MOS Electronic Nose for Alcohol and Solvents with Analysis by Hilbert-Huang Transform

by

Mark Lieberman

A Thesis
Presented to
The University of Guelph

In partial fulfillment of requirements
for the degree of
Master of Applied Science
In
Engineering

Guelph, Ontario, Canada
© 2013 Mark Lieberman
Abstract

MOS Electronic Nose for Alcohol and Solvents with Analysis by Hilbert-Huang Transform

Mark Lieberman
University of Guelph, 2013

Advisors:
Dr. Radu Muresan
Dr. Stefano Gregori

This work investigates the potential use of temperature modulation of MOS gas sensors combined with the Hilbert-Huang transform (HHT) as a feature extraction mechanism for MOS based electronic noses. It specifically targets environments which are traditionally considered unsuitable for MOS gas sensors, such as those with high concentrations of ethanol. The expected applications are in the alcoholic beverage industry, specifically quality control and blending of spirits such as whiskey. An electronic nose was designed which includes an array of 12 commercial MOS gas sensor cells. These MOS sensors were selected based on their ability to tolerate high concentrations of reactive species. This system includes specialized firmware and desktop computer software to support the operation of the electronic nose. Five samples each of ethyl acetate, ethanol and isopropanol were prepared. The response of each of four sensors in an array was decomposed using ensemble empirical mode decomposition (EEMD) and the marginal Hilbert spectrum (MHS) was computed. A set of 72 frequency components was extracted from marginal Hilbert spectrum response of each sensor in an array of four sensor to produce a 288 element odor signature of each sample. The signatures were successfully clustered using principal component analysis (PCA) and self-organizing map. A neural net classifier identified the samples with very high accuracy.
Acknowledgments

I would like to express my gratitude for the support of my supervisor Dr. Radu Muresan, whose guidance and trust made this work possible. I would also like to thank my co-advisors Dr. Stefano Gregori and Dr. Simon X. Yang, for their insight and advice throughout this project. I also extend my thanks for the support of Mr. David Doyle, Dr. Sapna Sharma and everyone else in the R&D department of Pernod-Ricard. I would like to thank my colleague Robert Littel for his contributions to this project. Lastly, I would like to extend a heartfelt thank you to my family which has supported me throughout my studies.

This work was supported by Ontario Centres of Excellence (OCE) and from Pernod-Ricard (Project No. TP-SW-11168-11).
1 Introduction
1.1 Objectives of this Thesis ........................................ 3
1.2 Contributions of this Thesis .................................... 3
1.3 Thesis Organization ............................................... 5

2 Background
2.1 The Electronic Nose .............................................. 6
2.2 The MOS Gas Sensor ............................................. 8
  2.2.1 Dynamic Temperature Mode of Operation ................. 9
2.3 The Hilbert-Huang Transform .................................... 9
  2.3.1 Instantaneous Frequency .................................. 10
  2.3.2 Empirical Mode Decomposition .......................... 11
  2.3.3 Ensemble Empirical Mode Decomposition ............... 14
  2.3.4 Hilbert Spectrum ......................................... 16
  2.3.5 A Comparison of Transforms .............................. 17
2.4 Summary .......................................................... 19

3 Literature Review
3.1 Temperature Modulation of MOS Sensors ..................... 22
3.2 Applications to the Alcohol Industry ......................... 23
3.3 Summary .......................................................... 27

4 Implementation
4.1 The Prototype Electronic Nose ................................. 29
  4.1.1 MOS Sensor Cells ......................................... 31
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2 Schematic Capture and PCB Layout</td>
<td>34</td>
</tr>
<tr>
<td>4.2.1 Regulator Board</td>
<td>34</td>
</tr>
<tr>
<td>4.2.2 Sensor Array Board</td>
<td>37</td>
</tr>
<tr>
<td>4.2.3 Relay Board</td>
<td>48</td>
</tr>
<tr>
<td>4.3 Microcontroller Firmware</td>
<td>49</td>
</tr>
<tr>
<td>4.3.1 Analog Sampling Engine</td>
<td>51</td>
</tr>
<tr>
<td>4.3.2 Temperature Modulation</td>
<td>56</td>
</tr>
<tr>
<td>4.3.3 Communication</td>
<td>59</td>
</tr>
<tr>
<td>4.4 Capture Software</td>
<td>67</td>
</tr>
<tr>
<td>4.4.1 Class Library</td>
<td>67</td>
</tr>
<tr>
<td>4.4.2 Application</td>
<td>70</td>
</tr>
<tr>
<td>4.5 Summary</td>
<td>75</td>
</tr>
<tr>
<td>5 Results</td>
<td>76</td>
</tr>
<tr>
<td>5.1 Experimental Setup</td>
<td>76</td>
</tr>
<tr>
<td>5.2 Recorded Data</td>
<td>79</td>
</tr>
<tr>
<td>5.3 Hilbert-Huang Analysis</td>
<td>82</td>
</tr>
<tr>
<td>5.4 Clustering and Classification</td>
<td>89</td>
</tr>
<tr>
<td>5.5 Sources of Error</td>
<td>93</td>
</tr>
<tr>
<td>5.6 Summary</td>
<td>94</td>
</tr>
<tr>
<td>6 Conclusion</td>
<td>96</td>
</tr>
<tr>
<td>6.1 Future Work</td>
<td>98</td>
</tr>
<tr>
<td>Bibliography</td>
<td>98</td>
</tr>
</tbody>
</table>
List of Tables

4.1 Table of measurable ranges for every configuration of the sensor circuits. 42

5.1 Pattern classifier average error at the output for 100 neural nets trained
with signature vectors such as those in fig. 5.9a. . . . . . . . . . . . . 92

5.2 Pattern classifier average error at the output for 100 neural nets trained
with signature vectors such as those in fig. 5.10a. . . . . . . . . . . . 92
# List of Figures

2.1 Operations in the analysis of odors by an electronic nose. ............... 8  
2.2 The empirical mode decomposition of a signal which is the sum of two sines with frequency 0.01 Hz and 0.06 Hz. ................................. 14  
2.3 Hilbert spectral analysis of the example signal in fig. 2.2. ............. 18  
2.4 A comparison of time-frequency-energy spectrum of Fourier, Wavelet and Hilbert-Huang transforms for a linear chirp signal. .............. 20  

4.1 An overview of the prototype electronic nose system. .................. 30  
4.2 The MOS sensor cells from left to right: TGS-2610-D, TGS-2610-C, TGS-2620-C, SB-15-00, and SB-AQ1-06. ............................... 33  
4.3 The power supply configuration for the prototype electronic nose. ... 35  
4.4 The schematic of the regulator board which produces the power supply rails for the electronic nose. ................................. 36  
4.5 Photographs of the regulator board project box. ....................... 36  
4.6 A photograph of the sensor array printed circuit board. An array of socketed sensor circuits populate the left half of the board, while the right contains the microcontroller board and support circuitry. 38  
4.7 The schematics for the MOS sensor circuits with a fixed heater voltage. 41  
4.8 The circuit schematic for the MOS sensors with a variable heater voltage. The power switching IC in fig. 4.7 is replaced by a programmable voltage PWMn generated by the highlighted section. ................. 44  
4.9 Simulated frequency plots for the 4th order low-pass filters in fig. 4.8. The −3dB cutoff frequency and attenuation at the PWM carrier frequency are indicated. The simulations were performed with PSpice. 46  
4.10 A temperature and relative humidity sensor module with its interface circuit. ................................................................. 46  
4.11 The interface circuit to control the relay boards which operate the vial heater and fan. ..................................................... 48
4.12 The schematic of the relay board in (a) and a photograph of the fabricated board in (b). .................................................. 49
4.13 An overview of the firmware application showing the MQX tasks and flow of information. ........................................... 50
4.14 A dataflow diagram describing the DMA driven analog conversion system for the ADC0 module. This process is executed once every sample period. .................................................. 52
4.15 Contents of the data structures after $P$ sample periods when three analog channels are being sampled. ...................... 54
4.16 A block diagram describing the components involved in communication between a computer and the Kinetis microcontroller. .... 60
4.17 Serial-to-TCP Relay is a simple TCP listen server which pipes any input and output from the virtual serial port to the first TCP client which is connected. ........................................... 61
4.18 Traffic from the microcontroller to the computer. Data written to rstdio is divided into packets which are reassembled by the computer. .................................................. 63
4.19 The packet format used by the transport application layer. .... 64
4.20 A screenshot showing a number of base64 encoded packets. The MCU sends a packet and the computer responds with an acknowledgment. ..................... 66
4.21 A class diagram of the most important classes in the class library project. 68
4.22 The system panel in (a) provides detailed information about the MOS sensor configuration. Right-click on a MOS sensor and select “Configure Heater” to access the wave designer dialog in (b) and edit the MOS sensor temperature profile. ........................................... 72
4.23 The capture panel in (a) charts the MOS sensor response. The Export dialog in (b) saves the recorded data in a number of formats. .... 73
4.24 The process panel is used to monitor and configure the control loop for the vial heater temperature. ............................. 74

5.1 The array response (above) to the changing heater voltage (below) in the presence of 1 µL ethyl acetate. ........................................... 79
5.2 The array response to 1 µL of ethyl acetate (a), ethanol (b), and isopropanol (c) with temperature modulation at 16.6 mHz. .............. 81
5.3 The ensemble empirical mode decomposition of the responses from sensor TGS-2610-C in fig. 5.2. ................................. 85
5.4 The ensemble empirical mode decomposition of the responses from sensor TGS-2620-C in fig. 5.2. ................................. 85
5.5 The ensemble empirical mode decomposition of the responses from sensor SB-15-00 in fig. 5.2.
5.6 The ensemble empirical mode decomposition of the responses from sensor SB-AQ1-06 in fig. 5.2.
5.7 The Hilbert Huang spectrum derived from each decomposition in figs. 5.3 to 5.6.
5.8 The marginal Hilbert spectrum computed from fig. 5.7 in red along with four additional trials in blue.
5.9 Clustering of samples based on odor signatures derived from the marginal Hilbert spectrums in fig. 5.8.
5.10 Clustering of samples based on odor signatures derived the marginal Hilbert spectrums which omit IMFs 6 and 7.
5.11 Self-organizing map hit plot with each neuron showing the number of input vectors that it classifies.
Chapter 1

Introduction

Analysis by a trained panel of experts is the most common form of quality control within the food and beverage industry. When a more rigorous analysis is required, the answer is often gas chromatography. An electronic nose is designed to combine the skill of a human expert with the objectivity and standardization of a machine. Most electronic noses are also able to complete an analysis in a fraction of the time required to perform GC or equivalent methods. One of the most common types of sensor employed in an electronic nose is the MOS gas sensor. These devices are popular because they are highly sensitive, easy to fabricate, and inexpensive. An electronic nose will typically employ an array of MOS gas sensors, each with a different structure or dopant to sensitize it to a variety of substances. Odors are identified by their characteristic response across the array of MOS sensors, often referred to as a fingerprint or signature.

With regard to the alcoholic beverage industry, there are a number of potential
applications for an electronic nose. Fake or counterfeit product is a growing concern within this industry [1]. Counterfeit whiskey may simply fail to meet the legal requirement of a minimum 40% alcohol by volume. Bottles which masquerade as legitimate brands can damage a brand or harm consumers. Illegal additives such as methanol and ethylene glycol can harm or kill unsuspecting consumers. An electronic nose could potentially detect such contaminants by measuring the headspace of the whiskey [2]. The origin of ingredients, fermentation process, and method of aging also leave chemical traces in beverages such as wine and other spirits which are detectable by an electronic nose [3, 4]. These traces may be used to detect sophisticated knock-offs.

Closer to the distillery, electronic noses have the potential to provide online monitoring of processes such as fermentation and blending. Measuring the composition of whiskey is one potential application of a properly designed electronic nose. Blended whiskeys are required to meet precise legal limits with regard to the minimum concentration of ethanol. The flavor and quality of whiskey is also determined primarily by compounds extracted from the wooden casks in which they are aged. These compounds which are liberated from the wood are often volatile organics which may be sensed by an electronic nose [5]. Furthermore, the ability to detect contaminants such as geosmin would also be of great importance to a distillery or bottler.
1.1 Objectives of this Thesis

Whiskey – which may contain between 40% and 90% alcohol by volume depending on the stage of production – is difficult to analyze with MOS sensors. Limitations in MOS sensor technology make it very difficult to perform rapid and accurate analysis of samples which contain large amounts of ethanol or solvent compounds. This work was an attempt to design an MOS sensor based electronic nose which is capable of discriminating between samples that contain a high concentration of alcohol or other solvents. Temperature modulation and novel analysis techniques are employed in an effort to overcome issues associated with using MOS sensors in high concentrations of ethanol. It is specifically targeting whiskey, vodka and other spirits in its application.

1.2 Contributions of this Thesis

The following list summarizes the main work units and accomplishments of this thesis:

- An electronic nose sensor board was designed with an array of 12 commercial off-the-shelf MOS sensors. This sensor board supports temperature modulation on a subset of the sensor array. Work on this element included schematic capture, circuit simulation, and PCB layout. The board itself was professionally fabricated.

- The firmware which operates the electronic nose is written in embedded C++. It includes a analog sampling component which exploits the Direct Memory
Access (DMA) module to enable high speed ADC performance without CPU intervention. It also includes support for MOS sensor temperature modulation using a fast interrupt-driven software component.

- A desktop application was developed to provide a data-driven interface to the electronic nose. It enables real-time control and visualization of the sensor array response and the status of the electronic nose. This application was written in C# and communicates with the firmware.

- Feature extraction based on the Hilbert-Huang transform was developed to process the data recorded from the sensor array board. This method was based on previous work, but was modified to solve noise corruption issues when applied to the type of signal recorded from the electronic nose.

- A basic classification experiment was designed to validate the functionality of the electronic nose. Information about the following compounds was recorded with the electronic nose: ethyl acetate, ethanol, and isopropanol. The compounds were successfully clustered and classified using PCA, self organizing map, and neural net classifiers, based on signature vectors derived from the Hilbert-Huang transform.

In additional to the work mentioned previously, a journal article regarding this work has been accepted for publication in the International Journal of Information Acquisition. A conference paper which will explore additional analysis of the data recorded by the electronic nose is also forthcoming.
1.3 Thesis Organization

This document is organized into a number of chapters. The theory of operation behind the electronic nose and the MOS sensors are presented in chapter 2. This chapter also presents the Hilbert-Huang transform along with some basic theory of its operation. Previous works which combine both temperature modulation of MOS sensors and applications of electronic noses to industry are essentially non-existent at this time. However, several studies related to individual elements of this work are discussed in chapter 3. Chapter 4 concerns the implementation of the electronic nose; it includes circuits schematics, PCB layout, firmware programming, and custom software for the personal computer. A basic classification problem is used to validate the function of the electronic nose in chapter 5. A number of solvents are clustered and classified using data recorded from the electronic nose, which has been processed using the Hilbert-Huang transform. The final chapter summarizes the work which has been performed. It concludes by suggesting improvements to the design and future experiments to continue improving the electronic nose.
Chapter 2

Background

This chapter begins with an overview of concepts related to electronic noses. Following this, the MOS gas sensor is introduced and the temperature modulation technique for MOS gas sensors is discussed. The remainder of the chapter describes the signal processing techniques intended to support the analysis of the temperature modulated sensor response.

2.1 The Electronic Nose

An electronic nose is a sensory device designed to detect odor or flavor. It is distinct from traditional chemical sensors in that it is designed in a biologically inspired manner. These devices are typically used to identify, compare or quantify complex odors in a manner which resembles biological olfaction. Typically, the odor is perceived as a global signature or fingerprint representing the scent.

Biological olfactory organs include millions of sensory proteins which bind to
specific odor molecules using a lock-and-key approach. This allows many animals to perceive remarkably faint odors with incredible specificity. Currently, only the most sophisticated bio-electronic noses can attempt to replicate this approach [6]. The majority of electronic noses employ a slightly different method.

Most electronic noses are constructed with an array of broadly cross-reactive, non-specific sensors. A number of different sensors are selected; each sensor in the array is fabricated differently to introduce variation in its response. Although, typically there will be a considerable overlap in the reactive species for each sensor. In this way, low selectivity can be overcome by pattern recognition techniques designed to extract information from differences in the response across the array. Although many suitable sensors exist, the MOS gas sensor is perhaps the most commonly used due to its combination of broad reactivity, high sensitivity, low circuit complexity, and moderate cost.

Analysis by electronic nose will typically proceed according to fig. 2.1. Dimensionality reduction (or feature extraction) is used to reduce the normalized measurements to a feature vector which represents the odor signature or footprint. Concepts and methods from the fields of digital signal processing (wavelet, Fourier, etc.) and chemometrics (PCA, PARAFAC [7], etc.) are often employed at this stage.

Soft computing techniques such as neural nets and fuzzy logic are typically applied to classify the feature vector. Such techniques often require a database of previously measured odors for comparison. For example, a support vector machine (SVM) classifier must be trained using an existing database of odors before it is able to classify new and unknown odors. Furthermore, training must be repeated should
any changes be made to the sensor array.

Figure 2.1: Operations in the analysis of odors by an electronic nose.

2.2 The MOS Gas Sensor

A metal-oxide semiconductor (MOS) gas sensor is constructed by fabricating a film of chemically reactive metal-oxide crystal such as SnO$_2$ on top of a micro-hotplate heater. The chemistry of MOS gas sensors is an area of active research, but they are generally believed to operate using the following principles. The heater is used to warm the film to temperatures between 100°C and 400°C. At this temperature, negatively charged oxygen from the air is absorbed into the crystal structure of the film. At grain boundaries within the crystal, charge separation occurs (due to the oxygen ions) which prevents charge carriers from moving freely. The sensor resistance is attributed to this effect. In the presence of a reducing/deoxidizing compound such as a carbon monoxide, ethanol, butane, etc., a chemical reaction takes place which produces free electrons and reduces the separation of charge. In this way, the resistance of the sensor is decreased as the concentration of the reducing compound increases.
2.2.1 Dynamic Temperature Mode of Operation

The temperature of the film affects the kinetics of the adsorption and reaction processes which take place within the sensor. Due to the optimum oxidation temperature and the stability of different oxygen species, each analyte gas produces a characteristic response [8]. Modulating the temperature of the film with a periodic signal will produce a characteristic response which has components proportional to the concentration of each compound in the analyte gas [9]. Aside from the analyte gas, a significant portion of the sensor response will be due to a thermistor effect as the temperature of the sensing film changes over time. Furthermore, the unique construction of each type of sensor may introduce additional features in the response. The signal which results will be extremely non-linear as well as non-stationary. Advanced signal processing techniques are required to decompose the response into components representative of the contribution from each compound.

2.3 The Hilbert-Huang Transform

The Hilbert-Huang transform is a signal processing and feature extraction technique which combines empirical mode decomposition with Hilbert spectral analysis by way of the Hilbert transform. The Hilbert-Huang transform is a modern technique which has applications in signal processing for non-stationary and non-linear systems. It is frequently used in applications similar to the short term Fourier transform and the discrete wavelet transform. Unlike the Fourier and wavelet transforms, there is currently no well understood theoretical framework which characterizes the em-
empirical mode decomposition. Rather, the empirical mode decomposition — and the
Hilbert-Huang transform by extension — resemble heuristic methods. The result of
performing a Hilbert-Huang transform on a signal is a time-frequency-energy distri-
bution which preserves the time localities of events in the signal. The first step in de-
termining the Hilbert-Huang transform of a signal is performing the empirical mode
decomposition. However, before considering the empirical mode decomposition, a
brief overview of the Hilbert transform and instantaneous frequency is presented.

2.3.1 Instantaneous Frequency

In the context of signal processing, the Hilbert transform is often used to derive an
analytic representation of a signal. The transform of the signal \( x(t) \) is defined as

\[
H(x)(t) = \frac{1}{\pi} P \int_{-\infty}^{\infty} \frac{X(\tau)}{t-\tau} d\tau, \tag{2.1}
\]

where \( P \) is the Cauchy principal value. The original signal can now be represented
as a complex conjugate pair

\[
z(t) = x(t) + jH(t) = a(t)e^{j\theta(t)}, \tag{2.2}
\]

where

\[
a(t) = \left[ x^2(t) + H^2(t) \right]^{\frac{1}{2}}, \tag{2.3}
\]

\[
\theta(t) = \arctan\left( \frac{H(t)}{x(t)} \right). \tag{2.4}
\]
Using eq. (2.4) it is now possible to define the instantaneous frequency as

\[ \omega(t) = \frac{d}{dt} \theta(t). \]  

(2.5)

This definition of the instantaneous frequency can only describe a single frequency at any given time \( t \). Therefore, this definition is only valid for signals which are monocomponent functions. The exact definition of a monocomponent function is not clearly defined. For lack of a better alternative, monocomponent functions are understood to be narrow band. Huang et al. consider this topic in greater depth in [10]. Fortunately, the empirical mode decomposition may be employed to derive a set of satisfactory functions from an arbitrary signal.

2.3.2 Empirical Mode Decomposition

Empirical mode decomposition is used to decompose an input signal into a set of intrinsic mode function (IMF) components. An intrinsic mode function can be considered as a simple oscillatory mode of a signal. It is similar to a simple harmonic function, but may have variable frequency and amplitude with respect to time. A signal which is purely frequency modulated or amplitude modulated can be considered as an IMF. The definition of an intrinsic mode function as stated by Huang et al. is as follows:

“An intrinsic mode function (IMF) is a function that satisfies two conditions: (1) in the whole data set, the number of extrema and the number of zero crossings must either equal or differ at most by one; and (2) at
any point, the mean value of the envelope defined by the local maxima and the envelope defined by the local minima is zero.” [10]

The method by which intrinsic mode functions are extracted from signal $x(t)$ is called sifting. Begin by identifying all the local extrema (local maxima and minima) in the signal. Define the upper envelope by connecting all the local maxima with a cubic spline. Define the lower envelope by connecting all the local minima with a cubic spline. At any given point, the upper and lower envelopes will have mean $m_1$. The first component $h_1$ is now extracted from the signal using

$$x(t) - m_1 = h_1. \quad (2.6)$$

The operation performed above should have made $h_1$ symmetric, having all positive maxima and all negative minima. If $h_1$ does not yet satisfy the definition of an IMF — it is merely a proto-IMF — the sifting processes is repeated with $h_1$ as the signal such that

$$h_1 - m_{11} = h_{11}. \quad (2.7)$$

The sifting process is repeated up to $k$ times, at which point $h_{1k}$ is an IMF and

$$h_{1(k-1)} - m_{1k} = h_{1k}. \quad (2.8)$$

A stopping criteria is selected to determine when the sifting process is complete. The sum of differences approach proposed by Huang et al. stops the sifting process
when the difference between sifting iterations $SD_k$ defined as

$$SD_k = \frac{\sum_{t=0}^T |h_{k-1}(t) - h_k(t)|^2}{\sum_{t=0}^T h_{k-1}^2(t)}$$  \hspace{1cm} (2.9)$$

is less than a pre-determined limit. An alternative stopping criteria which is commonly used is called the $S$-number. In this case, a value for $S$ is selected ahead of time and the sifting process stops only when the number of extrema has not changed for $S$ consecutive siftings. It is sometimes sufficient (though not necessarily efficient) to simply perform a fixed number of sifting iterations and forgo the use of a stopping criteria.

Once the stopping criteria have been met, the first intrinsic mode function $c_1$ has been identified as

$$c_1 = h_{1k}. \hspace{1cm} (2.10)$$

This intrinsic mode function should contain the components of the signal with the shortest period or finest detail. The intrinsic mode function $c_1$ can now be removed from the original signal to produce a residue $r_1$ by

$$x(t) - c_1 = r_1. \hspace{1cm} (2.11)$$

The residue may still contain some longer frequency components of the signal. Additional intrinsic mode functions may be extracted by repeating the entire sifting process with the residue in place of the signal. The entire process is repeated until

$$r_{n-1} - c_n = r_n, \hspace{1cm} (2.12)$$
The empirical mode decomposition of a signal which is the sum of two sines with frequency 0.01 Hz and 0.06 Hz.

and the final residue $r_n$ is a monotonic function. At this point, no more intrinsic mode functions can be extracted from the residue. The original signal $x(t)$ can be reconstructed from the intrinsic mode functions with

$$x(t) = \sum_{i=0}^{n} c_i + r_n.$$  \hspace{1cm} (2.13)

The empirical mode decomposition of a signal which is the sum of sines with frequency 0.01 Hz and 0.06 Hz is presented in fig. 2.2.

### 2.3.3 Ensemble Empirical Mode Decomposition

The empirical mode decomposition can be disturbed by a variety of mechanisms such as noise, mode mixing and spectral leak [11]. Significant corruption by interference
mechanisms may result in intrinsic mode functions which are not monocomponent. This obscures physical meaning of an intrinsic mode function and may adversely affect the instantaneous frequency estimate. Mode mixing occurs when one intrinsic mode function contains signals of widely disparate scale or when different intrinsic mode functions contain signals of similar scale. Signals with intermittent components are more susceptible to corruption by mode-mixing.

Ensemble empirical mode decomposition (EEMD) is a variation of empirical mode decomposition designed to reduce or eliminate the potential for mode-mixing to occur. Ensemble EMD is a type of noise-assisted data analysis which considers the true intrinsic mode functions as the mean of an ensemble of decompositions with added white-noise [12]. Since empirical mode decomposition is equivalent to a dyadic filter bank with an input of white noise, the addition of a finite amount of white noise to the signal makes it more likely that signals of similar scale will be extracted together.

The following additional steps are required to implement ensemble empirical mode decomposition.

1. Generate a white noise series $w(t)$. Add $w(t)$ to the signal $x(t)$ to get $x_w(t)$.

2. Decompose the signal $x_w(t)$ using empirical mode decomposition.

3. Repeat steps 1 and 2 with different realizations of the white noise series.

4. Obtain the means of corresponding intrinsic mode functions of the decompositions.
The EEMD method may be tuned somewhat by selecting different amplitudes of white noise. A noise amplitude of 0.1 to 0.2 standard deviations of the input signal amplitude is typically selected. The total number of white noise series which are employed is called the ensemble count. The ensemble count must be sufficient to remove the majority of added noise when the extracted IMFs are averaged. The white noise amplitude and ensemble count are determined experimentally using a trial-and-error approach to refine the decomposition.

2.3.4 Hilbert Spectrum

Having obtained the intrinsic mode functions, the Hilbert transform can now be applied. The magnitude and instantaneous frequency of each intrinsic mode function is determined using eq. (2.3) and eq. (2.5). The residue may be ignored since it is either a constant, or a monotonic function representing a trend in the data. Although the Hilbert transform can accommodate the trend as part of a longer oscillation, its energy content will likely be overpowering. In many cases, it is the oscillatory behavior of the signal which is of interest so the residue is omitted. The original signal can now be reconstructed from the Hilbert transforms by

\[ X(t) = Re \sum_{i=1}^{n} a_i(t)e^{i\theta_i(t)}. \]  

(2.14)
This is an equation in time \( t \), amplitude \( a_i(t) \) and frequency \( \omega_i(t) \). Thus, we can construct the Hilbert spectrum as

\[
H(\omega, t) = \text{Re} \sum_{i=1}^{n} a_i(t) e^{j \int \omega_i(t) dt}.
\] (2.15)

The marginal Hilbert spectrum (MHS) offers a measure of the total amplitude contribution from each frequency and is constructed from eq. (2.15). It measures the cumulative amplitude at each frequency across the entire signal in a probabilistic sense. The marginal Hilbert spectrum is calculated as

\[
H(\omega) = \int_{0}^{T} H(\omega, t) dt,
\] (2.16)

which integrates the Hilbert-Huang spectrum over time. The Hilbert-Huang transform and marginal Hilbert spectrum of the example in fig. 2.2 are plotted in fig. 2.3a and fig. 2.3b respectively. It can easily be observed that the example signal is composed of frequency components at 0.01 Hz and 0.06 Hz.

### 2.3.5 A Comparison of Transforms

Although the Hilbert-Huang transform lacks the complete theoretical understanding provided by Fourier and wavelet transforms, in practice a number of advantages are observed.

"The key advantage of using a Hilbert Transform, rather than FFT or wavelet processing, is that it allows the use of instantaneous frequency
(a) The Hilbert-Huang spectrum determined for the example signal in fig. 2.2.

(b) The marginal Hilbert spectrum calculated by integrating fig. 2.3a over time.

Figure 2.3: Hilbert spectral analysis of the example signal in fig. 2.2.
to display the data in a ‘time-frequency-energy’ format. This produces a more accurate ‘real-life’ representation of the data without any of the artifacts imposed by the non-locally-adaptive limitations of FFT or wavelet processing.” [13]

A comparison of common transforms of a linear chirp signal is presented in fig. 2.4. The chirp signal begins at 0.0001 Hz and increases linearly to 0.01 Hz at 1000 s. The Fourier transform is designed for signals which are both stationary and linear. Fourier only considers frequency and amplitude, so the transform has no time component. It cannot produce an accurate representation of the non-stationary chirp signal. The Wavelet transform can be considered as similar to a short-term Fourier transform. With this transform, the signal does not have to be stationary. The drawback to using the Wavelet transform is that frequency information is not directly encoded in the spectrum. Pseudo-frequency can be determined from the wavelet scale, but physical relationships are lost. The Wavelet transform also exhibits blurring of the energy content of the signal. However, the linear change in frequency is easily observed in the Hilbert-Huang spectrum.

2.4 Summary

This chapter presented concepts related to the construction of electronic noses based on MOS sensor technology. It also described the dynamic temperature mode of operation for MOS sensors. As a consequence of the highly non-linear response of the temperature modulated MOS sensors, the Hilbert-Huang transform is introduced.
Figure 2.4: A comparison of time-frequency-energy spectrum of Fourier, Wavelet and Hilbert-Huang transforms for a linear chirp signal.

The Hilbert-Huang transform — which combines empirical mode decomposition with Hilbert spectral analysis — was presented as an alternative to Fourier and wavelet analysis for the sensor response. The advantage of the Hilbert-Huang transform is that by using the instantaneous frequency the data can be represented as a time-frequency-energy spectrum.
Chapter 3

Literature Review

This chapter considers two kinds of work which apply to this thesis. The first is basic research into the application of temperature modulation to MOS sensors with the goal of increased sensitivity and selectivity of the sensor. There are two facets to this research: the type of modulation (sine, step, etc) and the analysis of the complex signals which result. Works in this area typically proceed by interrogating a sample with a temperature modulated MOS sensor, decomposing the response using a transform such as discrete wavelet transform, and feeding the coefficients into a classifier or clustering algorithm.

The second type of work under investigation is previous applications of electronic nose technology within the alcoholic beverage industry. The majority of this work focuses on wine due to its subtle characteristics, low concentration of ethanol, and prominent flavor and aroma profiles. Due to the harsh character of whiskey or other spirits with high concentrations of ethanol, MOS based electronic noses often perform
poorly. Most research in this area involves exotic dopants for the MOS sensors or abandons MOS for another technology altogether.

### 3.1 Temperature Modulation of MOS Sensors

The Hilbert-Huang transform is a relatively new method and its many applications are still being explored. Previously the Hilbert-Huang transform has been applied to complex non-linear data in the fields of: vibration signal analysis [14, 15]; seismic analysis [16], structural damage assessment [17], speech recognition [18], and so on. In 2010, Wen et al. [19] applied the Hilbert-Huang transform as an analysis technique for temperature modulated MOS gas sensors. In that work a single MOS sensor was able to discriminate between three gases: 3000 ppm methane, 150 ppm carbon monoxide and 15 ppm ethanol. The sensor used in the study was a micro-hotplate MOS sensor with a SnO$_2$ film thickness of 300 nm.

Most work involving temperature modulated MOS sensors employ traditional transforms such as the fast Fourier transform and the wavelet transform. The wavelet transform has been applied in a similar capacity to the Hilbert-Huang transform with considerable success. It is possible to produce a time-pseudofrequency-energy spectrum similar to the HHT using the wavelet coefficients. The wavelet coefficients are typically used as the input to a classifier such as a support vector machine (SVM) or other neural network. Methods taken from the field of chemometrics\(^1\) are also being applied to the analysis of the temperature modulated response as well.

\(^1\)Chemometrics is the science of extracting relevant information from chemical systems by data-driven means. [20]
Ge et al. [21] are able to achieve greater than 90% successful classification of mixed CO and H₂ based on wavelet processing and SVM classification using temperature modulated MOS sensors. A sinusoidal temperature profile is employed in this study. Similarly, Al-Khalifa et al. [22] achieve similar rates of classification of CO and NO₂ with a similar method. Jaisutti and Osotchan [23] employ a step-temperature modulation scheme with an array of eight commercial off-the-shelf MOS sensors to improve the classification of a variety of vapors. In this study, a Haar and Db4 wavelet was applied to decompose the signals and the coefficients used as input for PCA. Analysis of the transient signal was demonstrated to improve the selectivity and sensitivity of the MOS sensors. Ionescu and Llobet [24] found that the discrete wavelet transform outperformed the FFT in the classification of ethanol, acetone, and ammonia. A square wave temperature modulation signal was used and the transient response was analyzed. However, this experiment employed a tungsten trioxide sensor rather than the more common tin-dioxide MOS sensor. Chutia and Bhuyan [25] examine the performance of two commercial MOS sensors by analyzing the transient response to a step-temperature modulation. They determine that the transient response exposes additional information about the analyte.

3.2 Applications to the Alcohol Industry

Food and beverage industries are the primary utilizers of electronic nose technology. Electronic noses are employed in some the following capacities: detecting the shelf-life, ripeness and freshness of fruits; sensing spoilage or contamination in fish, seafood
and meat; determining maturity of cheese or other fermented products; classifying vintage or origin of beer, wine and liquor; and, classification of ingredients in blended products such as coffee and tea. The majority of work with electronic noses is still in the research phase; the sensors and techniques used in their construction are recent developments.

In the study by Martí et al. [26], a number of previous papers on quality control for alcoholic beverages are considered. The sensor technologies considered in each study include metal-oxide semiconductor, quartz micro-balance, and conductive polymer. Data is analyzed using treatments such as principal component analysis, artificial neural network, linear discriminant analysis, and discriminant function analysis. The study concludes that although MOS sensors demonstrate promise in many applications, they are rarely employed in the study of alcoholic beverages due to a number of drawbacks. Most work with MOS sensors requires extensive preparation of samples to remove ethanol and water from the sample or headspace gas. High concentrations of ethanol have an unfortunate tendency to “numb” MOS sensors [27]. In some studies, the MOS sensors experienced a delayed baseline recovery when exposed to high molecular weight compounds. Poisoning by sulphur containing compounds and weak-acids was also determined to be a problem. The study further concludes that strategies which are designed to mitigate the impact of high concentrations of ethanol tend to compromise the speed and simplicity associated with MOS sensor technology. As a result, relatively few studies of alcoholic beverages have been made with MOS sensor based electronic noses. The paper ultimately suggests that MOS gas sensors are better suited to screening products due to their advantages in cost
and simplicity, and more involved techniques should be considered for confirmation.

In an early example of the application of electronic noses to alcoholic beverages, Shurmer and Gardner [28] classified three different types of beer (two lagers and a bitter) using a 12-element array of tin-dioxide MOS sensors. These beers were successfully classified using a perceptron with back propagation learning.

Ragazzo-Sanchez et al. [29] employed a FOX model 4000 commercial electronic nose from AlphaMOS to identify a variety of alcoholic beverages including beer, wine, and spirits. Simultaneous analysis was performed by gas chromatography (Intersmat IGC 121C) equipped with a FID detector. Dehydration and dealcoholization procedures were performed before injection into the electronic nose. Two pattern analysis methods were applied: principal component analysis, and discriminant factorial analysis. Classification of beer and wines was the easiest, while confusion was most likely to occur between vodka, tequila, and whiskey.

Wine has complex and subtle flavor and fermentation chemistry. There is considerable interest within the wine industry in using electronic noses to produce qualitative analysis of wine products. Lozano et al. [30] performed an analysis of 29 common aromas found in white wine. Each chemical compound in the study was assigned to a class such as fruity, floral, chemical pungent, chemical oxidized, etc. The electronic nose apparatus consisted of an array of 16 tin-dioxide MOS sensors with film thickness between 200 and 800 nm at 250 °C. Using both PCA and probabilistic neural networks, perfect classification of floral, fruity, herbaceous, and microbiological aromas was obtained. Classification of chemical odors was less successful.

Classifying wines by vintage is a common challenge among producers and con-
sumers of fine wines. In [31], an array of five tin-dioxide MOS sensors was used to classify wine from five vintages (1989, 1990, 1991, 1992, and 1993) produced in a well delineated geographical location. Although the electronic nose had some confusion, it was able to identify a shift in the wine making procedure. Wines produced in 1990, 1991, and 1992 made using the barricatura\textsuperscript{2} process were identified by PCA. A similar approach is used in [32] to determine the vineyard in which four different red wine products originate. The analysis was performed using an array of four MOS sensors. In this case, the electronic nose was able to outperform the standard chemical analysis used by wine-makers for this application.

Electronic noses could potentially detect counterfeit products for the beverage industry. The chemistry of many alcoholic beverages depends on factors such as the type of wood cask in which the beverage is aged. Ethanol is responsible for liberating many compounds from the wood [5]. Lozano et al. [33] investigate wines made from the same grapes which are aged in French and American oak barrels. Analysis was performed with an array of 16-MOS sensors at at 250 °C. Age was determined to have a larger impact than type of wood on the wine chemistry. Clustering performance was found to be good, and a high rate of successful classification was achieved with a probabilistic neural network.

Analysis of whiskey is a much more daunting task as less than 1% of the chemical content contributes to the identification of the sample [34]. There are fewer examples of work involving MOS sensor electronic noses as applied to the analysis of whiskey or other strong spirits. At this time the majority of previous work focuses on beer

\textsuperscript{2}The barricatura process involves aging the wine in a small vessel which previously contained brandy, for a more flavorful wine.
and wine, but studies on whiskey are slowly emerging.

Counterfeit whiskey is also a growing problem in many markets. Whiskey is required to have an ethanol content of at least 40% ABV. Ashok et al. [35] have very successfully used an optofluidic chip with near-infrared spectroscopy to identify alcohol content in whiskey within 1% error. In another study, Wongchoosuk et al. [2] describe a portable electronic nose which includes tin-dioxide MOS sensors doped with carbon nanotubes to improve selectivity to methanol. They are able to detect whiskey contaminated with greater than 1% by volume concentration of methanol. Similar work involving MOS sensors is in its infancy due to the difficulty of sensing in high concentrations of ethanol.

### 3.3 Summary

The studies presented in this chapter demonstrate that temperature modulation is a valid method of improving the performance of MOS sensors. A number of studies have demonstrated that temperature modulation is a simple and inexpensive method to improve the sensitivity of MOS sensors. The difficulty lies primarily in performing a comprehensive analysis of the signals which result. There are many algorithms which have been applied in this area with varying success. Many methods taken from the field of chemometrics are currently under investigation. The Hilbert-Huang transform is a relatively new signal processing tool. Previous work with superficially similar methods such as the discrete Wavelet transform has demonstrated promise. Unfortunately, there is little work connecting these new methods to actual appli-
cations in the alcoholic beverage industry. Although beer, wine, coffee, and other beverages have been studied extensively, whiskey remains a difficult target for MOS-based electronic noses.
Chapter 4

Implementation

In this chapter, the design and implementation of the electronic nose is considered. The rationale for each design choice is presented, and ramifications of these choices are discussed. The chapter begins with an overview of the electronic nose system and its capability. The remainder of the chapter is an in depth discussion of the major work units in the design: schematic capture and PCB layout, microcontroller firmware, desktop capture software, and the communication between the desktop and the microcontroller.

4.1 The Prototype Electronic Nose

The prototype electronic nose was intended as a proof-of-concept for a later handheld device. The device is pictured in fig. 4.1 along with an exploded view indicating the various components of the electronic nose. The body of the electronic nose is an unpainted Hammond Mfg. R191-170-00 assembly. This is an airtight 2.5 L container.
manufactured from die-cast aluminum alloy. External wiring such as power and USB are introduced into the enclosure by way of airtight strain reliefs. A voltage regulator board installed in a black plastic project box provides additional power supply voltages for the sensor board.

An air flushing system is installed in the lid of the enclosure to exhaust the sample gases after analysis. At one end of the enclosure, a vial heater has been mounted. The vial heater has a port into which target vials containing sample may be introduced into the enclosure. The vial heater is temperature controlled by means of an integrated heater which can maintain the sample at any temperature between ambient and 80°C. Airflow between the vial heater and enclosure is controlled by an integrated ball-valve in the body of the vial heater. A fan is installed opposite the vial heater to promote mixing of the sample gases. Relay boards mounted on the
exterior of the enclosure allow the sensor board to operate the vial heater and fan.

A MOS sensor array consisting of 12 sensor cells has been constructed on a printed circuit board which is mounted inside the enclosure. A TWR-K40X256 microcontroller board powered by a Freescale Kinetis K40 CPU operates the MOS sensor array. This microcontroller integrates a 100 MHz ARM core, DMA memory controller, two high speed 16-bit ADC modules, three FlexTimer modules, and USB/serial communication. It is programmed to communicate with the laptop computer in real-time to visualize and record the array response.

4.1.1 MOS Sensor Cells

Figaro Engineering and FiS are two well established manufacturers of MOS sensor technology. Each company offers a product line which targets a wide variety of interesting compounds. Furthermore, the TGS-2600-series MOS gas sensors selected from Figaro Engineering are electrically compatible, making it easy to socket different sensors into the same circuit. The same is true of the SB-series sensors from FiS. Selecting from among these products means the sensor array need only support two series of sensor, but may target a variety of applications.

The sensor array consists of twelve MOS sensor circuits which are divided into two groups. The first group of six circuits are socketed to accept TGS-series sensor cells manufactured by Figaro Engineering. The remainder are socketed to accept SB-series sensor cells manufactured by FiS. Two circuits from each group include programmable circuitry to operate the sensors using dynamic temperature mode.

The array consists of twelve sensors as this was the largest number of sensors
that could be supported by the I/O available on the microcontroller. Being a prototype design, some sensor circuits were expected to fail. Therefore, the array was constructed with more positions than was strictly necessary. Ultimately, the sensor board proved more resilient than expected and no sensor circuits failed. Instead, the array was typically operated with multiple positions filled by identical MOS sensors. This is still useful as variations in manufacturing meant that even MOS sensors within the same product line had minor differences in response.

The MOS sensors purchased from each manufacturer were selected based on two criteria: (1) the ability to operate in high concentrations of ethanol/solvent vapour, and (2) sensitivity to alcohols and volatile organic compounds. High concentrations of solvent vapour typically destroy MOS sensors, so tolerance is important for this field of application. However, increased tolerance does trade-off sensitivity to lower concentrations (less than 500 ppm) of gas. The following list summarizes the properties of the MOS sensors selected for this electronic nose.

**TGS-2610-C:** This cell combines high sensitivity to LP gas with low power consumption and quick gas response. It is suitable for measuring concentrations up to 10,000 ppm.

**TGS-2610-D:** This cell has identical sensitivity to LP gas as the TGS-2610-C, but the cell housing includes a zeolite\(^1\) filter to reduce sensitivity to interference gases such as ethanol.

**TGS-2620:** This cell is sensitive to alcohol and solvent vapors up to 5000 ppm. It is suitable for measuring concentrations up to 5000 ppm. It

---

\(^1\)Zeolites are a class of microporous aluminosilicate minerals commonly used as commercial adsorbents.
is also sensitive to a variety of combustible gases such as carbon monoxide, making it a good general purpose sensor.

**SB-15-00:** This cell has high sensitivity and fast response to propane/butane and other LP gases up to 10,000 ppm.

**SB-AQ1-06:** This cell is suitable for monitoring of indoor air quality. It is sensitive to volatile organic compounds, solvent vapors, and pollutant gases between 10 ppm and 10,000 ppm.

The TGS-series MOS sensor cells are manufactured in standard TO-5 packages modified to include a mesh vent. The SB-series MOS sensor cells appear to use a propriety package. Mesh vents are used as anti-explosion features on sensors which measure combustible gases such as ethanol vapour, LP gas, etc. Each type of MOS sensor cell mentioned in the list above is pictured in fig. 4.2.

![Figure 4.2: The MOS sensor cells from left to right: TGS-2610-D, TGS-2610-C, TGS-2620-C, SB-15-00, and SB-AQ1-06.](image)
4.2 Schematic Capture and PCB Layout

Schematic capture and PCB layout were performed using the Cadence OrCAD suite of applications. Schematics were designed in OrCAD Capture and the netlists were exported for layout using OrCAD PCB Designer. The majority of the integrated circuits used in the design are selected from products manufactured by Analog Devices. The remaining components were purchased from various manufacturers via electronics supplier Digi-Key. Printed circuit boards were fabricated and assembled by Crest Circuit, a fabrication house located in Toronto, Ontario.

The design of the electronic components of the electronic nose include the following units: power supply, regulator board, sensor array board, and power-switching boards. The power supply and regulator board provide supply rails for the sensor array board. The sensor array board contains a variety of sensor circuits, control circuits and a microcontroller. The power switching boards allow the microcontroller to switch high current devices such as the vial heater and fan.

4.2.1 Regulator Board

A pair of dual output bench power supplies are used to power the electronic nose. Each supply is capable of providing up to 6 V at 2.5 A and up to 20 V at 0.5 A. One power supply provides all the power required by the sensor array. The other supply provides power for the vial heater and fan. The final configuration of the power supplies is shown in fig. 4.3.

The original design for the sensor array included 0.9 V, 3 V and ±5 V supply
rails. The 0.9 V rail was included as an option for the SB-series sensors, though it was never used. During testing of the prototype, the need for a 5.25 V supply rail was discovered (see section 4.2.2.) A regulator board was introduced which produces the required voltages using linear low-dropout (LDO) regulators. The schematic of the regulator board is displayed in fig. 4.4. The 3 V and 5 V rails are regulated by fixed output LDO regulators. The 5.25 V rail is generated by an adjustable LDO regulator where the voltage is selected by a precision potentiometer. A considerable current is drawn from each regulator, so they are installed on heat sinks. An LED on the output of each regulator sinks the minimum current necessary to enable regulation. Photographs of the completed regulator board are displayed in fig. 4.5.

Each TGS-series sensor cell requires 60 mA and each SB-series sensor cell requires 130 mA. The entire array and associated circuitry require approximately 1.2 A to operate. The SB-series sensor cells draw current from the 3 V rail. Four of the TGS-series sensors draw power from the 5 V supply rail. Two of the TGS-series sensor cells are powered by the 5.25 V rail. Sensors can be disabled either by jumper or software to reduce current consumption. The various multiplexers, IC switches and
Figure 4.4: The schematic of the regulator board which produces the power supply rails for the electronic nose.

(a) Linear regulators mounted on heatsinks on the exterior of the project box.  
(b) Internal circuits and wiring of the regulator board.

Figure 4.5: Photographs of the regulator board project box.
auxiliary circuits are primarily powered by the 5 V rail.

A second power supply is dedicated to the vial heater and fan. The resistive heater which warms the vials consumes a current in excess of 2.5 A so a dedicated supply is necessary. The separate supply also helps to isolate the power supply noise caused by the high current square wave used to control the heater temperature.

4.2.2 Sensor Array Board

The sensor array board is a densely populated 187 mm by 156 mm PCB which consists of mixed through-hole and surface mount components. Two etch layers are included as well as soldermask and silkscreen on the top and bottom of the PCB. The width of each trace is 10 mil, except power traces which are 20 mil wide. Up to 12 MOS sensors, two temperature and humidity modules, and one TWR-K40X256 evaluation board can be installed in the sensor array board. External connections to relay boards which control the vial heater and fan speed are also included.

The right-most section the circuit board (as pictured in fig. 4.6) may be removed to facilitate installation of the microcontroller board. Fortunately, the majority of connections are made on the left (primary) side of the microcontroller. A right-angle board-to-board connector bridges a handful of connections across the gap in the sensor board. Temperature and humidity sensor modules may be installed; one each in the small rectangular white headers above and below the leftmost PCIe connector. A four pin unshrouded white header located below the PCIe connector is used to interface with the relay boards which control the vial heater and fan. The large blue and yellow components (inductors, capacitors) on the top and bottom of the PCB
Figure 4.6: A photograph of the sensor array printed circuit board. An array of socketed sensor circuits populate the left half of the board, while the right contains the microcontroller board and support circuitry.
are power filters which support the sensor array. The left half of the sensor board consists of the 12 MOS sensor circuits.

The microcontroller interfaces with the various circuits using 59 digital and analog signals. Many of the microcontroller pins are multiplexed and may perform multiple functions depending on the microcontroller configuration. During the schematic capture, the assignment of pins to signals in the sensor circuits was fairly arbitrary. Once the placement of components was finalized in the PCB layout phase, these assignment were revised to optimize the routing of each signal trace.

All the circuits on this PCB are designed conservatively; jumpers and unpopulated pads are included such that potential defects in the circuits may be circumvented. Every IC has a 0.1 µF and 1 µF bypass capacitors located in adjacent to its power pins. Every analog line has a 0.01 µF bypass capacitor located as close to the microcontroller interface as possible.

The following sections will discuss in depth the nature and function of the circuits which constitute the sensor board. The majority of the sensor board consists of four variations of the MOS sensor circuit. The remaining circuits support additional sensors or functions. Additional information – particularly the interface between these circuits and the microcontroller – is available from the complete schematics.

**Fixed Voltage Sensor Circuit**

There are four instances of each version of the circuit in fig. 4.7 for a total of eight sensor circuits with fixed heater voltage in the sensor array. The ADG802 is a SPST power switching IC in a normally closed configuration. It is used to switch the power
to the heater in the sensor cell. The heater is powered whenever the SWn line is low, but a jumper can be used to override the CPU selection. The FiS version of the circuit also includes jumpers to select between 0.9 V and 3 V heater supply voltages.

The SB-series sensors require a 0.9 V heater supply voltage. Since this voltage is not available, a resistor is inserted in the heater circuit to limit the heater current when a 3 V supply is used instead. The typical heater current $I_H$ is 130 mA. To obtain the equivalent current at 3 V requires the circuit to present a resistance of $3 \text{ V} \div 130 \text{ mV} \simeq 23 \Omega$. A 22 Ω resistor (R77 in fig. 4.7b) rated for 1 W was installed in this capacity.

The AD810 opamp which buffers the analog signal from the sensor cell saturates at ±3 V. Also, the ADC module in the Kinetis CPU has a maximum analog reference voltage of 3.3 V. However, the sensor cell is installed in a half-bridge with a 5 V input. Since the output of the half-bridge may exceed 3 V, the analog signal from the sensor may be greater than the measurable range. The range of measurable resistances for each sensor depends on two parameters: which side (upper or lower) of the bridge the sensor is installed in, and the selection of the load resistor.

For sensor cells installed in the upper half of the bridge, the reading will decrease towards zero as the sensor resistance increases. However, the heater and the negative sense terminals are both internally connected to pin 3 on the SB-series sensors. Those sensors must be installed in the bottom half of the bridge so that pin 3 can be grounded. As a result, the reading will increase towards 5 V as the sensor resistance increases.

A bank of four resistors is available for use as a load resistor in each half-bridge.
Figure 4.7: The schematics for the MOS sensor circuits with a fixed heater voltage.
Table 4.1: Table of measurable ranges for every configuration of the sensor circuits.

<table>
<thead>
<tr>
<th>Resistor</th>
<th>RS Measurable Minimum</th>
<th>RS Measurable Maximum</th>
<th>RS Measurable Minimum</th>
<th>RS Measurable Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 kΩ</td>
<td>666.6 Ω</td>
<td>∞</td>
<td>6.04 kΩ</td>
<td>9.06 kΩ</td>
</tr>
<tr>
<td>2.49 kΩ</td>
<td>1.66 kΩ</td>
<td>∞</td>
<td>12 kΩ</td>
<td>18 kΩ</td>
</tr>
<tr>
<td>4.99 kΩ</td>
<td>3.326 kΩ</td>
<td>∞</td>
<td>18 kΩ</td>
<td>27 kΩ</td>
</tr>
<tr>
<td>10 kΩ</td>
<td>6.6 kΩ</td>
<td>∞</td>
<td>24.9 kΩ</td>
<td>37.35 kΩ</td>
</tr>
</tbody>
</table>

The resistor bank is multiplexed by an ADG1604 4-to-1 multiplexer. The CPU is able to select the desired resistor by configuring the MUXn lines. The multiplexer is enabled with the ENn line. When the multiplexer is not enabled, the input is high-impedance and the sensor becomes an open-circuit. The multiplexer enable can also be selected by jumper.

The load resistor in each half-bridge is multiplexed for two reasons. As a result of the manufacturing process, each batch of sensor cells has a different characteristic resistance. Using a multiplexer allows the CPU to select the load resistor which best suits the sensor installed in the circuit. Furthermore, if the signal from the sensor goes out of the measurable range, it may be possible to select a different load resistor to restore the output to a measurable value. The measurable range for each circuit configuration is indicated in table 4.1. It is also possible to install a new resistor using the unpopulated (DNP) footprint. To use this unpopulated footprint, the multiplexer IC must first be disabled by jumper.

An RC low-pass filter is installed to reject high frequency interference before the half-bridge output is buffered by the opamp. The cutoff frequency for this filter

\[ f_c = \frac{1}{2\pi RC} \]
is 256 kHz. Also, to protect the CPU and opamp from over voltage, an ADG465 channel protection IC is installed to clamp the half-bridge output at 3.6 V.

Also note that due to the limited number of GPIO lines, two circuits are controlled by each SWn and ENn line. The circuits are grouped as SEN2, SEN8; SEN3, SEN9; SEN4, SEN10; and SEN5, SEN11.

**Variable Voltage Sensor Circuit**

The variable voltage circuit is designed to facilitate the dynamic temperature mode of operation of the MOS sensor cell. There are two instances of each version of the circuit in fig. 4.8 for a total of four sensor circuits with variable heater voltage in the sensor array. In these circuits — which are otherwise identical to those in fig. 4.7 — the ADG802 and the SWn line are replaced by a programmable voltage PWMn. As before, due to the limited number of GPIO lines, two circuits are controlled by each ENn line. The circuits are grouped as SEN1, SEN6; and SEN7, SEN12. Since there are no other notable changes to the sensor circuit, this section will discuss the power regulating circuits which generate PWMn.

The PWMn voltages are generated in the highlighted sections of fig. 4.8 by pulse width modulation. The FTMn control signals are produced by the FlexTimer modules in the Kinetis CPU. These signals are 3.3V square waves with variable duty cycle. The ADG802 converts each logic level FTMn square wave into a high current 5V square wave. A flyback diode is introduced to protect the ADG802 from back-EMF generated by the inductors in the following filter section. The filter section is a 4th order low-pass filter which produces a DC voltage from the square wave. The
Figure 4.8: The circuit schematic for the MOS sensors with a variable heater voltage. The power switching IC in fig. 4.7 is replaced by a programmable voltage PWMn generated by the highlighted section.
output voltage is linearly related to the duty cycle of the FTMn square wave.

The FlexTimer module is essentially a counter which is incremented each cycle of the module clock $C_M$. Initially, FlexTimer output FTMn is high, but it goes low when the counter reaches the value $CV$. When the counter reaches the value $MOD$, the counter is reset. Therefore, the duty cycle $D_{FTM}$ of FTMn is determined by $D_{FTM} = \frac{MOD}{CV}$ and the carrier frequency $F_{FTM}$ is determined by $F_{FTM} = \frac{C_M}{MOD}$. With the fastest clock $C_{FTM} = 24$MHz and $MOD = 2000$, the carrier frequency is 24kHz and the resolution of the duty cycle is $\frac{1}{2000}$. At 5V, the resolution of the programmable voltage is $\frac{5V}{2000} = 2.5$ mV.

The low-pass filter is designed to provide suitable attenuation at 24 kHz. A limited selection of inductors were available with a reasonable cost, sufficiently small size, and large enough current rating. The frequency response of the filters is plotted in fig. 4.9. The TGS-series version of the filter provides 53 dB of attenuation which results in a current ripple of approximately 25 $\mu$A. The design of the filter for the SB-series heater was more difficult due to the much lower heater resistance. The SB-series version of the filter provides 47 dB of attenuation which results in a current ripple of approximately 500 $\mu$A.

Every effort was made to select inductors with the lowest possible equivalent series resistance. The ESR of the inductors causes a voltage drop which limits the output voltage to 4.8 V when the 5 V supply is used. An externally generated 5.25 V supply is introduced via pins 8 and 10 on jumpers J1 and J3. This external voltage compensates for the voltage drop due to the ESR of the inductors used in the filter. The ADG802 cannot switch voltages appreciably higher than the chip supply voltage,
Figure 4.9: Simulated frequency plots for the 4th order low-pass filters in fig. 4.8. The −3 dB cutoff frequency and attenuation at the PWM carrier frequency are indicated. The simulations were performed with PSpice.

(a) Filter for the TGS-series version of the circuit.

(b) Filter for the SB-series version of the circuit.

Figure 4.10: A temperature and relative humidity sensor module with its interface circuit.

(a) This circuit buffers the analog signals (b) An HTG3535CH temperature and relative humidity sensor module with cable.
so 5.3 V is the maximum voltage which can be introduced via J1 and J3. Nevertheless, 5.25 V provides a sufficient headroom to compensate for the voltage drop. The SB-series version of the circuit only produces voltages up to 0.9 V from the 3 V supply.

Miscellaneous Circuits

The Humirel HTG3500-series temperature and relative humidity sensor module is a dual output transducer which integrates a humidity sensor and NTC thermistor. The circuit in fig. 4.10a buffers the analog signals from the sensor module using a dual op-amp AD8032 integrated circuit. The relative humidity is output directly as a voltage on pin 4. The NTC thermistor – which is internally connected to pin 3 and ground – is installed in a half-bridge configuration using resistor R6. Two modules may be installed on the sensor board for redundancy or monitoring of specific locations within the electronic nose. The module itself (pictured in fig. 4.10b) plugs into the sensor board using a short cable.

A four pin header is used to interface with the two relay boards that switch the vial heater and fan. The header carries two digital signals to switch the relay boards, a temperature sense signal, and a common ground. The interface circuitry for this header is displayed in fig. 4.11. The microcontroller pins which interface to FTM4 and FTM5 may be configured for PWM or digital switching. Pulse width modulation is preferred such that the vial heater temperature may be controlled by PID loop. An NTC thermistor is mounted on the vial heater to measure the vial temperature. The thermistor is installed in a half-bridge configuration with resistor R5.
4.2.3 Relay Board

The relay board was introduced to switch the vial heater and fan. Both devices are powered by a power supply which is separate from the one which powers the sensor board. The vial heater consists of series of power resistors having a total resistance of 2.5Ω and consuming up to 2.8A of current. The fan consumes up to 0.5A at speeds normally employed. To facilitate the control of these devices, the relay board contains a simple circuit designed to switch high current devices using a logic level control signal.

The schematic is pictured in fig. 4.12a. It is essentially a push-pull transistor totem used to drive a power MOSFET. The totem is constructed from matched general purpose NPN and PNP transistors. The NPN power MOSFET is used in a low-side switch configuration to operate the load. A MOSFET is preferred over a relay to provide high speed switching for pulse-width modulation. The following connections are indicated for this board: J47 for the power supply, J48 to the sensor board for control, and J53 to the switched load. In the case of the vial heater, a ther-
mistor is connected to J51 to provide the temperature sense signal. A photographs of the relay board is printed in fig. 4.12.

The MOSFET on the relay board which operates the vial heater experiences a considerable switching loss due to the high current consumption of the heater. To prevent the MOSFET from overheating, a 250Hz carrier frequency was selected. This frequency is low enough to minimize losses but fast enough to ensure proper control of the temperature of the vial heater.

4.3 Microcontroller Firmware

The microcontroller firmware is a minimal C++ application designed to facilitate data collection. The firmware uses Freescale MQX as the real-time operating system and application host. This allows the firmware to divide the various functionalities into a number of real-time tasks which may run concurrently. A number of data structures provided by MQX (synchronization constructs, message queues, etc.) are
also employed. The firmware works in conjunction with the capture software to provide real-time visualization and online configuration of the electronic nose. A high level breakdown of the firmware is shown in fig. 4.13.

The following section will discuss the nature of the DSP task and how it makes use of the DMA hardware to achieve high sampling rates with very low CPU usage. The DSP task will communicate processed samples to the Sample Emit task using an IPC message queue provided by MQX. Sample Emit task is an intermediate task which provides samples to other parts of the application. For example, the vial heater temperature is communicated to the PID temperature control task using a similar message queue. An intermediate task is necessary because many actions may block, but the DSP task must not be blocked. The Sample Emit task also transmits samples to the computer as they are made available.
The MQX UART driver provides default handles stdin and stdout for communication with the computer. The TWR-K40X256 board connects the UART to the PEmicro debug chip such that all serial communication to the computer occurs over a virtual serial cable carried by the USB connection to the microcontroller.

A basic protocol stack is implemented using two tasks to facilitate guaranteed in-order delivery of bytes. A similar stack is implemented on the computer. A pair of MQX pipes are instantiated to function as “reliable stdin” and “reliable stdout.” Any bytes writted into these pipes (rstdin and rstdout) are transmitted reliably to the computer and vice-versa. This component of the firmware is discussed in section 4.3.3. Actual communication between the firmware and capture software is in the form of C-style message structs. The Control task parses the messages out of the byte stream provided by rstdin. It then carries out commands as specified in the message. Typically, this involves reconfiguring the electronic nose: enabling sensors, programming a new heater waveform, etc. The Control task may also generate messages in response to events which occur within the electronic nose.

4.3.1 Analog Sampling Engine

The parameters of the ADC module configuration were selected to maximize the expected number of bits. A 3 MHz clock was selected for the ADC modules. The conversion mode was set to 16-bit low speed operation and each conversion was performed with hardware averaging of 32 samples. This combination of clock speed and ADC options requires 260 µs per conversion\(^2\). ADC0 converts 11 channels per

\(^2\)This was determined using the calculator at http://www.freescale.com/webapp/sps/site/overview.jsp?code=ADC_CALCULATOR.
Figure 4.14: A dataflow diagram describing the DMA driven analog conversion system for the ADC0 module. This process is executed once every sample period.

...sample period, while ADC1 only converts 6 channels. Since the modules can operate concurrently, less than 4 ms are required to convert all 17 channels each sample period. Therefore, the maximum supported sample rate is about 250 Hz.

Using an interrupt driven ADC driver would require an interrupt to be handled for each conversion. A sample rate of 64 Hz would require 1088 interrupts to be handled each second. This is not feasible due to the overhead associated with handling an interrupt in MQX. Instead, the firmware is designed to offload as much of the analog conversion process to the DMA controller as possible. The DMA driven ADC driver architecture is described in fig. 4.14 and implemented in SensorCore.cpp.

The sample rate is determined by the PIT clock period. In the ideal configuration, DMA4 is periodically triggered by PIT-gated DMA request. However, due to Errata...
the PIT module must generate a software interrupt so the CPU can trigger DMA4 instead. At the start of each sample period, the PIT interrupt triggers DMA4 to transfer the first row from the command table to ADC0_SC1A. The command table contains $N$ rows of 32-bit register values for ADC0_SC1A which configure the desired channel and mode.

A conversion begins as soon as a write to ADC0_SC1A by DMA4 occurs, since the ADC module is in software trigger mode. The DMA controller is idle while the conversion takes place. The ADC module is configured to trigger DMA5 when the COCO flag is asserted at the end of a conversion. This causes the DMA controller to transfer the value from the ADC result register ADC0_RA to a row in the result table. When the DMA controller reads from ADC0_RA it also clears the COCO flag and deasserts the DMA request.

This completes the first conversion defined in the command table. However, $N - 1$ conversions still need to be performed. The minor loop ELINK function is used to create a loop wherein DMA5 triggers DMA4 at the conclusion of each minor loop. This causes the next command to be written to ADC0_SC1A, which restarts the conversion process. This repeats until the loop count CITER reaches zero. For this reason, the initial loop count BITER is equal to the number of elements $N$ in the command and result tables. Therefore, DMA4 and DMA5 are each triggered $N$ times per sample period. The source address for DMA4 is incremented by 4-bytes each time the channel is triggered. The destination address for DMA5 is incremented by 2-bytes each time the channel is triggered. When CITER reaches zero, DMA4 and

\footnote{Errata e4588: DMAMUX: When using PIT with "always enabled" request, DMA request does not deassert correctly.}
DMA5 addresses are reset to the start of the respective command and conversion tables.

DMA6 is triggered by major loop ELINK when the loop count CITER of DMA5 reaches zero. Thus, DMA6 is only triggered once per sample period. DMA6 copies the entire conversion table into the much larger conversion buffer. The size of the conversion buffer is $2^P N$ bytes where $P$ is always even. DMA6 has an initial loop count of $CITER = P$ and the destination address for DMA6 is incremented by $2 \times N$ bytes each time the channel is triggered. When CITER reaches zero the destination address is reset to the start of the conversion buffer. See fig. 4.15 for an example indicating the memory contents of the data structures when three analog channels are in use.

The conversion buffer is a split buffer which is accessed using a ping-pong approach. While the DMA controller is writing in the left half of the buffer the CPU may access the right half of the buffer, and vice-versa. The DMA controller is configured to generate an interrupt when the loop count for DMA6 reaches $\frac{P}{2}$ and zero.
A flag indicating which half of the conversion buffer is available is toggled each time the DMA6 interrupt is handled. A DSP task is also signaled to begin processing the conversion buffer.

The system described in this section applies to ADC0. Separate command and conversion tables are defined for ADC1 as well as a separate conversion buffer. DMA7, DMA8, and DMA9 are used instead of the DMA channels previously described. The software synchronizes the triggering and buffer access to sample all 17 channels across both ADC modules each period.

**DSP Task**

The DSP task sleeps until it is signaled by the DMA6 or DMA9 interrupt. Upon waking, it examines the conversion buffer flags to determine which data is ready to be processed. Each conversion buffer is large enough to hold 1 s of analog samples, that is $P$ is equal to the sample rate of 64 Hz. This means that the CPU has 0.5 s to process each half of the conversion buffer. The DSP task decimates each group of 16 samples in the conversion buffer to produce a final sample rate of $SR_D = 4$ Hz. The decimated samples are placed into a MQX message queue where a different task will eventually process them. The following code is an abridged version of the DSP task:

```c
/* TASK
  This is a DSP task which decimates the ADC buffers to produce the final decimated samples. The
  DMA interrupts set events which synchronize the buffer access. */

void SensorDspTask(uint_32 parameter) {
  /* Configure the ADC, DMA, PIT, and FTM peripherals and initialize MQX data structures. */
  HardwareSetup();
  // Begin task loop.
  while(true) {
    /* Wait for a half buffer to be ready. The flag bits indicate which halves of the sample
    buffers are ready. Left and right halves of the buffer for a module should never be
    ready at the same time. That would mean samples were dropped. */
```
4.3.2 Temperature Modulation

The code which produces the heater waveforms for the MOS sensor array is part of SensorCore.cpp. This is the part of the firmware application that supports the dynamic temperature mode of operation for the MOS sensors. Several daunting data structures are created in memory but the actual operation is quite straightforward. Only a single periodic interrupt is required to enable this function. The interrupt is enabled when the DSP task calls HardwareSetup(); all sensor heaters are initially configured to be off.
Each MOS sensor in the array contains an integrated heating element. The signals which control the heating element of each sensor in the array are assigned to one of eight possible heater channels. The sensors with programmable heater voltages are each assigned to an individual channel: HS01, HS06, HS07, and HS12. The remaining sensor heaters are placed two per channel: HS02S08, HS03S09, HS04S10, and HS05S11. These channels correspond to signals FTM[0-3] and SW[0-3] respectively on the schematic. Each channel has a control structure of type `HeaterConfig` in memory, which has the following format:

```c
/* Control struct for heater channels. */
typedef struct {
  /* True for FTM channels otherwise false for SW channels. */
  bool is_pwm_channel;
  /* The supply voltage for this channel. Must match the jumper selection on the PCB or the output
   * will be incorrect. */
  double v_supply;
  /* The maximum output for this channel. Software will not generate a voltage exceeding this
   * amount. */
  double v_limit;
  /* A copy of the current waveform configuration. */
  WaveformConfig waveform;
  /* A pointer to the waveform buffer and its length in samples. */
  uint16_t wave_size;
  uint16_t* wave_data;
  /* A calibration table to linearize the PWM voltage. Field linear_table is a 2D table of FTM.CV to
   * voltage. Disable interpolation by setting linear_size to zero. */
  uint16_t linear_size;
  uint16_t (*linear_table)[2];
} HeaterConfig;
```

The `wave_data` field is a pointer to 1D array of 16-bit unsigned integers which represent the samples of the heater voltage waveform for the channel. A periodic interrupt maintains a counter which indexes into `wave_data` using modular arithmetic to update the heater voltage each period of the interrupt. The entire code of the interrupt service routine is as follows:

```c
/* KERNEL ISR: PIT1
 * This ISR updates the FlexTimer channel value registers. */
void PIT1ServiceRoutine(void* state) {
  static uint32_t t = 0;
  // Update the PWM heater channels.
  FTM0_C3V = heater_s01.wave_data[t % heater_s01.wave_size];
```

57
Programmable channels use the entire 16-bit sample to update the associated FlexTimer channel value register. Fixed channels use only the first bit of each sample to set the corresponding GPIO output. A constant output can be configured by setting `wave_size` to one, and pointing `wave_data` to an array with one sample. Otherwise the period of the output is only limited by available memory to store the waveform samples. The sample rate of the heater voltage waveform is defined by `PWM_SAMPLE_RATE` constant in `SensorCore.h` and is currently 64 Hz.

The `ConfigureHeater()` function generates a new `wave_data` buffer and updates the `HeaterConfig` struct for the provided channel. A `WaveformConfig` struct is used to define the features of the desired waveform using common properties and a predefined enumeration of wave shapes such as constant, sin, square, etc. The amplitude and offset are both relative to the `v_supply` field in the `HeaterConfig` struct. Each sample is clamped to `v_limit` to prevent accidental damage to the sensors. Fixed heater channels may only be configured to output constant or square waveforms having samples of zero or one. `ConfigureHeater()` will reject any attempt to configure an invalid output such as a sinusoid on a fixed heater channel.
typedef struct {
    Waveforms shape; // The type of wave to generate: constant, sin, square, etc.
    double frequency; // The frequency of the wave in Hz.
    double amplitude; // The amplitude of the wave: 0 to 1.
    double offset; // The zero offset of the wave: 0 to 1.
    double duty_cycle; // The duty cycle of the wave if applicable.
    double phase; // The phase of the wave if applicable.
} WaveformConfig;

The voltages generated for the programmable channels are not perfectly linear, particularly at outputs near 0V. It is possible to construct a calibration table by measuring the actual output voltage for a number of different FlexTimer channel values. These tables are stored in the linear_table field of the HeaterConfig struct as a 2D array of channel value and output voltage data points. ConfigureHeater() will then use this table to interpolate the corrected sample values when generating a waveform.

4.3.3 Communication

The capture software and firmware communicate by exchanging message structs over a serial link. A transport layer implemented in the firmware guarantees ordered and error free delivery of the messages. The transport layer is designed to work around a number of shortcomings present in the built-in serial communication between the microcontroller and host computer.

Built-in Serial Communication

Communication between the microcontroller and a PC is not a straightforward connection of serial or USB transceivers. The TWR-K40X256 evaluation board includes
an OSBDM\textsuperscript{4} controller. This controller allows a computer to program and debug the Kinetis microcontroller over a USB connection. The debug controller also provides a serial interface which transmits data over the debug connection to facilitate serial communication between the microcontroller and a host computer. On the TWR-K40X256, the serial UART on the microcontroller is connected to the debug controller. On the PC, the OSBDM device driver installs a USB communications device class (CDC) virtual serial port on the host PC to support this function. The TWR-K40X256 board support package (BSP) creates default handles \texttt{stdin} and \texttt{stdout} for the UART associated with the OSBDM serial link. An overview of the debug and serial interface is shown in fig. 4.16.

On the host computer, an application obtains a handle to the virtual serial port which allows it to read from and write bytes to the microcontroller. The host com-

\textsuperscript{4}Background Debug Mode is an interface that provides in-circuit debug functionality for embedded systems. PEMicro uses an open source implementation known as OSBDM.
Kinetis K40 Microcontroller

C++ Developer Computer

Virtual Serial Cable
Serial UART
OSBDM CDC Serial Port
Serial-to-TCP Relay

TCP/IP
Capture Software

C# Developer Computer

Figure 4.17: Serial-to-TCP Relay is a simple TCP listen server which pipes any input and output from the virtual serial port to the first TCP client which is connected.

puter was a workstation configured for embedded C++ development. However, the capture software was written on another machine configured for C# development. A serial-to-TCP relay was installed on the embedded C++ machine to provide network access to the OSBDM CDC serial port. This setup allows the electronic nose to remain connected to the embedded C++ developer with all debugging facilities intact at all times. The capture software may be installed on the C++ machine (in which case it is connected to localhost) or it can run on the C# machine using an Ethernet connection to communicate with the electronic nose. The diagram in fig. 4.17 demonstrates this configuration. This means that the firmware communicates using MQX file handles stdin and stdout while the capture software communicates using a .NET NetworkStream instance.

The connection between the microcontroller and host PC is not always reliable and bytes are frequently dropped from the connection. These losses are independent of the UART queue size, baud rate or driver mode (polled/interrupt) as configured on the microcontroller. A transport layer has been implemented to guarantee in-order
and error-free transmission of bytes between the microcontroller and PC.

**Transport**

Reliable, ordered, and error-checked transmission of binary data is provided by the transport layer implemented in `SerialCore.cpp` of the firmware. The transport layer installs two MQX pipe drivers called `rstdin` and `rstdout` which wrap the standard input and output file handles. Two MQX tasks are created; one task manages the transmit and the other the receive. A C# class library has been written to provide an implementation of the transport layer for PC. The implementation on the computer is nearly identical. It uses analogous structures such as threads instead of MQX tasks and streams instead of MQX pipes, for example.

The transport layer is a basic implementation of the stop-and-wait repeat-request (ARQ) error-control algorithm. Under stop-and-wait ARQ, the sender transmits one packet at a time. After receiving a valid packet, the receiver transmits an acknowledgment. If the sender does not receive an acknowledgment before a timeout has elapsed, the sender transmits the packet again. To prevent the receiver from processing duplicate packets (perhaps the acknowledgment was lost or delayed) each packet is assigned a sequence number. A 1-bit sequence number is sufficient to identify the packet order under stop-and-wait ARQ. An example communication sequence is displayed in fig. 4.18.

The packets generated by the transport layer employ a combination of ASCII and Base64 encoded fields to avoid transmitting binary data directly over the serial connection. The packet format is indicated in fig. 4.19. The packet preamble is used
Figure 4.18: Traffic from the microcontroller to the computer. Data written to \texttt{rstdout} is divided into packets which are reassembled by the computer.

to synchronize the data stream (particularly when characters are dropped.) When a packet preamble is detected, the data stream is buffered until the packet postamble is encountered. If a postamble is not encountered before the packet buffer is full, the buffer is discarded. The preamble, postamble, and flag fields are non-printable ASCII control characters\textsuperscript{5}. A data packet contains a NULL character in the flag field and the packet sequence (ASCII ‘0’ or ‘1’) in the sequence field. An acknowledgment packet contains an ACK character in the flag field and the sequence of the acknowledged packet in the sequence field. The checksum and data fields may be omitted when transmitting an ACK packet. Otherwise, the checksum is calculated as the sum of the bytes in the data field before Base64 encoding. In the future, the CRC module in the Kinetis CPU might be utilized to generate a more robust checksum.

\textsuperscript{5}See the ASCII table at http://en.wikipedia.org/wiki/ASCII\#ASCII_control_code_chart for more information.
The transport layer ensures that bytes written into the transport stream are received in order and error-free. The message protocol which the microcontroller and capture software use to communicate exists on top of the transport layer.

**Messaging**

Within the firmware, the Control task in `ControlCore.cpp` is responsible for responding to messages and performing the requested operations. The messages exchanged by the microcontroller and PC are specially crafted C/C++ style POD\(^6\) structs. The structs are crafted in the sense that the packing of members in memory is relevant and occasionally specified manually.

The format of each type of message struct is defined in `ControlCore.h`. The transmitter outputs a byte which indicates to the receiver the type of the message, followed by the message struct. The receiver knows the length and format of the message struct based on the first byte and reads the appropriate number of bytes into memory. The following listing demonstrates how a message struct is defined in C++ and C#, as well as how the message struct is serialized for transmit.

The Kinetis microcontroller requires double word alignment for floating point data types and will include padding in the struct layout to ensure correct alignment.

---

\(^6\)Plain Old Data: A data structure that is represented only as passive collections of field values, without using object-oriented features.
/* Identification bytes for each type of control packet. */
const static uint8_t MSG_SAMPLE = 0x0;
const static uint8_t MSG_HEATER_CONFIG = 0x5;
/* ... */

// Message used to configure the temperature modulation waveform.
typedef struct {
  HeaterChannels channel;
  struct WaveformConfig {
    Waveforms shape;
    double frequency;
    double amplitude;
    double offset;
    double duty_cycle;
    double phase;
  } waveform;
} MsgHeaterConfig;

// Send a message to the computer.
void WriteMessage(uint8_t packet_id, void* data, int length) {
  _lwsem_wait(&emit_sync);
  _io_write(rstdout, &packet_id, 1);
  _io_write(rstdout, data, length);
  _lwsem_post(&emit_sync);
}

// Create and transmit a message.
MsgHeaterConfig msg;
msg.channel = HeaterChannels::HS03S08;
msg.waveform.shape = /* ... */
WriteMessage(MSG_HEATER_CONFIG, &msg, sizeof(MsgHeaterConfig));

If this padding is removed (i.e. to make the struct smaller), the microcontroller incurs a penalty as the fields must be copied to an aligned memory address before access.

Since C# is a memory managed language, attributes must be used to manually specify the layout of the structure in unmanaged memory. Padding is explicitly added to the C# version of the message struct to keep the unmanaged layout consistent with the microcontroller.
Serial-to-TCP Relay

Serial Relay is a very simple C# application. At startup, it will attempt to open a handle to the OSBDM CDC virtual serial port. Once the serial port handle has been successfully obtained, it will start a TCP listen server and wait for a client to connect. After a client has connected, the program reads bytes from the serial stream and writes those bytes to the TCP stream and vice versa. Data from each stream is also printed to the application console using ASCII encoding, an example of which is displayed in fig. 4.20. This output is also useful when debugging the sequence and format of communication between the microcontroller and the computer. Lines which begin with “MCU -” indicate data which originates on the microcontroller, and “PC -” data from the capture software. The strange symbols are substituted for non-printable ASCII characters.

![Image](file:///C:/Workspace/Electronic%20Nose%20for%20Spirits/SerialRelay/SerialRelay/bin/Debug/SerialRelay.EXE)

Figure 4.20: A screenshot showing a number of base64 encoded packets. The MCU sends a packet and the computer responds with an acknowledgment.
4.4 Capture Software

The capture software is written in C#/WPF using Visual Studio 2012. The solution file contains two projects: a class library, and an application. The class library project provides the object model which represents the current state and capability of the electronic nose. It also implements the communications layer which interfaces with the firmware on electronic nose. The application project depends on the class library and provides a user interface which allows the operator to monitor and control the electronic nose.

4.4.1 Class Library

The class library provides an object model which represents the current state of the MOS sensor array and the firmware. The most important classes in the library are presented in fig. 4.21. The \texttt{McuSystem} class is implemented using a singleton design pattern since it represents the electronic nose system. It implements methods to configure the electronic nose and events to notify consumers of changes to the electronic nose. \texttt{McuSystem} is partially composed of a collection of \texttt{SensorBase} instances, which represent the sensors installed in the sensor array.

Each type of sensor circuit on the electronic nose is represented by a class derived from \texttt{SensorBase}. These classes provide an object model which represents the current state and capability of the sensor circuit. However, the only sensors with any significant capability are the MOS sensor circuits. The \texttt{MuxSensor} class represents a MOS sensor circuit and has methods to configure the selected resistor
Figure 4.21: A class diagram of the most important classes in the class library project.
on the multiplexer. Some properties of SensorBase like IsMuxAmpOn are read-only. Any property which may result in side-effects to another sensor must be changed from McuSystem using a function like ChangeMuxEnable(). These side-effects occur because some sensor circuits share GPIO lines like ENn and SWn. Some instances of SensorBase are associated with a CellInfo class which provide calibration and conversion methods which are specific to an individual sensor cell.

The CellInfo class is designed to contain cell-specific information for each sensor technology. For example, the MosCellInfo derived class is programmed with the response curves for ethanol. The Make() method of MosCellInfo is used to derived a calibrated representation (using the measured $\frac{R_s}{R_o}$ value for the cell) from the prototype MosCellInfo instances. A number of prototype MosCellInfo instances are available as static members of the MosCellInfo class. The NtcCellInfo class works similarly, but contains the Steinhart-Hart coefficients (or Beta value) of the NTC thermistor it represents. The CellInfo classes are instrumental in performing conversions from ADC voltage to unit output such as gas concentration or degrees Celsius.

Lastly, the ReliableClient and MessageClient classes implement the transport layer and messaging protocol respectively. The implementation of each is discussed in section 4.3.3; however, equivalent C# structures and objects are used instead of C++ structures and objects.
4.4.2 Application

The application project is written in C# and uses Windows Presentation Foundation (WPF) to provide a rich user interface. The windows, dialogs and controls are designed using XAML with behavior provided by C# in code behind. Binding properties are used extensively to push updates to the user interface.

High speed charting is provided by the System.Windows.Forms.DataVisualization.Charting namespace. However, this is a WinForms control library, so the charts must be hosted in WPF by a WindowsFormsHost control. WinForms controls cannot be edited in the design view of a WPF project, so a small workaround is employed. The charts are actually part of the class library project, which is a C#/WinForms project despite having no user interface. This allows the charts to be edited in the design view. The charts can then be referenced and instantiated from the application project.

The capture application interface consists of a single window with three tabs. The System tab (screenshot in fig. 4.22a) provides a detailed view of the configuration of the MOS sensor array using a list view format. The cell type is populated from the CellInfo instance associated with the sensor. The enabled/disabled state and selected resistor for each multiplexer is indicated. The heater state and a description of the temperature profile is also provided. The last three columns present the latest sample in 16-bit unsigned integer, voltage, and sensor units. The configuration of

---

7A native charting control is available as part of the WPF Toolkit. However, the WPF Toolkit charting control suffers from a known memory leak which occurs when the chart is updated. Due to the large number of data points being plotted, the application quickly becomes unresponsive. Other third-party charting controls are available, but they are mostly paid software.
each sensor circuit can be modified by right-clicking on the appropriate row and selecting an option from the context menu. Heater temperature profiles are designed by selecting Configure Heater from the context menu. This will present a Wave Designer dialog (in fig. 4.22b) which may be used to design a temperature profile. The dialog is aware of the capability of each sensor and will enable or disable controls to reflect that capability.

The Capture tab (screenshot in fig. 4.23a) charts the sensor array response in a customizable plot. The axis may be configured by right-clicking on the chart. Each series on the plot can be hidden by unchecking the checkbox in the list view on the right. The list view also provides a summary of the information from the System tab using background colors and hover tooltips. Some sensors will also provide an estimate of the equivalent concentration of ethanol using the curves in the MosCellInfo instance.

Additional commands are available through the Capture menu in the menu bar. The recorded information can be cleared using Reset Capture. The recorded data can be exported to a file using Export Capture. At this moment, Excel Spreadsheet using Microsoft Office Interop is the only supported method. However, the export system is modular and additional exporters may be added easily. Exporting directly to a Matlab .mat file is a priority for a future work. Lastly, the Automation menu can automate a number of tasks using scripts included at compile time. Automation is mainly used in two ways: the first is to play alert tones when an operator action such as closing a valve is required, and the second is to configure different temperature profiles at predefined intervals.
Figure 4.22: The system panel in (a) provides detailed information about the MOS sensor configuration. Right-click on a MOS sensor and select “Configure Heater” to access the wave designer dialog in (b) and edit the MOS sensor temperature profile.
Figure 4.23: The capture panel in (a) charts the MOS sensor response. The Export dialog in (b) saves the recorded data in a number of formats.
The Process panel (screenshot in fig. 4.24) is used to monitor and configure control loops. At the moment, only the vial heater temperature is controlled. The vial heater control is implemented using a velocity PID algorithm whose parameters may be configured using controls on the right. The PID parameters are stored in non-volatile flash memory on the microcontroller, so the algorithm does not need to be tuned each time the electronic nose is powered on. The process value, control action, and reference are updated at 1 Hz intervals for the vial heater.

![Figure 4.24: The process panel is used to monitor and configure the control loop for the vial heater temperature.](image)
4.5 Summary

In this chapter, the design of an electronic nose prototype was presented. The prototype electronic nose includes an array of 12 MOS gas sensors. A subset of the sensor array may be operated with a dynamic temperature mode. The temperature of the sensor is modulated by varying the integrated heater voltage using a programmable DC-DC voltage converter circuit based on PWM. Each sensor circuit also includes a bank of multiplexed resistors such that an optimal load resistor may be selected for any type of cell which may be installed in the circuit. The sensor board also integrates a pair of temperature and humidity sensor modules.

A firmware application was developed for the TWR-K40X256 microcontroller board which operates the electronic nose. This board includes a Kinetis K40-series CPU which integrates a number of useful hardware modules such as a DMA controller, two 16-bit ADC modules, and several FlexTimer modules. A high-speed DMA driven analog sampling engine was implemented using the hardware modules of the microcontroller to achieve a sample rate of 64 Hz over 17 analog channels. The DSP component of the application decimates the samples to final sample rate of 4 Hz to reduce noise. Support for programmable heater temperature profiles on the MOS sensors was implemented using a single fast periodic interrupt.

A transport layer was written to allow the microcontroller to reliably communicate with a custom C# application running on a computer. This application provides a user-friendly interface to: (1) view and configure the status of the electronic nose, (2) record and visualize in real-time the data generated by the electronic nose, and (3) manage controlled processes such as the vial heater temperature.
Chapter 5

Results

This chapter considers the experimental methods and results used to verify the operation of the electronic nose and the validity of the method of analysis. The applicability of the Hilbert-Huang transform to the recorded dataset is discussed, and initial steps in classification are presented.

5.1 Experimental Setup

To ensure reliable operation, the electronic nose must be allowed to warm up before samples are taken. MOS sensors which have not been energized for a period of several hours will exhibit an effect known as initial action, during which they will have a lower than expected resistance for up to 10 minutes after being re-energized. Also, the temperature and relative humidity will often change rapidly during the first 10 to 20 minutes of operation due to heat produced by the MOS sensors. During warm up and between samples, the MOS sensors operate in a static temperature mode so
that the response may be observed to reach a steady state. This warm-up phase also provides time for the vial-heater to reach its temperature setpoint.

During warm up and between samples, the electronic nose was flushed with clean air for 30 minutes. Due to the small volume, cool air introduced by the flushing system will affect the temperature inside the electronic nose. Therefore, a gentle airflow was preferred, since it perturbs the array less at the expense of increased flushing time. The MOS sensor response was also monitored to determine if the previous sample has been sufficiently purged. The analyte was typically undetectable after 10 minutes of flushing. Temperature and humidity sensors were also monitored to ensure identical starting conditions for each experimental trial.

A microliter syringe was used to deposit 1 µL of the analyte in a 3 mL target vial. Working quickly, the vial was inserted into the vial-heater. The vial-heater was warmed to 60°C and so the samples rapidly evaporated. After five minutes, the valve separating the vial-heater and sample chamber was released. The evaporate was allowed to diffuse evenly into the sample chamber for another five minutes. Finally, the microcontroller began to modulate the sensor temperature such that data recording could proceed.

When operating in dynamic temperature mode, the microcontroller has enough free memory (32 kB) to store 240 s of modulation signal data. With four variable voltage sensor circuits operating, the longest supported period is 60 s per sensor. Gutierrez-Osuna et al. [36] indicate that a slow changing temperature is desirable as it provides more discriminatory features. Although a longer period will increase the overall sample time, 60 s or 16.6 mHz modulation was a natural choice for initial work
with the electronic nose. A sinusoidal modulation profile was selected to generate a gradual temperature change throughout the modulation cycle. The TGS-series sensors are specified to operate at 5 V and so they are programmed to cycle between 2 V and 5 V. The SB-series sensors are specified to operate at 0.9 V, however they were programmed to cycle between 0.2 V and 0.8 V. This was a compromise to guarantee that the response during the entire cycle would remain within a measurable range (see table 4.1.)

This electronic nose is intended for use in the beverage industry, where the properties of the liquid sample are of interest. For this reason, all samples were prepared using a constant liquid volume. However, it is possible to determine the vapour concentration of pure samples using

\[
V_{vapor} = \frac{\rho_{liquid} V_{liquid} RT}{M_{liquid} P},
\]

\[
c_{ppm} = \frac{V_{vapor}}{V_{chamber}} 10^6,
\]

where \(\rho_{liquid}, V_{liquid}\) and \(M_{liquid}\) are the density, volume and molar mass of the liquid sample respectively. The volume of vapor produced by evaporating the liquid is \(V_{vapor}\) and the volume of the electronic nose is \(V_{chamber}\). Parameter \(R\) is the ideal gas constant, while parameters \(T\) and \(P\) are the ambient temperature and pressure. Since the volume of the electronic nose is approximately 2.2 L, the expected concentrations of ethyl acetate, ethanol and isopropanol are 120 ppm, 200 ppm, and 150 ppm, respectively.
Figure 5.1: The array response (above) to the changing heater voltage (below) in the presence of 1 µL ethyl acetate.

5.2 Recorded Data

During each trial, ten minutes of data was collected. The firmware sampled the resistance of each sensor in the array at 64 Hz and decimated the data to 4 Hz. The capture software recorded the samples from the electronic nose and exported the information to a file. The capture file contained four signals (one per active MOS sensor) with 2400 samples per signal. Each capture file was then imported into Matlab, where five minutes of continuous signal was selected for analysis. The selected segment of the recorded signal represents five periods of the temperature modulation signal. In order to visualize the data more intuitively, the each sensor’s resistance has been converted to conductance. In this way, the peak conductance corresponds to the strongest response. (As opposed to the lowest resistance corresponding to the
Each of the following compounds were sampled five times for a total of 15 trials: ethyl acetate, ethanol, and isopropanol.

An example (taken from the recorded data) of the temperature modulated response and the heater modulation signal are plotted together in fig. 5.1. Comparing the responses in fig. 5.2, observe that the magnitude of the peak as well as the surrounding features are different between samples. Generally speaking, the response exhibits a peak, a secondary peak or plateau, and a valley. The peak and valley coinciding with the maximum and minimum temperature, respectively. This indicates that the features of the response encode additional information about the analyte. These features are produced when oxidized species of the analyte preferentially react to the sensor due to the temperature dependency [37].
Figure 5.2: The array response to 1 µL of ethyl acetate (a), ethanol (b), and isopropanol (c) with temperature modulation at 16.6 mHz.
5.3 Hilbert-Huang Analysis

The ensemble empirical mode decomposition of the response for sensor TGS-2610-C is plotted in figure fig. 5.3. Three samples are plotted on the same set of axis, one sample for each compound in fig. 5.2. Similar decompositions are plotted for sensor TGS-2620-C in fig. 5.4, sensor SB-15-00 in fig. 5.5, and sensor SB-AQ1-06 in fig. 5.6. Note that the scale of adjacent IMFs (both within and between figures) may not be similar. The parameters for the EEMD algorithm were determined experimentally: 0.2 standard deviations of white noise amplitude, ensemble count of 2000 iterations, and up to 7 IMFs may be extracted using exactly 20 sifting operations per IMF.

The majority of white noise is extracted from the signal within the first two IMFs. The white noise is both noise from the original signal and components of the ensemble mean which were not perfectly canceled by averaging. The noise signals are the first to be extracted since they have the shortest scale. The third IMF is typically transitional, containing both signal and noise components. This is evident in figs. 5.3 and 5.4 in which IMF 3 is quite noisy but has periodic elements corresponding to the following IMF.

Note that the components of the EMD are often physically meaningful as the characteristic scales are defined by the physical data [10]. The seventh IMF is an excellent example of this property; this IMF is strongly correlated with the temperature modulation signal in fig. 5.1. Therefore, it can be inferred that the differences in the remaining IMFs must encode some information about the analyte. For example, the fourth IMF typically contains a characteristic shape which occurs during the nadir just before the peak conductance.
In fig. 5.4 IMF 6, the signal for ethanol has an increasing amplitude over the course of the sampling period. Meanwhile in IMF 7, the amplitude is being reduced. Components of the signal appear to have been mixed throughout IMF 6 and 7. However, adjacent IMFs are expected to contain related information and the heuristic nature of EMD does not guarantee perfect separation in all cases. This anomaly does not significantly affect the remaining steps in the analysis.

The Hilbert-Huang spectrum is constructed for each sample and sensor in fig. 5.7. The magnitude and instantaneous frequency of each IMF is determined using the equations described in section 2.3.1. The Hilbert-Huang spectrum is then constructed by plotting all the magnitudes and instantaneous frequencies on the same axis. IMF 1 and 2 are omitted when generating the spectral plots since they contain no useful instantaneous frequency data (because they are primarily noise.)

In each spectral plot, a constant 16.6 mHz frequency component is observed. This is an expected consequence of the thermistor effect and the temperature modulation signal. Each plot also has a component which is reminiscent of a rectified sinusoid between 0 mHz and 20 mHz. This component – similar in cause but distinct from the thermistor effect – appears to be derived from the change in sensitivity as the sensor temperature varies. (i.e.: The adsorption chemistry is altered as the temperature varies leading to changes in conductivity.) Of course, this component will also contain some information about the analyte. Frequency components between 20 mHz and 60 mHz appear to be the most telling. These frequency components have unique and easily identifiable shapes. In most cases, the weaker components above 60 mHz are also useful. Interestingly, sensor SB-AQ1-06 – the only sensor which does not target
LP or solvent gas – produced the most unique and difficult to interpret Hilbert-Huang spectrum. Nevertheless, each spectral plot contains uniquely shaped instantaneous frequency components.

A summary of each spectrum may be determined by integrating each frequency component over time to produce the marginal Hilbert spectrum. The marginal Hilbert spectrums for each plot in fig. 5.7 have been plotted in red in fig. 5.8. The remaining four samples from each compound are plotted in shades of blue for comparison. During the determination of the marginal Hilbert spectrum, the instantaneous frequencies in the Hilbert-Huang spectrum are quantized and binned before integration. Integrating over time has reduced the dimensionality of the data considerably. However, the instantaneous frequency components which were previously trivial to identify are now somewhat more difficult to discern.

At this point, some variation in the spectrum can be observed between samples. The TGS-series sensors appear to be more consistent in terms of repeatability between trials. The SB-sensors are less consistent but features of the spectrum (number of peaks, valleys, etc.) indicate they may capture more information than the TGS-series sensors. The response of SB-AQ1-06 to isopropanol is particularly inconsistent. Possible sources of error are discussed in section 5.5. It is also worth noting that many discriminatory details are a magnitude of order smaller than the peak. Thus, they are somewhat obscured due to the scale of the axis.

Further processing may be applied to the marginal Hilbert spectrum, or the spectrum itself may be used as an input to a classifier. The following section will investigate classification based on the data from the marginal Hilbert spectrums.
Figure 5.3: The ensemble empirical mode decomposition of the responses from sensor TGS-2610-C in fig. 5.2.

Figure 5.4: The ensemble empirical mode decomposition of the responses from sensor TGS-2620-C in fig. 5.2.
Figure 5.5: The ensemble empirical mode decomposition of the responses from sensor SB-15-00 in fig. 5.2.

Figure 5.6: The ensemble empirical mode decomposition of the responses from sensor SB-AQ1-06 in fig. 5.2.
Figure 5.7: The Hilbert Huang spectrum derived from each decomposition in figs. 5.3 to 5.6.
Figure 5.8: The marginal Hilbert spectrum computed from fig. 5.7 in red along with four additional trials in blue.
5.4 Clustering and Classification

Information from each sensor in the array must now be integrated to facilitate classification of the samples. An odor signature vector is assembled from sections of the marginal Hilbert spectrum taken from each sensor. Each section consists of the magnitude of the spectrum sampled every 1 mHz between 8 mHz and 80 mHz. Each 72-element section is normalized and the four sections are concatenated to form a 288-element odor signature vector.

Uniformly sampling the marginal Hilbert spectrum is straightforward but not necessarily optimal. Frequency ranges which change very little between samples could be eliminated from the vector. Conversely, frequency ranges which change very much could be sampled more densely. The ultimate goal being to improve the performance of the classifier by distilling relevant information.

Odor signature vectors for ethyl acetate, ethanol and isopropanol are plotted in fig. 5.9a. Clustering of the samples based on the signature vector is assessed using principal component analysis. PCA scores for each of the 15 samples available have been plotted in fig. 5.9b. Ethyl acetate and ethanol cluster well; however there appears to be some overlap between ethanol and isopropanol.

The majority of the variance in the signature vector is due to components which are derived from IMFs 6 and 7. The subtle and highly discriminatory features of IMFs 3 to 5 are a magnitude of order smaller than IMFs 6 and 7. The clustering performance may be improved by omitting IMFs 6 and 7 from the Hilbert spectrum (and marginal Hilbert spectrum.) New odor signature vectors for ethyl acetate, ethanol and isopropanol are plotted in fig. 5.10a.
Figure 5.9: Clustering of samples based on odor signatures derived from the marginal Hilbert spectrums in fig. 5.8.

Figure 5.10: Clustering of samples based on odor signatures derived the marginal Hilbert spectrums which omit IMFs 6 and 7.
PCA scores derived from the new odor signature vectors for each of the 15 samples available have been plotted in fig. 5.10b. Although the tightness of the clusters has been increased, it is the separation between clusters that has seen the most dramatic improvement.

The self-organizing map is an alternative clustering technique which maps an input vector onto a 2D plane of neurons. Clusters are determined by the regions (neighborhoods) of adjacent neurons which are activated by the input vectors. A hit map plots the number of input vectors which are classified by a given neuron. The `selforgmap` function in Matlab was used to generate the SOM. In fig. 5.11, both the original (all IMFs) and new (IMFs 3, 4 and 5) odor signature vectors are clustered within a plane of 64 neurons. In the case of fig. 5.11a, there are two distinct hit clusters in the top left and right. The remaining hits are spread out much like in fig. 5.9b. In fig. 5.11b, the clusters are well defined much like those in fig. 5.10b.

It is difficult to generalize the performance of a pattern classifier neural net due to the limited number of samples. The performance of the classifier will depend almost entirely on the initial conditions and the samples chosen for testing and validation. Instead, the performance will be determined by the average error of 100 different neural nets. The `patternnet` function in Matlab was used to create the classifier neural net. The neural net has 288 inputs, 10 hidden nodes and 3 outputs. The output nodes correspond to one compound: ethyl acetate (1,0,0), ethanol (0,1,0), and isopropanol (0,0,1). Each neural net is trained using 11 training, 2 validation, and 2 testing samples which are randomly selected at the start of each training. The average error at the output is indicated in table 5.1 and table 5.2.
Figure 5.11: Self-organizing map hit plot with each neuron showing the number of input vectors that it classifies.

Table 5.1: Pattern classifier average error at the output for 100 neural nets trained with signature vectors such as those in fig. 5.9a.

<table>
<thead>
<tr>
<th></th>
<th>Ethyl Acetate</th>
<th>Ethanol</th>
<th>Isopropanol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output 1</td>
<td>-0.0393</td>
<td>-0.0620</td>
<td>0.0899</td>
</tr>
<tr>
<td>Output 2</td>
<td>-0.0160</td>
<td>0.0481</td>
<td>-0.0214</td>
</tr>
<tr>
<td>Output 3</td>
<td>0.0762</td>
<td>-0.0616</td>
<td>-0.0772</td>
</tr>
</tbody>
</table>

Table 5.2: Pattern classifier average error at the output for 100 neural nets trained with signature vectors such as those in fig. 5.10a.

<table>
<thead>
<tr>
<th></th>
<th>Ethyl Acetate</th>
<th>Ethanol</th>
<th>Isopropanol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output 1</td>
<td>-0.0269</td>
<td>-0.0144</td>
<td>0.0381</td>
</tr>
<tr>
<td>Output 2</td>
<td>-0.0357</td>
<td>0.0190</td>
<td>-0.0282</td>
</tr>
<tr>
<td>Output 3</td>
<td>0.0548</td>
<td>-0.0168</td>
<td>-0.0328</td>
</tr>
</tbody>
</table>
The difference between table 5.1 and table 5.2 is small. Depending on the initial conditions, the majority of networks trained are able to classify every sample correctly. Unlike previous examples, removing IMF 6 and 7 from the spectrum has a marginal impact. By its nature, the supervised learning algorithm quickly learns to de-emphasize the larger low frequency components in favor of the more discriminatory high frequency components.

These classifications serve to demonstrate that the electronic nose hardware and the analysis methods are sound. In the future, both the hardware and the analysis must be refined to improve the consistency of the data and by extension the information which may be extracted from the data. The following section will discuss sources of error which must be addressed in future revisions of the electronic nose.

5.5 Sources of Error

The marginal Hilbert spectrums in fig. 5.8 are consistent enough to classify samples containing a single compound. However, in a real application the sample will consist of many compounds, as many as several hundred. The response will be much more subtle, and so the consistency of the recorded data must be improved considerably. Much of the error in these samples is believed to be a result of a failure to adequately control variables such as temperature and relative humidity.

Compressed air was used to flush the electronic nose since sufficient volumes of zero gas could not be obtained. Compressed air contains contaminants such as compressor oil in addition to any pollutants naturally present in the air. Nor are
the temperature and humidity of the compressed air regulated in any way. Although every effort was made to restore the electronic nose to the same initial conditions between each sample, these factors were not as tightly controlled as was desirable.

A second and perhaps equally significant source of error was the preparation of samples. Preparing microliter samples of volatile compounds is difficult since a significant portion of the sample can be lost to evaporation almost immediately. This is particularly challenging with volatile compounds such as isopropanol.

Finally, there are also a number of issues with MOS sensor technology. MOS sensors exhibit drift over periods on the order of several weeks. Some sensors also exhibit changes or reduced life when operated in a dynamic temperature mode, although this is usually the result of exceeding the manufacturers specification for the heater temperature. High concentrations of solvent vapor may damage or alter the reversibility of the sensor cell. Although in this case, the sensors were not subjected to concentrations greater than those specified in the manufacturers data sheet.

5.6 Summary

The prototype electronic nose was used to record responses to three solvent vapors which may be present in whiskey: ethyl acetate, ethanol, and isopropanol. Fifteen 1 µL samples (five of each compound) were analyzed using the Hilbert-Huang transform with ensemble empirical mode decomposition. Five minutes of recorded data was analyzed from each sample, corresponding to five periods of the sinusoidal temperature modulation signal. Each compound was decomposed into a set of seven
IMFs. The features of the decomposition and the resulting Hilbert spectrums were discussed; the shorter scale and intermittent features of decomposition were determined to be highly discriminatory. Some IMFs were attributed to a physical basis in the response. Odor signature vectors were created by concatenating sections of the marginal Hilbert spectrum from each sensor in the array. The samples were successfully clustered using principal component analysis as well as self-organizing maps. Clustering performance was improved by isolating components of the decomposition with shorter scale from the Hilbert spectrums. Samples were also successfully classified by multi-layer perception. Finally, sources of error and the impact of that error on the analysis was discussed. The Hilbert-Huang transform with EEMD in concert with temperature modulated MOS sensors has demonstrated the potential to improve the classification of analytes in the electronic nose.
Chapter 6

Conclusion

An electronic nose was designed which combines temperature modulation of an array of MOS sensors with the Hilbert-Huang transform for feature extraction. This was done in an effort to overcome the problems inherent in sensing with MOS sensors in high concentrations of solvent vapors such as ethanol. A number of previous studies have determined that temperature modulation is an effective way to increase the sensitivity and selectivity of MOS gas sensors. However, analysis of the temperature modulated sensor response is very difficult due to the highly non-linear and non-stationary signal which is produced. The Hilbert-Huang transform – itself a combination of EMD and Hilbert spectral analysis – was introduced as a new and relatively unexplored method for decomposing the response. The transform was modified to use an ensemble approach which defeated the problem of mode mixing in the decomposition of the particular responses produced by the electronic nose.

The sensor array in the prototype electronic nose consists of a 12-element array
of commercial MOS gas sensors manufactured by Figaro Engineering Inc. and FiS Inc. The selected sensors were all based on thick film doped tin-dioxide sensing layers and targeted compounds such as LP gas and volatile organic compounds. To survive exposure to whiskey and other spirits, each sensor was specified to operate in concentrations of at least 5000 ppm ethanol. Each sensor circuit also includes a bank of multiplexed resistors such that an optimal load resistor may be selected for any type of MOS cell which may be installed in the circuit. Four of the sensor circuits in the sensor array support temperature modulation using programmable DC-DC voltage converters based on PWM. The sensor array is operated by a TWR-K40X256 microcontroller board which includes a Kinetis K40-series CPU. This processor incorporates a number of useful components such as a pair of 16-bit ADC modules, a DMA controller and FlexTimer modules. The microcontroller samples the sensor array at 64 Hz which is then decimated to 4 Hz in software.

An experiment was designed to validate the operation of the electronic nose and the feasibility of the Hilbert-Huang transform in this application. The compounds ethyl acetate, ethanol and isopropanol were selected for analysis due to their frequent appearance in the alcohol industry. For example, all three compounds may be present in whiskey. Fifteen 1µL samples (five of each compound) were subjected to analysis by the electronic nose using sinusoidal temperature modulation. The modulated response of each sensor was rich in features which uniquely identify each type of sample. The Hilbert-Huang transform was performed and odor signature vectors were created by combining selected portions of the marginal Hilbert spectrums. The Hilbert-Huang transform with EEMD successfully extracted components of the
response having minuscule amplitude, intermittent features and short scale. These components were extremely useful in assisting the classification of each sample. The nature of the decomposition also makes it trivial to discard IMFs that encode signal components which are undesirable; or conversely, to focus on components which are of interest. The marginal Hilbert spectrum is a natural way to summarize the Hilbert spectrum and produce a convenient odor signature. Ultimately, the odor signatures formed well-defined clusters and were easily classified.

6.1 Future Work

A number of experiments are yet to be performed: determining the lower limit of sensitivity to various compounds, for example. The true test of the Hilbert-Huang transform will be to isolate components from samples which contain combinations of compounds. Identifying binary mixtures with varying proportions of ethanol and another compound of interested would be an important milestone in the development of an electronic nose with this application. It would also be prudent to examine other temperature modulation profiles besides sinusoidal. In fact for some classes of MOS sensor there is evidence that a high temperature pulse after exposure to gas can shorten the recovery time of the sensor [38]. No doubt this would be a useful phenomenon to exploit for a MOS sensor which is routinely exposed to high concentrations of gas.
Bibliography


