Pivit Technologies Inc

School of Engineering Science ♦ Burnaby, BC ♦ V5A 1S6 Project
malajube@googlegroups.com

February 19th, 2007

Mr. Lakshman One
School of Engineering Science
Simon Fraser University
Burnaby, BC
V5A 1S6

Re: ENSC440 Design Specifications for a Tactile Hearing Aid

Dear Mr. One:

Hearing loss is a growing concern for many Canadians. Hearing aids are ineffective for individuals with severe inner ear damage and methods utilizing sight are limited by the necessity to observe everything that one wishes to hear. At Pivit Technologies Inc., we aim to deliver acoustic information using the sense of touch to assist the hearing impaired with daily activities requiring an awareness of noise. We believe the ability to interact and "feel" the sounds of their environment will result in an improved standard of living for the hearing impaired.

Our goal is to design and build a PC controlled tactile hearing pad capable of stimulating the skin through vibrations to determine the viability of delivering acoustic information through the sense of touch. We detail the design specifications for our prototype in the attached document, *Design Specifications for a Tactile Hearing Aid*.

Pivit Technologies Inc. is comprised of four dedicated, hard-working individuals: Ryan Dickie, David Dickin, Mehran Eghtesad, and Merle Kinkade. Please feel free to contact me by phone at 604-787-4871 or send any questions, comments, or concerns via email to malajube@googlegroups.com.

Sincerely,

David Dickin

David Dickin
CEO Pivit Technologies Inc.

Enclosure: *Design Specification for a Tactile Hearing Aid*

Pivit Technologies Inc

Design Specification for a

Tactile Hearing Aid

Project Team: *Ryan Dickie*
David Dickin
Mehran Eghtesad
Merle Kinkade

Contact: malajube@googlegroups.com

Submitted to: Lakshman One – ENSC 440
Steve Whitmore – ENSC 305
School of Engineering Science
Simon Fraser University

Issued: March 5, 2007

Revision: 1.0

Executive Summary

In an effort to improve the quality of life and acoustic awareness of the hearing impaired we are developing a Tactile Hearing Aid. It will allow individuals who are hard of hearing the opportunity to interact with sound in the world around them by transferring audio information to the sense of touch. By increasing awareness of the hearing impaired to environmental noise we hope to increase both their safety, by providing a means of notification for sound based alarms, and for enjoyment by providing an interface to music and verbal communication.

The development of the Tactile Hearing Aid will occur in two phases. After the completion of the first phase of development, the Tactile Hearing Aid will be capable of translating sound into tactile sensations using a non-portable implementation. The device will have the following features:

1. The VibraPad, an adjustable article of clothing containing an array of vibrotactile transducers.

2. The capability of mapping input sounds frequency components into control signals for the VibraPad transducer array.

3. A software GUI allowing the user to monitor and control the input sound and software response.

After the second phase of development, the device will also:

4. Be powered by a portable power source.

5. Be lightweight and contained within a durable, portable package.

6. Run the software components off of a mobile platform.

The first phase of development in the Tactile Hearing Aid will be completed in April 2007.

Table of Contents

List of Figures

List of Tables

1 Introduction

The Tactile Hearing Aid is an assistive technology intended to help the hearing impaired perform their daily tasks safely and efficiently. Incoming sounds are mapped to an array of vibrators strategically placed on the skin. With extended use, the user would ideally recognize and categorize frequencies associated with everyday noises as well as sounds indicating potential hazards. The initial prototyping and testing phase resulting in a proof-of-concept device will be completed in mid April; after which, further research and preparations for the production model will begin.

1.1 Scope

This document describes the design decisions necessary to build a working prototype that is able to demonstrate the goals of this project. It will also list and itemize specific tests that must be conducted to determine the effectiveness of our proposed method of sound recognition. Modifications intended for the production model will not be covered within the scope of this document.

1.2 Acronyms

DLL	Dynamic Link Library
FFT	Fast Fourier Transform
IBM	International Business Machines
IC	Integrated Circuit
PC	Personal Computer
PCB	Printed Circuit Board
PWM	Pulse Width Modulation
SPP	Standard Parallel Port
THA	Tactile Hearing Aid
TTL	Transistor-Transistor Logic
USB	Universal Serial Bus

1.3 Referenced Documents

[1] LM317: 1.5 A Adjustable Output, Positive Voltage Regulator. "alldatasheets.com." [11 Feb. 2007]

[2] 2N3904: NPN General Purpose Amplifier "alldatasheets.com." [16 Feb. 2007]

[3] K. A. Kaczmarek and P. Bach-y-Rita, "Tactile displays," in Virtual Environments and Advanced Interface Design. W. Barfield and T. Furness, Eds. Oxford University Press, 1995. (invited review chapter)

[4] Win32 and COM Development. Internet: http://msdn2.microsoft.com/en-us/library/aa139672.aspx [3 Feb. 2007]

[5] Windows GDI. Internet: http://msdn2.microsoft.com/en-us/library/ms536795.aspx [3 Feb. 2007]

[6] Windows Multimedia Start Page. Internet: http://msdn2.microsoft.com/en-us/library/ms713771.aspx [5 Feb. 2007]

[7] Boost.org C++ Libraries. Internet: http://www.boost.org/ [7 Feb. 2007]

[8] JTC1/SC22/WG21 - The C++ Standards Committee. Internet: http://www.open-std.org/jtc1/sc22/wg21/ [6 Feb. 2007]

[9] FFTW Home Page. Internet: http://www.fftw.org/ [12 Feb. 2007]

[10] J.S. Jacaba. “Audio Compression using Modified Discrete Cosine Transform: The MP3 Coding Standard.” B.Sc Math, University of the Philippines, Diliman, Quezon City, 2001.

[11] J.O. Smith III and J.S. Abel. (1999, Nov.). “Bark and ERB Bilinear Transforms.” IEEE Transactions on Speech and Audio Processing. [Online]. pp. 6. Available: http://ccrma.stanford.edu/~jos/bbt/bbt.pdf [Feb. 23, 2007].

[12] Islands of Music. “Specific Loudness Sensation.” Internet: http://www.ofai.at/~elias.pampalk/music/tech/loudness.html [2 Mar. 2007].

[13] Interfacing the Standard Parallel Port. “Beyond Logic.” Internet: http://www.beyondlogic.org/spp/parallel.htm [4 Feb. 2007].

[14] Parallel Port Configuration. Internet: http://www.geocities.com/nozomsite/parallel.htm [4 Feb. 2007].

[15] Standard TTL Logic Levels. “Twisted Pair.” Internet: http://www.twysted-pair.com/74xx.htm [4 Feb. 2007].

[16] Inpout32.dll for WIN NT/2000/XP. "Logix4u.net." Internet: http://www.logix4u.net/inpout32.htm [6 Feb. 2007].

[17] 74S373 Octal D-Type Transparent Latches And Edge-Triggered Flip-Flops. "alldatasheets.com." [10 Feb. 2007]

[18] Audacity: The Free, Cross-Platform Sound Editor. Internet: http://audacity.sourceforge.net [22 Feb. 2007]

1.4 Intended Audience

This document is mainly intended to assist design engineers to realize the specifications relating to the prototype as outlined in the functional specifications report. They will utilize the detailed guidelines to assist in building the proof-of-concept prototype.

Test Engineers will use the test plans detailed in this document to analyze the prototype and ensure that it performs according to specifications.

Project managers will use this document to ensure that productivity levels are maintained and that all functional specifications pertaining to the prototype model are properly implemented.

Quality Control personnel will find this document useful to determine areas of improvement and to ensure that all applicable quality standards are met.

2 System Overview

Figure 2-1 shows the general system overview of the THA.

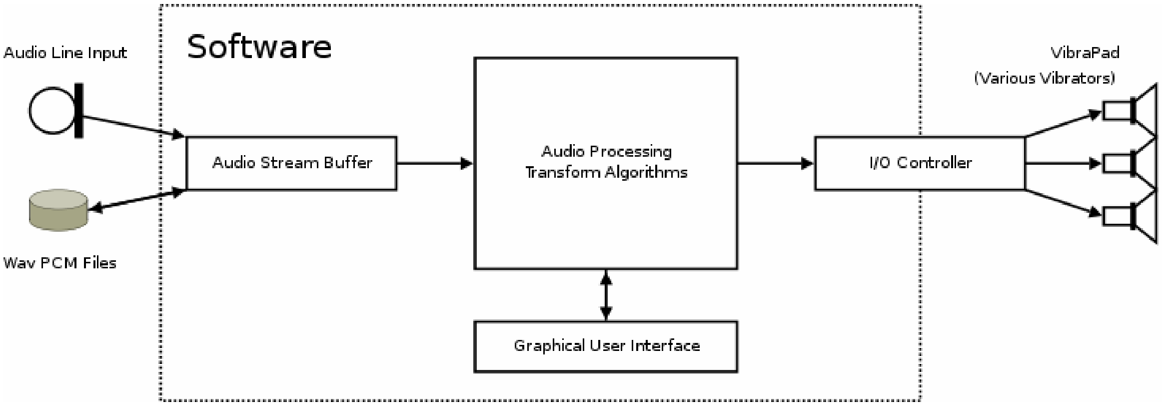


Figure 2-1: Tactile Hearing Aid System Overview

The Audio Stream Buffer is a software module that takes in acoustic information by an audio line input or a previously saved .wav sound file. The Audio Processing Transform Algorithms are also implemented in software with the purpose of analyzing the audio stream and providing control signals for the VibraPad. The Audio Processing block passes information to both the Graphical User Interface and the I/O Controller. For this initial prototype, the software block will be run on a PC and an external I/O control board will be used to drive the vibrotactile transducers. A commercial device, however, will have its software integrated onto a mobile platform.

I/O control from the PC will be achieved via the parallel port, which is used as input to the external control circuit. The output of the control circuit is connected to the VibraPad and used to drive the vibrotactile transducers.

The following sections describe each of the above subsystems in greater detail. The subsystems are categorized in two main subcategories: hardware and software.

3 System Hardware Design

3.1 I/O Control Board Design

3.1.1 Overview

The I/O Control Board is a PCB that will be used to interface the parallel port of the PC with the vibrotactile transducer array. Control signals sent from the PC through the parallel port will be used to enable or disable the drive circuitry of individual transducers. The I/O Control Board must provide sufficient power to operate all 16 transducers in the array as the parallel port cannot source much current. A simple user interface will also be provided to allow the user control over the vibration intensity of the transducers and to turn the board on and off. Figure 3-3-1 depicts a high level block diagram of the I/O Control Board. Appendix A - Schematics contains the schematic for the I/O Control Board.

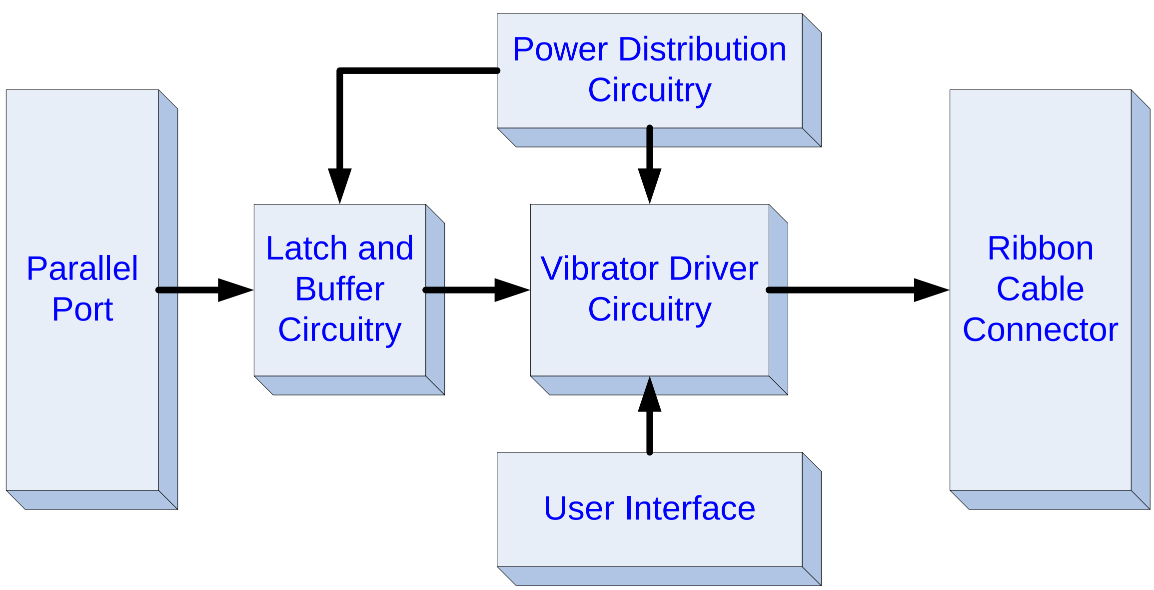


Figure 3-3-1: I/O Control Board Block Diagram

3.1.2 Parallel Port Interface

The control signals coming from the parallel port consist of eight Data signals, D0-D7, and two select signals, SEL_1 and SEL_2. These signals are the inputs to the Latch and Buffer Circuitry block of the I/O Control Board. The D0-D7 signals are used as the inputs to two octal latches, 74S373 IC’s. The high output voltage level of the parallel port pins is 5V, which the TTL latches are capable of handling without sustaining damage. The SEL_1 and SEL_2 signals control when each of the latches acts transparent and when they latch the inputs. The SEL signals are required to time multiplex the eight Data signal such that control of all 16 vibrotactile transducers is possible from 8 control signals. This multiplexing scheme is implemented in software and is discussed in detail in the Parallel Port section later in this document. A block diagram of the latch and buffer circuitry is shown in

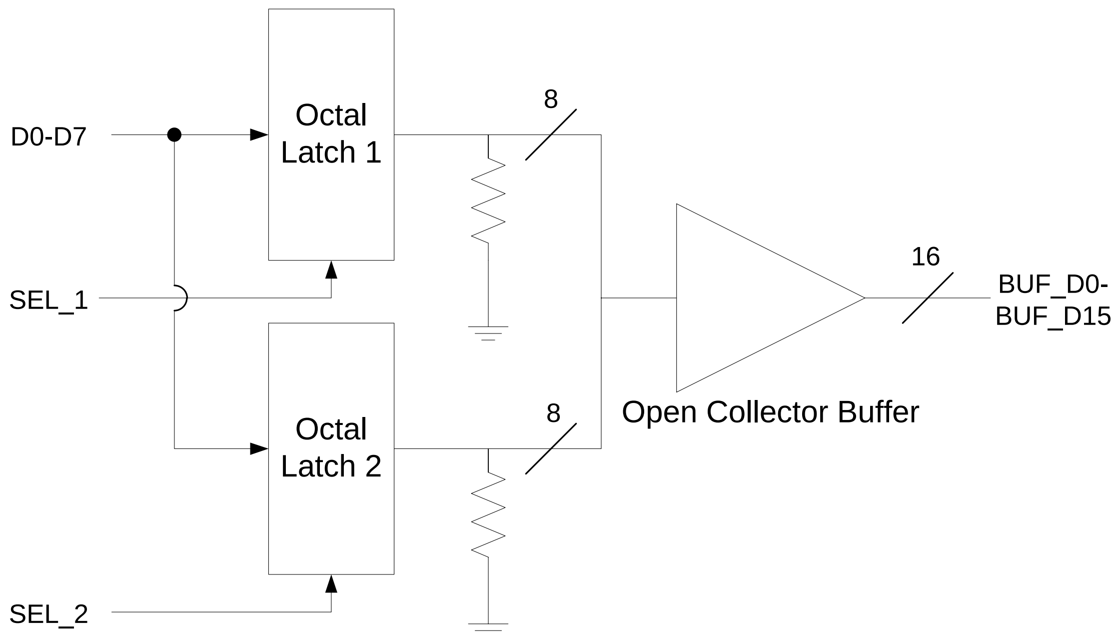


Figure 3-3-2: Latch and Buffer Circuitry

The 16 output signals of the latches are fed into buffers with open collector outputs. The open collector outputs are necessary for the vibrator driver circuitry. Three 7407 IC's, hex buffers with open collector outputs, will be used. The pull down resistors on the outputs of the octal latches are required to prevent the transducers from turning on when power is applied to the board and no parallel port is connected. If these resistors are not present then BUF_D0 – BUF_D15 would float when the latch inputs are not driven. A high impedance buffer output turns on a transducer, which is undesirable when no software control is present.

3.1.3 Power Scheme

The power scheme for the I/O Control Board requires that sufficient power is provided to drive all 16 transducers at full output as well as the various IC's necessary for software control. To accomplish this, a 12V, 1.25A wall adaptor will be used as the board's power source. As we are driving 16 vibrators, which are essentially 16 small motors and motors are current driven devices, we must ensure that our supply can source enough current to meet our needs. Table 3-1 summarizes the major current consuming components on the I/O Control Board.

Component	Current (mA)	Quantity	Total Current(mA)
Paging Motor (Vibrotactile Transducer)	50	16	800
74S373 Latch IC	80	2	160
7407 Buffer IC	30	3	90
TOTAL			**1050**

Table 3-1: Component Current Consumption Summary.

There is an extra 200 mA available from the wall adaptor for losses and smaller components. Since it is unlikely that all 16 vibrators will be active at full intensity at the same time, there will typically be more excess current available.

Two voltage regulators will be used on the I/O control board. The first one is a LM7805 5V regulator to provide VCC for the logic IC's. The second regulator is a LM317 adjustable regulator, it will provide a user controlled voltage rail for powering the transducer array. The adjustable rail will range from 1.2V to 4V, and the IC is capable of providing up to 1.5A (assuming we had 1.5A available from the wall adaptor). However, we only need 800mA from the rail in a worst case scenario. The control scheme for the adjustable regulator is shown in Figure 3-3.

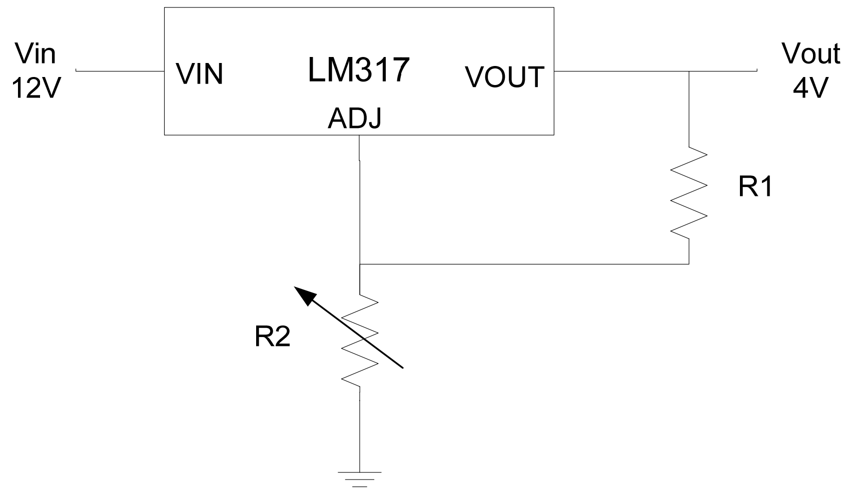


Figure 3-3: Control of the adjustable voltage regulator LM317. It is a 3 terminal device; Vin, Vout, Adj. And attempts to maintain a 1.2V difference between Vout and Adj.

The LM317 datasheet [1] provides Equation (1) for determining the output voltage of the regulator.

$$V_{out} = 1.25\left(1 + \frac{R_2}{R_1}\right) + I_{adj}(R_2) \tag{1}$$

We required V_{out} = 4V, and selected R_1 to be 470Ω. We will assume I_{adj} is small, and neglect that term from the equation. This assumption is acceptable because we do not need our V_{out} rail to be exactly 4V, the vibrators operating characteristics do not vary greatly when the voltage rail is around 4V. Solving for R_2 we get 1.03kΩ, or approximately 1kΩ. Now the R_2 symbol in Figure 3-3 represents a potentiometer. In our design R_2 will be a linear varying, 5kΩ maximum, potentiometer and a fixed resistor in parallel to reduce the maximum resistance of R_2 to the 1kΩ we require. The fixed resistor value is determined using Equation (2) to be 1.2kΩ.

$$1k\Omega = \frac{R_{fix} R_{pot}}{R_{fix} + R_{pot}} \tag{2}$$

With this addition our adjustable voltage rail can be tuned by the user using the potentiometer from 1.2V to 4V, the higher the voltage on the line, the more intense the vibrations. This adjustable rail is called VDD in the schematic, and will be referred to as such for the remainder of this document.

3.1.4 Vibrator Drive Control

The datasheet for the paging motors we intend on using as vibrators was lacking in details (see Appendix B – Vibrator Data Sheet). It specified that a maximum of 3V should be used to drive one vibrator. Since motors are largely current driven devices we desired to know how much one vibrator would draw. Thus, to help facilitate the design process a sample paging motor was obtained and its current voltage characteristics were measured. Figure 3-4 is a graph of the current drawn by the paging motor, I_{vib}, versus the power supply voltage, VDD, applied across its terminals.

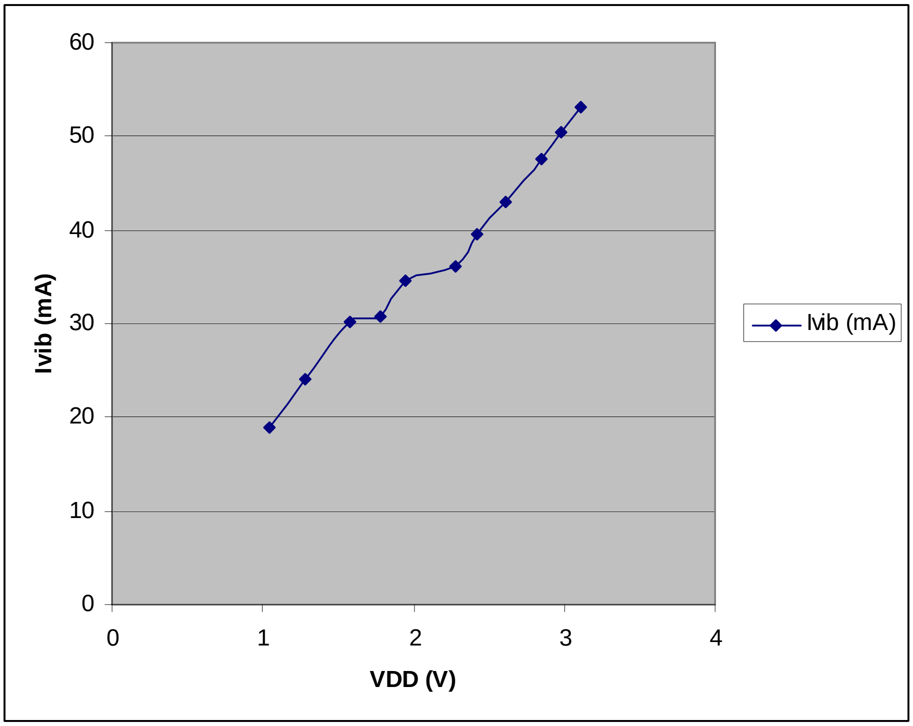


Figure 3-4: Characterization of the Vibrator. I_{vib} is the current drawn by the vibrator from a power source with a voltage of VDD.

With VDD = 2.98V, the current draw is 50.4mA. The motor appears to work with higher voltages as well, but the difference in how the vibrator feels at these higher voltage levels is comparable to the 3V operation point. Also, the datasheet specifies a maximum voltage of 3V. Therefore the maximum voltage to be applied is 3V and the expected current draw is 50mA.

Each drive circuit contains a BJT, a resistor, and one of the BUF_D signals. Figure 3-5 shows a diagram of an individual vibrator drive circuit.

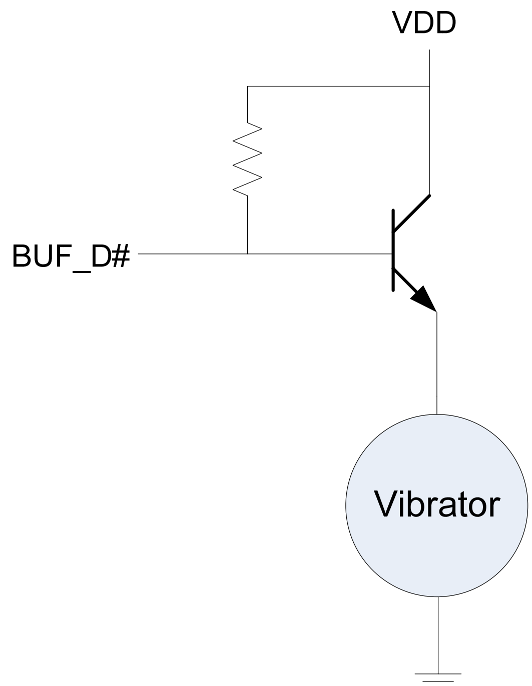


Figure 3-5: Vibrator Drive Circuit

The BJT is a 2N3904, a general purpose NPN transistor. The resistor between the base and collector is 100Ω. From the 2N3904 datasheet [2], with I_c = 50mA and V_{CE} = 1V the DC current gain, h_{fe} has a minimum value of 60. This is the operating point we desire as the collector is tied to VDD, which is 4V, so the V_{CE} drop of 1V yields 3V at the vibrator terminals, and I_c = 50mA is sufficient to drive the paging motor. The h_{fe} of 60 means the base will be drawing approximately 0.83mA, resulting in a .083V drop over the 100Ω resistor.

The BUF_D# signal is one of the 16 open collector output signals from the buffers. When software wants to turn the buffer off this signal is driven low, turning the transistor off. When software wants to turn this pin on it asserts a high, which means the buffer output will be high impedance. This allows the resistor to pull the base voltage up and turn the motor on.

The outputs of these drive circuits connect to a 2x40 pin ribbon cable connector. The ribbon cable connects to the VibraPad and powers the transducer array.

3.1.5 User Interface

The user interface for the I/O Control Board is quite simple. It includes the potentiometer previously mentioned for adjusting the intensity of the vibrators. There is also a switch for turning the board on and off. This switch will disconnect the wall adaptor power from the rest of the board. Finally there are two LED's that will light up to indicate that the two regulators are on, this will only occur if the wall adaptor is connected to the board and the switch is in the on position.

3.1.6 Fabrication

We plan on fabricating the I/O Control Board using a presensitized PCB and etching chemicals. This was chosen to reduce costs as initial inquiries about prototype pricing at fabrication houses were about $200 for two layer boards. Since we are etching the board ourselves we will need to ensure that the traces are quite large in the layout to allow for any over etching that may occur, and because our mask resolution will be quite low. The masks themselves will be on transparencies. Also, all mounting holes and via's will need to be drilled by hand, and all via's will have to be connected by hand as we have no solder bath available.

3.2 Physical Enclosure (VibraPad) Design

The physical enclosure referred to as the VibraPad consists of two components: a belt and an array of vibrators. The belt functions to secure the transducer array in a specified pattern to ensure proper skin-to-surface contact with the vibrators. The array will be worn facing the torso due to the available area and density of nerve endings within this region. The VibraPad will also act to shield the vibrators from direct impact and to ensure mobility for the user.

The initial prototype of the belt will be constructed with dimensions that cover a maximum area of the torso so as to accommodate a variety of arrays for purposes of testing. Because of its size, testing will likely require that the user remain relatively still (either standing or sitting up straight) to ensure proper skin-to-surface contact. After preliminary tests determine the best vibrator layout on the belt, each transducer will be secured in place and their wires passed through small slots at the front of the pad to prepare the prototype for demonstration purposes.

3.2.1 Vibrator Specifications

For the transducer array, we will utilize the Vibrating Disk Motor VPM2 by Solarbotics. These devices were selected for their size (12mm in diameter) and cost. The data sheet is included in Appendix B – Vibrator Data Sheet for reference. Due to the fragile nature of the wires and solder contacts, this area will be taped to strengthen the connections and ensure they last throughout the testing process.

3.2.2 Belt Design

The belt component of the VibraPad functions to secure the vibrators in repositionable arrays around the waist of the user. It provides the ability to place the vibrators in various patterns on the torso to determine the array that will produce the best results among the most test subjects. This flexibility is important to determine the minimum distance between vibrators that will allow a user to distinguish between vibrations at different frequencies. It is also necessary so as to determine the ideal number of vibrators needed to produce the desired results. The length of the belt must be able to accommodate persons of average size while ensuring proper skin-to-surface contact with the surfaces of every vibrator.

Construction of the belt consists of three sections connected by elastic as illustrated in Figure 3-6; where the circles represent individual vibrators arranged in one of several possible arrays. The center section will be covered with the softer "catching" Velcro layer to avoid irritating the skin during testing. This layer ensures the ability to reposition, add, or remove vibrators as necessary. The remaining two sections are layered with three strips of Velcro and will wrap around the waist of the test subject.

Each vibrator of the finalized array will be secured in place and the center partition covered with an overlay with holes corresponding to the position of each vibrator so that they maintain contact with the skin. This final touch is to improve the overall comfort and appearance of the device for demonstration purposes.

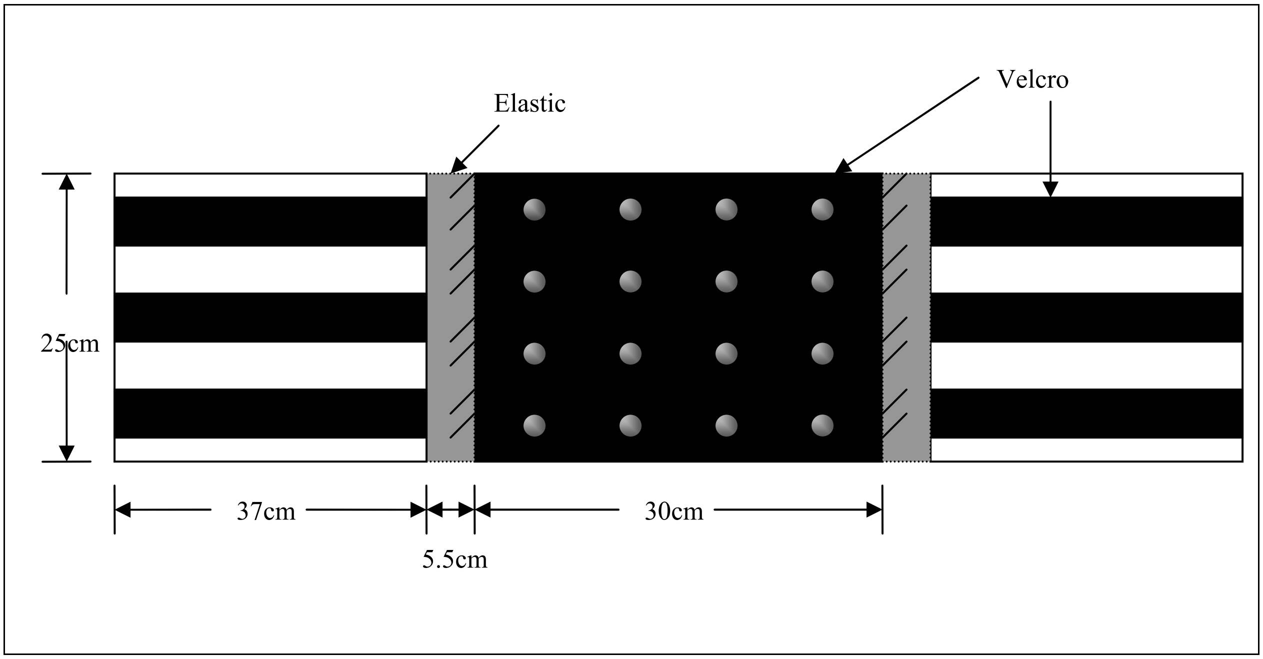


Figure 3-6: VibraPad Belt Design

3.2.3 Vibrator Placement Schemes

Throughout the testing process, it is important to have the flexibility to reposition, add or remove vibrators as necessary to determine the layout that will produce the most desirable results in each test case. While testing may produce a completely unique array that provides the most consist results among the chosen test subjects, we have devised two possible patterns with which to begin testing.

3.2.3.1 Evenly Spaced Pattern using 16 Vibrators

The array illustrated in Figure 3-7 places 16 vibrators evenly over the entire space of the center pad. Research [3] has shown that for a vibro-tactile stimulus on the back, the simultaneous two point discrimination threshold is 1.1 - 1.8 cm. This is the distance between two transducers required for each stimulus to be distinguishable. Although our project utilizes vibrotactile stimuli on the abdomen, we will assume the skin's ability to distinguish between individual points is approximately the same. Because our transducers have a diameter of 1.2cm, the distance between vibrators will need to be greater than the recommended 1.1 - 1.8 cm. The pattern shown in Figure 3-7 allocates 4.5cm horizontal spacing and 3.5cm vertical spacing between transducers; it utilizes the given pad size most efficiently and provides a good starting point for testing.

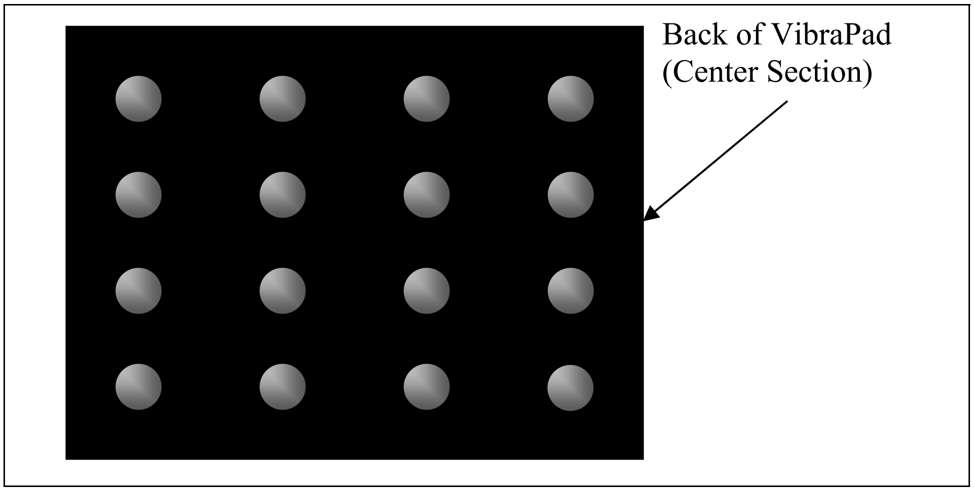


Figure 3-7: Evenly Spaced 16 Vibrator Array

3.2.3.2 Offset Pattern using 14 Vibrators

It is suspected that the array shown in Figure 3-7 may cause desensitization of the skin if transducers in the same column or row are activated for extended periods of time. The offset between transducers of the array shown in Figure 3-8 accommodates for the possibility of desensitization. By offsetting the vibrators from each other, the same area of the skin will not be vibrated constantly in the case of similar sound stimulus. This pattern requires the removal of two vibrators to accommodate the offset which will result in a smaller range of available frequencies. To compensate, the vibrators associated with the highest and lowest frequency bands will be removed since these bands are not associated with the majority of sound encountered on a daily basis. While this array does not provide the most efficient usage of space, the added distance between vibrators may allow a user to more effectively distinguish between vibrations. Testing will determine the effectiveness of the resulting reduced frequency range.

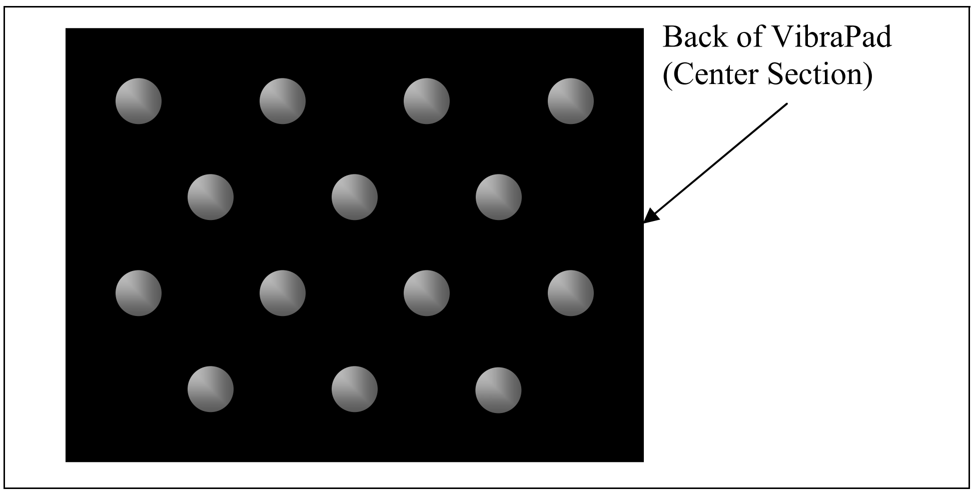


Figure 3-8: Offset 14 Vibrator Array

3.2.4 Wiring Pattern

To allow for the repositioning of vibrators during the testing phase, the back of the VibraPad will be "open" and as a result, the wires will be arranged loosely. However, when the pattern has been finalized, the wires of each vibrator will be passed through slots located on the front of the pad at intervals determined by the placement scheme. Figure 3-9 illustrates the general concept assuming the array pattern illustrated in Figure 3-7. The final wiring pattern will be dependent upon the number of vibrators used and their placements. Wiring through the front of the pad will provide a better aesthetic for demonstration purposes and allow easy access and identification of individual vibrator connections.

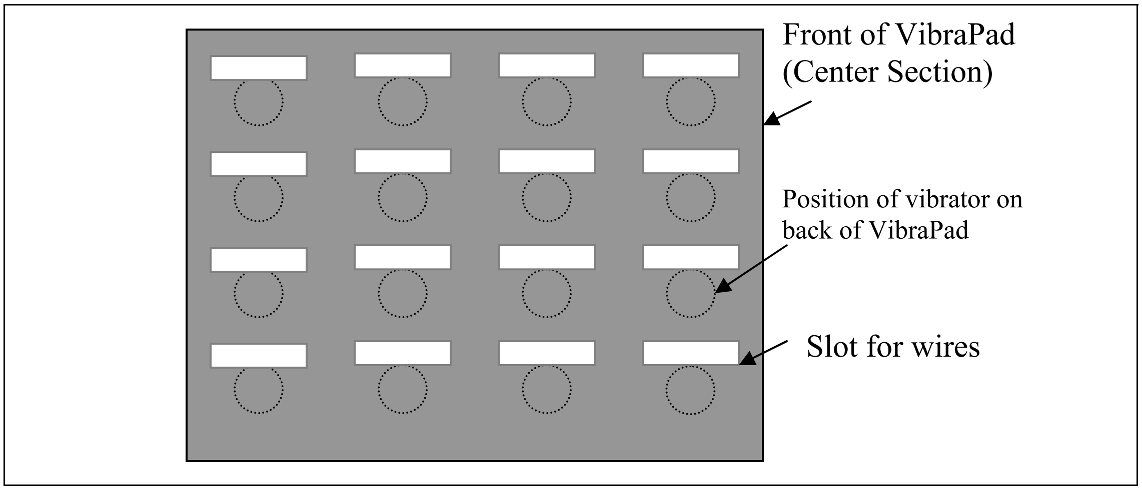


Figure 3-9: Wiring Pattern

4 System Software Design

4.1 Architectural Overview

The software follows an event-driven architecture. At any given moment there are several events occurring. There is incoming audio from the microphone/line-input device, there is outgoing audio to the speaker system, there is outgoing data to the parallel ports, and there are GUI events to be handled such as paint event and button presses. There are also additional events created on initiation and destruction of the main program window.

Since we require access to the wave input and wave output device on Windows there were two candidates: DirectX or Windows Multimedia. Windows Multimedia is the simplest choice by far and integrates well with Windows API for GUI elements and GDI for drawing. WindowsAPI [4], GDI [5], and WinMM [6] are also very fast and safe to initialize across all windows versions since Windows 95. This will allow the required startup time of under 3 seconds as per required by the functional specifications. The natural choice in programming language for these chosen libraries is C though C++ will be used to bring additional libraries from Boost [7] and standard C++ [8] to simplify things greatly.

A standard WinAPI WindowProcedure message loop provides the main loop for the program where events are processed. The WinAPI window events used are WM_CREATE, WM_DESTROY, WM_TIMER, WM_PAINT, and WM_COMMAND. Most events are posted to this loop while some events have dedicated callbacks and run in their own thread.

The event WM_TIMER is sent roughly every 15ms. It polls the stream buffer to see if there is data to extract. This value was chosen because it prevents buffer overflows by extracting values too fast. It is also roughly the resolution of the regular performance windows timer due to system level scheduling. Whenever data is extracted it is passed through the transform methods and the appropriate graphical display elements are updated. The buzzer values are also set using the parportIO functions. One of them represents the time domain values and is shown as an oscilloscope. The second one represents the FFT and displays a log-frequency vs amplitude plot. The third display shows the status of the buzzers graphically. Each buzzer will be shown as a numbered circle. The fill colour of the buzzer will represent the status. Black represents OFF and the colours green, yellow, and red represent increasing duty cycle for the PWM of the buzzer.

The stream class is a buffer for incoming and outgoing audio for the application. It allows data to be sent to it at a different rate and from different sources than data can be pulled from it. This offers great design flexibility and further optimizations. It behaves like a queue except that push and pop are designed for reading and writing blocks of data rather than single elements. It was made thread safe since all push and pop events are made atomic by having scoped mutex locks. The combination of getSize() and pop() is not an atomic operation. Thus, to simplify things, pop would check internally and return false if it could not pop and return true if it did thus creating an atomic check-pop operation. There are three stream objects. There is an input stream class, a playback stream class, and a recording stream class. The main stream class is the central buffer from which all transform data is pulled. Here is where the microphone input is added, and wav files are added. The streamOut object is a short buffer used for sending the samples currently being transformed and displayed to the speakers. Data that is transformed is added to data out and then sent to the PC speakers in the same way that audio is acquired from the input device. The recording stream receives a copy of incoming microphone data if the program is in the recording state. Data is not pulled from this until recording has stopped. When it is stopped, the data is written to a wav file and the buffer is cleared.

In order to keep pulling from buffers at a steady rate a millisecond resolution timer is utilized. The number of samples pulled from the buffer is compared to the total time elapsed in milliseconds. The 15ms WM_TIMER interval ensures that polling is rapid enough to prevent overflow and pulling from the buffer too slowly. This long-term timer ensures that data is not pulled too fast. This long-term timer does not send events or evaluate a callback; it merely returns the time elapsed.

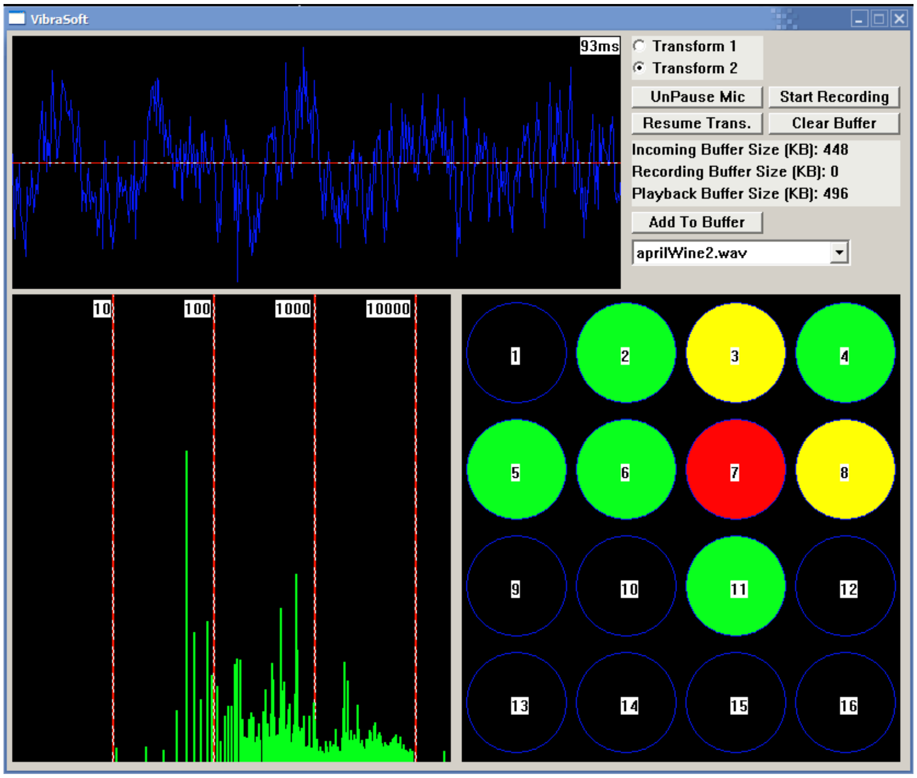


Figure 4-1: Software GUI Screenshot

Figure 4-1 shows the GUI layout. It is in a window set to a fixed size of 800x700 pixels. The pause microphone button pauses microphone input. The pause transform button will pause the pulling of data from the main stream. Clear will clear all three streams. There are two transform radio buttons. Switching the transform will only become active when the next set of samples is transformed. The button "Add To Buffer" will append the data from the wave file to the end of the main stream. The wave file is located in the wavs directory. The combobox shows a list of the available wav files. This list is only updated on initial load of the program.

The start recording button enables recording from the microphone input. When clicked it will change text label to indicate it is used to stop recording as well. When it is clicked in the stop recording state it will dump the recording buffer to a hard-coded "recording.wav" file on disk. The user will have the option to rename the file using the standard file manager.

There are three text labels showing performance statistics for the audio/transform streams. This shows the size each stream is currently using in KB's. It is constantly updated during every WM_TIMER event.

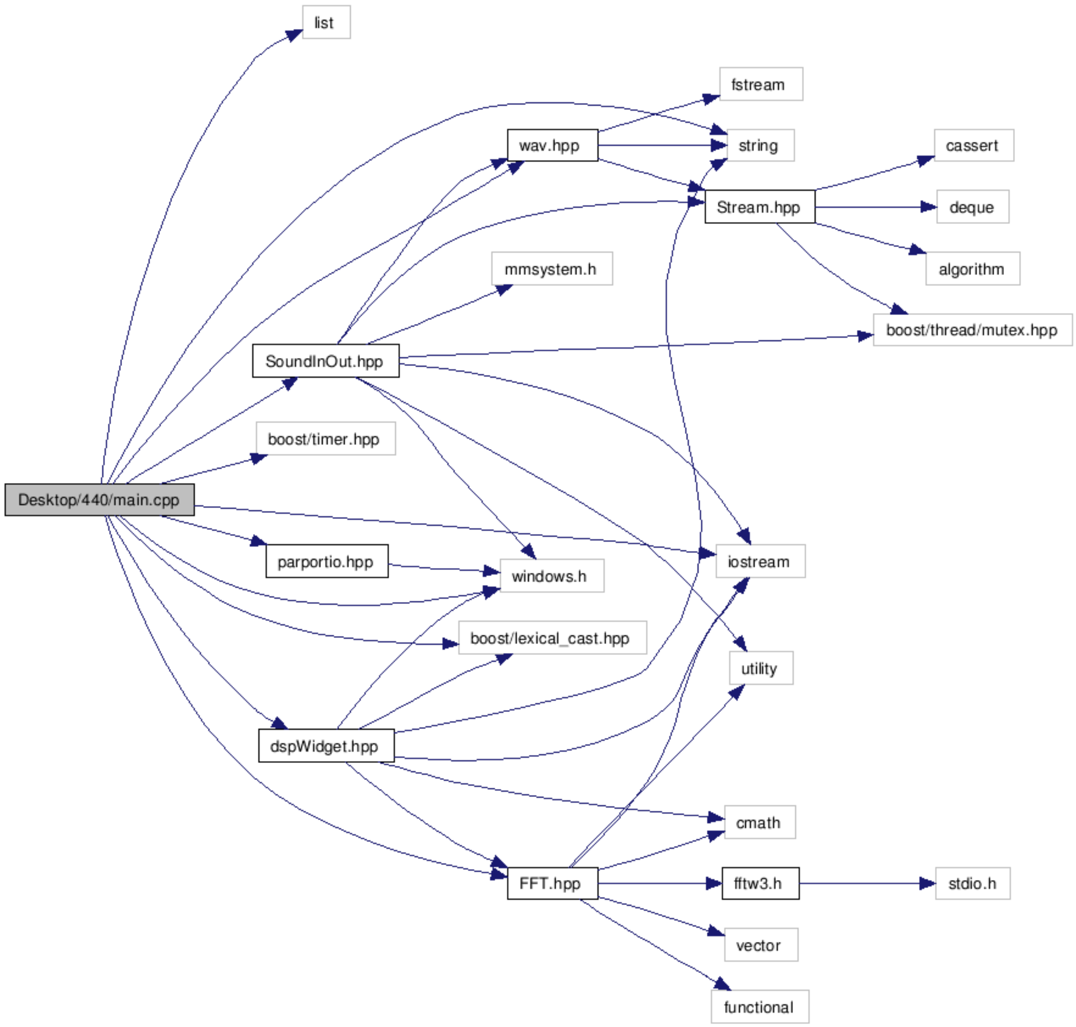


Figure 4-2: File Dependency Graph for thFigure 4-3e main file.

Figure 4-2 shows the dependency graph for the main file. The complete-detail documentation for the class, method, and test units are available in html format upon request.

4.2 Input/Output Sampling

Input was sampled as standard uncompressed PCM wav audio at a sampling frequency of 44100 Hz and 16-bit mono. The input device is automatically chosen by Windows Multimedia as the primary input device which is set by the user in windows. Typically this is the microphone but may also include line-in, cd-in, etc.

The sound sampling was done in a separate thread handled by windows and the audio device driver. After 250 samples have been acquired it evaluates the callback callbackFull(). The system is triple buffered to avoid skipping and gaps on the incoming audio. The incoming audio would then be added to the appropriate stream container for later retrieval and processing. The sample rate of 250 was carefully chosen to minimize delay between acquisition and processing while allowing the windows scheduler to correctly process without lagging. Windows does not have a real-time scheduler.

The sound out is currently handled alongside the transform. Whenever data is pulled to perform a transform data it is also sent to the output stream. The output audio methods pull the data from this stream and send it to the output audio device. The wave output is done using 8 audio buffers with a buffer size of N/8. This relationship corresponds nicely with the display being shown on screen and the buzzer being shown. The sound out is helpful to people performing tests and designing experiments. If there is a stream output underflow the callback loop will be maintained by sending strings of '0' to the device. This will prevent stuttering by sending complete silence to the device.

Figure 4-4 demonstrates the dependency graph. The SoundInOut class is implemented using the singleton design pattern. The resources are allocated using lazy instantiation and are freed by calling the destroy method. Figure 4-5 demonstrates the call graph. Figure 4-6 shows the UML class diagram for SoundInOut and Stream.

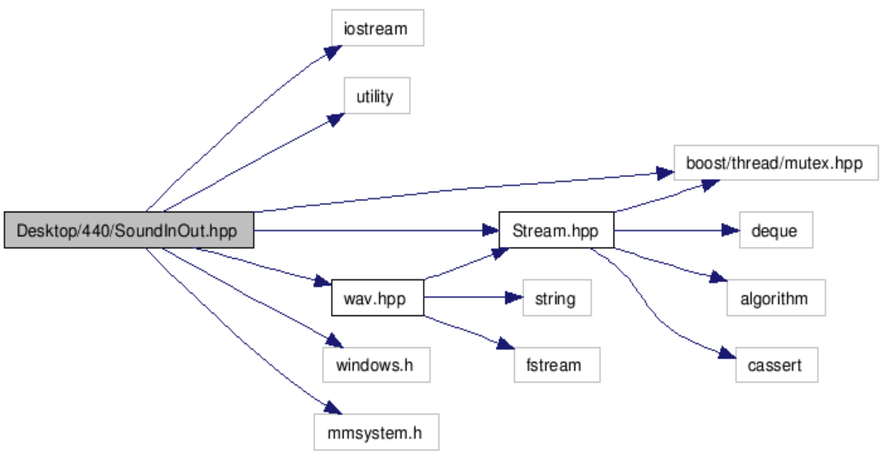


Figure 4-4: SoundInOut.hpp Dependency Graph

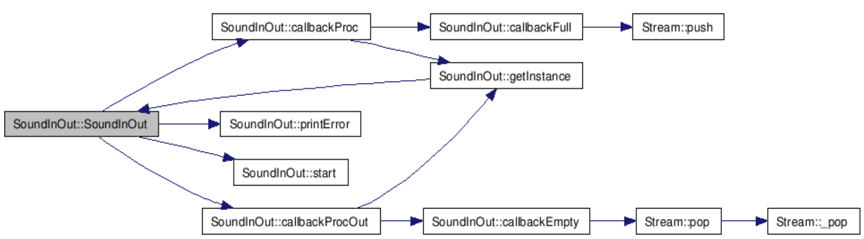


Figure 4-5: SoundInOut Call Graph

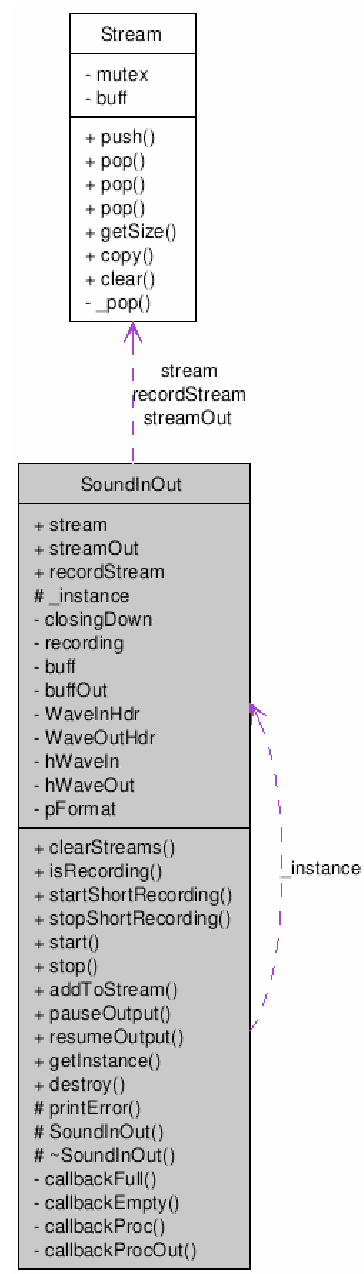


Figure 4-6: SoundInOut UML Class

4.3 FFT

The FFT code falls under the Transform namespace. This namespace includes a windowing class, an FFT class, and several transform functions for pre and post-processing the FFT data. The windowing class will contain coefficients for both a rectangular and a Hamming window. The Hamming window was chosen because of its excellent frequency isolation features.

The FFT class wraps the powerful FFTW [9] library. In particular the r2r_1d function was used. It takes in real-valued input data while producing output in the half-complex format. It is a very efficient library and FFTW is capable of determining the fastest implementation on a per system basis.

A frame size of N = 4096 was used. This is a power of 2 as required for efficiency purposes by FFTW. This is a common requirement across most FFT implementations. The frame size N was chosen because it offers the best combination of frequency and temporal resolution. If N was too small the frequency resolution would be very poor while if N was too large the frequency composition would change too rapidly and the temporal resolution would suffer. This value of N produces a period between transforms of approximately 93ms.

The FFT itself performed in the construction of the class. Then the genPVector member will generate an array of points representing the frequency in Hz and raw magnitude. Several transforms exist in the transform namespace to calculate the logarithm of the frequency, the dB of the amplitude, and threshold out low and numerically unstable values. Figure 4-7 shows a block diagram of the FFT namespace.

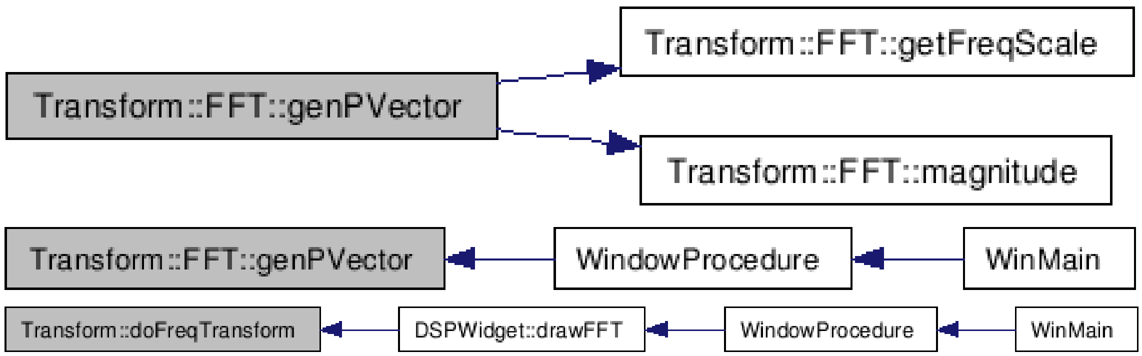


Figure 4-7: FFT Call Diagram

4.4 Wav

The program is capable of reading and writing wav files using the standard RIFF header and then subsequent PCM header and data sections. It will be used to write the recording stream to a wav file and used to load wav files into the input stream.

In the following figures the call and caller graphs are demonstrates for load and save. In these figures main is a function from one of the test benches.

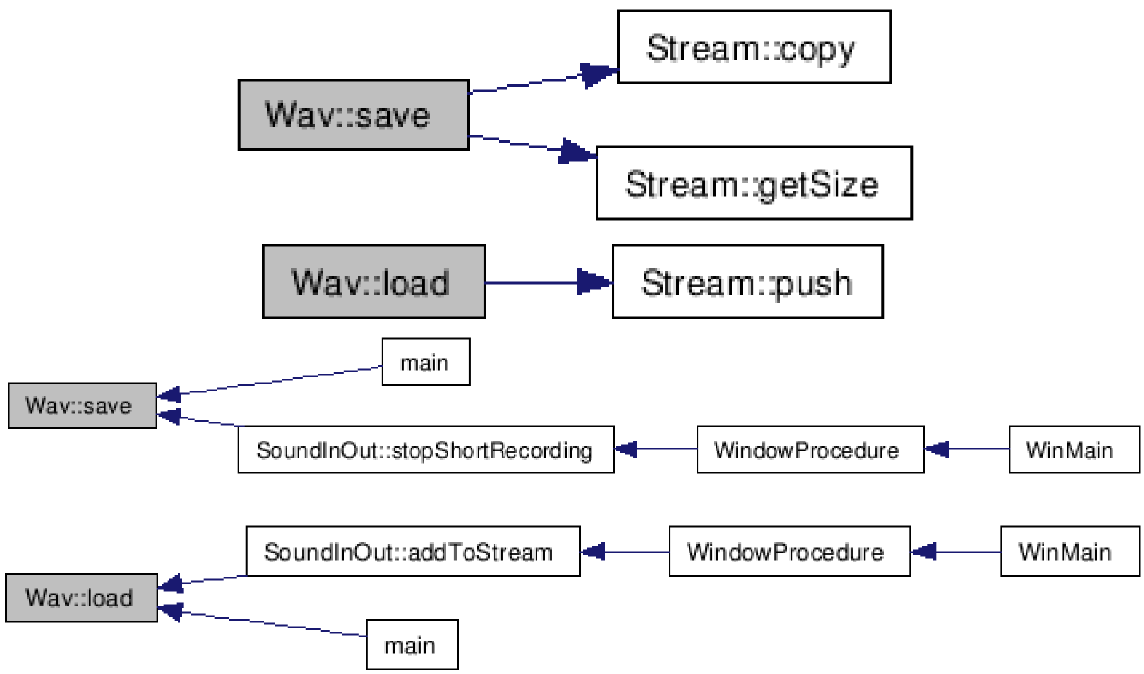


Figure 4-8: Wav Namespace Call diagrams

4.5 Drawing

There are a few controls on the window that are hand drawn. The buzzer display, the oscilloscope, and the FFT display are all drawn using GDI using the double buffering technique for smooth display. Whenever a transform occurs the appropriate dspWidget object is called to draw the data to a memory bitmap. The entire rectangle is invalidated sending the WM_PAINT message. Windows uses a dirty rectangles scheme to update the screen display. Whenever the WM_PAINT message is sent the dirty rectangle is updated using the blit function. And HDC for the window obtained in the WM_PAINT provides the surface to blit to. Windows itself also sends the WM_PAINT message under many situations including when another windows is being dragged on top of it or the window is maximized. Figure 4-9 shows the DSPWidget namespace organization.

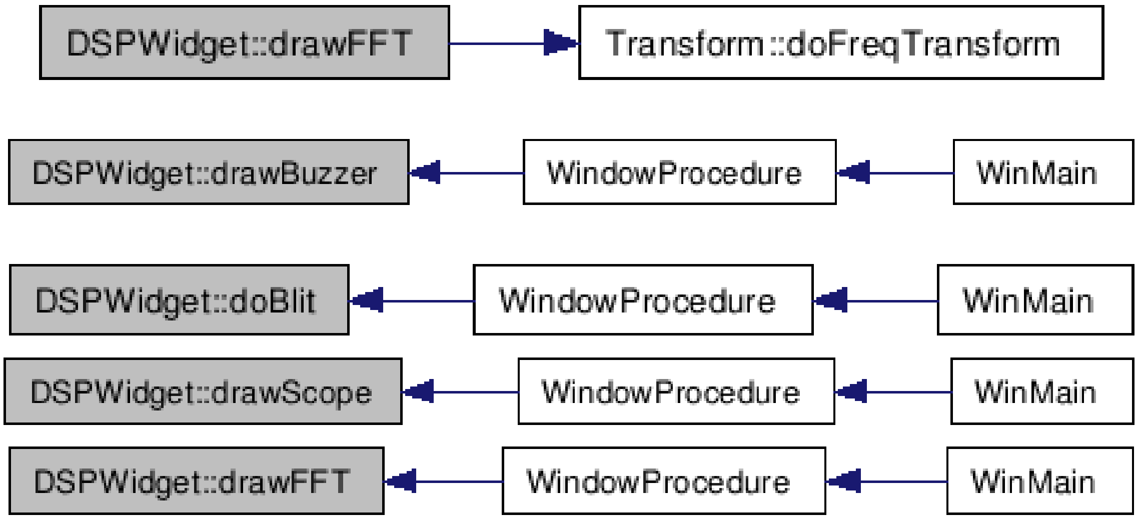


Figure 4-9: DSPWidget Namespace

4.6 The Transform Mapping

4.6.1 Critical Band Transform

Human hearing has a typical range of 20Hz-20kHz [10]; each frequency within this span can be categorized into one of 24 critical bands of hearing. The transform will utilize the Critical Band scale to categorize incoming sound stimuli into one of the 24 critical bands.

4.6.1.1 The Bark Frequency Scale

One method used to determine the frequencies of the Critical Band scale is the Bark Scale [11]. The 24 frequencies are summarized in Table 4-1.

Table 4-1: Critical Band Frequencies based on the Bark Frequency Scale [11]

Critical Band	Edge Frequency (Hz)	Center Frequency (Hz)
1	100	50
2	200	150
3	300	250
4	400	350
5	510	450
6	630	570
7	770	700
8	920	840
9	1080	1000
10	1270	1170
11	1480	1370
12	1720	1600
13	2000	1850
14	2320	2150
15	2700	2500
16	3150	2900
17	3700	3400
18	4400	4000
19	5300	4800
20	6400	5800
21	7700	7000
22	9500	8500
23	12000	10500
24	15500	13500

4.6.1.2 Mapping

Although the Critical Band scale spans the range of typical human hearing, most sounds encountered on a daily basis fall below 4kHz. Research has indicated that sounds are most easily distinguishable from each other at frequencies below 500Hz [12]. The human ear functions best with sounds in the range of 2kHz – 4kHz where the average human voice spans from 500Hz – 2kHz [10]. As frequencies increase, the bandwidths of sound increase significantly such that tones at high frequencies must be very different to be distinguishable [12]. For this reason, the critical bands above 4kHz will be grouped together and designated to high-pitched noises, typically associated with sirens, alarms, or other emergency indicators. The frequency bands will be mapped as specified in Table 4-2 to one of 16 vibrators on the transducer array illustrated in Figure 3-7.

Table 4-2: Critical Band-Vibrator Mapping

Frequency Range (Hz)	Associated Vibrator
0-200	1
201-400	2
401-630	3
631-770	4
771-920	5
921-1080	6
1081-1270	7
1271-1480	8
1481-1720	9
1721-2000	10
2001-2320	11
2321-2700	12
2701-3150	13
3151-3700	14
3701-4400	15
4401-15500	16

The critical band transform will be implemented under the Transform namespace as a function for post-processing of the FFT data. It will accept as an input an array of frequencies in hertz (generated with genPVector) and turn ON the vibrator associated with the corresponding band. The algorithm for this process is illustrated in Figure 4-10.

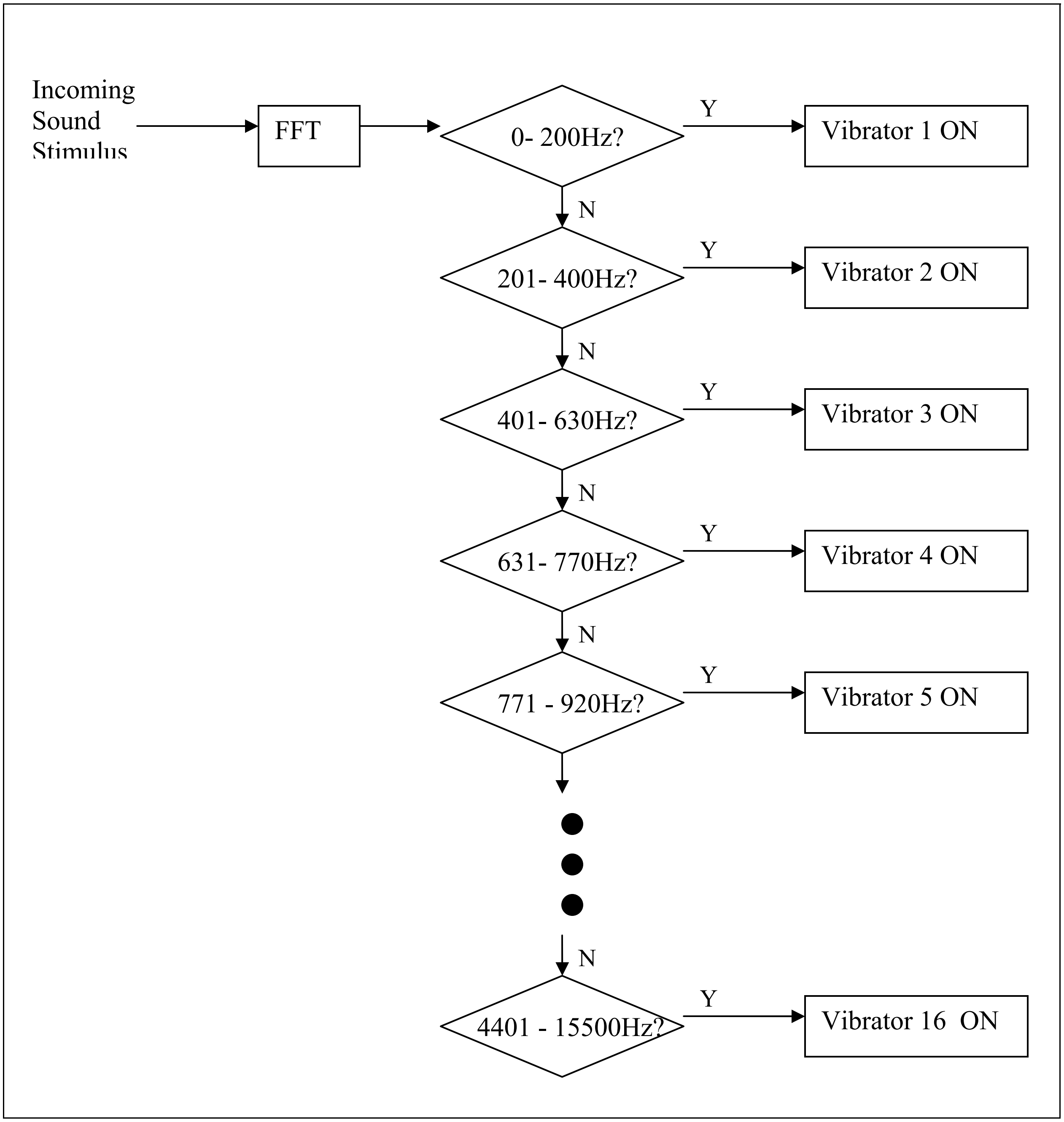


Figure 4-10: Flow Chart for Critical Band Mapping

To perform the distribution shown in Figure 4-10, the frequencies for the incoming stimuli will be compared with the lower limits of each range specified in Table 4-2 until a condition is found to be false. At this point, the vibrator associated with the range one level prior to the "false" condition will be turned on. For the first level, if the incoming freqency is found to be below 0Hz, no vibrators will be activated since there should be no negative frequencies. The pseudocode for this process would be as follows:

```
Check if frequency of stimulus • 0Hz
if 'true' check next level
if 'false' No vibrators turned ON

Check if frequency of stimuls • 201Hz
if 'true' check next level
if 'false' turn vibrator 1 ON

Check if frequency of stimuls • 401Hz
if 'true' check next level
if 'false' turn vibrator 2 ON
.
.
.
Check if frequency of stimuls • 4401Hz
if 'true' turn vibrator 16 ON
```

In the case of the final vibrator, if the input frequency is determined to be greater than or equal to 4401Hz, vibrator 16 will be activated. There should be no false condition for this case since it would indicate a frequency below 4401Hz and should have been used to activate the a transducer at the appropriate lower level.

4.7 Parallel Port

4.7.1 Introduction

There are a few user accessible standard I/O ports on a PC. The most common are the Universal Serial Bus (USB), RS232 Serial port, and the parallel port. Both USB and RS232 are serial standards and would require specific control hardware to interpret and de-serialize the data. Considering this added layer and of complexity the choice was made to use the parallel port.

The standard IBM parallel port uses a DB25 connector as the hardware interface between the computer and the peripheral device. Of these 25 pins, 8 are used for data, 9 for various control signals, and 8 for ground. There are various different configurations for different peripherals, allowing for unidirectional and bidirectional data transmission and different data rates [13].

In its simplest configuration, known as Standard Parallel Port or SPP, the parallel port has 8 output data pins, 5 input control pins, and 4 bidirectional control pins [13]. For this project, we are not concerned with the data rate as the refresh rate for the VibraPad will be on the order of a few times per second. We are also not concerned with any inputs from the VibraPad as there are no feedback signals in our design. Hence this simple configuration will be adequate.

4.7.2 Pin Configuration

The hardware pin-out of the female DB25 port of a typical IBM PC is shown in Figure 4-11.

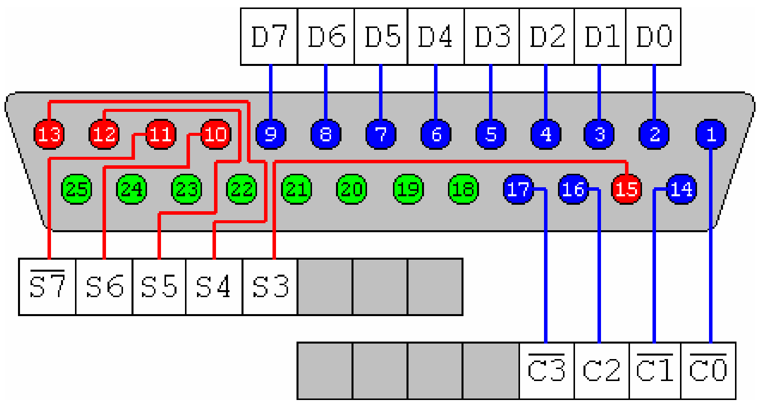


Figure 4-11: DB25 Pin-out [14]

Table 4-3 shows the configuration of each pin. The pin numbers correspond to the pin numbers given in Figure 4-11. The signal names are Standard Parallel Port (SPP) signal names as defined by the IEEE 1284 standard [13]. An “n” in front of the signal name indicates that it is hardware inverted or active low. Direction column indicates weather each pin can be used as an output, input, or both in SPP mode. The Address column indicates the memory mapped address of each pin where *Base* is the base address for the parallel port. This base address can be found by reading the computer’s communication port configurations. In most PC’s the parallel port is assigned a base address of 0x378, which is usually configurable in the bios.

Table 4-3: Standard Parallel Port (SPP) Pin-out [13]

Pin No.	SPP Signal	Direction	Register	Addr
1	nStrobe	In/Out	Control	Base+2 [0]
2	Data 0	Out	Data	Base+0 [0]
3	Data 1	Out	Data	Base+0 [1]
4	Data 2	Out	Data	Base+0 [2]
5	Data 3	Out	Data	Base+0 [3]
6	Data 4	Out	Data	Base+0 [4]
7	Data 5	Out	Data	Base+0 [5]
8	Data 6	Out	Data	Base+0 [6]
9	Data 7	Out	Data	Base+0 [7]
10	nAck	In	Status	Base+1 [6]
11	Busy	In	Status	Base+1 [7]
12	Paper-Out	In	Status	Base+1 [5]
13	Select	In	Status	Base+1 [4]
14	nAuto-Linefeed	In/Out	Control	Base+2 [1]
15	nError	In	Status	Base+1 [3]
16	nInitialize	In/Out	Control	Base+2 [2]
17	nSelect-Printer	In/Out	Control	Base+2 [3]
18 - 25	Ground	Gnd		

The parallel port interface uses TTL voltage levels. TTL logic defines a logic "low" for voltages between 0-0.8V, and a logic "high" for voltages between 2-5V [15]. The control board which directly interfaces with the parallel port uses latches that operate at TTL logic levels. This is discussed in more detail in the hardware design section of this document. It is sufficient to mention that no voltage translation circuitry is required between the parallel port and I/O control circuit since it is compatible with the hardware.

4.7.3 Port Access

In newer version of the Windows operating system, such as Windows XP, the parallel port is not directly accessible due to more stringent security measures. A low level driver is required to access the port for reading and writing data. Many such drivers are available as Dynamic Link Libraries (DLL's) and provide simple read/write capabilities to the parallel port. The DLL used for this project is *inpout32.dll* [16] and provides the following functions:

```
short Inp32(short PortAddress);
void Out32(short PortAddress, short data);
```

This DLL is integrated into the program using the `LoadLibrary` function and mapped to two local functions:

```
short inp32(short PortAddress);
void out32(short PortAddress, short data);
```

The port address is simply the address of the parallel port pin you wish to write to or read from. As mentioned in the previous section, the base address usually starts at 0x378 and each pin can be accessed using the mapping given in Table 4-3.

4.7.4 Data Multiplexing

The VibraPad consists of 16 vibrating transducers, each of which needs to be controlled by the software independently. However, the parallel port supports only 8 output data pins and hence some multiplexing is required to control each motor independently.

This is achieved through the use of the bidirectional control pins. The hardware control board contains two latches each with 8-bit data bus and single bit select pin (details explained in section 3.1.2). The data input pins of both latches are connected to the data pins of the parallel port, but each latch is controlled using a different select pin. Figure 4-12 shows the high level design.

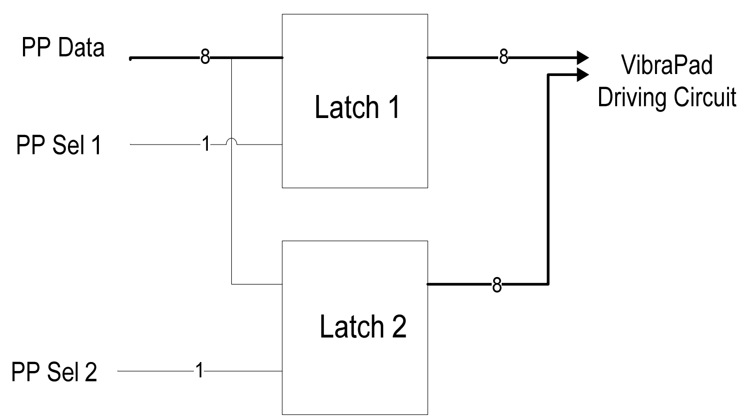


Figure 4-12: Parallel Port Data Multiplexing

During each data refresh cycle, the transform software determines the desired value of each vibrating transducer and passes these values in a 16-element array to the parallel port write function. This function goes through the following sequence of actions: select the first latch by enabling the corresponding select signal, put the first 8 bits of data on the bus, then unselect the first latch and select the second latch, and finally put the second 8 bits of data on the bus.

The set-up and hold times of the latches used are on the order of nanoseconds [17]. The `out32()` and `inp32()` functions have inherent delays on the order of 10's of microseconds. It is therefore not necessary to apply any delay in the software between the select and data outputs.

4.7.5 Pulse Width Modulation (PWM)

Up to this point, we have discussed the fact that the 16 vibrating transducers are controlled by a binary signal, which means that they only have two states: on and off. Given the nature of these vibrating motors, it is possible to change the intensity of their vibration by varying the amplitude DC voltage applied to them; however using the parallel port highly restricts this degree of freedom.

Since the amplitude of the input signals cannot be changed, we have opted to use pulse width modulation to change the intensity of vibration of each motor. Using a function generator, it has been determined that a pulse width modulated square wave can be used to change this intensity. And at a frequency of about 50-100Hz, it is possible for the skin to distinguish changes of about 30% in duty cycle. Therefore, we can have various intensities of vibration by modifying the duty cycle. If we are able to implement PWM in the software we gain an extra degree of freedom to transmit amplitude information to the user.

In terms of the software, it is simple to generate a clock signal with frequency as high as 500Hz, using a 1ms timer to toggle one of the control pins. The problem, however, lies in the latching scheme of the control board. We need to control each vibrator independently, even though each pair of vibrators is being controlled by the same data pin which is simply multiplexed via the latches.

In order to overcome this, each latch select pin will receive a software generated 500Hz clock signal. Each signal will be from a separate parallel port control pin and they will be inverted with respect to each other. Each 10 clock cycle represents 1 cycle of the output and hence the output signal will have a frequency of 50Hz. The output duty cycle is controlled by changing the input data pin when the clock of the respective latch is in the high position. This duty cycle can be set to anywhere from 0 to 100%, in 10% increments.

Figure 4-13 shows the waveform of the PWM implementation. Given the two select signals, the input signal can be toggled such that each two latch output signals sharing a single input signal can maintain different duty cycles (30% and 70% for Dout1 and Dout2, respectively). The same algorithm can be applied to control the two output signals independently for any duty cycle. The algorithm simply needs to ensure that the input signal is high only when the output signal is high AND when its corresponding select signal is high, and vice versa.

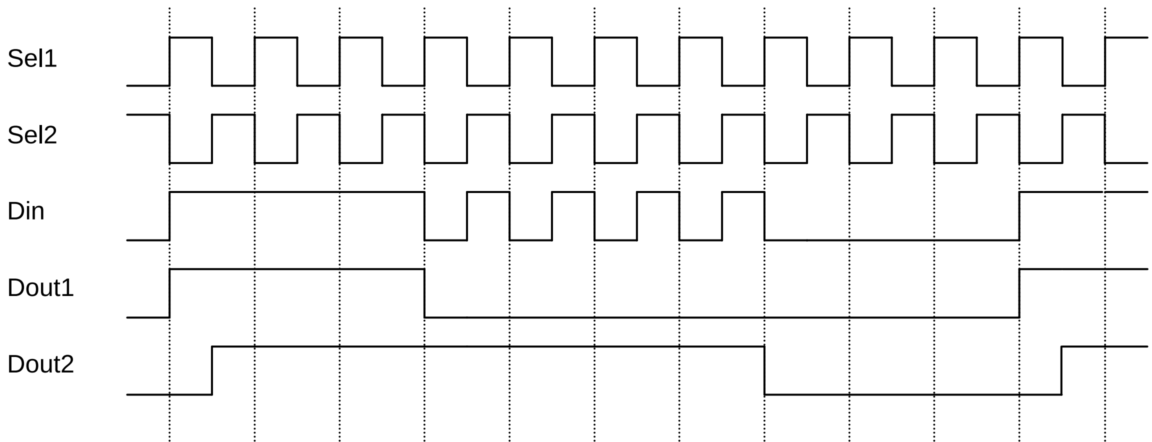


Figure 4-13: PWM Scheme Waveform

The decision has been made to use four states of vibration intensity labeled from 0 to 3. Where 0, 1, 2, and 3 correspond to the vibrating motor getting 0%, 30%, 70%, and 100% duty cycle on the output, respectively.

It is important to note that using PWM changes the software structure significantly. In the simpler Boolean implementation, it was possible to call parallel port write function from within the main software thread every time the transducer array is being updated. Using PWM, however, it is necessary to create a new thread dedicated to producing the desired select (or clock) signals and toggling the data pins as necessary. The data from the transform software is written to a shared memory location and a mutex mechanism is used by the I/O thread to read the data.

Due to this added complexity, the simpler form of the IO (Boolean form without PWM) will be implemented in the first revision of the software. After extensive testing of the PWM routine, and given sufficient time to integrate it with the transform software and resolve all multithreading issues, the PWM module will be used in the second revision.

5 Test Plan

5.1 Hardware Test Plan

5.1.1 I/O Control Board Testing

Once the I/O Control Board has been fabricated and populated several tests will be done to ensure that it is operating correctly.

5.1.1.1 Connection Check

Verify using a digital multimeter that all connections on the PCB were fabricated correctly. Specifically,

- All power and ground connections
- Parallel port to latch connections
- Latch to buffer connections
- Buffer to drive circuit connections
- Drive circuit to ribbon cable connections

5.1.1.2 Component Placement Check

Do a visual inspection to ensure that all components placed are the correct components and that all orientations are correct.

5.1.1.3 Voltage Rail Check

Apply power to the board and check that all voltage rails are at expected levels. Ensure that the switch disables power to the regulators and that the potentiometer adjusts the VDD rail voltage.

5.1.1.4 Functional Verification

Using a power supply, a digital multimeter, and one vibrator, test that each of the VIB_D# signals turns a vibrator on when D# is active high and that the vibrator turns off when D# is low. The digital multimeter can be used to measure the current being delivered to the vibrator and ensure that it is within the expected range.

5.1.2 VibraPad Testing

Initial testing of the various components of the VibraPad will be performed on an associate at Pivit Technologies. It will require a test engineer to perform the following:

- Ensure that all connections between the pad and the transducers are correct and secure.
- Ensure that all connections from the pad to the interfacing hardware are correct and secure such that sudden movements of the pad will not cause the transducers to shift or damage to the electrical connections.
- Test the operation of each transducer by applying power individually. Each transducer must operate at the lower and upper limits according to our specifications.
- Confirm that the pad can be secured to the body such that all transducers make proper contact with the skin without causing undue discomfort.
- Confirm that each vibrator can be operated at its upper limit without shifting from its array position on the pad.

5.1.3 Integrated Testing

The integrated testing will be performed with the software, I/O Control Board, and the VibraPad all connected to each other. This testing will be performed after the individual test plans have been completed and the sub modules are integrated together.

5.1.3.1 Functional Verification

A set of prerecorded sounds will be used to verify the bare functionality of the integrated system. These sounds will be tones at specific frequencies corresponding to the quantization bins used by the software FFT. When a test clip is played, it should turn on the corresponding vibrator. These tones will be used to verify the following,

- Software can control each vibrator independently
- Verify that software cannot turn vibrators on without the appropriate select signal active.
- Also verify that activating one of the first eight vibrators does not activate the vibrator in the last eight, sharing its parallel port data pin.

Once bare functionality has been verified then more complex sound clips will be played to ensure that multiple vibrators can be turned on simultaneously.

5.1.3.2 Power Verification

All 16 vibrators will be turned on simultaneously and the voltage on the VDD rail will be monitored. If all of the vibrators turn on, the voltage on the VDD rail stays constant, and nothing blows up, then the maximum power load of the system has been verified.

5.1.3.3 User Safety

To ensure that no short term unhealthy side effects arise with use of the integrated system one of Pivit Technologies associates will wear the VibraPad and have test clips played. This is to ensure that the maximum strength vibrations are not uncomfortable when pressed against a user's skin. Also, we wish to verify that the vibrators do not become so hot that they burn, that no rash develops, or any other unnecessary physical discomfort is inflicted upon the user.

5.2 Software Test Plan

5.2.1 Resource Usage

Due to the real-time nature of software, the following minimum system requirement will be required:

- CPU: Pentium 4, 2GHz or greater/equivalent
- RAM: 256Mb or greater
- HD: 100Mb or more free space
- I/O: Standard Parallel Port (SPP) capable
- SOUND: Capable of audio input and output
- OS: Windows 2000 or XP

Passing criteria: The software in its entirety (including future integration of PWM I/O) will not consume more than 10% of processor resources and 100Mb of memory at any time during execution.

5.2.2 Audio Stream Support

5.2.2.1 Audio Input/Output Stream correctness

The audio input and output will be tested for correctness from a white-box software level testing. The first test is simply to play back the audio that is incoming. Feeding an FM radio into the line-input port will provide a data source. Playback will be tested and recorded for half-hour sessions under various system usage scenarios. Playback will be examined for errors, gaps, anomalies, and stuttering. In addition, regular microphone usage will be tested using voice. Audio output will be tested under buffer underflow conditions by pausing the microphone input and allowing the buffer to empty. During underflow a null sound should be sent to the speaker. A loaded wave file will be appended using the "Add to Buffer" button. This test will ensure the sound sample will be appended to the end of the stream and any incoming audio after the button is pressed will be added after that wav file. This will be ensured correct by listening for the proper sequence of events from the playback mechanism.

5.2.2.2 Wav File Read/Write Support

There will be two simple tests aside from daily usage of the software itself. The first test is a simple test of correctness. A wav file will be loaded, stored in a stream, and then saved back to disk. A binary diff command will be run on the files to determine if even 1 byte is different.

The second test will rely upon correctness of the first and upon the software system in general. A recording will be made of a known and parameterized audio stream using the software. The program will load and play back the recording and the test will examine for any abnormalities.

5.2.3 FFT/Mapping and Controls

5.2.3.1 Pause Controls

The two pause controls are the microphone pause and the transform pause. The microphone input will be paused and then resumed. The test will examine if any data has been added to the main audio stream while it has been paused. The transform pause test will ensure that data can be added to the streams without removing data. An FM radio will be plugged into line-input. Proper functioning will allow the radio signal to be paused and resumed without missing any audio.

The FFT and associated mappings will be tested by loading wav files with a known frequency composition. These files will be generated using Audacity [18]. The predicted results will be compared with expected results. The transform pause buttons will be periodically used to examine specifics parts of the audio stream in great detail.

5.2.3.2 Clear Control

This test is quite simple. After the clear button has been pressed, all three audio buffers must be emptied. The transform and microphone input will both be paused during this test to ensure no audio is added or removed.

5.2.3.3 Recording Control

The recording functionality testing will begin by feeding a known audio clip into the input device. The recording will be stopped at a very specific point upon which the file should be written to disk. The audio file will be played back to ensure that it works properly.

5.2.4 Parallel Port and PWM

5.2.4.1 Timing

Each pin is capable of producing a square wave with maximum a frequency of 500Hz. Each pin is toggled once every 1ms. Testing will be done using a software test routine and verified using the oscilloscope. When toggling the period is set to 1ms in software, to pass this test the output signal must not deviate from 500Hz by more than 5% (25Hz).

5.2.4.2 Time Multiplexed Output

A test function will continuously call the parallel port write function with predetermined values to be written to each transducer. Output is monitored with the oscilloscope to ensure the correct select and data signals are selected with each write. To pass this test the correct select and data signal is pulsed when the corresponding vibrator is selected.

5.2.4.3 Time Multiplexing

This test is to be conducted along with the I/O control board and the VibraPad. The procedure is the same as the previous test, however the objective is to ensure the data is properly latched and held and the correct transducers turn on. To pass this test the data must be properly latched and the correct vibrator turns on.

5.2.4.4 Pulse Width Modulation

Run function continuously in independent thread. Write a test function that will update the vibrator values and ensure the output has the correct duty cycle. This test is to be done in conjunction with the latches since they are an integral part of the PWM design. Latches can be mocked up on a breadboard or the entire I/O control board can be used and the output of the latches monitored. To pass this test each latched signal must have the correct duty cycle depending on the test function. This is to be confirmed using an oscilloscope.

6 Experimental Procedures and Performance Analysis

6.1 Introduction

The main goal of this research project is to determine the effectiveness of a tactile hearing aid and the extent to which it can help someone interpret sound information using the VibraPad.

In order to quantify this goal, we plan to conduct a series of experiments which increase in complexity as the test subjects become more familiar and comfortable using the prototype and we become more familiar with its capabilities. Due to the many uncertainties surrounding the nature of this research, the data collected from initial experiments will be used to adjust the test plans and create new test cases.

Initially, we expect to conduct all the following test cases on at least two individuals, most likely members of our design team. If the results are deemed acceptable, and after refining the test plans, further experiments can be conducted on other subjects who are unfamiliar with the prototype and also hearing impaired individuals.

6.2 General Experiment Parameters

6.2.1 Setup

1. Select a quiet area and ensure comfort of the test subject.
2. PC is set up with speaker and clear sound. Software is launched and ready to be used.
3. All subsystems are turned on and properly configured for the experiment.
4. A selection of 10 wave files, each no longer than 5 seconds in length, representing every day sounds. For example these sounds can include a barking dog, ambulance sirens, fire bells, cars honking, door bells, telephones ringing, and others. The particular sound files will not be specified here, however upon selection, the same set of files shall be used for all subsequent iterations of this experiment.
5. Test subject puts on the VibraPad. Ensure all transducers make contact with the skin. The test subject may need to stand or sit on a high stool and keep back straight if necessary.
6. Run experimental setup routine, which turns each vibrator on one by one to ensure proper contact with the skin. Allow the test subject to adjust the maximum vibration level by fine-tuning the potentiometer on the control board.

6.2.2 Procedure

1. Familiarize test subject with the software GUI if he/she is unfamiliar.
2. Turn off the control board.

3. Direct test subject to listen to each 10 sound at least twice.
4. Turn on control board.
5. Direct test subject to select each sound in the predetermined order, listening to each twice.
6. Direct subject to carefully listen to the sound, observe the visual queues on the GUI interface, and pay attention to the vibration at the same time.
7. Repeat the previous 2 steps 10 times.
8. Rearrange the order of the files, or direct the test subject to select them at random.
9. Allow the user 10 minutes to listen and get familiar with the sounds in any order they wish.
10. Turn off the speaker, and direct the user to repeat the process, observing the sounds using only the visual queues and the tactile sensory.
11. Allow another 10 minutes of training without the speakers.
12. Ensure that comfort of the test subject is maintained throughout the training process. Direct test subjects to indicate if at any point they are becoming desensitized to the VibraPad.
13. After the training period, turn the monitor such that the test subject cannot see which sound file is being selected.
14. Select sound files randomly and ask the test subject to identify each clip. Do not select the same file twice. Record the number of successful identifications.
15. This test case can be iteratively performed with the same test subject in order to observe any improvement in hit rate as the result of more training. After each iteration of the above procedure, the test subject most not use the THA for a minimum of 10 minutes. In order to record the training time as a cumulative number, there cannot be a gap of longer than 48 hours between iterations.

6.2.3 Results

The results of each experiment can be quantified as hit rates. Hit rate is defined as the percentage of the number of successful sound clip identifications versus the total number of sound clips played.

To deem a series of experiments successful, a minimum hit rate of **80%** must be achieved after a maximum of **100** minutes of cumulative training, or 5 iterations of the above experiment.

6.2.4 Experiment Parameters

The previous section defines the general guidelines for conducting an experiment. The following parameters can be changed to create various different test cases:

- Signal transforms: several signal transforms are designed and they will need to be tested under similar conditions.

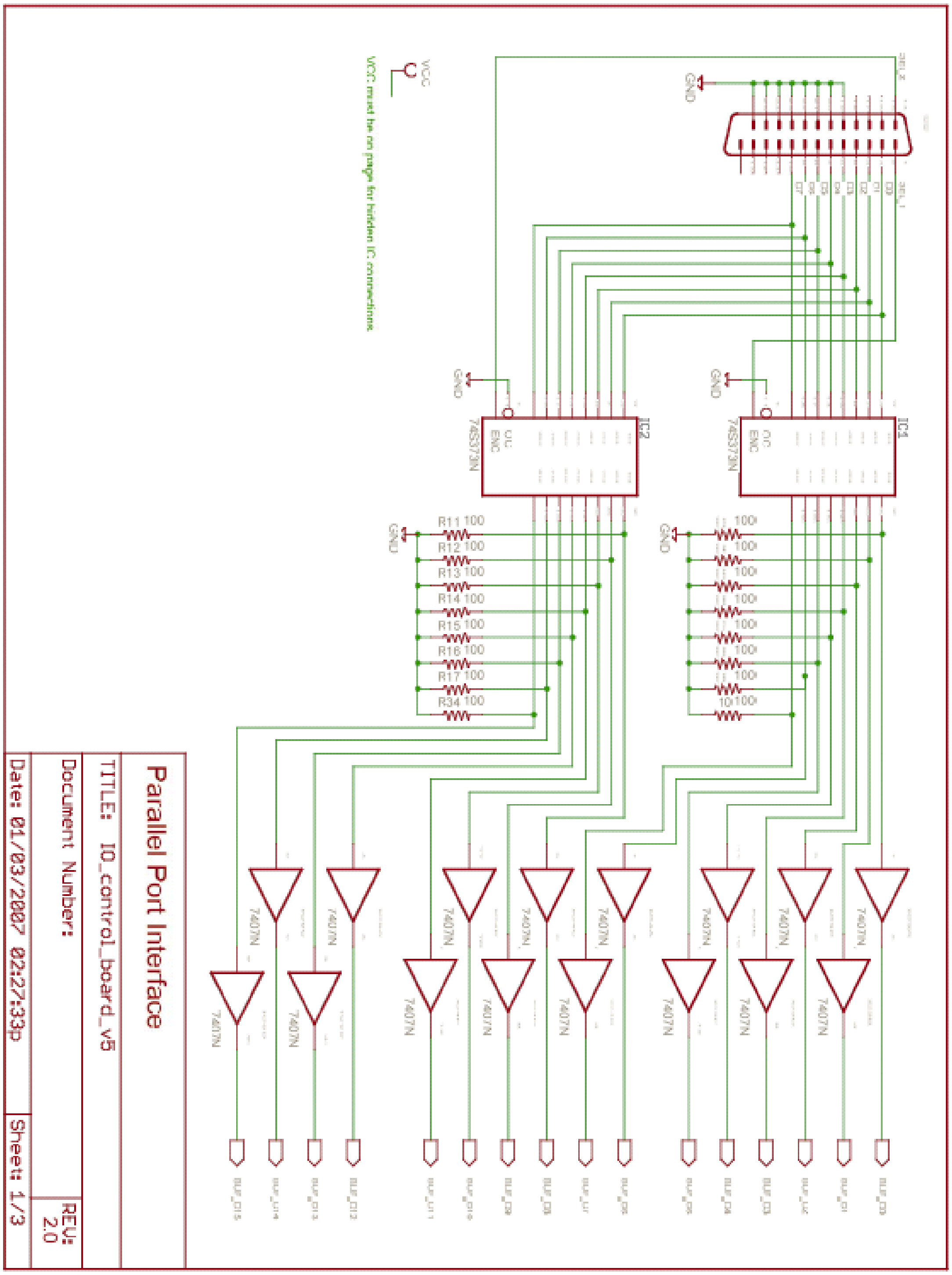
- Transform mapping: each transform can be mapped to the VibraPad array in different ways. The arrangement of the transducers can be changed to different patterns and spacing.
- Number of sound clips: This number can be reduced to 5 if a desirable hit rate of 80% is not achieved within the training time constraint. The number can be increased to 20 if the hit rate is achieved easily within a short training period.

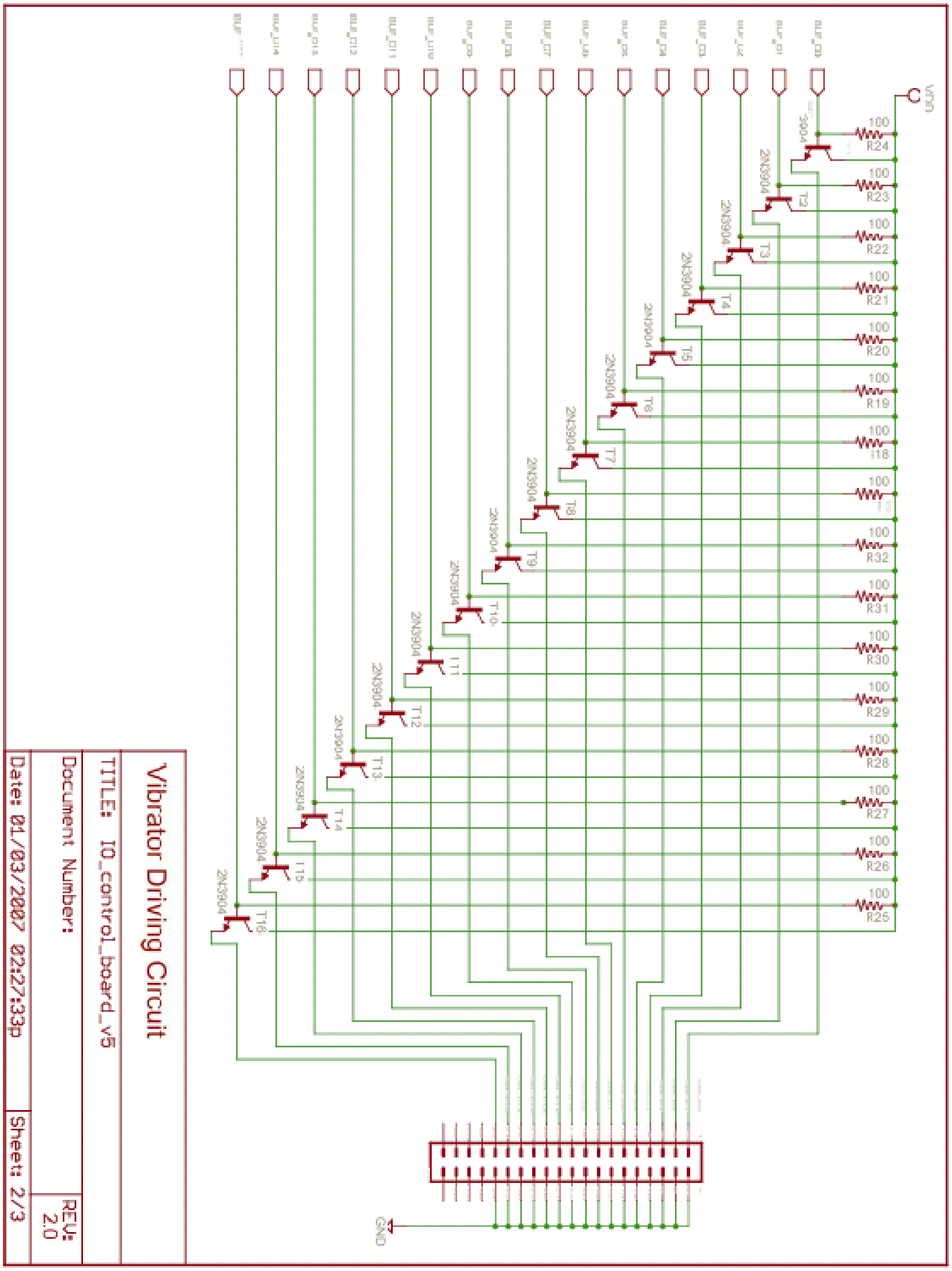
7 Conclusion

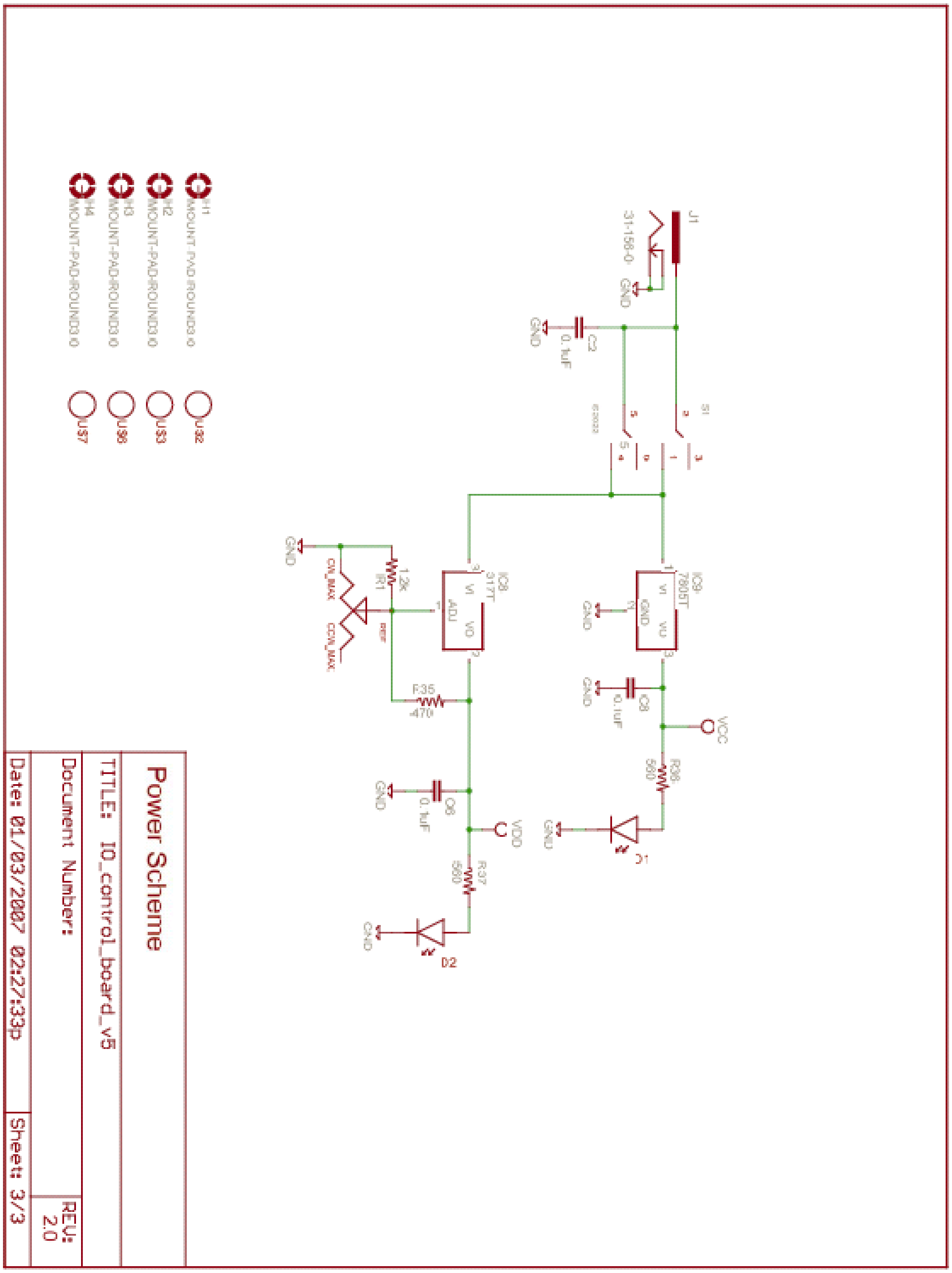
The goal of our project is to produce an assistive technology that will aid the hearing impaired in their daily activities. Our project consists of the research and design phase of the production cycle and will produce a proof-of-concept prototype to verify our proposed method of sound recognition.

This document details the design specifications needed to meet the requirements intended for the initial prototype as outlined in the Functional Specifications document. It encompasses the hardware and software design decisions and provides detailed descriptions of the tests necessary to analyze and determine the feasibility of our prototype. It is after this testing phase that we will determine whether our proposed method of sound recognition is feasible and whether the information it provides is beneficial so as to move on to the production phase of our project. The expected completion date for the proof-of-concept prototype is in mid April and if testing results prove favorable, preparations for production will commence shortly thereafter.

8 Appendix A - Schematics




Vibrator Driving Circuit
TITLE: IO_control_board_v5
Document Number:
REV: 2.0
Date: 01/03/2007 02:27:33p
Sheet: 2/3
VDD
GND


Power Scheme
TITLE: IO_control_board_v5
Document Number:
REV: 2.0
Date: 01/03/2007 02:27:33p
Sheet: 3/3
J1
GND
C2
0.1uF
S1
IC9
7805T
VI
VO
GND
C8
VCC
R36
560
D1
IC8
317T
ADJ
R1
1.2k
R35
470
C6
VDD
R37
D2

9 Appendix B – Vibrator Data Sheet

SOLARBOTICS Ltd

Motor Specification

Flat Type Vibration Motor **VPM2**

1. STANDARD OPERATING CONDITION

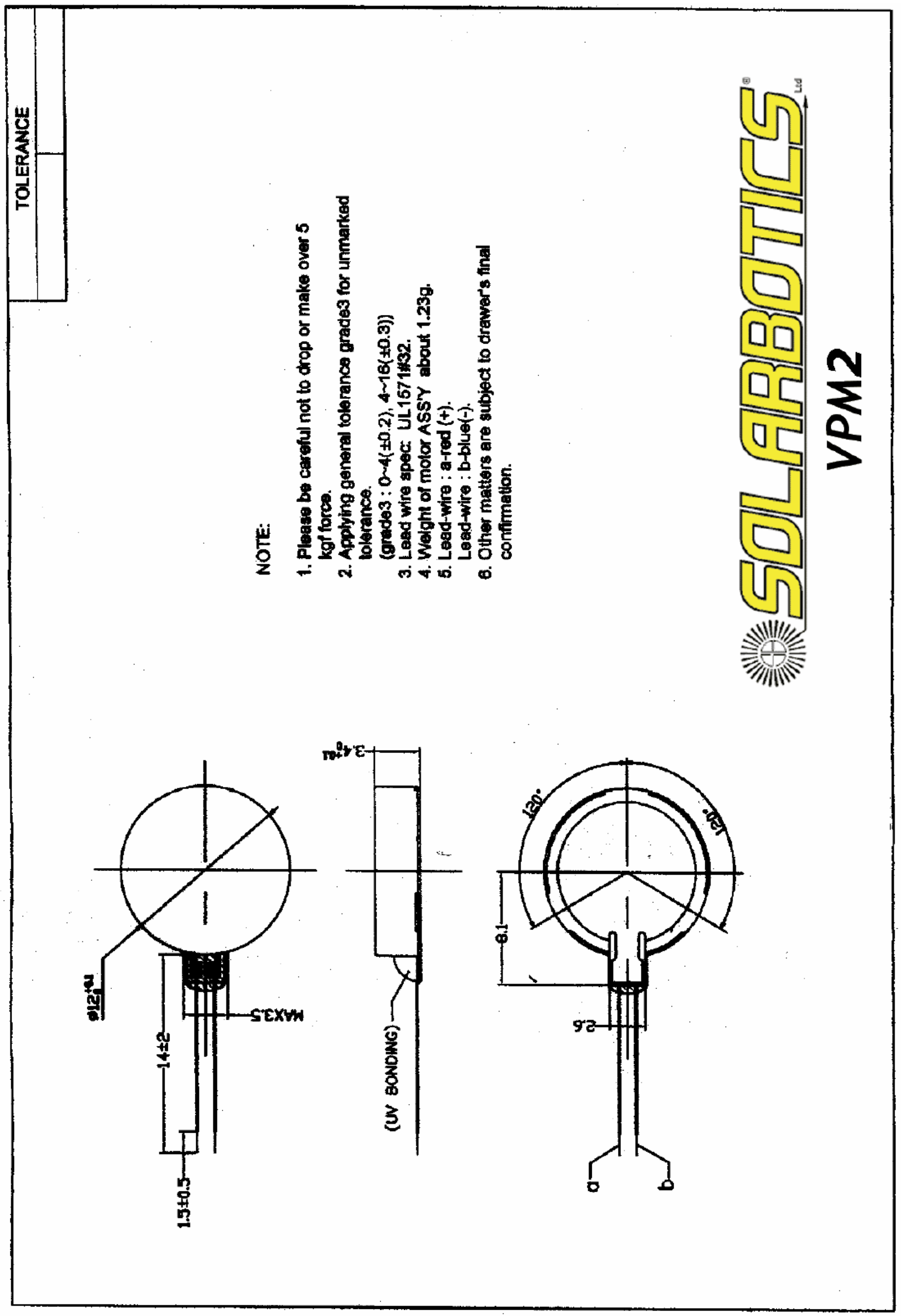
NO	ITEM	RATED CONDITION
2-1	STANDARD VOLTAGE	DC3.0V
2-2	OPERATING VOLTAGE GANGE	DC 2.5 ~ 3.5V
2-3	ROTATING DIRECTION	CW , CCW
2-4	OPERATING TEMPERATURE RANGR	-10℃ ~ +60℃
2-5	STORAGE TEMPERATURE RANGE	-30℃ ~+70℃

2. MEASURING CONDITION

NO	ITEM	RATED CONDITION
3-1	TEMPERATURE	5℃ ~35℃
3-2	HUMIDITY	35% ~75%RH
3-3	POWER SUPPLY,VOLTAGE SOURCE	DC POWER SUPPLY OR BATTERY3.0V
3-4	POSTURE OF MOTOR	STATE OF STANDARD MEASUREMENT

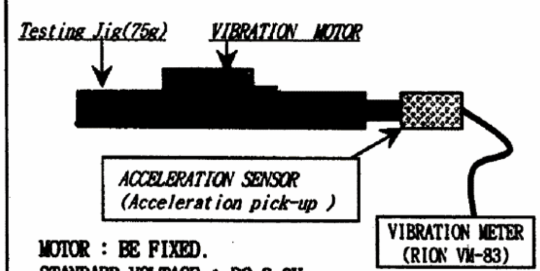
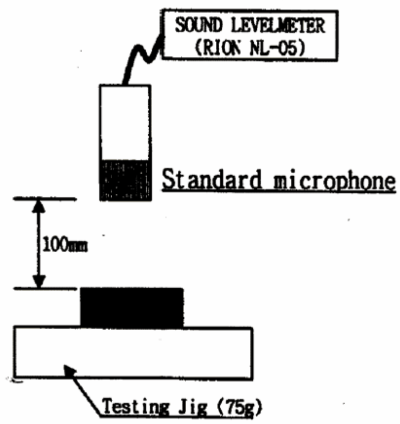
3. ELECTRICAL CHARACTERISTIC

NO	ITEM	UNIT	SPEC	CONDITION
4-1	STANDARD SPEED	rpm	12.000±3,000	STANDARD VOLTAGE : DC 3.0V
4-2	STANDARD CURRENT	mA	80 MAX	STANDARD VOLTAGE : DC 3.0V
4-3	MIN.STARTING VOLTAGE	V	2.3MAX	ON/OFF-1CYCLE,TTL5CYCLES UNDER DC3V
4-4	TERMINAL RESISTANCE	Ω	32 ± 20%	EACH BRUSH CONTACTS EACH POLE OF COMMUTATOR
4-5	STARTING CURRENT	mA	120MAX	MOTOR LOCKING
4-6	INSULATION RESISTANCE	MΩ	10MIN	MEASURING BETWEEN CASE AND TERMINAL.


TOLERANCE
NOTE:
1. Please be careful not to drop or make over 5 kgf force.
2. Applying general tolerance grade3 for unmarked tolerance.
(grade3 : 0~4(±0.2), 4~16(±0.3))
3. Lead wire spec: UL1571#32.
4. Weight of motor ASS'Y about 1.23g.
5. Lead-wire : a-red (+).
Lead-wire : b-blue(-).
6. Other matters are subject to drawer's final confirmation.
SOLARBOTICS
VPM2
1.5±0.5
14±2
MAX3.5
(UV BONDING)
120°
120°
8.1
2.6
a
b

VPM2 SOLARBOTICS Ltd

4. MECHANICAL CHARACTERISTICS

NO	ITEM	UNIT	SPEC	CONDITION
4-1	VIBRATION STRENGTH	G	(1.0)	 MOTOR : BE FIXED. STANDARD VOLTAGE : DC 3.0V . STIPULATION AT ROTATING SPEED .
4-2	MECHANICAL NOISE	dB	50 MAX	 MOTOR : BE FIXED . STANDARD VOLTAGR : DC 3.0 V BACK GROUND NOISE 25dB MAX.-A SCALE MICRO PHONE SHOULD BE VERTICAL.
4-3	SHAFT PULL STRENGTH	gf	500 MIN	DEMOLITION TEST BY PUSH-PULL GAUGE.
4-4	BRACKET DEFLECTION STRENGTH	gf	500MIN	DEMOLITION TEST BY PUSH-PULL GAUGE.