Boston University
Electrical & Computer Engineering

EC464 Senior Design Project

User's Manual

OBSIDAS

On-Board Sound Intensity Data-Acquisition System

by

Team 8 / Team Volpe
(Client: U.S. DOT Volpe Center, Cambridge, MA)

Team Members

Muhammed Abdalla muhabda@bu.edu
Aurojit Chakraborty aurojit@bu.edu
Daniel Cardosi dcardosi@bu.edu
Javier González Santamaría javiergs@bu.edu
Joseph Lucatorto III jlucator@bu.edu

April 19, 2024

Table of Contents

1 Executive Summary (Aurojit Chakraborty)

There is an ever-growing body of evidence that demonstrates the negative effects of noise pollution: not only does it harm local wildlife near major roadways, but it is also linked to health issues such as tinnitus, hearing loss, and heart disease. Automobile traffic is a major driver of noise pollution, yet the current sound-measurement tools used by the U.S. Department of Transportation's Volpe Center consist in outdated and cumbersome programs. The developed software (hereafter OBSIDAS), requested by Volpe, aims to make their road-acoustics research more streamlined and scalable by interfacing with an external microphone array to perform fully automated measurements of tire-pavement noise via the official On-Board Sound Intensity (OBSI) method, standardized by the American Association of State Highway and Transportation Officials. OBSIDAS is a LabVIEW executable that leverages a C++ subroutine for all computations on sound data.

2 Introduction (Joseph Lucatorto)

The design of cars and roads demands consideration of their harmony with drivers, with the environment, and with society. Among the variables influencing this harmony, noise production is paramount. Given that tire-pavement noise is the predominant source of roadway noise at speeds above 30 miles per hour [1] and increases logarithmically with increasing speed [2], it is especially useful to measure on-board sound intensity because it corresponds to the energy generated at the interface of a vehicle's tire and the pavement beneath it and thus correlates with a vehicle's contribution to roadway noise. Access to a reliable system for this form of measurement may aid in the creation of design constraints for automobile manufacturers and traffic engineers alike, which will in turn produce quieter traffic. Organizations such as the Volpe Center adhere to an existing methodological standard for road-acoustics research, known as the On-Board Sound Intensity (OBSI) method (*q.v.* section 6.1). This standard defines the entire measurement procedure, including specific numerical data, necessary calculations, and the configuration of all involved hardware. Per Fig. 1 below, sound-pressure data is collected using two pairs of specialized probes in a standardized array configuration, and a single measurement trial proceeds as follows: over a user-prescribed duration of time (most commonly 5 seconds), the probes feed acoustic data to a laptop in the vehicle as it drives.

The software currently used by Volpe to gather OBSI data has a cluttered interface, only functions on legacy operating systems, and has extremely limited sensor compatibility, leading to increased costs for researchers. OBSIDAS adheres completely to the AASHTO-standardized process for OBSI measurements and collects the same data as the Volpe Center's current software, with the added

benefits of greater hardware flexibility, easy distribution, no overhead for license fees, a user-friendly GUI, and access to its source code.

Figure 1: OBSI sensor array mounted on a car

3 System Overview and Installation

3.1 Top-Level Functional Description (Muhammed Abdalla)

The user-facing function of OBSIDAS is supported by two main capabilities: microphone calibration, wherein users may calibrate each of the four microphones via individually editable sensitivity parameters; and trial execution, wherein users may specify a measurement-trial duration and thereafter initiate a trial, the data from which is automatically exported as a timestamped *.csv file. For each microphone, OBSIDAS further allows users to select from a variety of spectral representations (*q.v.* section 3.3 below) a form in which recorded input signals are graphically displayed after each trial on the front panel; this function is additional to the requirements of Volpe and has no bearing on the primary function of OBSIDAS.

The acoustical data in each generated *.csv are computed automatically. Under a black-box description for a single trial, the backend of the OBSIDAS takes as its input a sample of tire-pavement noise and returns as its output (formatted in the saved *.csv) the following metrics:

In ⅓-octave bands, with centers from 250–5040 Hz
(quantities are of 14-element arrays, each element corresponding to a single ⅓-octave band)

1. Single-microphone sound pressure levels (x4)
2. Single-probe sound pressure levels (x2), each probe being a pair of colocated microphones
3. Overall (all-microphone) sound pressure levels (x2)
4. Inter-microphone (single-probe) magnitude-squared coherences (x2)

For the entire frequency spectrum

5. Single-probe sound pressure levels (x2)
6. Overall sound pressure levels (x1)
7. A-weighted versions of (5) and (6) above

3.2 Requisite Hardware (Javier González Santamaría)

The following hardware items together constitute the input-signal chain of the OBSIDAS in the form of a quad-microphone sensory array (quantities are indicated in parentheses):

1. GRAS 26AK preamplifiers (x4)
2. GRAS 40AI externally polarized paired microphones (x4; i.e, 2 pairs)
3. GRAS 12AQ 2-channel power modules (x2)
4. Brüel and Kjær 4231 calibrator (x1)
5. GRAS 7-pin LEMO 10-meter extension cables (x4)
6. BNC 25-foot extension cables (x4)
7. Native Instruments USB-4431 4-input data-acquisition device (x1)

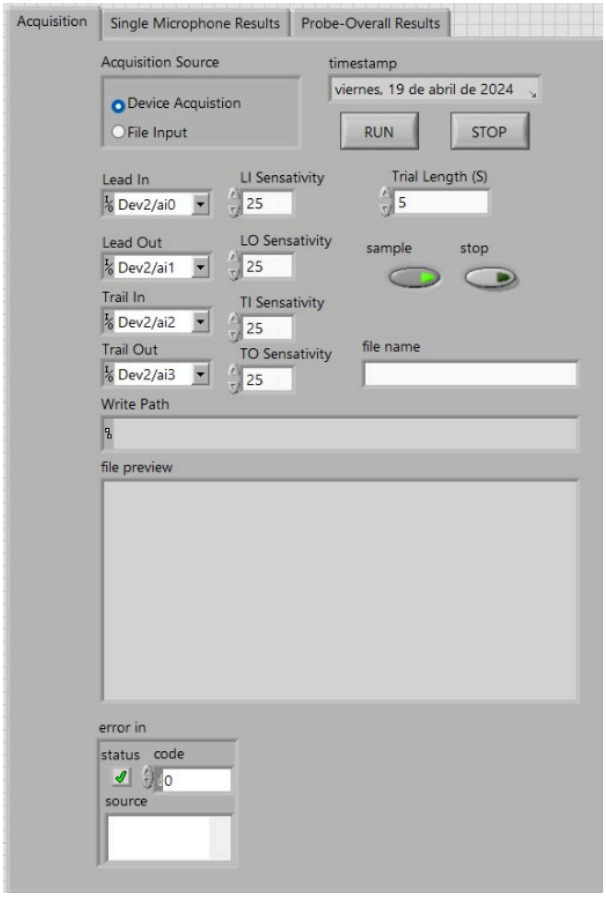
NB: wheel-mounting apparati were not provided by Volpe and thus are not listed above.

3.3 User Interface (Javier González Santamaría)

The user interface consists of three general sections (see Figs. 2–4): on the left are controls for input-device selection and for measurement-trial configuration and initiation, including spaces to define trial duration and microphone sensitivities; on the right is a preview of the information to be exported in *.csv form; and in the center are graphs generated from recorded input data of the previous trial, each of which displays for a single microphone one of the following items (selectable by the user):

1. Plot of raw signal (amplitude vs. time)
2. Plot of Fourier Transform (positive-frequency half of spectrum)
3. Table of sound pressure levels (of unit dB), in ⅓-octave bands
4. Full-spectrum averages of single-band sound pressure levels and of RMS voltages

In Figs. 5–7 are shown examples of items 1–3, respectively.



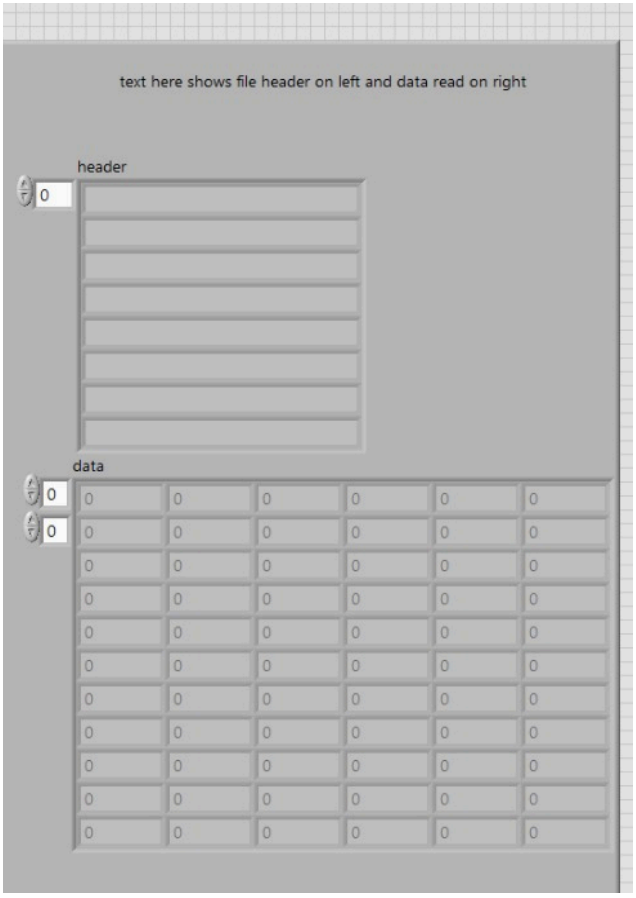


Figure 2: Leftmost portion of user interface

Figure 3: Rightmost portion of user interface (uninitialized)

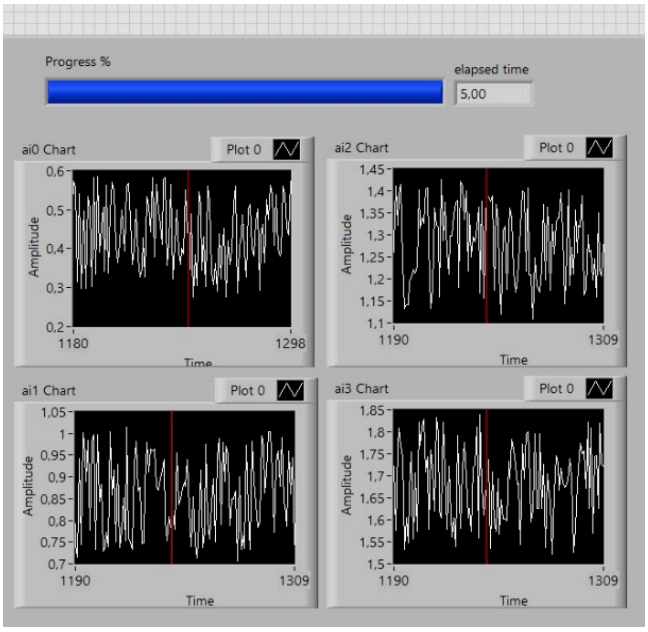


Figure 4: Central portion of user interface

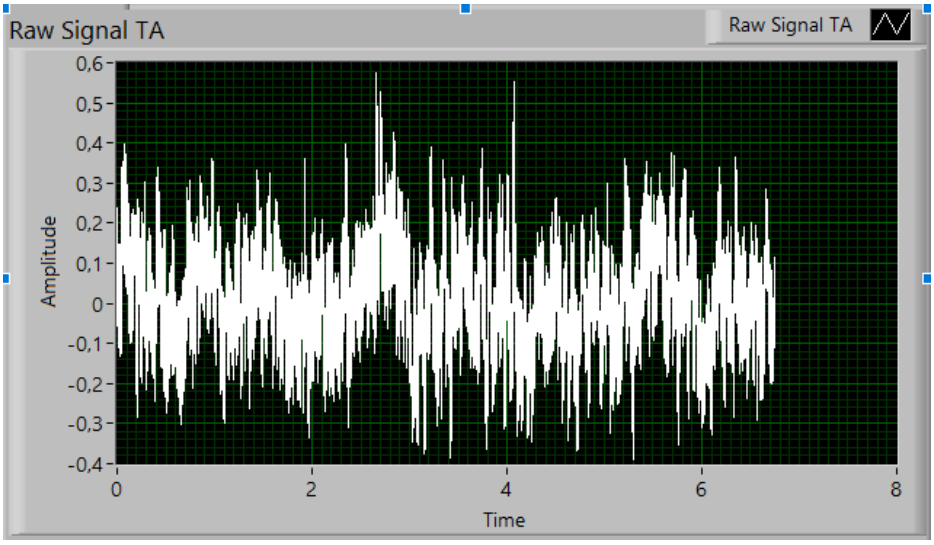


Figure 5: Raw signal plot (amplitude vs. time)

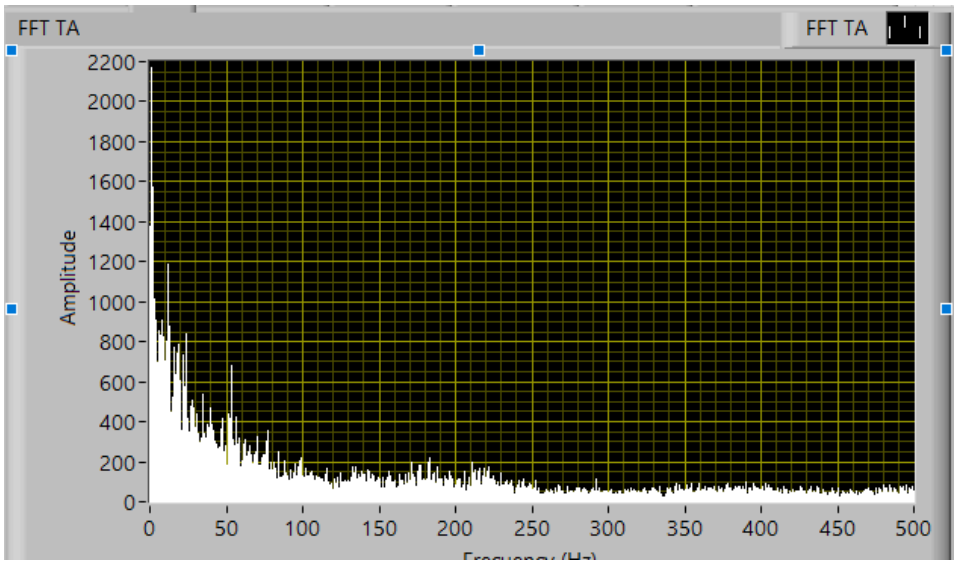


Figure 6: Fourier transform plot (positive-frequency half of spectrum)

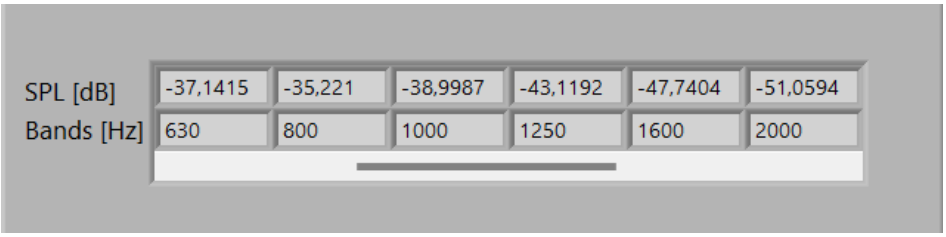


Figure 7: Single-band SPL data table (unscaled)

3.4 Overview Block Diagram

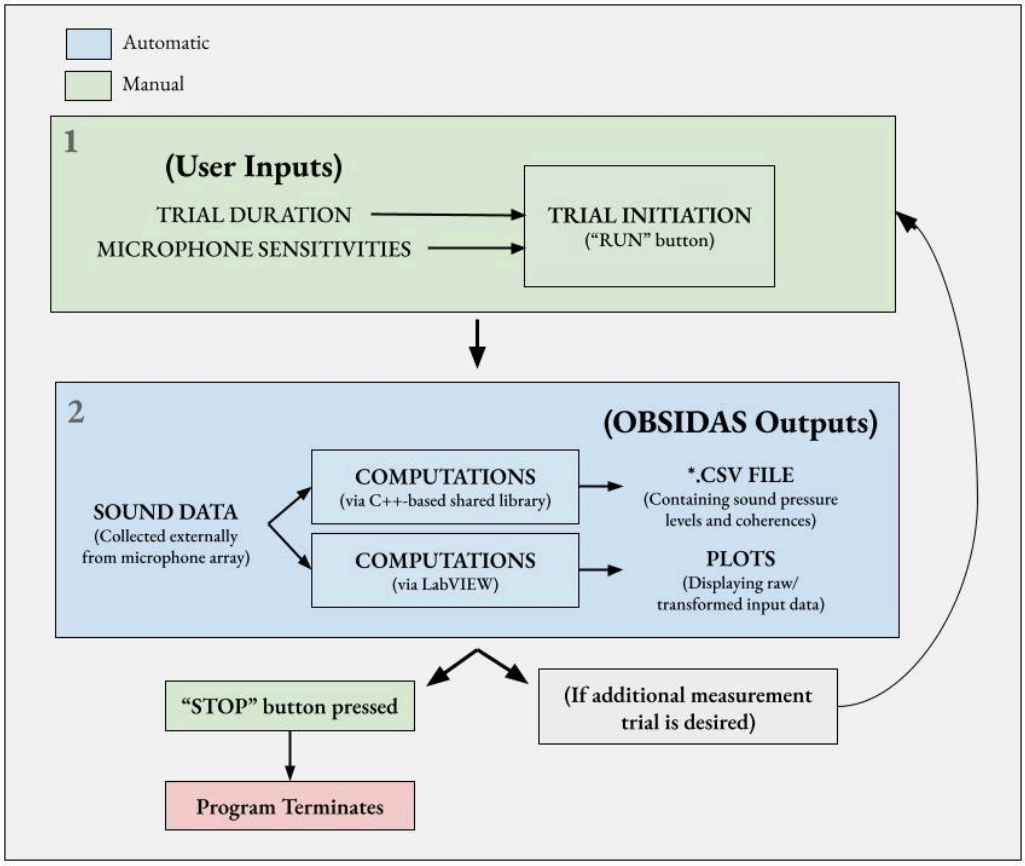


Figure 8: Schematized system overview

3.5 Installation: Hardware (Muhammed Abdalla)

The hardware input chain is constructed as follows: each 40AI microphone is attached to a 26AK preamplifier, which is then connected to a single input channel of a 12AQ power module via a LEMO cable. The corresponding output of the power module is then routed to the USB-4431 DAQ, which is connected via USB to a computer loaded with the OBSIDAS software.

3.6 Installation: Software (Muhammed Abdalla)

The software consists entirely of a LabVIEW 2017 Virtual Instrument and a Windows dynamic-link library on which the executable depends. The OBSIDAS executable is compiled using Application Builder, which takes the VI and DLL and creates a 2017 executable in order for it to run independently on any windows desktop machine given LabVIEW 2017 runtime is installed.

4 Operation

4.1 Software Setup (Joseph Lucatorto)

For OBSIDAS to obtain accurate data, all microphones must be calibrated prior to a measurement session. Calibration proceeds by positioning the output driver of the 4231 calibrator directly against a 40AI microphone capsule and digitally adjusting the sensitivity parameter (the default being 25 mV/Pa) corresponding to the microphone (across 1 or more successive trials) until the sound-level monitor reads the desired value; for maximal accuracy in calibration, users should fix the fifth user-interface tab to display the trial-average sound pressure level corresponding to the 7th lowest ⅓-octave band (centered at 1 kHz, the frequency of the calibration tone) and monitor the recorded level for each microphone thusly. Prior to each measurement trial, the duration of recording must be preset by inputting a value in seconds.

4.2 Measurement (Aurojit Chakraborty)

Users may initiate a measurement trial by clicking the button labeled “RUN” on the front panel of OBSIDAS, and thereafter begins the automatic process of recording, computation, and export as outlined in section 3.1 above.

4.3 Front Panel and Data Display (Javier González Santamaría)

In the user interface, there are three main tabs where the user can interact with the program. The first one is called "Acquisition," and it contains all the controls for sound acquisition through the microphones. Users can choose the source from which the program acquires the sound, either through the hardware device (the microphone array, by way of the USB-4431) or from a specified local audio file. When the USB-4431 is selected as the input device, users can subsequently map the four channels of the USB-4431 (each connected to a single microphone) to the four input channels visible on the front panel; for usage of OBSIDAS per the AASHTO-compliant OBSI method of measurement, the uppermost front-panel input always corresponds to the leading, inside microphone, the input directly thereunder to the leading, outside microphone, the lowermost front-panel input to the trailing, outside microphone, and the input directly thereabove to the trailing, inside microphone. Lastly, users can individually adjust the sensitivity of the microphones and specify a trial duration in seconds.

In the same tab, there are also the *.csv file data: both the path where the file will be located and a preview of the data that will be passed to the file. Below the progress bar, which informs the user as to the degree of measurement-trial completion, there are four real-time graphs of the different

microphones displaying what the system acquires from the input device, and the text below these graphs consists in both the file header of the *.csv and the data read into the *.csv.

In the second tab, named “Single Microphone Results,” are graphs for each of the four microphones. For each microphone, users can select between the items listed above (see section 3.3) to populate the graph using the input data of the previous trial.

In the last tab, named "Probe Overall Results," are the results of the data from each pair of microphones (i.e., each probe). This includes the single probe SPL of each pair, the overall SPL, the single probe magnitude-squared coherence (MSC) of each pair, and the overall MSC.

5 Technical Background

5.1 Software: System Architecture (Muhammed Abdalla)

The OBSIDAS executable opens a LabVIEW 2017 virtual instrument (VI) constructed with a sequential structure that allows for repeated measurement trials conducted on a single running instance of the VI. In Figs. 9–13 are block diagrams for each of the main elements, grouped separately in the system-wide block diagram as clusters of related/cofunctional subVIs and processes:

1. A data-acquisition group, responsible for interfacing with the NI USB-4431 to collect sound data in the the form of voltage-value arrays
2. A numerical-computation group, responsible for returning a set of arrays together containing the required acoustically significant metrics, each of which it computes from the voltage-value input arrays via an external C++-based library (*q.v.* section 5.2 below)
3. A write-to-file group, responsible for generating and exporting a timestamped *.csv file containing the computed data from the numerical-computation group
4. An error-handling group, responsible for monitoring the occurrence of errors throughout all levels of the program, especially those consequential from user errors and improper usage, and for providing descriptions thereof (in the form of pop-up messages) to guide the resolution of such errors

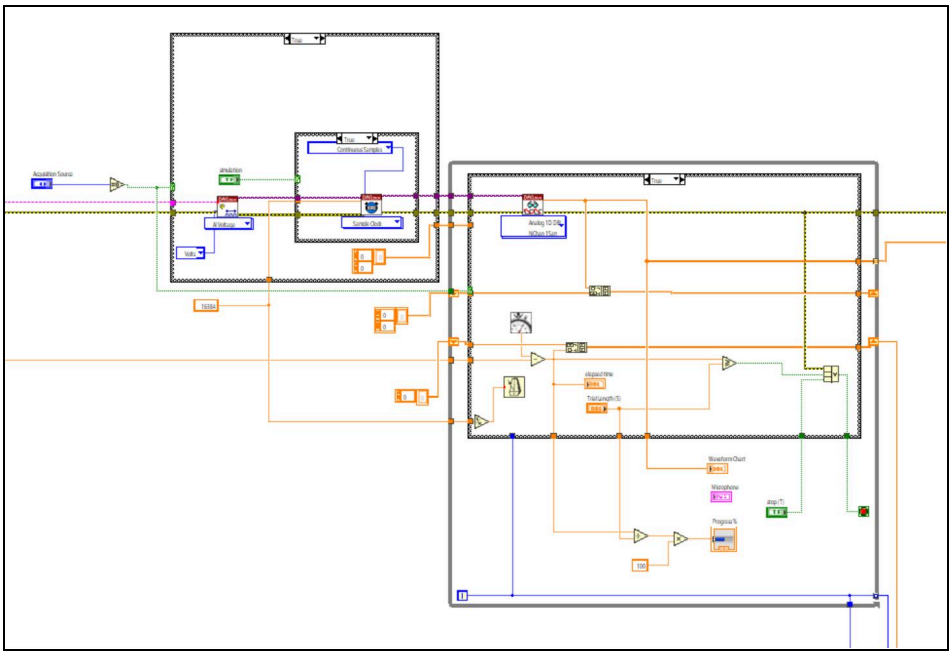


Figure 9: Data-acquisition sequence of LabVIEW VI

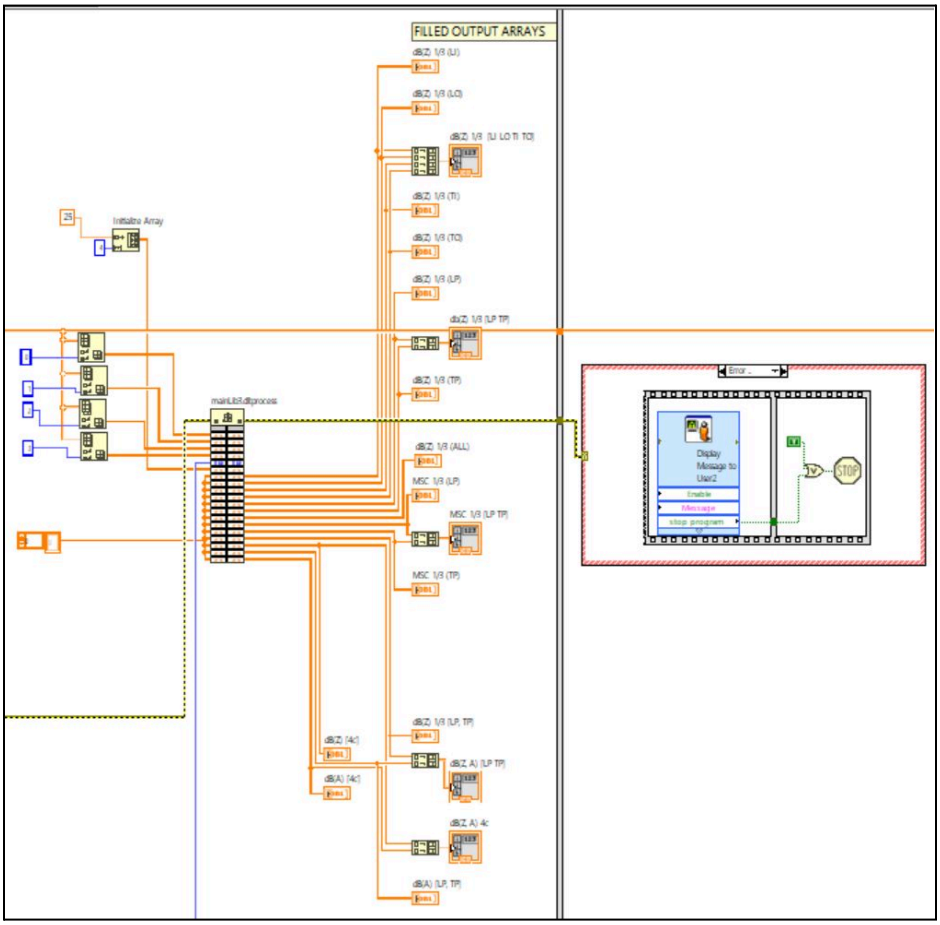


Figure 10: Numerical-computation (library-dependent) sequence of LabVIEW VI

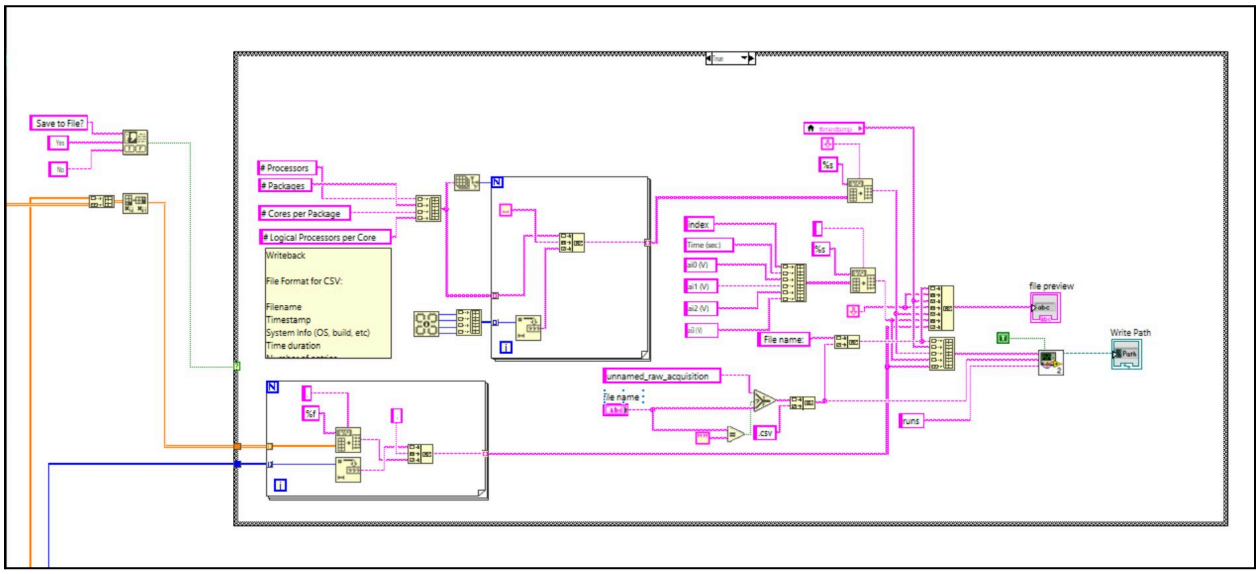


Figure 11: Write-to-file sequence of LabVIEW VI

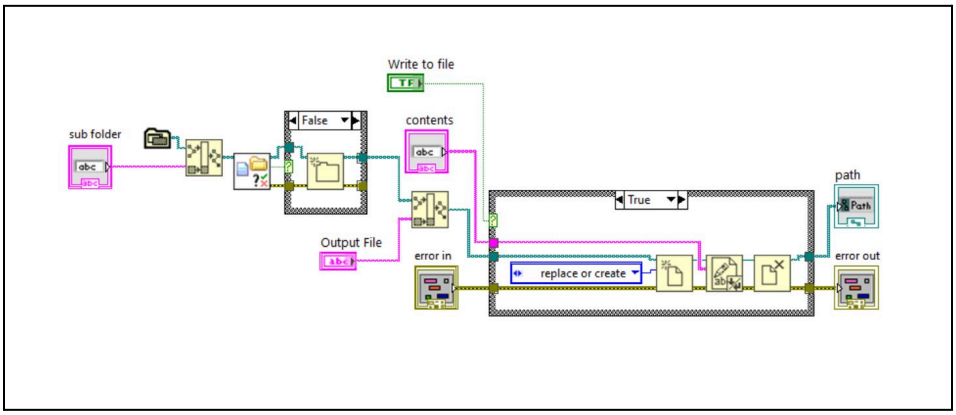


Figure 12: Custom write-to-file submodule utilized in LabVIEW VI

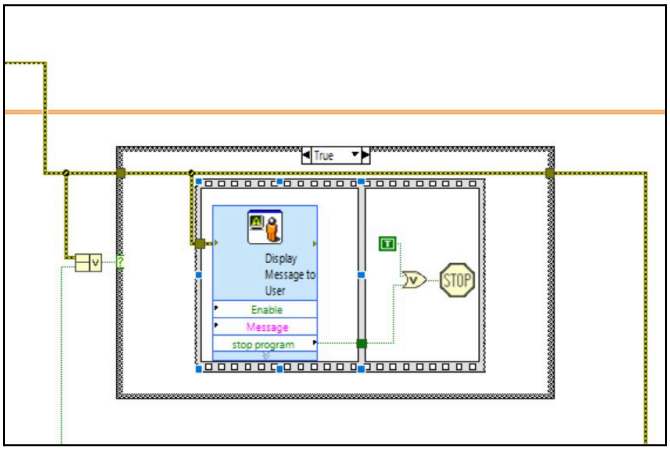


Figure 13: Error-handling module of LabVIEW VI

5.2 Software: Numerical-Computation Subroutine (Daniel Cardosi)

As outlined above, at the lowest level of the OBSIDAS' internal structure is a C++ subroutine, in the form of a Windows dynamic-link library (of extension *.dll) executed from within LabVIEW via a "Call Library Function" node. This library, compiled from a C++ script and a static build of the FFTW C library (version 3.3.10), is responsible for the computation of all listed metrics (*q.v.* section 3.1 above) from the 4 raw voltage-value input arrays on each trial.

Inputted to the script on each trial are 4 voltage-value arrays containing the raw sound data as recorded by the 4 microphones, in addition to an array containing the user-defined sensitivity values for each microphone. Computations of all metrics follow an initial series of discrete Fourier transforms (DFTs) applied to each voltage-value input array via a set of functions implementing the FFTW library. All DFTs are n-point, n being the number of samples in each microphone-input array as determined on a trial-by-trial basis, and are configured via the real-to-real (r2r) one-dimensional (1d) FFTW plan. The execution of each DFT utilizes both the half-complex FFTW_R2HC output format, which exploits Hermitian redundancy to reduce computation time without loss of information, and the maximally parsimonious FFTW_ESTIMATE planning-rigor flag, similarly economical computationally; these two options were determined during testing to maximize efficiency at runtime. All considered alternatives as regards DFT computation are detailed in the Appendix (section 9.3).

All dB-unit computations are performed on root-mean-square (RMS) values of the raw voltage-unit sound-input values, which are themselves calculated by Parseval's theorem from computed Fourier coefficients. Coherences are likewise computed via DFT results and complex conjugations thereof, per the usual definition of coherence as a form of non-stochastic first-order correlation wherein statistical ensemble averages become instead averages across time; squaring the magnitudes of such coherences then yields the magnitude-squared coherences outputted in the exported *.csv. All other sub-computations involved in generating the required acoustically significant values follow from basic mathematical operations and definitions.

5.3 Hardware (Joseph Lucatorto)

All hardware components, including the USB-4431 unit via which the OBSIDAS software interfaces with the input-signal chain, were selected and provided by Volpe per their internal standards and requirements.

6 Relevant Standards (Daniel Cardosi)

6.1 Tire-Pavement Noise

The procedures by which tire-pavement noise is measured per the On-Board Sound Intensity (OBSI) method are detailed in AASHTO TP 76-10.

6.2 Decibels

The auditory-threshold-based reference pressure of 0.0002 dynes/cm^2 inherent in each of the values OBSIDAS presents in dB(SPL), equivalently dB re 20 μPa, appears in ISO 1996-1:2016.

6.3 One-Third Octaves

The base-two frequency ratio defining the one-third octave as implemented in OBSIDAS' computations follows ISO 18405:2017 and thus results in 14 bands whose center frequencies range from 250 Hz to 5039.684 Hz.

6.4 Psophometric Filter Curves

The mathematical weighting functions utilized to produce sound-pressure-level data per an A-weighted amplitude-frequency spectrum (*q.v.* item 7 of section 3.1 above) are described in IEC 61672-1:2013.

6.5 Programming Language

The C++17 standard to which all written code in the *.cpp script conformed has identifier ISO/IEC 14882:2017; this script constitutes the entirety of non-LabVIEW code utilized by OBSIDAS.

7 Cost Breakdown (Aurojit Chakraborty)

Item	Description	Cost
1	LabVIEW 2017 Full	$1664
2	LabVIEW Application Builder	$518
3	LabVIEW Sound and Vibration Toolkit	$824
-----	**Total Cost:**	**$3006**

By request of the client, our program will make use of LabView 2017 along with the following two packages: LabVIEW Application Builder and LabVIEW Sound and Vibration Toolkit. As National Instruments requires that both subscriptions and addons be purchased, Boston University defrayed

these costs for our development purposes; however, given that Volpe is already in possession of all required licenses, our final software will necessitate no further monetary expenses on their behalf.

8 Bibliography

[1] R. O. Rasmussen, R. J. Bernhard, U. Sandberg, and E. P. Mun, “The Little Book of Quieter Pavements,” *FHWA-IF-08-004*, Federal Highway Administration, Washington, DC, 2008.

[2] U. Sandberg, "Tyre/Road Noise Myths and Realities," *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, 2001.

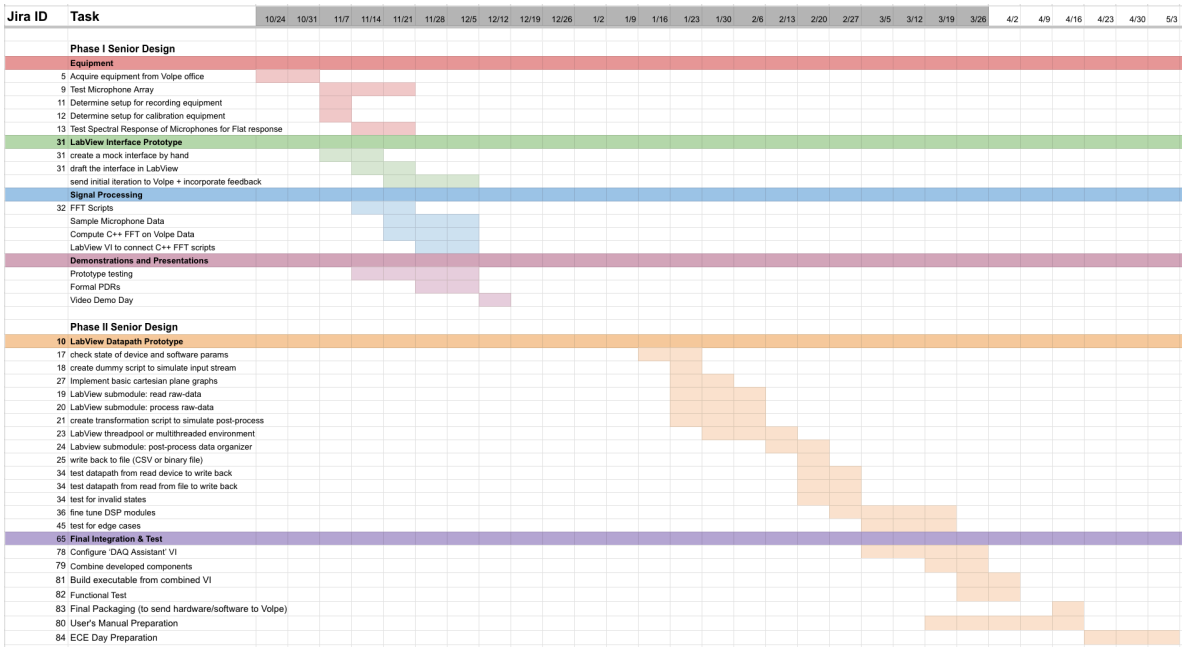
9 Appendix

9.1 Engineering Requirements

OBSIDAS fulfills each of the requirements specified by Volpe, all reproduced in the table below.

Requirement	Value, range, tolerance, units
User-editable sensitivity-parameter inputs for microphone calibration	Precision of 1/10 mV/Pa for sensitivities
Manually triggered measurement trials of user-defined durations	Range of 3–60 sec and precision of 1/10 sec for durations
Data export to a human-readable file	*.csv format
Computation of single-microphone SPLs (x4), single-probe SPLs (x2), overall SPLs (x1) in third-octave bands with centers from 250–5040 Hz; computation of single-probe SPLs (x2) and overall SPLs (x1) for entire spectrum	Precision of 1/10 dB for all SPLs
Computation of inter-microphone coherences in ⅓-octave bands with centers from 250–5040 Hz	Precision of 1/10 (unitless) for all coherences
LabVIEW-2017-compatible executable (containing entire program)	*.exe format

9.2 Gantt Chart



Jira ID	Task
	Phase I Senior Design
	Equipment
5	Acquire equipment from Volpe office
9	Test Microphone Array
11	Determine setup for recording equipment
12	Determine setup for calibration equipment
13	Test Spectral Response of Microphones for Flat response
31	**LabView Interface Prototype**
31	create a mock interface by hand
31	draft the interface in LabView
	send initial iteration to Volpe + incorporate feedback
	Signal Processing
32	FFT Scripts
	Sample Microphone Data
	Compute C++ FFT on Volpe Data
	LabView VI to connect C++ FFT scripts
	Demonstrations and Presentations
	Prototype testing
	Formal PDRs
	Video Demo Day
	Phase II Senior Design
10	**LabView Datapath Prototype**
17	check state of device and software params
18	create dummy script to simulate input stream
27	Implement basic cartesian plane graphs
19	LabView submodule: read raw-data
20	LabView submodule: process raw-data
21	create transformation script to simulate post-process
23	LabView threadpool or multithreaded environment
24	Labview submodule: post-process data organizer
25	write back to file (CSV or binary file)
34	test datapath from read device to write back
34	test datapath from read from file to write back
34	test for invalid states
36	fine tune DSP modules
45	test for edge cases
65	**Final Integration & Test**
78	Configure 'DAQ Assistant' VI
79	Combine developed components
81	Build executable from combined VI
82	Functional Test
83	Final Packaging (to send hardware/software to Volpe)
80	User's Manual Preparation
84	ECE Day Preparation

9.3 Alternatives Considered for DFT Computation (Daniel Cardosi)

The internal use of Python for numerical computation was considered alongside C++, but the latter was chosen over the former to avoid possible issues arising from incompatibility between Python and Native Instruments' LabVIEW, which only introduced native capability for calling Python scripts in the version released in 2018. Also considered in place of the C++ subroutine was LabVIEW itself, which provides a built-in VI for DFT calculation; this VI, however, utilizes the canonical matrix formulation of the DFT in its computation, which has a time-complexity of $O(n^2)$; the development of the subroutine in C++ therefore allowed a variety of more efficient $O(n\log(n))$ algorithms to be considered, of which there were eight:

1. Danielson-Lanczos/Runge-König (radix-2)
2. Cooley-Tukey (mixed-radix, decimation-in-time)
3. Gentleman-Sande (mixed-radix, decimation-in-frequency)
4. Transposed Stockham (in-place)
5. Pease (row-oriented Cooley-Tukey)
6. Stockham (row-oriented Transposed Stockham)
7. Good-Thomas, Rader, Bluestein (prime sizes)
8. Duhamel-Hollmann (split-radix/unsymmetric-butterfly)

From analyses of the above schemes for FFT-based DFT computation was selected the FFTW3 C library, specifically given the library's recursive implementation of the Cooley-Tukey algorithm incorporating stages of decimation both in time and in frequency, a modified split-radix architecture (after the Bernstein tangent algorithm) with adaptively chosen radices, and subroutines for prime-length inputs per the Good-Thomas (prime-factor) and Rader algorithms. C++ thus enabled the development of a more efficient program than did the other solutions, both through the control it allows over DFT computations and in its low-latency interoperation with LabVIEW 2017.