Hourly rate	Total hours	Subtotal x 2.5
Kristine Cabrera	$35	130	$11375
Kevin Chen	$35	130	$11375
Joseph Shim	$35	130	$11375
Total			$34125

Table 2: Parts

Parts	Quantity	Cost per Unit	Total
Stellaris LM4F120*	1	$12.99	$12.99
KS0108 LCD display*	2	$35.00	$70.00
Ti Op-Amps	5	$0.643	$3.22
PCB**	2	$0.00	$0.00
LEDs	10	$0.262	$2.62
RCL Components****	N/A	$0.00	$0.00
LiPo Battery	1	$22.50	$22.50
Mic Probe***	1	$0.00	$0.00
BNC Connectors	3	$5.57	$16.71
Pushbuttons	10	$3.06	$30.60
Total			$158.64

* may be provided by Texas Instruments Stellaris Group

** provided by UIUC ECE Dept.

*** provided by CERL

****provided by the ECE Service shop

Table 3: Grand Total

Labor	Parts	Total
$34125	$158.64	$34283.60

3.2 Timeline

Table 4: Schedule

Week	Task	Member Assigned
2/04	Work on proposal	Kristine, Kevin, Joseph
2/11	Rough Design for Analog Input	Joseph
	Rough Design for Power	Kristine
	Rough Systems Design + Program	Kevin
2/18	Finalize and Test Analog Input	Joseph
	Finalize and Test Power	Kristine
	StellarisWare Operation Proof of Concept	Kevin
2/25	Prototype Interface Combination	Joseph, Kristine
	Program DSP functions	Kevin
3/04	Setup LCD screens	Joseph, Kevin
	Design User Interface	Kristine
3/11	Program GUI	Joseph, Kevin
	Finalize User Interface	Kristine
3/18	Finalize Hardware	Joseph, Kevin, Kristine
3/25	Iron out software	Joseph, Kevin
	Additional Functionality	Kristine
4/01	FlexTime*	Kristine, Kevin, Joseph
4/08	FlexTime*	Kristine, Kevin, Joseph
4/15	FlexTime*	Kristine, Kevin, Joseph

* Due to murphys law (”Anything that can go wrong will go wrong”) we will be adding FlexTime which can be used for debugging or damage control. In the event that we are on schedule we will be refining our design to improve user experience.