University of Alberta

Ultraviolet Stabilization and Performance Enhancement of Nanostructured Humidity Sensors

by

Daniel Patrick Smetaniuk

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Microsystems and Nanodevices

Department of Electrical and Computer Engineering

Spring 2012
Edmonton, Alberta

Abstract

Glancing angle deposition (GLAD) was used to fabricate nanostructured TiO_2 capacitive relative humidity (RH) sensors. These sensors exhibit sub-second response times and large sensitivities, but are susceptible to ageing. Ultraviolet (UV) treatment of the sensors has been found to reverse ageing and enhance sensor performance. This thesis presents research investigating the UV stabilization and performance enhancement of GLAD RH sensors. The UV treatment was characterized using a mercury vapour lamp and optical filters to isolate UV wavelengths. Treatment and long-term stabilization with UV light-emitting diodes (LEDs) was studied and 370 nm was identified as the optimum LED wavelength for stabilization. A custom 8-channel impedance analyzer that was built will allow for parallel sensor testing in future experimentation to optimize the UV treatment. The goal is to eventually combine a capacitive sensor with UV LEDs in a self-stabilizing sensing platform.

Table of Contents

List of Tables

List of Figures

List of Symbols and Abbreviations

α	Angle of vapour incidence	Degrees
A	FOM maximum value	
AC	Alternating current	
ADC	Analog to digital converter	
β	Column tilt angle	Degrees
c	FOM fit parameter	s
C	Capacitance	F
C_0	Reference capacitive response	F
C_g	IDE geometric capacitance	F
C_s	Capacitive sensor response	F
$C_x[\%RH]$	Capacitive response curve after treatment or ageing	F
CMOS	Complimentary metal oxide semiconductor	
CW	Continuous wave	
ΔE_v	Valence energy level offset	eV
ΔE_c	Conduction energy level offset	eV
Δf_m	Measured frequency spacing size	Hz
D	Diffusivity	m^2/s
d	Film thickness	m
$d_{barrier}$	Barrier layer width	m
d_{pore}	Pore diameter	m
d_{QW}	Quantum well layer width	m
DAC	Digital to analog converter	
DDS	Direct digital synthesis	

DFT	Discrete fourier transform	
DUT	Device under test	
ϵ_{RH}	Absolute error for relative humidit measurements	
f	Frequency	Hz
f_clk	System clock frequency	Hz
f_m	DDS output (measuring) frequency	Hz
f_s	ADC sampling frequency	Hz
FET	Field effect transistor	
FIFO	First-in-first-out	
FOM	Figure of Merit (relative)	
$FOM*$	Figure of Merit (absolute)	
FWHM	Full width at half maximum	
γ	Surface tension of water	N/m
GLAD	Glancing angle deposition	
GRIN	Graded index	
i_R	Alternating current through the range resistor	A
i_x	Alternating current through the unknown impedance	A
I	Current	A
I_r	Current through the feedback resistance	A
I_x	Current through the unknown impedance	A
IDC	Interdigitated capacitor	
IDE	Interdigitated electrode	
λ	Wavelength	m
L	Diffusion length	m
L	Inductance	H
L	Number of sampled periods	
LED	Light emitting diode	
LCR	Inductance (L), capacitance (C), resistance (R)	
LPF	Low-pass filter	
m	FOM fit parameter	1/s

M_w	Molar mass of water	g/mol
MFC	Mass flow controller	
N	Number of samples	
ω	Angular frequency	rad/s
ϕ	Substrate rotation angle	Degrees
P_s	Saturation water vapour partial pressure	Pa
P_w	Water vapour partial pressure	Pa
PIR	Phase increment register	
PGA	Programmable gain amplifier	
QCM	Quartz crystal microbalance	
ρ	Density of water	g/cm^3
R	Molar gas constant	J K^{-1} mol^{-1}
R	Resistance	Ω
R_r	Feedback resistance	Ω
R_{out}	Series output resistance	Ω
R_R	Range resistor	Ω
RH	Relative humidity	%
RH_{min}	Minimum relative humidity for FOM calculation	%
RH_{max}	Maximum relative humidity for FOM calculation	%
r_k	Kelvin radius	m
SAW	Surface acoustice wave	
SCL	Serial clock	
SDA	Serial data line	
SoC	System-on-a-chip	
τ	Response time	s
τ	Time constant	s
θ	Impedance phase angle	Degrees
θ_x	Unknown impedance phase angle	Degrees
t	Time	s
T	Temperature	Kelvin

T_{meas}	Measurement time	s
TFT	Thin film transistor	
u_0	Test signal	V
u_i	Measured current signal, differential amp output	V
u_R	Signal across range resistor	V
u_u	Measured voltage signal, buffer output	V
u_x	Signal across unknown impedance	V
U_i	Complex result of DFT on signal u_i	
U_u	Complex result of DFT on signal u_u	
UV	Ultraviolet	
V	Voltage	V
V_r	Voltage across the feedback resistance	V
V_x	Voltage across the unknown impedance	V
X	Reactance	Ω
X_C	Capacitor reactance	Ω
X_L	Inductor reactance	Ω
XRD	X-ray diffraction	
Z	Impedance	Ω
Z_x	Unknown impedance	Ω

Chapter 1

Introduction

This thesis presents research on high-speed, high-sensitivity humidity sensors. Chapter 1 provides a definition of relative humidity and gives examples of humidity sensing applications, specifically those requiring sensors with fast response times. The motivation and scope of the research are discussed and an outline of the thesis is given.

1.1 Definition of Relative Humidity

Relative humidity is a measure of the water content in air and is defined as the ratio of the partial pressure of water vapour in an air mixture to the saturated water vapour partial pressure, described as a percent:

$$\text{RH} = \frac{P_w}{P_s} \times 100\% \tag{1.1}$$

where RH is the relative humidity, P_w is the water vapour partial pressure, and P_s is the saturated water vapour partial pressure.

1.2 Humidity Measurement Applications

Relative humidity sensors are crucial to measurement and control systems in a wide variety of industries. The textbook ”Water Vapor Measurement Methods and Instrumentation” [1] provides many application examples, several of which are given below.

Moisture sensing is critical to the microelectronics fabrication industry. Even trace levels of water vapour in a process can reduce yields drastically. Highly sensitive humidity sensors are important for monitoring these processes. Museums and library archives housing centuries-old pieces of art and literature need to carefully control humidity levels and maintain environment conditions to preserve artifacts. Humidity control is important for indoor air quality. At room temperature, the optimum humidity environment for humans is between 40 % RH and 60% RH. Proper humidity control is necessary to prevent harmful fungi and other microbial growth, prevent indoor smog, and reduce allergen levels that may exacerbate asthma. Numerous medical applications require humidity measurements including incubators, artificial heart growth, medical gas supplies, and sterilizers. Heat treatment processes for metals and alloys aim to create hard surface coatings, making metal alloys more resistant to fatigue and corrosion. During treatment, careful measurement and control of gases, including water vapour, is required to achieve the desired properties. Industrial driers account for significant energy use in many industries and often suffer from poor thermal efficiency. Humidity control systems can improve efficiency and reduce energy and fuel use. In food processing, the amount of moisture during processing and packageing can affect taste, color, and the shelf life of the foods. Humidity is critical to many meteorological applications. Ground-based and aerial humidity sensors are important for weather forecasting. The lift produced by airplanes during takeoff and landing depends on humidity, and proper measurements are important for lift calculations. There are many more humidity sensing applications than the small subset given here.

1.2.1 Applications Requiring Fast Response Times

There are some applications which require high-speed humidity sensing. The example applications discussed below can either be improved by faster and more sensitive humidity sensor technology, or have yet to be fully realized because the required technology is not yet available.

Tropospheric water vapour is an important factor in climate and weather systems and measurement of this water vapour is important for meteorology. Water vapour concentrations are routinely measured by launching radiosondes; however, these offer poor spacial and temporal resolution of the complex three-dimensional water vapour profiles [2]. High-speed humidity sensors with a 10 Hz bandwidth fixed to aircraft can map atmospheric water vapour with high spatial and temporal resolution. High-resolution and high accuracy mapping of water vapour profiles in the atmosphere will assist with areas such as atmospheric chemistry, hydrology, severe weather prediction, climate research, and polar studies [2].

Analysis of human perspiration measurements can provide valuable information on patient health, and is also used for designing clothing for use under various climatic conditions [1]. There is a need for researchers to measure perspiration rates during exercise and physical exertion. Humidity sensors for measuring perspiration rates must be portable, have high accuracy, fast response times, and be highly sensitive to humidity changes. These sensors must be able to accurately and quickly track small perspiration fluctuations that result from changes in the physical exertion of the patient.

Spirometers measure the volume of air inhaled and exhaled by the lungs and are used to diagnose and treat asthma and chronic obstructive pulmonary disease, which affect between 10-20% of the population worldwide [3]. Currently spirometers use mechanical peak expiratory flow meters, which often display large discrepancies and have limited accuracy. Development of a multisensor microsystem to replace the mechanical systems requires high-speed humidity sensors with sub-second response times [3–6].

High-speed humidity sensors may also be used for replacing devices requiring direct airway connections for respiration monitoring of neonates, patients under risk of airway obstruction or exhibiting apnea, or patients undergoing anesthesia [7, 8]. These devices are used to alert when apnea or hypoapnea events occur. Apnea results in a lower oxygen content in the blood, which can lead to brain damage, so quick detection is crucial. With newborns, raised respiratory rates in the range of 60 breaths per minute may be a sign of illness or infection [9]. Humidity sensors with very fast response times are needed to detect these breathing rates.

1.3 Motivation and Scope of Thesis Research

Improving humidity sensor performance generally requires modifying factors such as material, thickness, porosity, surface area, and morphology. Our group has developed a physical vapour deposition technique, Glancing Angle Deposition (GLAD) that allows for the deposition of high porosity nanostructured thin films [10–14]. We have fabricated capacitive relative humidity sensors by depositing porous GLAD thin films on top of interdigitated electrodes (IDEs) to form a capacitive relative humidity sensor [15,16]. Response times as fast as 50 ms have been demonstrated with a TiO_2 sensing layer [17–19]. Typically these sensors have a large dynamic range with capacitive changes of 1 nF to 1 μF from 2 %RH to 95 %RH. However, as is characteristic with metal oxide sensing layers, the sensors are vulnerable to ageing over time [20]. The GLAD process and GLAD RH sensors are described in Chapter 3. Many commercial humidity sensors use polymer sensing layers, which are often stable to under 2% RH a year. These sensors typically have a sensitivity under 1 pF/%RH, an accuracy of 2-3% RH and response times of 5 - 60 seconds [21–25].

TiO_2 is well known as a photocatalyst which is active under ultraviolet (UV) light. Because of this effect titanium dioxide is used in a wide range of applications such as water purification, air cleaning, anti-fogging coatings, and self-cleaning surfaces [26, 27]. Previously our group has studied the effect of UV treatments with a mercury vapour lamp on ageing and sensor performance with GLAD TiO_2 relative humidity sensors [28, 29]. The results of the study, presented in Chapter 3, found that a 48 hour UV treatment not only reverses the effects of ageing in the sensors, but also improves performance.

Recent advancements and innovations in AlGaN semiconductors, described in Chapter 4, have resulted in commercial availability of deep UV LEDs, with wavelengths down to approximately 200 nm now achievable. The cost of UV LEDs continues to decrease while the available power increases. UV LEDs offer a small, low power alternative to mercury vapour lamps and do not have environmental hazard disposal issues.

This thesis presents research investigating the effects of UV treatment on GLAD

RH sensors. The objective of the work is to enhance humidity sensor performance and long term stability by combining a capacitive sensor with a UV LED (Figure 1.1).

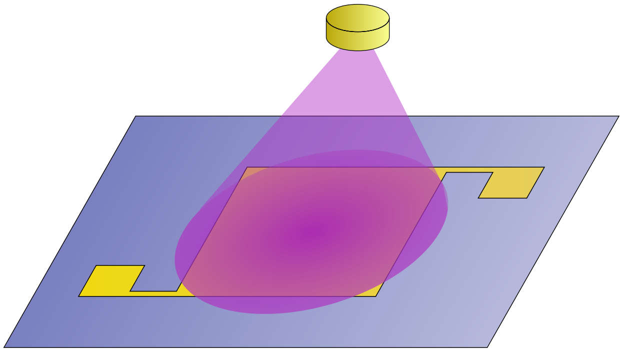


Figure 1.1: Concept of the integrated UV LED for a self-regenerating sensor.

Chapter 2 provides definitions of performance characteristics of humidity sensors and a background of types of technologies including the limitations. The physical mechanisms governing the performance of metal oxide humidity sensors are explained.

In Chapter 3 the GLAD process for fabricating nanostructured thin films is described. The development of GLAD relative humidity sensors is discussed and the issue of sensor ageing is addressed.

A background of the UV LED technology used for sensor treatment experiments is provided in Chapter 4.

Chapter 5 reports the results from a study characterizing the effect of wavelength during UV treatment on sensor performance and ageing. Optical filters were used with a mercury vapour lamp to isolate individual emission lines from the spectra of the lamp. Sensors were treated with different filtered wavelengths and the effect on sensor response and ageing was analyzed.

A study investigating UV treatment with UV LEDs with the goal of identifying an optimum UV LED wavelength is presented in Chapter 6. The long-term stability of sensors illuminated with UV LEDs is examined. The performance of GLAD RH sensors stabilized with UV LEDs is compared to commercial RH sensors.

Chapter 7 describes the design and testing of an 8-channel impedance measuring device. The device will be used in future experiments beyond the scope of this thesis.

Chapter 8 summarizes the research presented in this thesis. Suggestions for future work and research are given.

Chapter 2

Relative Humidity Sensing

This chapter describes a number of methods and types of humidity sensing, and how the devices work. The mechanisms of water adsorption on metal oxides and the effect of pore size on water capillary condensation are discussed. There are several terms used throughout the chapter to describe and compare the performance characteristics of sensors. These are defined below:

- **accuracy** — The combined error in the measured value compared to the actual value being measured. It usually incorporates sensor hysteresis, linearity, drift, and calibration in the figure and is quoted as a percent.

- **sensitivity** — The change in a sensor parameter with a change in humidity. e.g. for a capacitive sensor: nF / %RH

- **selectivity (specificity)** — The ability of a sensor to detect a particular analyte without detecting others.

- **hysteresis** — The maximum difference (in percent) between the rising and falling sensor output at a particular humidity.

- **stability** — The deviation of the output of the sensor across a range of conditions such as temperature or pressure, or the drift in sensor output over time.

- **response time** — Upon a step change in humidity, the time it takes the sensor output to rise or fall past a specific value.

- **linearity** — The maximum difference, in percent, of the measured output from a calibrated line of best fit for the device.

- **lifetime** — The length of time under which the sensor reliably and effectively operates within the specifications, after which it must be replaced.

The design trade-offs that exist between these different performance characteristics have led to a wide variety of different approaches and types of relative humidity sensors.

2.1 Types of Relative Humidity Sensors

2.1.1 LiCl Dewpoint Sensor

The Dunmore or LiCl dewpoint sensor was invented in 1938 and uses a fine glass or plastic tube coated with a LiCl solution (or sometimes other salts such as LiBr) around which two copper wires are wound. Voltage is applied to one of the wires. When the relative humidity is above approximately 12%, the LiCl solution becomes conductive and is heated by the applied voltage, causing water to evaporate from the solution. This in return decreases the conductivity of the solution. This process continues until an equilibrium is reached between the humidity in the air and the amount of water in the solution. The second wire is then used to measure the temperature, from which the dew point or relative humidity may be measured using LiCl saturation curves [30]. Lithium chloride sensors have the advantage of being low cost and having a low sensitivity to contaminants; however, they are limited by the fact that they cannot measure below 11% RH and have a low to moderate accuracy. Additionally they have slow response times, require frequent maintenance and calibration, and generate significant heat which restricts them from being used in small contained environments [1].

2.1.2 Gravimetric

Gravimetric humidity sensors rely on the change of mass of a structure during water adsorption or desorption. The very first hygrometer was invented by Nicholas de

Cusa in 1450 and used a ball of sheep's wool on a balance scale with weights on the other side to measure the change in mass of adsorbed water. Leonardo da Vinci later drew sketches of this device with sponges instead of wool for a better sensitivity to humidity [1]. A popular modern gravimetric humidity sensor is the quartz crystal microbalance (QCM), which uses the resonance of piezoelectric quartz with a hygroscopic layer on top. Water adsorbed in the hygroscopic layer changes the mass of the QCM, changing the resonant frequency, which is in the MHz range. Humidity is measured by the change in frequency. Polyimide, porous ceramics, sol-gel silica, and fullerene layers may be used as the hygroscopic layer. A second non-coated QCM can be used as a reference to account for frequency changes resulting from temperature or pressure changes. Surface acoustic wave (SAW) devices are also another type of gravimetric sensor. The operation of SAW devices relies on the change in phase velocity of surface waves resulting from adsorbed water. SAW devices usually consist of two pairs of interdigitated electrodes (IDEs) on a piezoelectric substrate, one the emitter and one the receiver, with a hygroscopic layer between the two electrodes. Similar to the QCM sensors, two SAW devices may be used, one without a hygroscopic layer, to account for cross-sensitivities. These types of sensors have wide operating ranges, low contamination sensitivity, and fast response times, but are generally more expensive and are not suitable for *in situ* measurements since they require a sample gas flow and often a reference gas flow [1].

2.1.3 Hygrometric

Hygrometric humidity sensing is another old method that utilizes the mechanical expansion or contraction of a material during water adsorption or desorption to measure relative humidity. An early hygrometer invented by Santorrio Santorre in 1625 used the expansion and contraction of a lyre string with humidity to raise and lower a weight on one end [1]. Many hygrometers use the expansion or contraction of human or animal hair in their mechanism. A recent example of such a device was developed by Ha *et al.* and uses a strand of hair with one end fixed, and the other connected to a thin metal sheet with a small window in it, supported by a spring [31]. Light from a LED is transmitted through the window and is measured with a photodiode. When the humidity changes, the hair strand expands or contracts, moving the mirror and changing the amount of light transmitted. Other devices use polymer films which swell and contract under changing humidity. A sensor designed by Gerlach and Sager uses a polyimide layer on top of a silicone membrane. Stresses on the membrane caused by the swelling and contracting of the polyimide film with humidity are converted into a voltage output with piezoresistors located at points of high stress [32]. Another sensor designed by Singamaneni *et al.* uses a plasma polymerized methacrylonitrile layer on top of a silicon microcantilever to cause deflections in the cantilever in response to humidity changes because of tensile or compressive stresses that develop in the polymer layer [33].

2.1.4 Optical

There are several optical techniques that are used for humidity sensing. One technique, laser absorption spectroscopy, makes use of the absorption spectrum of water. Schirmer *et al.* transmited laser light through a sample of air, and measured the absorption at certain wavelengths to determine the water content in the air sample [34]. Another technique relies on changes in the refractive index of a material as water is adsorbed. A Fabry-Perot interferometer measures changes in the resonant frequency of light in a structure with a medium between two reflective surfaces. A device presented by Mitschke uses a stack of SiO_2 and TiO_2 layers as a humidity sensor coupled with an optical fibre [35]. Light is passed through the fibre to the sensor, where it is reflected back to a detector. These devices have a high sensitivity and offer the potential of remote monitoring with signals passed over long optical fibers, but require fine control in the processing of the microporous optical layers. Our group has used glancing angle deposition (GLAD) to create a TiO_2 film designed as a narrow bandpass filter by varying the porosity and thus the refractive index throughout the film [36]. Water adsorption and desorption in the film shifts the center wavelength of the bandpass filter. These sensors can be more complicated to fabricate and for accurate measurements require more expensive optical equipment. Other optical methods for sensing include materials that change color during water adsorption, fluorescent and luminescent compounds, and light scattering [37].

2.1.5 Capacitive and Resistive

One of the most popular methods to measure humidity is with capacitive or resistive sensors in which adsorbed water results in a change in the capacitance or resistance of the device. Usually ceramics or hygroscopic polymers are used as the dielectric in capacitive sensors, while ceramics, polymers and electrolytes are used with resistive sensors. Al_2O_3, TiO_2, and SiO_2 are commonly used metal oxides. The adsorption and desorption of water on metal oxides is described in detail in Section 2.2 of this chapter, and Traversa has extensively reviewed metal oxide relative humidity sensors [38]. Polymers for resistive sensors respond to humidity with conductance changes while polymers for capacitive sensors respond with a dielectric constant change. The advantage of polymers over ceramic sensors is their better linearity and long-term stability. However, polymers are very sensitive to heat and usually only operate at room temperature, whereas ceramics are able to operate at higher temperatures. Suitable polymers for capacitive sensors include polyimides, poly(methyl methacrylate) (PMMA), cellulose acetate butyrate (CAB), poly(ethylene terephthalate) (PET), and poly(vinyl crotonate) [39]. Suitable conductive polymers for resistive sensors include poly(*p*-diethynylbenzene) (PDEB) and poly(propargyl benzoate) (PPBT) [39].

The properties of capacitive sensors depends on the dielectric layer as well as the electrode geometry. Interdigitated electrodes (IDEs) allow the dielectric layer to remain exposed allowing for quick adsorption/desorption and fast response times. However, only half of the IDE electric field passes through the dielectric, while the other half passes through the substrate resulting in a smaller sensitivity. Top and bottom electrodes with the dielectric between results in a uniform electric field that is contained within the electrodes. The top electrode is usually fabricated very thin (10 - 80 nm of gold) to allow water to pass through [30]. Figure 2.1 shows schematics of interdigitated and parallel plate electrode configurations [40]. Other electode configurations include vertically offset interdigitated, spiral, and grid (or porous electrode) [41]. The design of resistive sensors is usually similar to that of

Figure 2.1 removed due
to copyright restrictions.

Figure 2.1: Top view and side view schematics of parallel plate and interdigitated electrode capacitor sensors. Source: [40]

capacitive sensors.

Field effect transistors (FETs) may also be used as gas sensors [42–44]. Many thin film transistor (TFT) sensors are compatible with standard complimentary metal oxide semiconductor (CMOS) technology commonly used to fabricate integrated circuits. With the proper sensing layer, these may be used as relative humidity sensors. A sensing layer is deposited on top of a bottom gate electrode. A permeable top electrode is deposited on top of the sensing layer, similar to the capacitive sensor design. The threshold voltage is related to the capacitance of the sensing layer, which changes as water is adsorbed. This results in a change in the drain current of the device.

2.1.6 Limitations of Current Technologies

The humidity sensing technologies described above each address different requirements such as dynamic range, accuracy, cost, and size for a range of humidity sensing applications. However response time is an important performance characteristic that current technology has been deficient in addressing. Most commercial humidity sensors have response times on the order of seconds to minutes. Some groups have been able to obtain sub-second response times, but the sensors have very low sensitivities and often have longer desorption times [3,4,6,8].

Porous metal oxide capacitive sensors are simple to fabricate using a variety of methods and promise very fast response times. Our group has demonstrated nanostructured porous metal oxide sensors with sub-second response times and dynamic ranges over three orders of magnitude. The fabrication process and the characteristics of these sensors is described in Chapter 3. The remainder of the current chapter explains some of the mechanisms governing humidity sensing with metal oxides.

2.2 Metal Oxide Relative Humidity Sensors

In principle, any metal oxide can be used for relative humidity sensing because of water adsorption on the surface. As the relative humidity increases, water condenses in the pores and capillaries of the metal oxide, resulting in a large increase in the dielectric constant and increase in conductivity.

2.2.1 Water Adsorption

There are two types of water adsorption that occur at the surface of a metal oxide, chemisorption and physisorption [42]. The first layer of water molecules are chemisorbed onto the oxide surface. This process starts with a water molecule physisorbed onto an activation site (Figure 2.2a-b). The water molecule then dissociates [45]:

$$\mathrm{H_2O} \rightarrow \mathrm{H^+} + \mathrm{OH^-} \tag{2.1}$$

resulting in a hydroxyl and hydrogen proton. The hydroxyl from the water molecule reacts with a metal cation at the surface, while the hydrogen proton reacts with the adjacent O^{2-} to form another hydroxyl OH^- group (Figure 2.2c) [38]. For every water molecule that dissociates the result is two hydroxyls on the oxide surface. Next a water molecule physisorbs to two chemisorbed surface hydroxyl groups with two hydrogen bonds (Figure 2.2d). The double hydrogen bonds in the first physisorbed layer prevent the physisorbed water molecules from being free to move or rotate. However, subsequent layers of physisorbed water only have single hydrogen bonds and are more free to move and reorient (Figure 2.3).

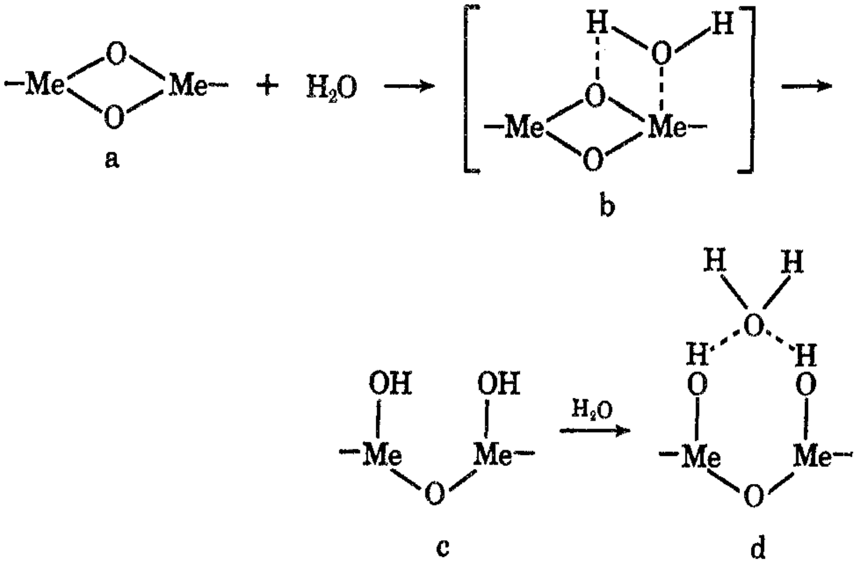


Figure 2.2: Mechanism through which water adsorbs on a metal oxide surface. a-b) a water molecule is physisorbed onto an activation site. c) The water molecule dissociates and reacts to form two surface hydroxyls. d) A water molecule is physisorbed by double hydrogen bonding to the two surface hydroxyls. Reprinted with permission from [46]. Copyright 1969 American Chemical Society.

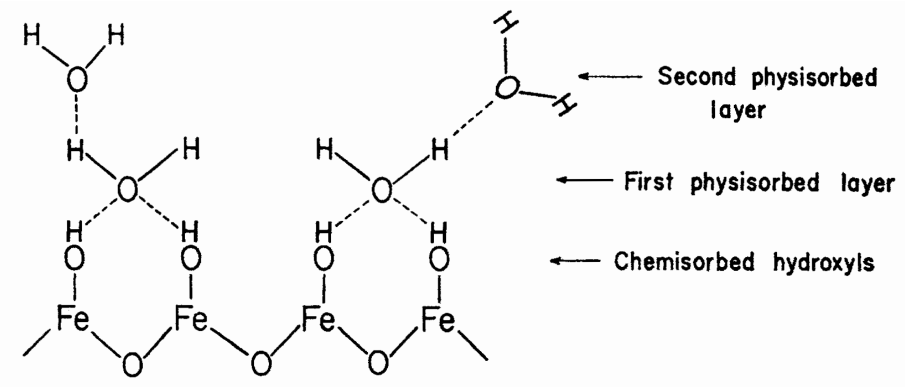


Figure 2.3: Example of adsorbed layers on an α-Fe_2O_3 surface. Additional adsorbed layers form with single hydrogen bonding in the same way as the second physisorbed layer. Image from [47]. Reproduced by permission of the Royal Society of Chemistry.

As more layers are adsorbed, the water molecules become less ordered and more liquid-like in their behavior. This means that under an electric field the water molecules can form dipoles and orient themselves to the field, increasing the dielectric constant [38]. In this liquid-like layer, charge transfer is dominated by the Grotthuss mechanism [48, 49], in which hydrogen protons may hop between water molecules in a chain, temporarily forming H_3O^+ groups. The carrier concentration increases with the number of physisorbed layers, increasing conductivity.

2.2.2 Water Condensation in Pores and Capillaries

As the relative humidity increases, water vapour condensation occurs in pores and capillaries. Capillary condensation occurs for cylindrical pores closed at one end with a radius smaller than the Kelvin radius, r_k, according to the Kelvin equation:

$$r_k = \frac{2\gamma M_w}{\rho RT \ln P_s/P_w} \tag{2.2}$$

where γ is the surface tension of water, ρ is the density of water, M_w is the molecular weight of water, R is the universal gas constant, T is temperature, and P_w and P_s are the actual and saturation water vapour pressures respectively. The Kelvin radius will differ between hydrophillic and hydrophobic surfaces since the surface tension of water is related to the water contact angle. With pores that are open on both ends, hysteresis occurs between adsorption and desorption because water condenses in pores with radii up to r_k/2 while desorption still occurs according to r_k [42,50]. This is because in tubular pores open at both ends, capillary condensation first occurs along the inside edges of the pore until a cylindrically-shaped meniscus is formed, after which condensation will occur corresponding to an effective radius that is twice the pore radius [51]. In reality porous materials are made up of a network of open and closed pores.

Increasing the porosity or the surface area of a metal oxide will increase its sensitivity. Sensitivity may also be increased by increasing the number of surface sites available for water dissociation, such as oxygen vacancies on the surface [45]. Over time, stable chemisorbed OH^- groups form on the surface of metal oxides, reducing the number of sites available for water to dissociate and causing drift in the response [20]. Sensor response may be regenerated by thermally heating the oxide to over 400 °C, causing desorption of OH^- groups [46]. Impurities present in the atmosphere may also adsorb or adhere to the surface, causing permanent changes to the sensor's response, however contaminants that adsorb in the same way as water may also be removed by heating [38].

Chapter 3

Sensors by Glancing Angle Deposition: Prior Work

3.1 Introduction

Several types of sensors have been fabricated using the Glancing Angle Deposition (GLAD) process. This chapter first describes the GLAD process and how deposited film morphologies may be controlled with the process, and then explains sensor development using GLAD. Previous work on GLAD sensors, interdigitated electrode development, comparison of sensing materials, factors affecting response time, and sensor ageing are discussed.

3.2 Glancing Angle Deposition

GLAD is a physical vapour deposition technique developed by the Brett group for growing nanostructured porous thin films [10–14]. It employs oblique incident vapour flux angles to create a variety of microstructures. With GLAD, film morphology may be controlled on a 10 nm scale. GLAD films may be made using virtually any material that can be deposited with physical vapour techniques and may be used in a variety of applications such as optical filters, nanosprings and sensors.

At oblique deposition angles columnar film growth results from adatom shadowing on the substrate surface. Initial nucleation sites of the deposited material shadow areas of the substrate where no material will be deposited (Figure 3.1).

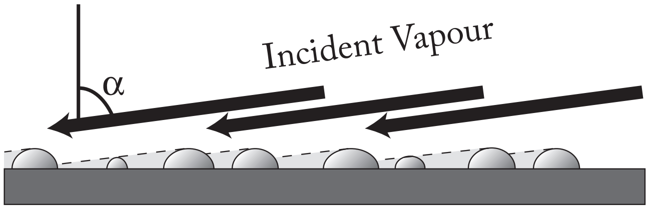


Figure 3.1: Shadowing by nucleation centers at oblique deposition angles. Reprinted with permission from [10] Copyright 2007, American Vacuum Society.

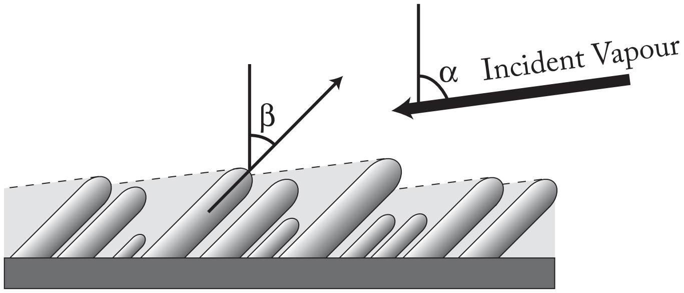


Figure 3.2: Tilted columnar growth of GLAD films. Reprinted with permission from [10] Copyright 2007, American Vacuum Society.

Rather than forming a continuous film, further material is deposited on the top areas of the nucleation sites and tilted columns grow out from the substrate (Figure 3.2). The column tilt angle β is different from the incident vapour flux angle α.

Columnar film growth using GLAD requires a directional incident vapour flux with little divergence in order for shadowing to occur. Thermal and electron beam evaporation as well as long-throw sputtering provide suitable collimated vapour fluxes. Low adatom diffusion lengths on the substrate surface are also required for columnar film growth. This requires lower substrate temperatures that are generally less than 0.3 times the source material melting point, corresponding with zone 1 film growth [52]. At higher temperatures adatom diffusivity is large enough that shadowed areas are filled in, and separate columnar structures are not formed.

Figure 3.3 shows a typical setup for a GLAD system. The substrate is mounted on a chuck that is positioned above the vapour flux source. The chuck is moved around two degrees of freedom: the angle of incidence α and the angle of rotation ϕ. A quartz crystal microbalance (QCM) measures the vapour flux near the substrate, from which the deposition rate at the chuck is derived. A computer monitors the deposition rate and controls the substrate rotation.

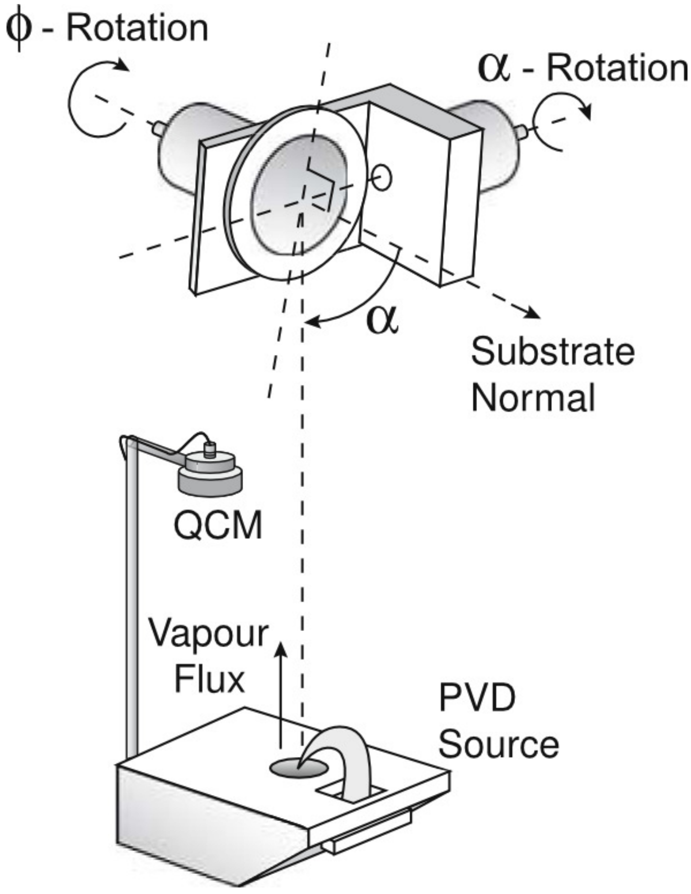


Figure 3.3: Typical GLAD system setup. Image from [12]. Reprinted with kind permission of Springer Science and Business Media.

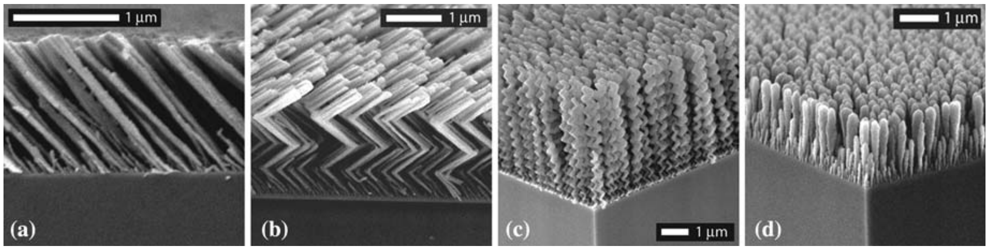


Figure 3.4: Effect of ϕ rotation speed on film structure: (a) tilted columns with no rotation, (b)chevron film by increasing ϕ by 180° between segments (c) helices at slow rotation speeds, (d) vertical columns at fast rotation speeds. Images from [12]. Reprinted with kind permission of Springer Science and Business Media.

Figure 3.5 removed due to copyright restrictions.

Figure 3.5: Effect of α on film density. α increases from (a)-(c) resulting in increased columnar separation and column width. Image from [40]

The structure of the deposited film may be controlled by adjusting the speed of substrate rotation about ϕ (Figure 3.4). When the substrate is stationary, ballistic shadowing results in slanted columns growing outward from the substrate (Figure 3.4a) at an angle β, which is smaller than the deposition angle α because of the geometry of the shadowing. Rotating the substrate 180° causes the columns to grow in the opposite direction. Chevron or zigzag structures can be created by periodically increasing ϕ by 180° (Figure 3.4b). If ϕ is continually increased at a slow rate, the direction of column growth follows the rotation and vertical helices or springs are formed (Figure 3.4c). When ϕ is increased at a faster rate, the direction of column growth cannot respond fast enough to the changing vapour flux, and the structure degenerates into vertical columns (Figure 3.4d).

The density of posts in the film is controlled by changing the angle of incidence α (Figure 3.5). As α is increased the columnar separation also increases. The film morphology changes from small closely-packed columns to larger, more loosely spaced columns. The density - α relationship and the surface area of GLAD films have been characterized [53]. The surface area of GLAD films was found to peak at around 60° - 75°, depending on the type of material. At larger deposition angles there are larger voids between columns because of increased columnar separation, which results in an overall decrease in surface area from the peak at lower angles.

3.3 GLAD Relative Humidity Sensors

3.3.1 Previous GLAD Sensor Development

GLAD films are very suitable for use as the sensing layer in sensors because of their high porosity. Hydrocarbon sensors have been fabricated using Pt GLAD films [54]. Hydrocarbons were sensed by measuring the temperature change when the hydrocarbons were catalytically oxidized on the surface of the Pt films. The first capacitive GLAD relative humidity sensor was demonstrated by Wu *et al.* using SiO films [55]. The films were deposited on top of aluminum electrodes with both helical and columnar microstructures and were capped with a gold electrode. A dynamic response range over 5 orders of magnitude and a response time down to 3 minutes was reported. Analysis of an equivalent circuit model showed that the change in sensor capacitance could not be attributed to just changes in the dielectric constant of the layer from pores filling with water. Later a theoretical model was developed that explained the large dynamic range [56]. It was determined that sensor microstructure, and properties and interactions at the surface of the posts are related to sensor performance. It was suggested that imperfect pore microstructure can lead to condensed water being trapped which results in a larger hysteresis. With parallel electrode sensors, the response time is usually dictated by the time it takes water molecules to diffuse through the top electrode. To address this issue, Harris *et al.* obtained response times in the range of 100 ms by using a continuous top electrode perforated with pores to allow for easier diffusion [57]. Two types of sensors were fabricated with this electrode. The first was vertical SiO vertical posts 2.3 μm thick. The second type was the inverse of the first and consisted of pores embedded in a photoresist polymer 1.4 μm thick. These types of films are known as perforated thin films. Optical and capacitive relative humidity sensors have been made using silica (SiO_2), titania (TiO_2), and alumina (Al_2O_3) GLAD films [16].

3.3.2 Optical GLAD RH Sensor

Sensor response times of 270 ms and 160 ms for adsorption and desorption have been obtained with optical titania GLAD RH sensors [36]. An optical filter sensitive to humidity was made using a graded index (GRIN) titanium dioxide GLAD film. The filter uses sinusoidal variation in the refractive index to produce a stopband of frequencies with low optical transmission. Introduction of a π-phase shift defect in the index profile creates a very narrow passband of frequencies with very little transmittance attenuation (Figure 3.6). The sinusoidal refractive index profile was created by periodically modulating the deposition angle α while depositing a vertical column film, creating variations in the column diameter. The effective index of refraction of the film is a combination of the indices of the columns and voids and depends on the relative amount of each. Water adsorption changes the index of the void and thus the effective index of the film. Adsorption of water increases the effective index, which shifts the center bandpass frequency to a lower frequency. Conversely, desorption of water decreases the effective index, resulting in a shift to a higher frequency. Humidity was measured by measuring the center frequency of the bandpass filter.

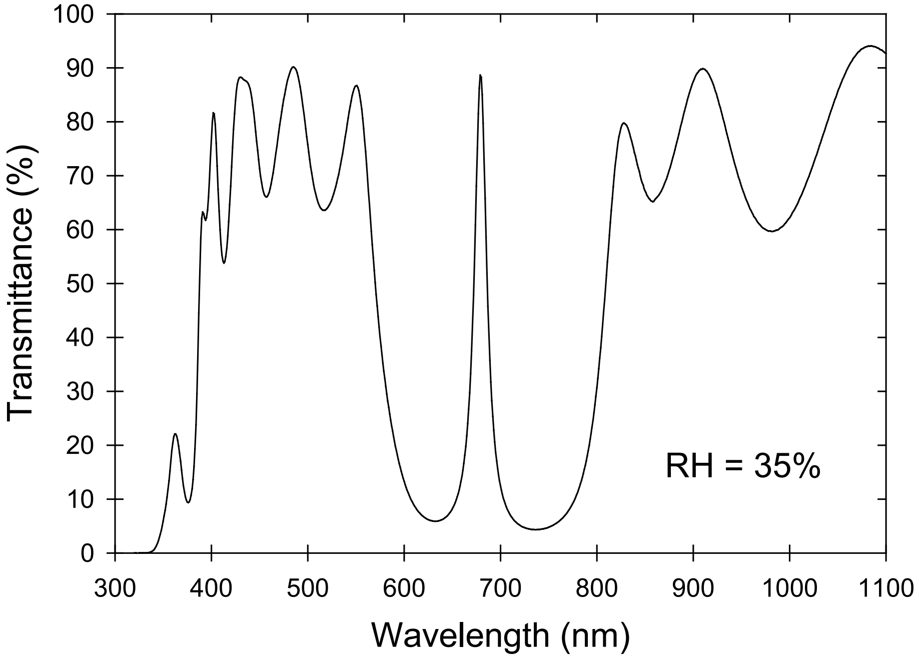


Figure 3.6: Transmittance spectrum of an optical GLAD sensor. The stop band spans from approximately 550 nm to 830 nm with a narrow pass band centered at approximately 680 nm. Reprinted from [36]. Copyright 2006, with permission from Elsevier.

3.3.3 Interdigitated Electrode Development

The development of GLAD RH sensors using interdigitated electrodes (IDEs) as opposed to parallel plate sensors is desirable because the design leaves the porous sensing material open to the air, increasing response time, as well as removing the capping layer step in the GLAD fabrication process. The first IDEs for GLAD RH sensors used a large geometry with 20 μm digit widths and 10 μm digit separation for a 30 μm period and were fabricated using standard photolithography techniques [40]. Figure 3.7 shows a diagram of the IDE layout. Modelling of IDE electrical fields has shown that most of the electrical field is contained within a distance of half the electrode period above the IDE [58]. In the case of the large geometry IDEs, this means the field is contained within 15 μm of the IDEs. GLAD films are typically grown up to several microns thick, thus much of the electric field is not contained within the film and the sensor will have a reduced sensitivity. Two possible approaches to improve sensitivity are to increase the film's thickness, or reduce the IDE digit period. Increasing the film thickness is undesirable because the deposition time is increased and column broadening and extinction create a dramatic change in film microstructure. Additionally, sensor response time scales with film thickness, as will be described in Section 3.3.5. The more desirable solution is to reduce the IDE digit period and to use a smaller geometry, while continuing to use film thicknesses in the micrometer range.

Figure 3.7 removed due to copyright restrictions.

Figure 3.7: Diagram of an interdigitated electrode. Image from [40]

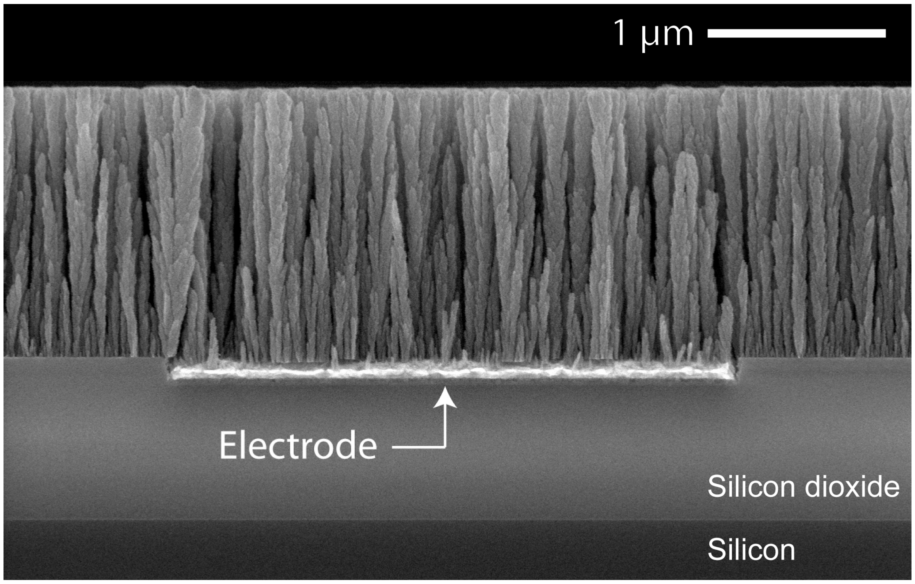


Figure 3.8: A countersunk electrode with a GLAD film deposited on top. Image from [16] © 2008 IEEE.

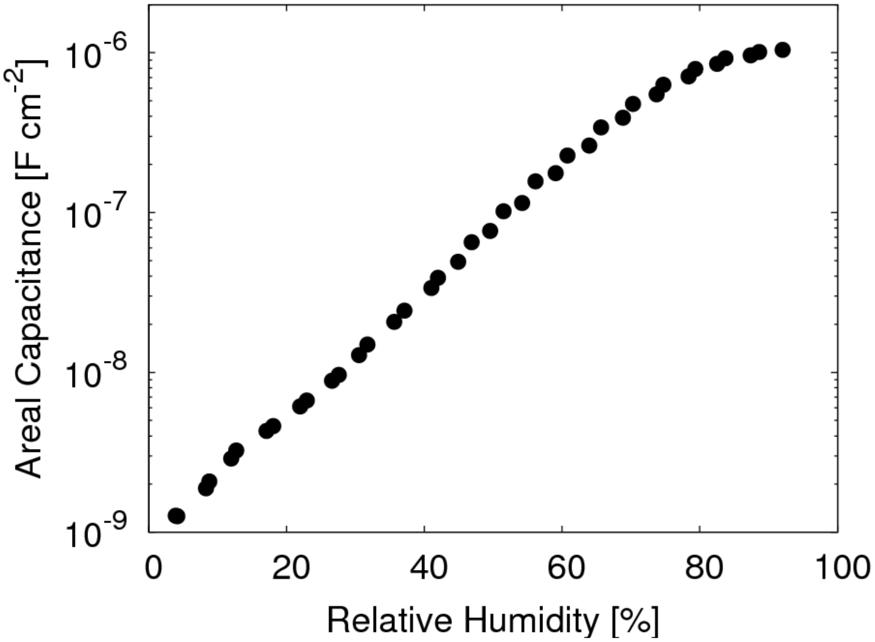


Figure 3.9: Responsivity of a sensor with a 1.5 μm thick TiO_2 sensing layer and 8 μm IDE digit period. Capacitive changes are observed over three orders of magnitude. Image from [16] © 2008 IEEE.

An issue with the IDEs that is compounded as they are reduced in size is shadowing from the electrode boundaries during film growth. This shadowing results in large film irregularities and defects at the electrode boundaries, which reduces the overall surface area and sensitivity. To solve this problem research partner Micralyne, Inc. developed a proprietary process to fabricate electrodes that are countersunk into the substrate so that they maintain a planar surface. Figure 3.8 shows a cross-sectional SEM of a GLAD film deposited on top of a planar countersunk IDE. IDEs with 3 μm digit widths and 3 and 5 μm digit separation were fabricated and tested. It was found that reducing the digit period from the large geometry IDEs increased the sensitivity, and even a small decrease in digit spacing from 5 to 3 μm resulted in significant increase in sensitivity. Figure 3.9 shows the responsivity of a sensor with a 1.5 μm thick TiO_2 sensing layer and 8 μm IDE digit period. Capacitive changes are observed over three orders of magnitude.

3.3.4 Comparison of Different Metal Oxides

Capacitive RH sensors have been fabricated with GLAD films of silica, titania, and alumina on top of IDEs [16]. Figure 3.10 shows x-ray diffraction (XRD) patterns of these films deposited on a silicon substrate. The lack of significant peaks in these patterns shows that films were amorphous. Titania films were found to produce the largest change in capacitance from 0 % to 100 % relative humidity and have the best sensitivity (nF / %RH) (Figure 3.9). Sensor response times for the 1.5 μm thick films are summarized in Table 3.1. Alumina had the best response time for both adsorption and desorption of water.

Table 3.1: Response times for 1.5 μm thick films. Reproduced from [16] © 2008 IEEE.

Material	Adsorption (ms)	Desorption (ms)
TiO_2	275 ± 1	297 ± 2
SiO_2	231 ± 7	229 ± 4
Al_2O_3	87 ± 1	104 ± 1

Figure 3.10 removed due to copyright restrictions.

Figure 3.10: XRD patterns for 1.5 μm thick TiO_2, SiO_2 and Al_2O_3 films deposited at an angle of α = 81° on a Si (100) substrate. Image from [40]

3.3.5 Response Time Effects of Film Porosity and Thickness

The relationships between film thickness and deposition angle and the response curve and response times have been studied [11]. It was found that the response time scales linearly with thicknesses below 4 μm; however, the sensitivity also increases with film thickness because more electric field lines are contained within the film. Response times decrease with increasing deposition angle because the larger column separation allows for quicker water adsorption and desorption. There is also a dependence between sensitivity and deposition angle, with different deposition angles having higher responsivities in different ranges. This means that the film thickness and deposition angle may be tuned differently depending on the sensor application. Empirical models have been developed for the effects of film thickness and deposition angle on response time.

A study by Krause *et al.* examined the macroporosity and mesoporosity of TiO_2 and SiO_2 GLAD films using top-down SEM images and krypton gas adsorption isotherms at 87.3 K [59]. To estimate the macroporosity, SEM images of films

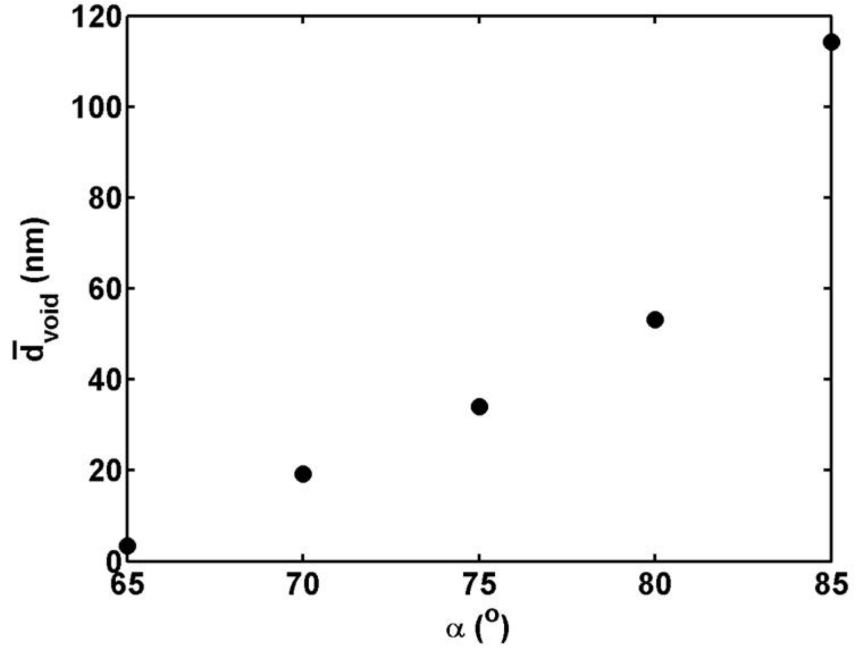


Figure 3.11: The relationship between mean pore diameter and deposition angle α. Reprinted from [59]. Copyright 2011, with permission from Elsevier.

fabricated with α = 65° to 85° were processed by applying thresholding to separate columns from voids. The void distances between columns were then measured and the mean of the distribution calculated (Figure 3.11). The macroporous void distances between the tops of the columns increase with deposition angle. For angles of 70° to 80° the void distance is in the range of tens of nanometers. At 85° the distance is greater than 100 nm. Krypton adsorption isotherms were analyzed for GLAD films deposited with α = 45° to 85° to examine the mesoporosity. Below a deposition angle of 65° mesoporosity dominates, and the pore size distributions were found to be log normal. The peak pore diameter ranges from 2-3 nm at 50° to 4-5 nm at 65°. Above 70° films are dominated by columnar voids but still exhibit mesoporous roughness and porosity.

Pore size is a factor in how fast water can diffuse into the film and thus it affects sensor response time. Knudsen diffusion describes the diffusion that occurs through long, narrow pores 2 - 50 nm in diameter in which collisions with pore walls are frequent. The diffusivity D in pores is determined by the following equation [60]:

$$D = \frac{d_{pore}}{3}\sqrt{\frac{8RT}{\pi M_w}} \tag{3.1}$$

where d_{pore} is the pore diameter, R is the molar gas constant, T is the temperature in Kelvin, and M_w is the molar mass of water. A pore size of 10 nm has a diffusivity of $\approx 2 \times 10^{-6}$ m^{2}s^{-1}. The diffusivity may be combined with the Fickian diffusivity model [61] to obtain the time it takes water to diffuse through the film. The diffusion length is described by:

$$L = 2\sqrt{Dt} \tag{3.2}$$

Rearranging for t,

$$t = \frac{4L^2}{D} \tag{3.3}$$

For a 1.5 μm thick film, the diffusion time through the pores is approximately 285 ns, which is much lower than the reported response times for GLAD IDE sensors. However the model for Fickian diffusion does not take into account molecular interactions with the pore walls. As described in Chapter 2, there are strong surface interactions between TiO_2 and water, so the effective diffusivity of water through the pores is likely reduced. Rapid diffusion of water to the bottom of pores, closest to the IDE, is crucial for sensor performance since electric field lines are the most concentrated closest to the IDE.

The effect of deposition angle and film thickness on sensor response time has been studied [18]. The response time was found to decrease with increasing deposition angle. For films 1.5 μm thick, the response time decreased with a slope of approximately -16 ms degree^{-1}. The response time increased with film thickness by 162 ±4 ms μm^{-1} at a deposition angle of 81°. These two correlations can be combined into the follow relation describing response time as a function of film thickness and deposition angle:

$$\tau(d, \alpha) = (162 \text{ ms } \mu\text{m}^{-1})d - (16 \text{ ms degree}^{-1})(\alpha - 81 \text{ degrees}) \tag{3.4}$$

where τ is the sensor response time, d is the film thickness, and α is the deposition angle. This relation does not take into account the possibility of any cross dependence. The decrease in response time with increasing deposition angle can be related to the increase in pore width that occurs with alpha, as described above. This is predicted by the Knudsen diffusion model in Equation 3.1. According to Equation 3.3 the response time should be related to the square of the film thickness if Fickian diffusion dominates with respect to the pore length, however this is not the case as a linear correlation is observed. This suggests that the response time is instead dominated by surface interactions rather than diffusion since surface area roughly scales with film thickness to first order.

Kupsta *et al.* have developed an etching procedure for the films with the aim of decreasing the response time by increasing the effective diffusivity of water vapour by smoothing the columns [19]. TiO_2 films were etched using a CF_4 etch gas along with a polytetrafluoroethylene (PTFE) spacer to absorb free flourine radicals in the plasma so that the TiO_2 columns don't clump together. The etch process was able to reduce the adsorption response time of the sensors from 150 ms to 50 ms, and although initial desorption rates were improved, long desorption tails still remained. The desorption tails could be because of nodular defect columns that grow at the edges of the electrodes. These columns have different volume and surface properties than the rest of the film and likely respond differently to humidity than the rest of the column population. The etching process also introduced large hysteresis in the sensor response. Applying a 48-hour UV treatment significantly reduces the hysteresis introduced by the etch, but does not affect the improvements in response time.

3.3.6 Sensor Ageing

Ageing is observed in GLAD RH sensors over time. A decrease in the response and hysteresis in the response curve of aged sensors is observed (Figure 3.12). It has been found that ageing in titania films can be reversed by treating the sensors with UV light [28]. Sensors that have received a UV treatment exhibit a capacitance change over a wider range than untreated sensors and also have an increased sensitivity at lower humidities (Figure 3.13). The effects of this treatment are reproducible. It is possible that organic contaminants or a decrease in pore distribu-

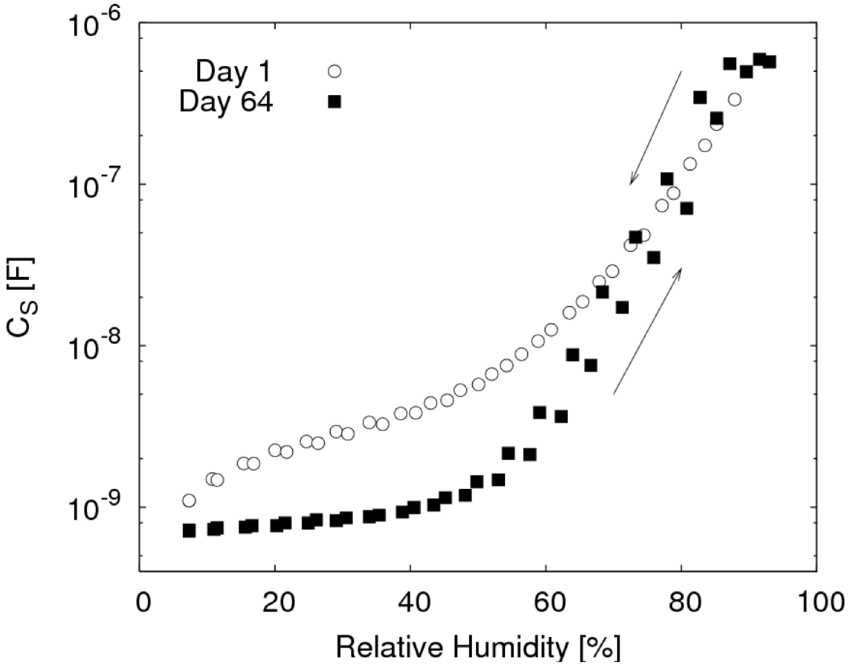


Figure 3.12: Hysteresis and decreased responsivity in an aged sensor. Reprinted from [28]. Copyright 2008, with permission from Elsevier.

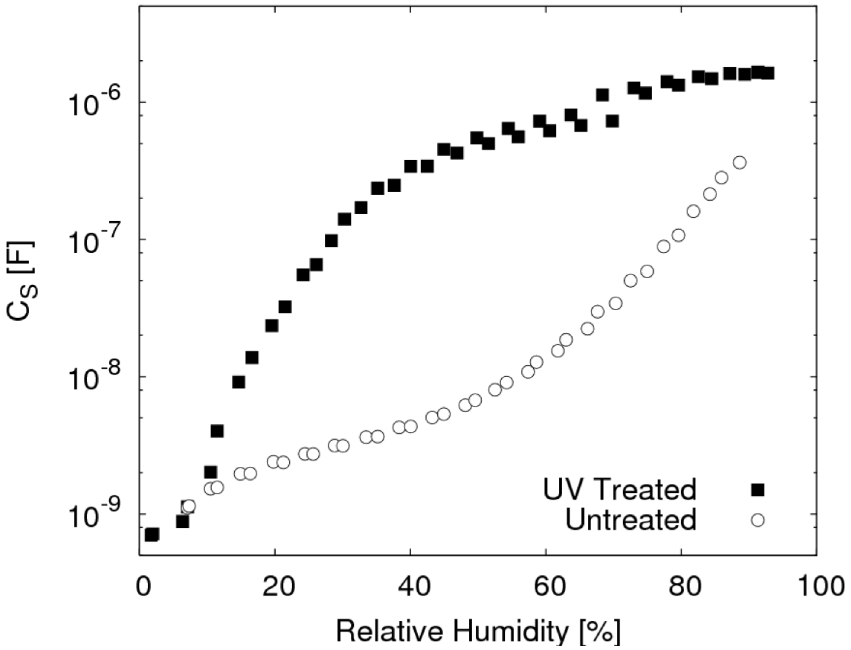


Figure 3.13: Response curves for UV treated and untreated sensors. Reprinted from [28]. Copyright 2008, with permission from Elsevier.

tion or number of sites is responsible for the ageing in the sensors. However the physical and chemical mechanisms behind the ageing and UV treatment are largely unknown.

3.4 Conclusion

GLAD is a physical vapour deposition process that allows for fine control over the morphology of deposited porous thin films. Capacitive relative humidity sensors have been fabricated with GLAD films on top of countersunk IDEs. Because of the high porosity of the films the sensors exhibit extremely fast response times and show capacitive changes over three orders of magnitude. Titania, silica, and alumina films have been studied, with titania films found to produce the largest change in capacitance and have the highest sensitivity. The dependence of response time on film thickness and deposition angle has been characterized, with a linear relationship between response time and thickness established. The relationships between sensor response and sensitivity and film thickness and deposition angle have also been characterized. Based on this research, film thickness and deposition angle may be tuned and optimized depending on the sensor application. The effects of pore sizes in the films on response time have been examined, and it is possible to improve response time by using a reactive ion etch treatment to smooth the columns.

Currently the largest deficiency of GLAD RH sensors is ageing over time and the associated drift in the sensor response. It has been found that a UV treatment can reverse the ageing. The research described in Chapter 5 and Chapter 6 aims to characterize the effects of UV treatment on sensor performance and long term stability. Initial experiments have used a mercury vapour lamp as the UV source, however compact UV LEDs have recently become commercially available. The goal is to combine UV LEDs with capacitive humidity sensors to create a device with improved long term stability. The available UV LED technology is discussed next in Chapter 4.

Chapter 4

UV LED Technology

4.1 Introduction

Deep ultraviolet (UV) light emitting diodes (LEDs) offer the possibility of replacing mercury vapour lamps in a variety of applications such as anti-microbial treatments, sterilization, decontamination, DNA decomposition and fluorescence biological agent detection. Mercury vapour lamps are large, bulky, power hungry, environmentally unfriendly and require expensive optical filters to isolate individual emission lines at specific wavelengths. UV LEDs are small, compact and span a wide range of UV wavelengths.

UV LEDs are a very young technology compared to other visible and infrared LEDs, which have been available for decades. The very first 285 nm UV LED was announced in 2002 [62] followed by the 265 nm UV LED in 2003 [63]. Development was slow because there are many materials issues with group III-nitride semiconductors, which are the most suitable material for UV light emitting devices and have a bandgap range that covers wavelengths down to 200 nm. Developments of growth techniques such as metalorganic vapour phase epitaxial growth (MOVPE) [64] and migration-enhanced metalorganic chemical vapour deposition (MEMOCVD) [65] have allowed the fabrication of these devices. Although UV LED output efficiencies are quite low compared to other more common wavelengths, UV LED efficiencies have been steadily increasing while prices have been decreasing.

4.2 Materials Physics in UV Devices

4.2.1 UV LED Semiconductor Materials

The UV spectrum spans wavelengths from 10 nm to 400 nm, corresponding to bandgap energies from 124 eV to 3.1 eV. Fabrication of UV light-emitting devices requires materials with direct bandgaps greater than 3.1 eV. Si and Ge are both very important materials to the semiconductor industry, but have indirect bandgaps that are too small, making them unsuitable for UV light-emitting devices. Direct bandgap materials are required for efficient light-emitting devices.

Table 4.1: III-V binary semiconductor bandgap energies, wavelengths and type (D = direct, I = indirect) at T = 300 K. Reproduced from [66].

Material	Bandgap Energy E_g (eV)	Bandgap Wavelength λ_g (μm)	Type
Ge	0.66	1.88	I
Si	1.11	1.15	I
AlN	6.2	0.2	D
AlP	2.45	0.52	I
AlAs	2.16	0.57	I
AlSb	1.58	0.75	I
GaN	3.39	0.365	D
GaP	2.26	0.55	I
GaAs	1.42	0.87	D
GaSb	0.73	1.70	D
InN	0.7	1.77	D
InP	1.35	0.92	D
InAs	0.36	3.5	D
InSb	0.17	7.3	D

Group III-V semiconductors combine group III elements (Al, Ga, In) with group V elements (N, P, As, Sb). Table 4.1 summarizes the bandgap energies, wavelengths and type of group III-V binary semiconductors. Group III-Nitride materials stand out from the group because they are direct-bandgap semiconductors and cover a wide range, from 0.7 eV for InN to 6.2 eV for AlN [67].

Bandgaps may be shifted with ternary and quaternary alloys of these materials. This involves replacing some group III atoms in the lattice with one or two different group III atoms. The crystal structure is still the same, however the elements have

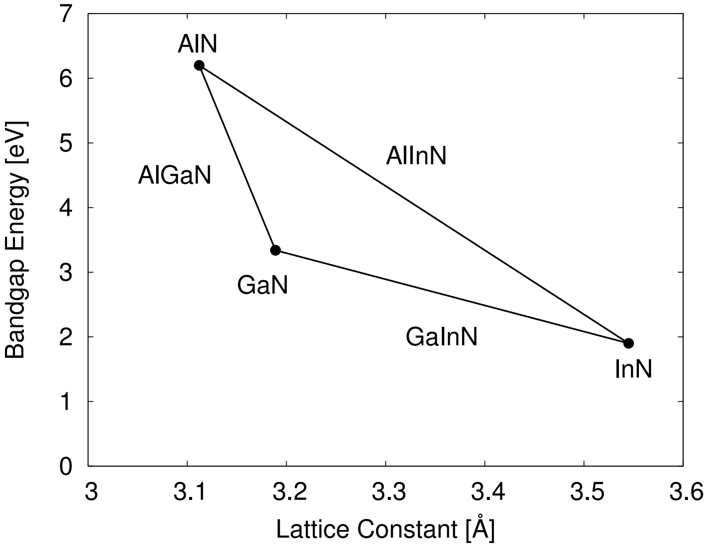


Figure 4.1: Bandgap versus lattice constant of GaN, AlN, InN and their alloys. Values from [68].

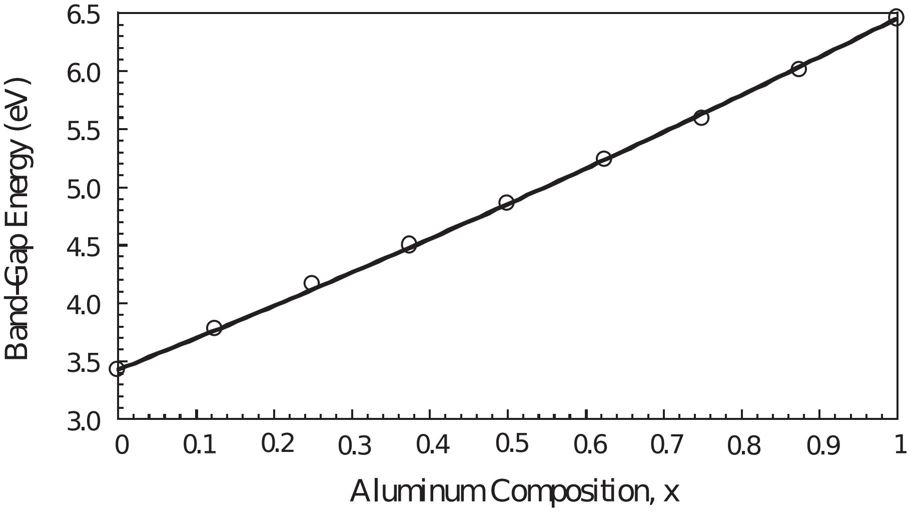


Figure 4.2: Change of bandgap energy of $Al_xGa_{1-x}N$ with respect to aluminum composition. Image from [69]. Copyright 2002 The Japan Society of Applied Physics.

different electronic structures and band energies so the material bandgap and lattice constant properties are modified. Figure 4.1 shows the relationship between bandgap energies and lattice constant for changing alloys of group III-nitrides. $Al_xGa_{1-x}N$ is a ternary alloy and $Al_xIn_yGa_{1-x-y}N$ is a quaternary alloy with the x and y subscripts denoting the elemental composition. Changing the elemental composition of Al in $Al_xGa_{1-x}N$ can create bandgaps spanning the range between those of GaN (3.4 eV) and AlN (6.2 eV) (Figure 4.2). Figure 4.3 shows the optical characterization of a range of $Al_xGa_{1-x}N$ LEDs. $Al_xIn_yGa_{1-x-y}N$ alloys are sometimes used to lattice-match to GaN substrates. Different lattice constants between the substrate and epitaxially grown layers puts strain on the crystal lattice. This results in polarization fields in the crystal, which reduce the overall optical emission of the device [70, 71]

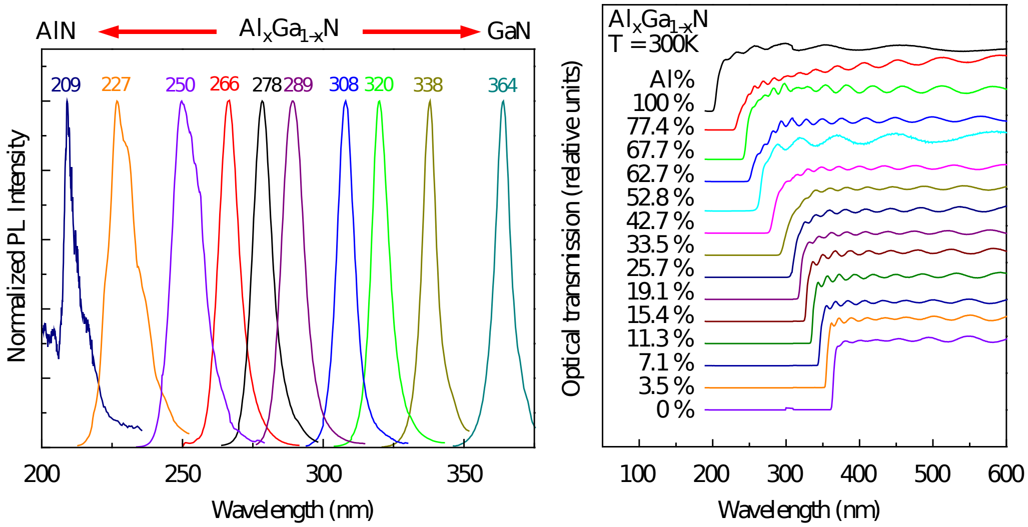


Figure 4.3: Optical characterization of $Al_xGa_{1-x}N$ material with different compositional levels of Al. Left shows the photoluminescence and right shows the optical transmission (which shows fundamental absorption edges). Reprinted from [72]. Copyright 2009, with permission from Elsevier.

4.2.2 Quantum Well Structures

A heterojunction is formed when two crystalline semiconductors with different band structures are interfaced. An ideal heterojunction creates an abrupt discontinuity in the material energy bandgap at the interface. Quantum wells (QWs) are important structures for the active regions of LEDs and consist of alternating series of thin-layer heterojunctions. A multiple quantum well structure consists of n QW layers (GaN) of width d_{QW} and (n + 1) barrier layers (AlGaN) of width $d_{barrier}$ [71]. The quantum wells confine electrons and holes within the layer, effectively confining them to 2-dimensional motion rather than 3-dimensional. This increases photon-emitting electron-hole recombination and increases the internal quantum efficiency of the device.

4.3 UV LED Devices

4.3.1 Typical UV LED structure

Figure 4.4 shows the structure of an AlGaN UV LED device [73]. The structure is the same for all ultraviolet wavelengths, with the Al composition in some layers determining the wavelength. It is fabricated on a sapphire substrate and employs an AlInGaN multiple quantum well active region. Most UV LED devices have very similar structures.

Sapphire substrates are typically used for UV LED devices. This requires a buffer or nucleation layer, which is Al for deep UV devices and may be GaN for near UV devices. Many devices use an AlN/AlGaN superlattice layer next to relieve strain in the lattice to reduce piezoelectric polarization fields and to maintain the bandgaps of the materials. On top of this a thick conducting n^+-AlGaN layer doped with Si is grown and acts as the n-contact, sometimes followed by a more lightly doped n-AlGaN layer. Most devices have a single AlGaN or AlInGaN quantum well or multiple quantum well active region. A Mg-doped p-type AlGaN current blocking layer prevents electrons from overflowing out of the active region [74]. This layer has an Al composition about 5-10% higher than the barrier layer in the quantum well [75]. The entire structure is capped off with a graded p-type AlInGaN [73] or GaN [72, 75]) ohmic p-contact layer.

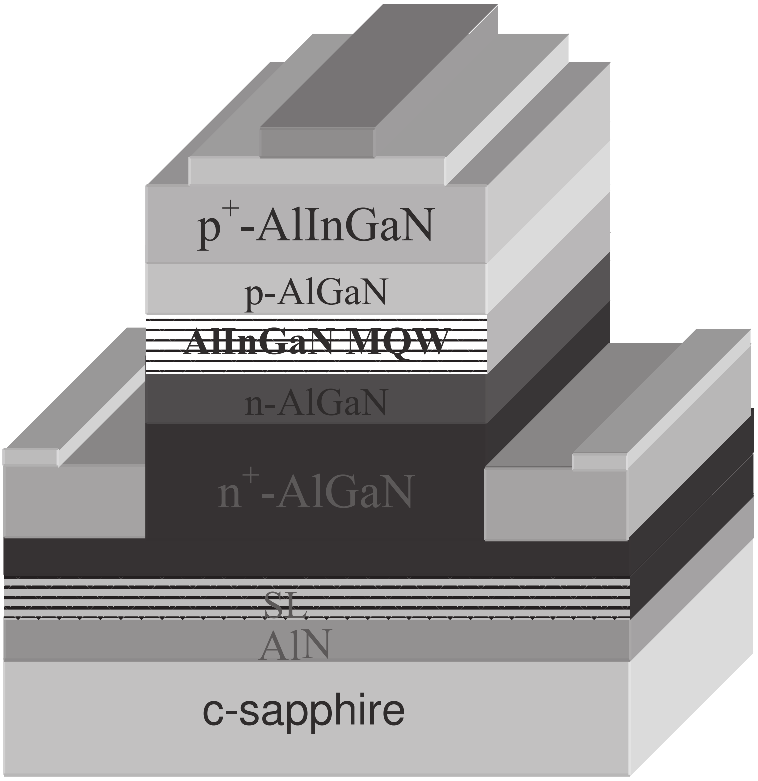


Figure 4.4: Structure of an AlGaN UV LED device. Image from [73]. Copyright 2005 The Japan Society of Applied Physics.

4.3.2 Device Performance

UV LEDs have been built spanning a range of wavelengths from 210 nm - 365 nm (Figure 4.5). The output powers follow a general trend of decreasing output power with decreasing wavelength. Below 250 nm LED powers drop off considerably. For example, a 210 nm LED from the University of South Carolina has an output power of only 0.02 μW, compared to powers of 3.3 mW at 273 nm and 8.4 mW at 346 nm from RIKEN [76].

LEDs are usually operated in two modes: Continuous-wave (CW) mode where a DC current is applied to the device, and pulsed mode where a larger DC current is pulsed through the device. 20 mA is a common current for reporting the CW performance of devices. Pulsed currents allow for larger currents to be passed through the device, usually at a duty cycle of 1-2 %, and for greater output powers to be obtained. Figure 4.6 shows the CW power versus current performance for sev-

Figure 4.5 removed due to copyright restrictions.

Figure 4.5: Output power of AlGaN based deep UV LEDs. Copyright 2010 Wiley. Used with permission from [76]

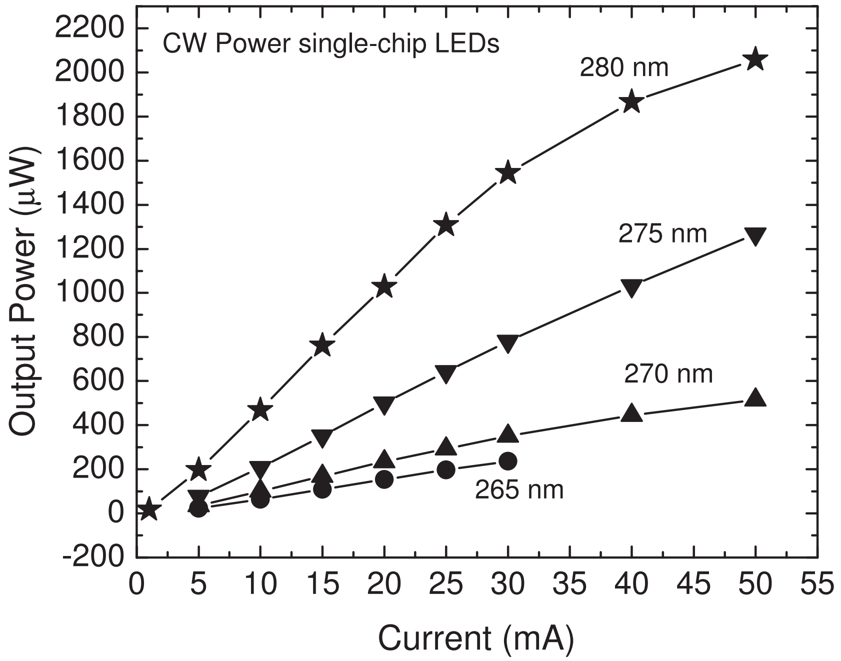


Figure 4.6: (a) CW power-current plot for 265, 270, 275, and 280 nm single-chip packaged LEDs. Image from [73]. Copyright 2005 The Japan Society of Applied Physics.

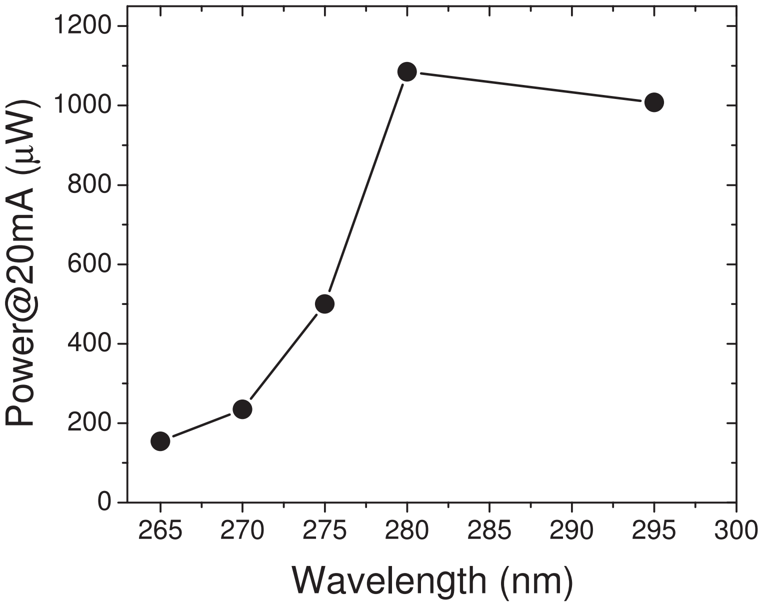


Figure 4.7: Plot of power at 20 mA DC vs. emission wavelength. Image from [73]. Copyright 2005 The Japan Society of Applied Physics.

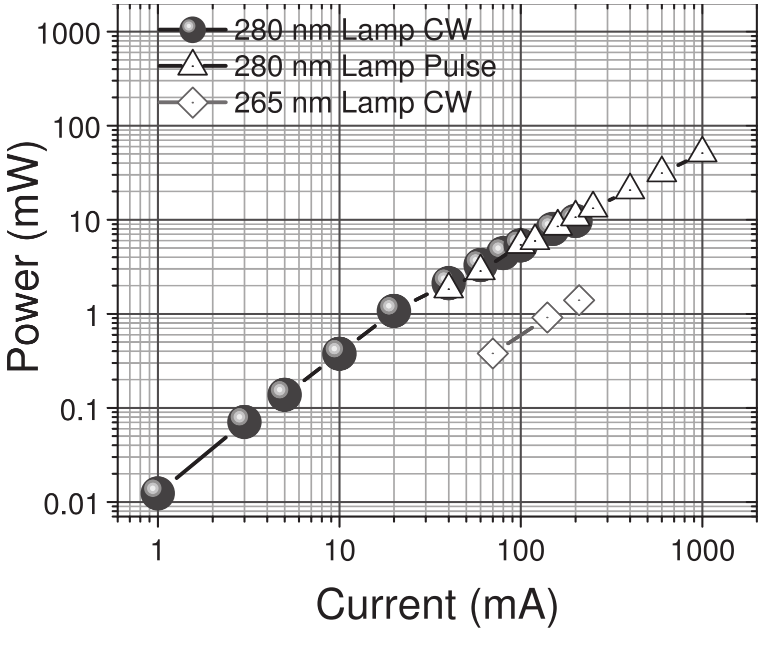


Figure 4.8: Power-current plot for 265 and 280 nm UV lamps consisting of three chips connected in parallel. A duty cycle of 2 % at 10 kHz was used for the lamp pulse. Solid circles represent 280 nm CW powers, open triangles represent 280 nm pulse powers, and open diamonds represent 265 nm CW powers. Image from [73]. Copyright 2005 The Japan Society of Applied Physics.

eral wavelengths of devices from Sensor Electronic Technology Inc. Note that the power-current relationship of the LEDs is linear. Figure 4.7 shows the relationship between power and wavelength at 20 mA. This plot shows that output powers start to decrease considerably below 280 nm.

Figure 4.8 shows the output power for 280 nm and 265 nm UV lamps from Sensor Electronic Technologies Inc. consisting of three LED chips connected in parallel. The 280 nm lamp is operated in both CW and pulsed modes. Pulsed operation at currents over 50 mA at 10 kHz and a duty cycle of 2% allows for an output power over 56 mW to be obtained [73].

Figure 4.9 shows that there is little drift in the emission peak position of the LEDs during operation, only increasing significantly at higher DC currents, which tend to damage the LED. The spectral width, or full-width-half-maximum (FWHM) value, stayed below 10 nm for both modes of operation.

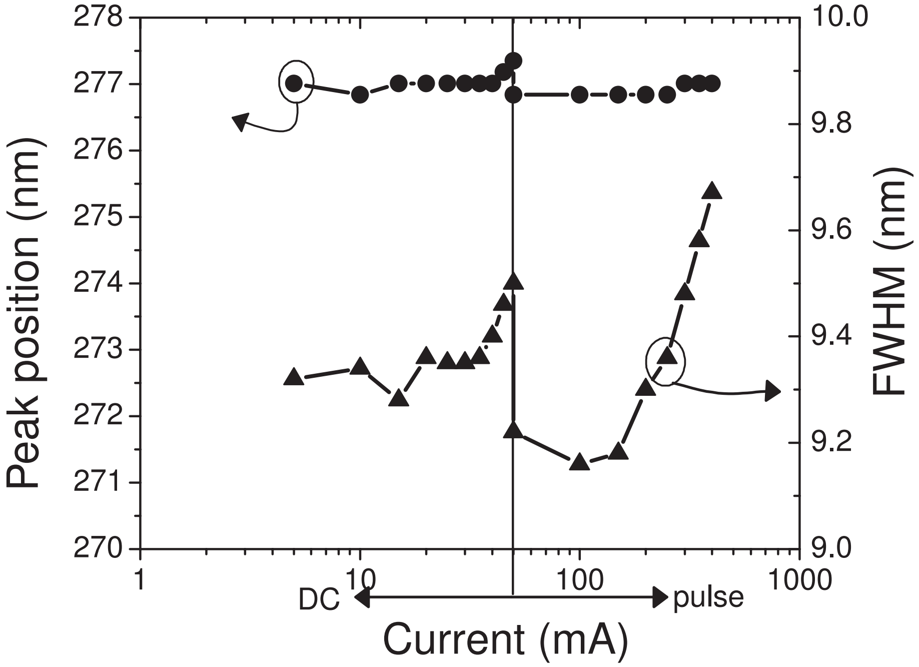


Figure 4.9: Emission peak position and spectral width (FWHM) vs. current. Image from [73]. Copyright 2005 The Japan Society of Applied Physics.

4.4 Conclusion

Advancements in techniques such as MOVPE and MEMOCVD for growing high quality AlInGaN based layers have led to the fabrication of deep-UV LEDs. Other design techniques such as superlattice growth and the growth of high-quality AlN buffer layers on sapphire substrates have mitigated some of the issues with using group III nitride materials. LEDs with emission wavelengths ranging from 210nm - 365nm have been fabricated and a very wide range of wavelengths are now commercially available. The price of UV LEDs continues to decrease and the output power increases as processes and techniques for fabrication are improved. UV LEDs are an excellent low-power replacement for bulky mercury vapour lamps and their use for UV treatment of GLAD RH sensors is demonstrated in Chapter 5. The research presented in Chapter 6 characterizes different UV LED wavelengths for sensor treatment.

Chapter 5

UV Regeneration Characterization [1]

5.1 Introduction

Chapter 2 explained how metal oxide humidity sensors can drift (age) over time because of surface hydration from continued exposure to humidity and from the buildup of organic contaminants. Chapter 3 discussed how the sensitivity and performance of metal oxide GLAD humidity sensors was observed to decrease over time. It was seen that treating titanium dioxide (TiO_2) GLAD humidity sensors with ultraviolet light reverses this ageing process and enhances the sensor response [28]. This may be related to the well-known photoactivity of titanium dioxide under UV irradiation. Because of this effect, titanium dioxide is used in a wide range of applications such as water purification, air cleaning, anti-fogging coatings, and self-cleaning surfaces [26, 27]. Photocatalytic activity in GLAD titanium dioxide films has been demonstrated and studied by many groups [77, 78]. Recently it was found that titanium dioxide surfaces become hydrophilic under UV irradiation [79, 80], which may also affect the performance of humidity sensors.

The research presented in this chapter investigates the effect of UV irradiation wavelength with a mercury vapour lamp on sensor performance and ageing. The regenerative effect is also demonstrated using commercially available UV light emitting diodes (LEDs). The objective is to establish a baseline wavelength characterization of the UV treatment.

[1] Part of this chapter has been published in: D.P. Smetaniuk, M.T. Taschuk, and M.J. Brett. ”Photocatalytic titanium dioxide nanostructures for self-regenerating relative humidity”, *IEEE Sensors Journal*, 11(8):1713-1719, August 2011. [29]

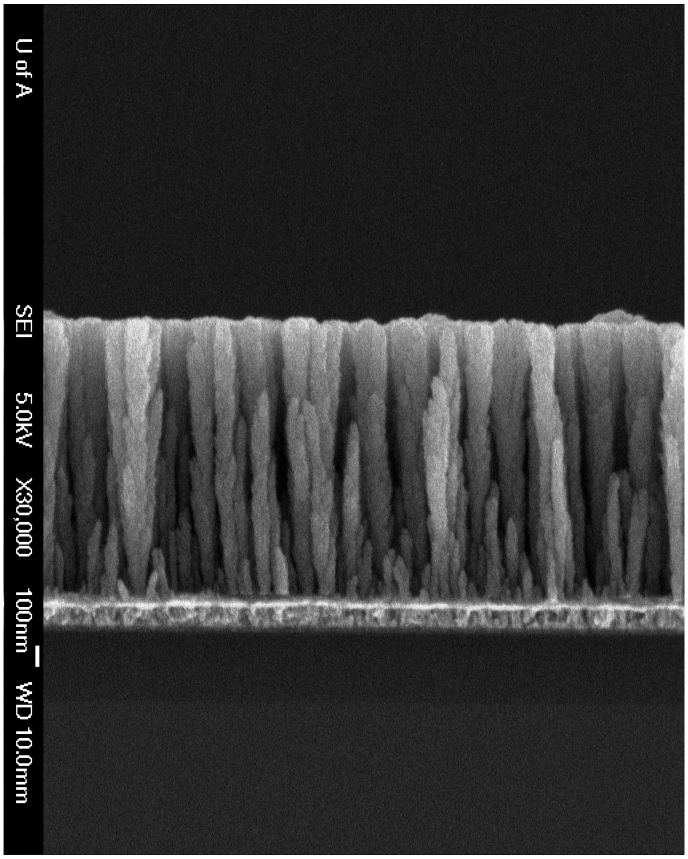


Figure 5.1: SEM image of a 1.5 μm thick TiO_2 film deposited at α = 80° on top of an electrode.

5.2 Experimental Setup

The sensors studied consist of interdigitated electrodes (IDEs) countersunk into a substrate (Micralyne, Inc.) with GLAD films deposited on top [17]. The tops of the countersunk IDEs were planar with the substrate to prevent shadowing during film deposition. The GLAD films were TiO_2 vertical posts 1.5 μm tall and several hundred nanometers in diameter. Rutile TiO_2 (99.9 %, Cerac, Inc.) was used as a source material. A film thickness of 1.5 μm was chosen so that the results of the study can be compared to other studies using the same thickness. It also provides a reasonable balance between a high sensitivity and a fast response time. An incident vapour flux angle α = 80° was chosen because it provides a good balance between fast response time, and good sensitivity at lower humidities. The film parameters may be tuned for a specific application, but for the purposes of this study were held constant so that results may be directly compared. The films were deposited at a vapour flux deposition rate of 0.5 nm s^{-1} between pressures of 6×10^{-5} Torr and 8×10^{-5} Torr using an electron beam evaporation system (Kurt J. Lesker, AXXIS). To grow a more stoichiometric film oxygen was added during deposition at a flow rate of 2.2 sccm - 3.6 sccm using a mass flow controller (MFC) (MKS 1179A). After deposition the samples were thermally oxidized in atmosphere at 100°C for 24 hours. Figure 5.1 shows an SEM image of the film on an electrode. A planar 1.0 μm thick TiO_2 film was deposited on a Si substrate by depositing at α = 30° for water contact angle measurements.

The sensor impedance response to humidity was tested using a custom humidity chamber [40], shown schematically in Figure 5.2, able to achieve relative humidities between 2 %RH and 95 %RH. Wet air was created by bubbling dry nitrogen through a water reservoir. Two mass flow controllers (MKS M100B) mix ratios of dry nitrogen and the wet air by controlling the flow rates in order to achieve the range of humidities. Flow rates vary between 0 - 500 standard cubic centimeters per minute (sccm). The air mixture flows into a sealed chamber containing the humidity sensor being tested and a commercial humidity probe (Vaisala HMP100). The leads for the sensor and the humidity probe are passed through a small hole in the top of the chamber to be accessible outside. The humidity in the chamber was measured using the commercial probe and the impedance of the sensors was measured using an LCR meter (Quadtech 1715 Digibridge).

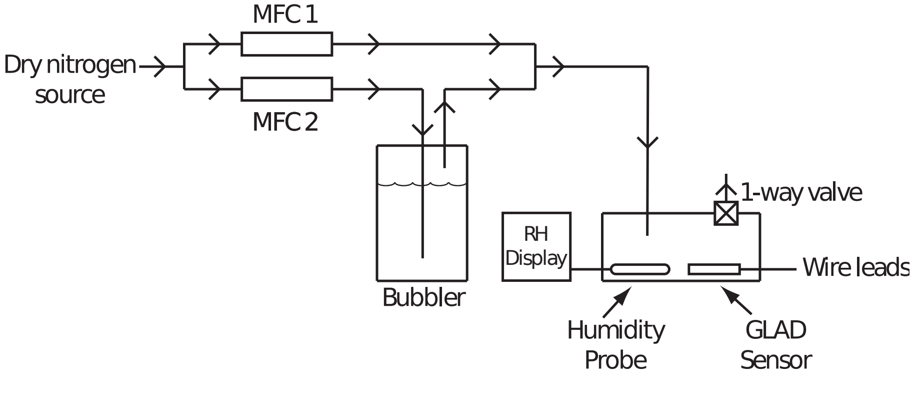


Figure 5.2: Relative humidity testing chamber setup. Image from [16] © 2008 IEEE.

Calibration of the commercial humidity probe and LCR meter were done previously and are described in detail in [40]. The commercial humidity probe was calibrated using saturated salt solutions of LiCl, $MgCl_2$, NaCl, and K_2SO_4 [81,82]. A large number of capacitors and resistors from different vendors were used to calibrate the LCR meter. The series capacitance, impedance magnitude and phase of the sensors was recorded at 1 V_{rms} and at frequencies ranging logarithmically from 20 Hz to 1 MHz. The capacitance measurement error of the LCR meter was found to be 2.4% (95% confidence level). The measurement error of the commercial humidity probe depends on relative humidity and is given by (95% confidence level):

$$\epsilon_{RH}^2 = (1.7 \times 10^{-4})\mathrm{RH}^2 + (3.5 \times 10^{-4})\mathrm{RH} + 1.04 \qquad (5.1)$$

where ϵ_{RH} is the percent RH error in the probe reading, and RH is the nominal relative humidity in percent.

The water contact angles on the planar films were recorded with a Contact Angle Measurement System (First Ten Angstroms, Inc.). For each UV wavelength a single sample was used for measurements before and after UV treatment.

An 8 W mercury vapour lamp (UVP MRL-58) was used for UV treatment of the sensors. The lamp emission spectra is shown in [28]. A set of optical bandpass filters (Melles Griot) centered at 254 nm (F1M016), 289 nm (F1M022), 313 nm (F1M024), 365 nm (F1M028), and 360 ± 50 nm (Schott Glass UG3) was used to isolate individual emission lines of the lamp to expose sensors to a specific wavelength or wavelength range. Sensor A was treated with 254 nm, Sensor B with 289 nm, Sensor C with 360 ± 50 nm (wideband), Sensor D with 313 nm, and Sensor E with 365 nm. The sensors were placed under holes approximately 2 cm × 2 cm in a sheet of black corrugated plastic. An optical filter covered each hole and the entire arrangement was placed under the mercury vapour lamp. The tube of the lamp is approximately 25 cm long and was situated 10 cm above the samples. Initial UV treatment tests were conducted using UV LEDs (UVTOP295, UVTOP310, Sensor Electronic Technology Inc.) with peak emissions centered at wavelengths of 295 nm (12 nm FWHM) and 310 nm (10 nm FWHM). The 295 nm and 310 nm LEDs both have total output powers of 500 μW at a diode current of 20 mA. The samples were placed 1.5 cm away from the UV LEDs, producing an irradiance of approximately 500 μW cm^{-2} at the surface. All treatments were conducted within an enclosed steel storage cabinet for 48 hours. When the samples were not being treated they were stored in a separate steel cabinet. Relative humidity scans of the sensors were conducted before and after UV treatment and compared.

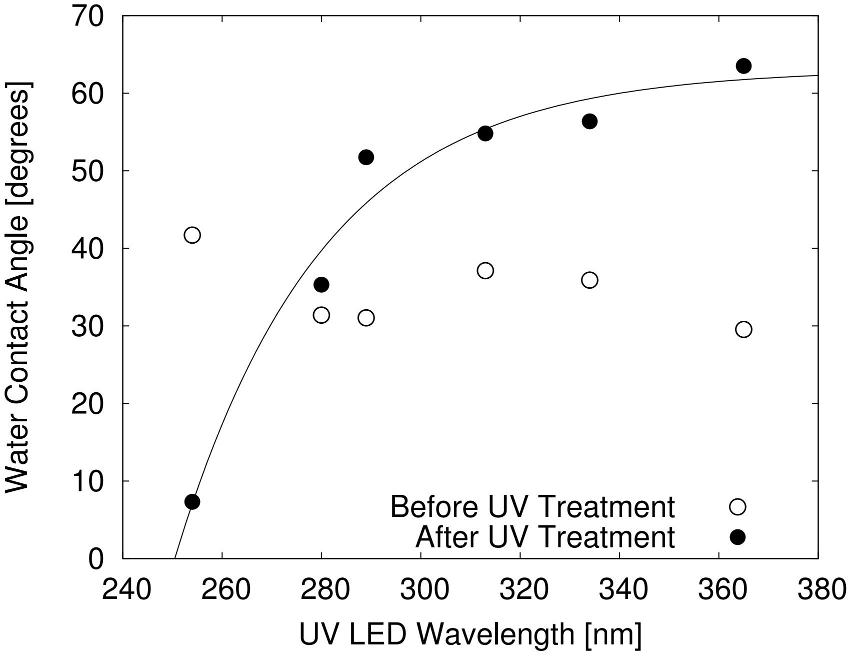


Figure 5.3: Water contact angle of planar films before and after UV treatment. The trend line serves as a guide for the eye.

5.3 Results

A difference was observed in water contact angles on the planar films before and after UV treatment (Figure 5.3). Smaller water contact angles correspond to a greater hydrophilicity of the film. The contact angles before UV treatment ranged from 30° to 42°. The contact angle was greater after UV treatment for all wavelengths except for 254 nm which saw a large decrease in the contact angle. The contact angles after UV treatment decreased slightly with decreasing wavelength down to about 300 nm after which a more significant decrease was seen down to 254 nm.

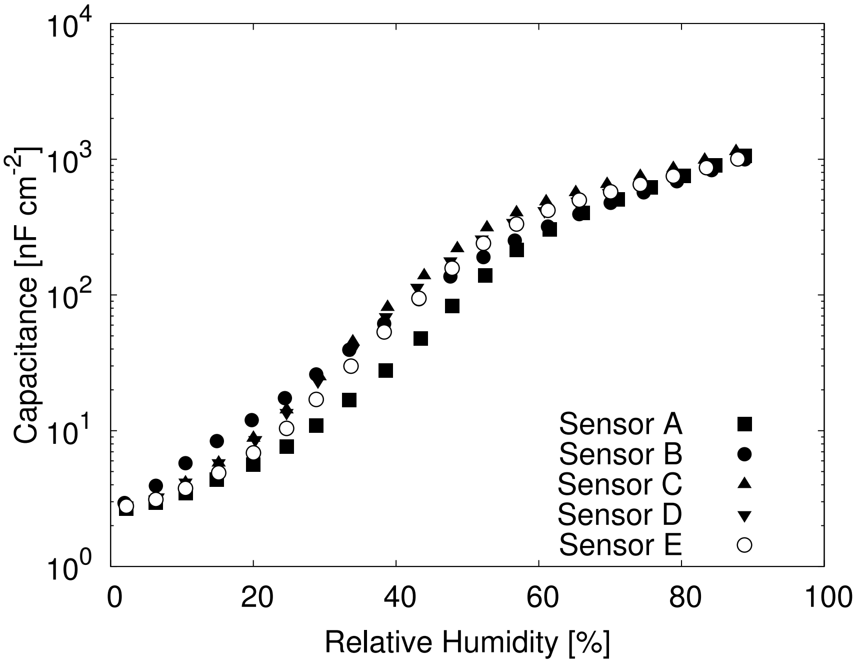


Figure 5.4: Sensor responses before UV treatment showing device reproducibility.

New, as-deposited sensors show little variability in the response curves (Figure 5.4.) After UV treatment with the mercury lamp, a spread in the response curves is observed (Figure 5.5.) The change in responsivity is larger for smaller wavelengths. The steepest region of the responsivity curve is found at lower relative humidities with lower UV wavelengths. The UV treated sensors were then left to age between 87 and 96 days. After ageing the steep region of the response curve shifts towards higher relative humidities as seen in Figure 5.6. The relative spread between the curves after ageing is similar to the spread before ageing. Even after 90 days of ageing the response of the sensor treated with 254 nm light is still above the response of the original sensor before treatment.

The sensitivity of a sensor is defined as the change in areal capacitance with a change in relative humidity in units of nF cm^{-2} %RH^{-1}. Figure 5.7a shows the change in the sensitivity curves after UV treatment for all wavelengths. The sensitivity curves of the sensors have a constant high sensitivity region at greater humidities that saturates at a value of approximately 20 nF cm^{-2} %RH^{-1}. This region is extended toward lower humidities with decreasing treatment wavelengths. In Figure 5.7b after 87-96 days ageing the high sensitivity region shifts to higher

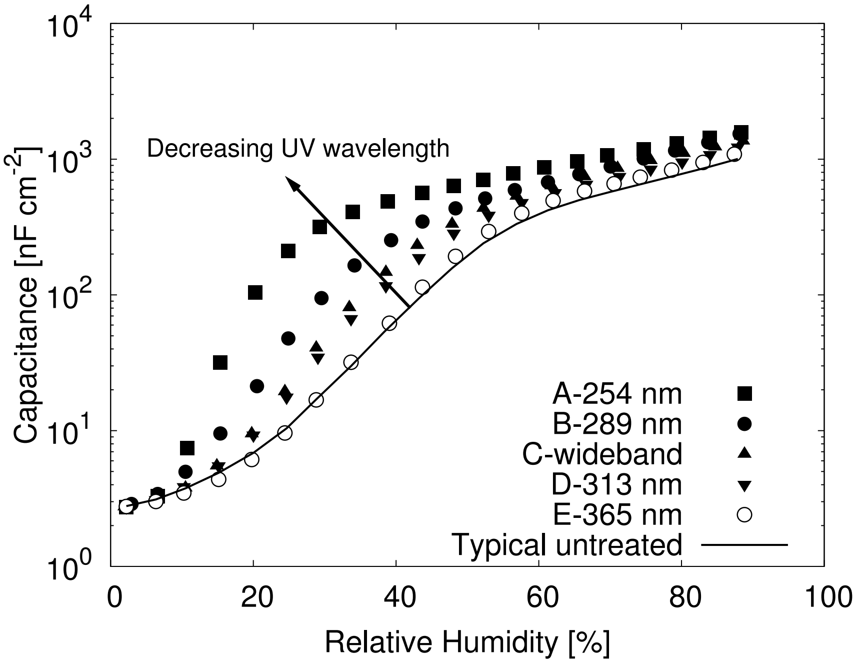


Figure 5.5: Sensor responses after 48 hour UV treatment showing that lower wavelengths create a larger change.

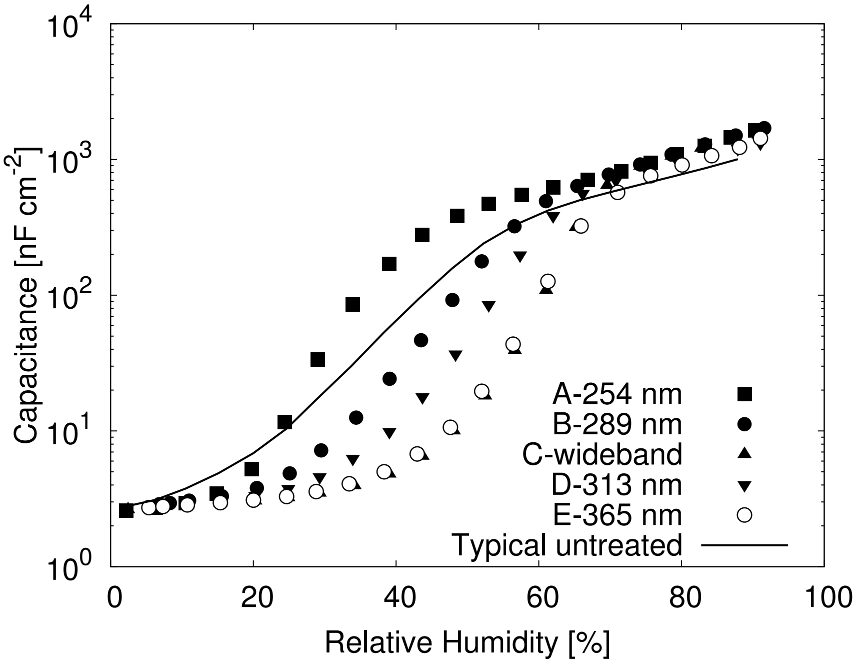


Figure 5.6: Sensor responses after 87-96 days ageing. The 254 nm response still exceeds the original untreated sensor.

humidities and the sensitivity at lower humidities is decreased.

Figure 5.8 shows the changes in the responsivity curves before and after treatment with the 295 nm and 310 nm UV LEDs . The largest change in the responsivity was observed below 50 %RH for both cases.

A figure of merit (FOM) was defined in order to quantify the change in responsivity of the sensors after UV treatment and ageing:

$$FOM = \int_{5}^{85} (\log_{10} C_x[\%RH] - \log_{10} C_0) dRH \tag{5.2}$$

where C_0 is the reference capacitive response curve of the original as-deposited sensor before UV treatment or ageing and $C_x[\%RH]$ is the capacitive response curve after UV treatment or ageing. This is shown as the shaded area in Figure 5.9. The area is integrated between 5 %RH and 85 %RH because humidities below 5 %RH and above 85 %RH are sometimes difficult to reach with the humidity setup. A larger FOM corresponds to a larger change in the response curves of the sensors. Aged sensors in which the response curve is below the reference correspond to a negative FOM. A positive FOM corresponds to an improved sensor response and a negative FOM a decreased response with respect to the original sensor response.

The FOM as a function of illumination wavelength is shown in Figure 5.10 for sensors treated with the mercury lamp and two sensors treated with UV LEDs. With the exclusion of the UV LED treated sensors, an exponentially decreasing FOM with illumination wavelength is observed. The FOM values range from 2.1 for 365 nm to 61 for 254 nm. Treatment with the UV LEDs resulted in significantly larger FOMs than similar mercury lamp wavelengths. Figure 5.11 shows FOMs normalized by dividing by the power delivered at each wavelength. The resulting trend of the normalized FOM peaks around 275 nm and drops off sharply below 260 nm. The normalized FOMs for UV LED treated sensors are significantly lower than the trend due to the high power of the LEDs. The relation between FOM and illumination power is shown in Figure 5.12. A high power regime around 500 μW produced the largest FOMs. The 254 nm mercury lamp line and UV LED treatments fall in this regime. The power delivered by these was 2-3 orders of magnitude greater than all other treatments. In the low power regime below 10

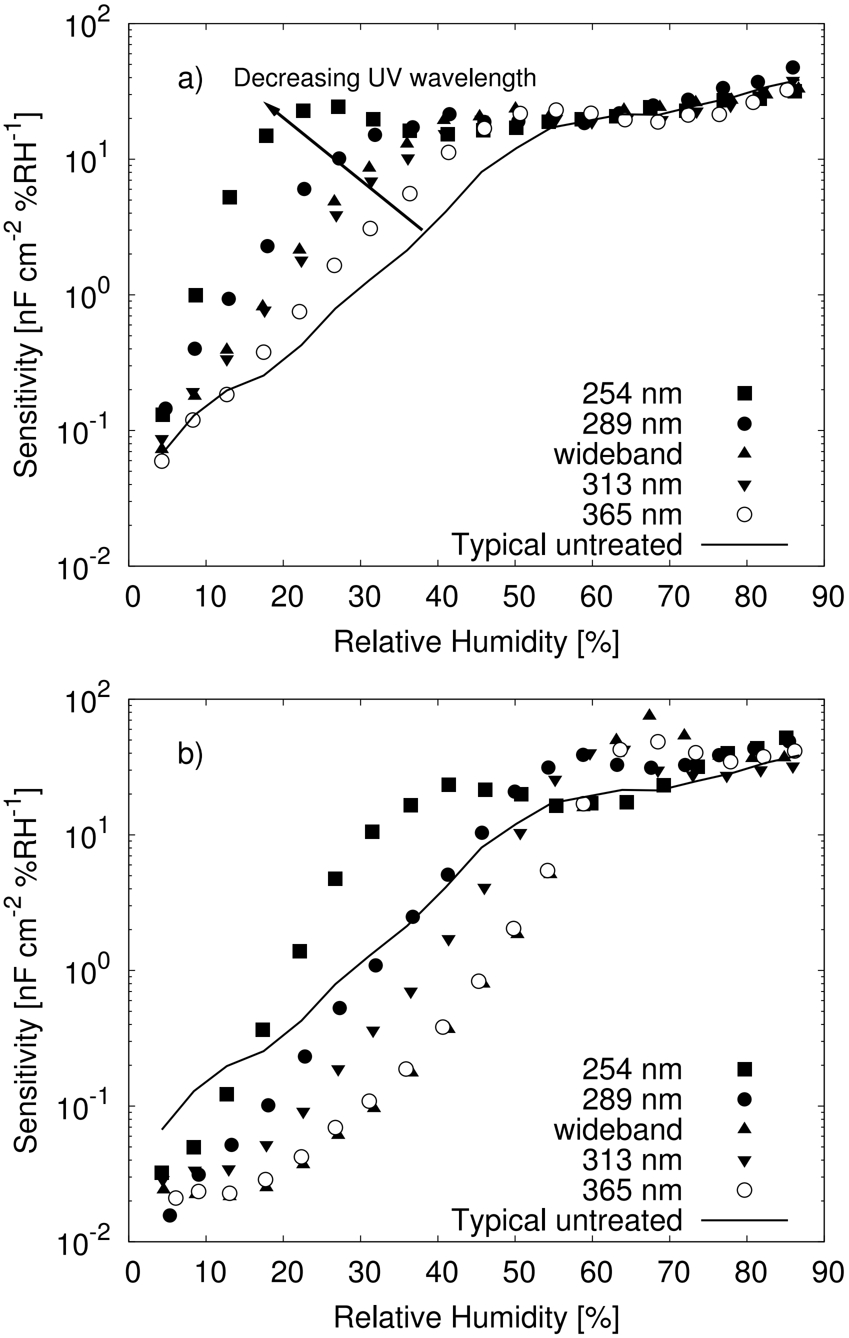


Figure 5.7: Sensor sensitivity (a) after UV treatment and (b) after 87-96 days ageing. The UV treatment extends the high sensitivity range of the sensors, with a wider high sensitivity range obtained for lower wavelengths. The high sensitivity performance saturates at a value of approximately 20 nF cm^{-2} %RH^{-1}.

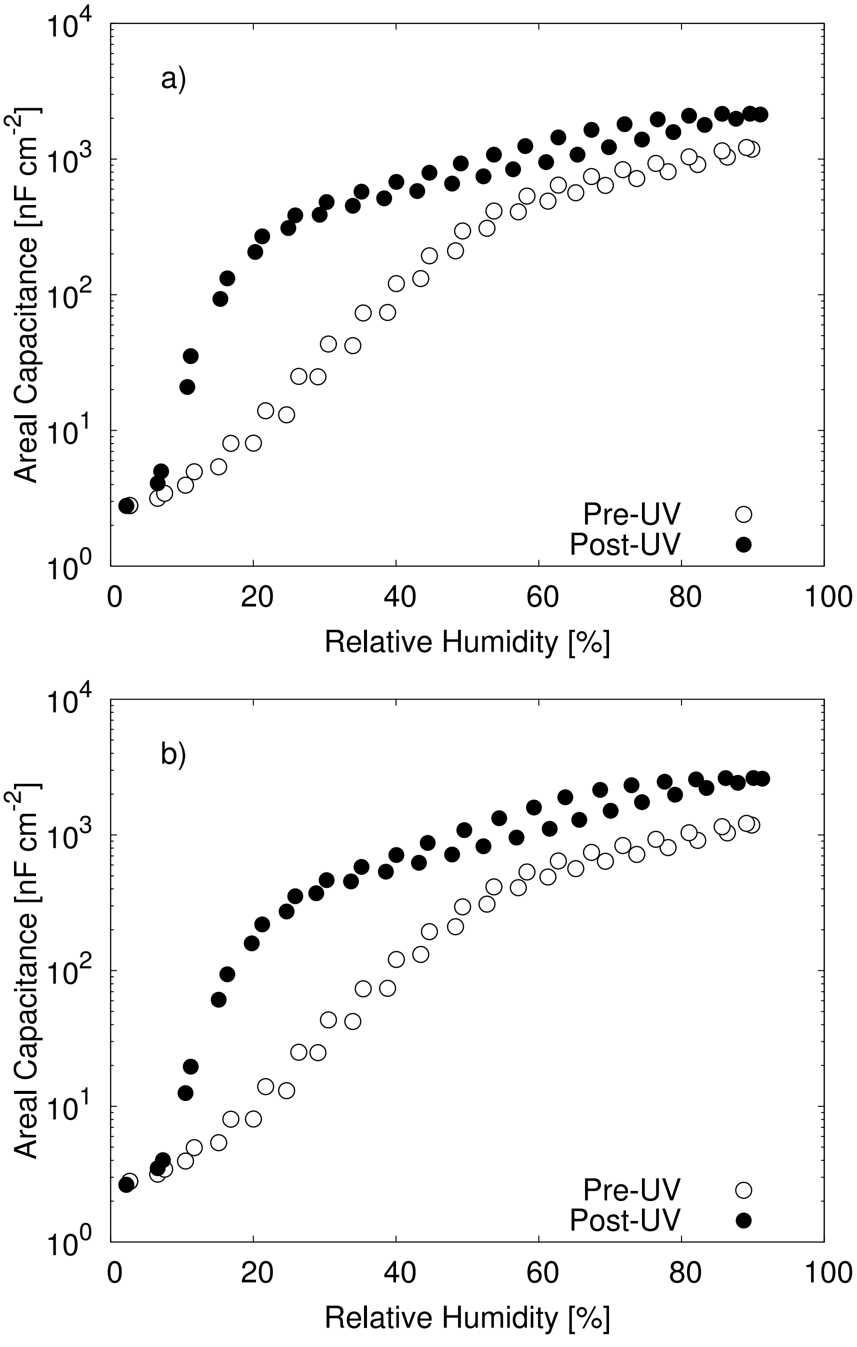


Figure 5.8: Responsivity of RH sensors treated with (a) 295 nm and (b) 310 nm UV LEDs

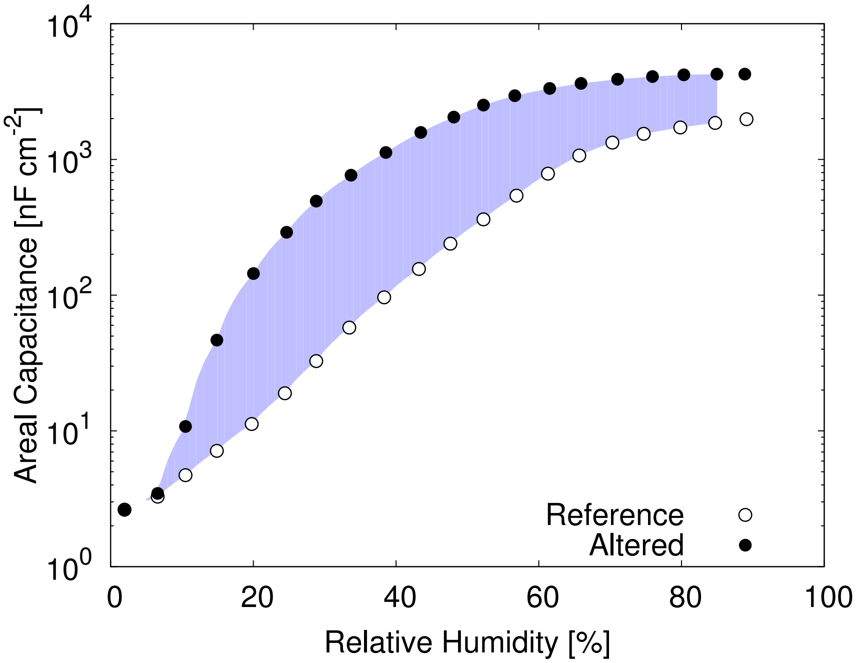


Figure 5.9: Graphical representation of the figure of merit (FOM), defined as the area between a reference sensor response curve (the original as-deposited sensor) and the altered sensor response curve. A UV-treated device has a FOM greater than zero, while an aged device has a FOM less than zero.

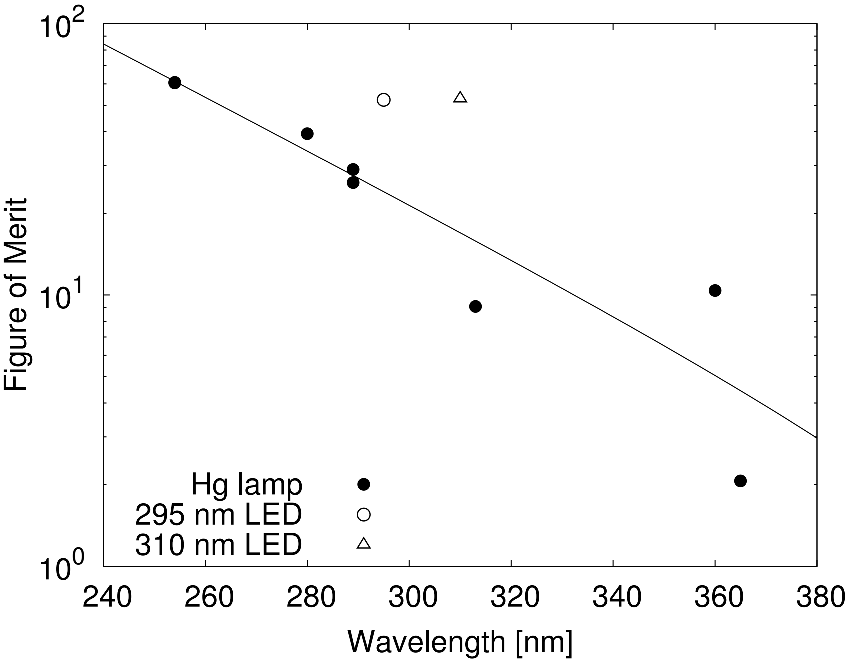


Figure 5.10: FOM as a function of illumination wavelength. The curve is a guide for the eye.

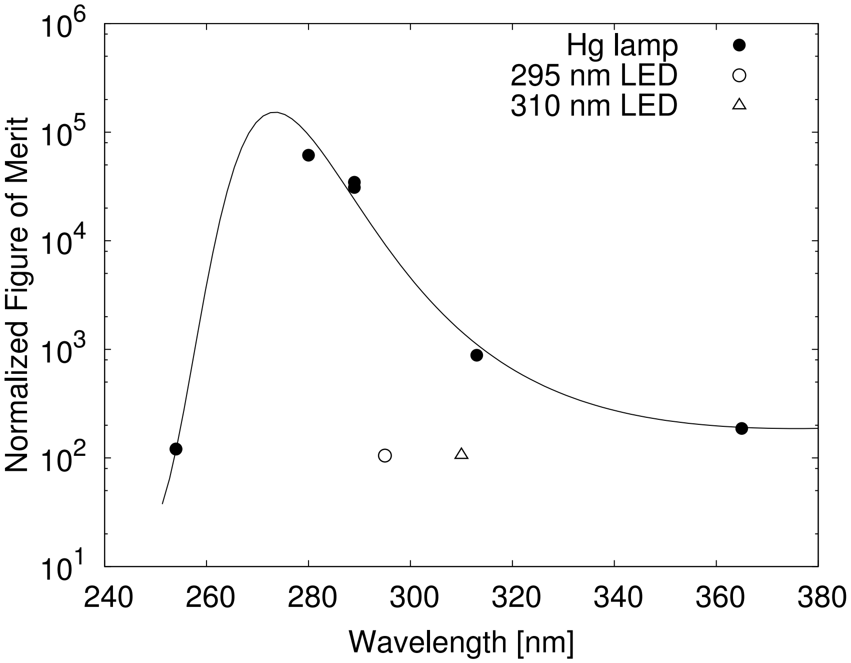


Figure 5.11: FOM divided by power delivered at that wavelength. The curve is a guide for the eye.

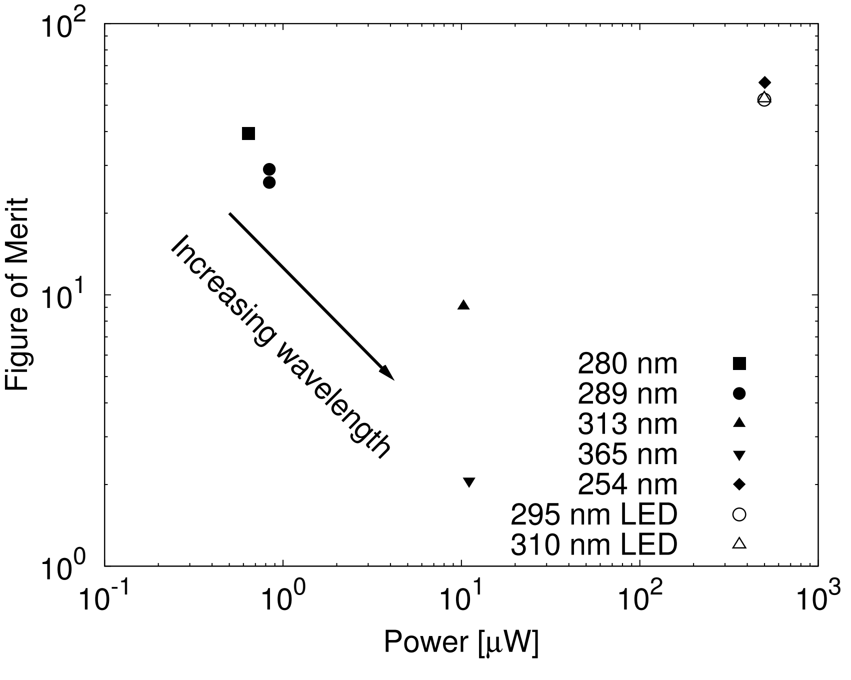


Figure 5.12: FOM as a function of the illumination power.

μW the FOM decreased with increasing power since higher powers correspond to longer treatment wavelengths.

After UV treatment the devices were left to age for 87-96 days before being tested again. Ageing resulted in a decrease in the FOM (Figure 5.13a). Devices in the low-power regime had a negative FOM while the devices in the high-power regime retained positive FOMs. The change in FOM ranged from -56.9 to -32.6 with a mean of -36.3. Figure 5.13b shows time-resolved device ageing under several different conditions. The samples treated with filtered wavelengths were the most stable and aged the slowest, at an average rate of -0.43 ± 0.03 FOM/day. An untreated sample aged at a rate of -0.93 ± 0.04 FOM/day. The least stable sample was treated with an unfiltered lamp (data from a previous study [28]) and aged at a rate of -1.38 ± 0.03 FOM/day.

5.4 Discussion

The increase in the water contact angle after UV treatment at all wavelengths except for 254 nm is contrary to what was expected. It is possible that the sensors were not exposed for long enough at higher wavelengths to induce hydrophilicity. The power of the 254 nm emission line is significantly greater than the other lines and produced the smallest contact angle and largest change. Sakai *et al.* found that the rate of change in the water contact angle is proportional to the irradiation intensity [83]. They also reported that there is a strong relation between the reciprocal of the contact angle and the density of surface hydroxyl groups [84], which will be explained in more detail below. Lin *et al.* reported an average water contact angle of 24° for fresh titanium oxide films deposited by reactive DC magnetron sputtering [85], which is smaller but compares to the average of 34° for the new films examined here. Lin *et al.* also found that after ageing for 2 days the average contact angle was over 60°, after ageing for 6 days the average contact angle was over 80°, and UV irradiation quickly changed the contact angle back to a range around 33°. For the films reported here, after the 48 hour treatment the contact angle for all films except for the one treated at 365 nm were below 60° and all except for the

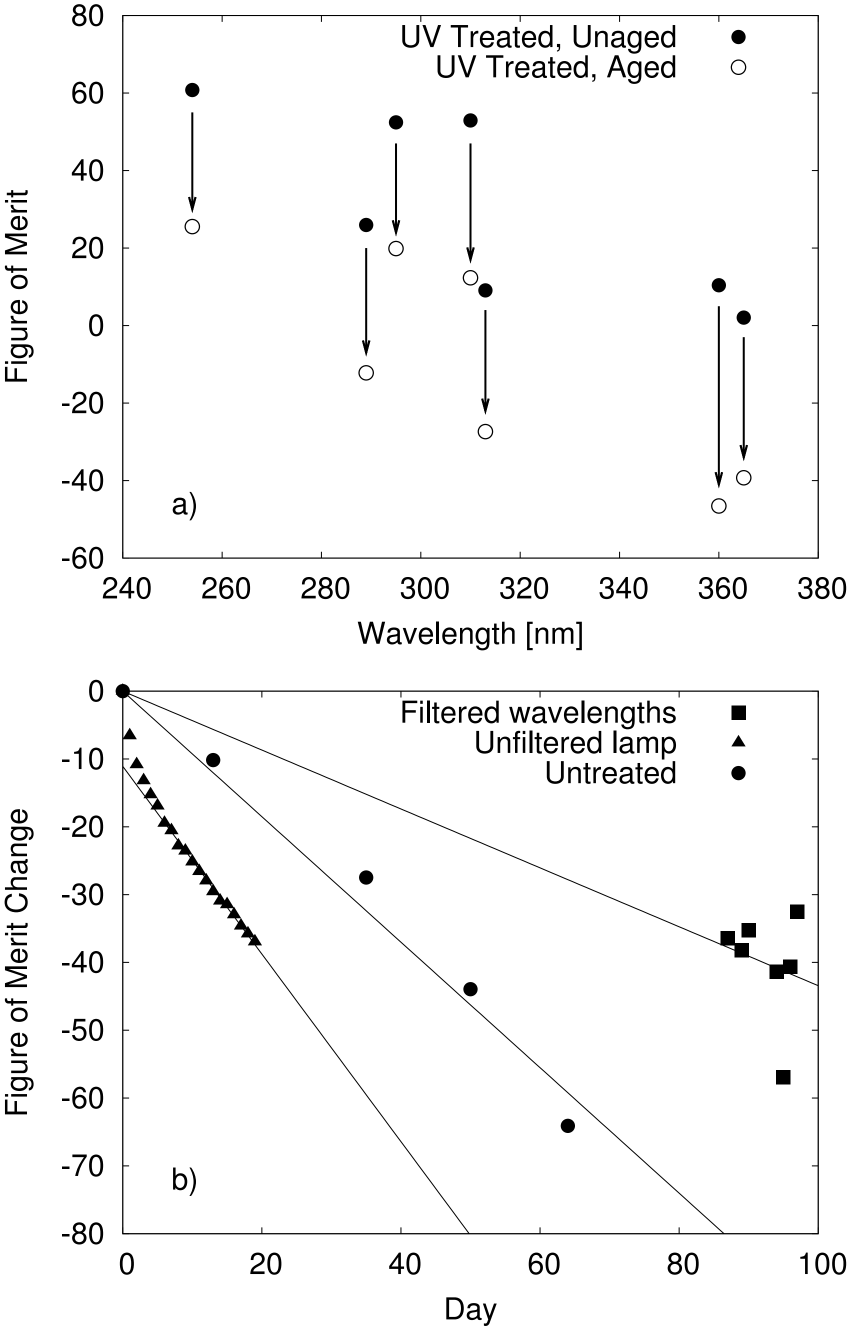


Figure 5.13: (a) FOM of unaged sensors and sensors aged approximately 90 days as a function of illumination wavelength. (b) Change in the FOM over time for different applied treatments. The sensors treated with the filtered mercury lamp are the most stable (show the smallest change in FOM) followed by the untreated sensor and then by the sensor treated with an unfiltered mercury lamp (data from a previous study [28]). The untreated sensor starts at a FOM of 0 while the treated sensors start with a FOM of between 25 and 100.

one treated at 254 nm had increased. This could be because the irradiation intensity was too weak and the films aged faster than they were regenerated over the 48 hour treatment period. With sufficient irradiation intensity a point may be reached where the ageing is countered and the sensors are regenerated, since the rate of change in water contact angle is proportional to the intensity.

Despite the increases observed in water contact angle after UV treatment, the decreasing trend in angle with decreasing wavelength after treatment shows some similarity to the FOM trend. As the water contact angle decreases, the FOM increases. Both trends suggest that shorter UV wavelengths have a greater effect on the films and sensors. The effect is greatest in both cases at wavelengths below 300 nm. These results form the basis of the UV characterization and will be useful when selecting a UV LED wavelength for sensor regeneration and stabilization.

Analysis of the FOM versus illumination wavelength in Figure 5.10 is somewhat misleading because the trend suggests that a greater FOM may be achieved with continually shorter illumination wavelengths. The FOMs from the UV LEDs contradict this trend. The reason for this is because the delivered power is not taken into account. When the FOMs are normalized with respect to the power it becomes apparent that the 254 nm and UV LED treatments are less efficient treatments when compared to the others. However the trend observed in Figure 5.11 does not necessarily mean that the optimum wavelength is 275 nm. Since the 254 nm and UV LED treatments deliver 2-3 orders of magnitude more power than the other treatment, and that the FOMs in this high power regime are similar values, it is likely that the treatments were saturated and more power was delivered than necessary to achieve the results. The fact that the (relatively) high-powered LEDs produced significantly larger FOMs than low-power treatments at similar wavelengths supports this. This means that although the optimum LED wavelength in terms of FOM versus power delivered might be at shorter wavelengths, cheaper LEDs at longer wavelengths may be used to achieve good sensor results.

In Figure 5.13b the sensor treated with an unfiltered mercury lamp aged about 3 times as fast as sensors treated with filtered wavelengths. Two plausible reasons for this difference have been identified. The first is the frequency of testing. The RH was continuously cycled and impedance data collected for the duration of several weeks for the sensor treated with the unfiltered lamp. Extreme RH cycling has been observed to age sensors faster than normal [28]. This is contrasted with the sensors treated with filtered wavelengths, which were left unilluminated inside a cabinet for approximately 3 months before being tested again.

The second potential reason for the difference in the rate of ageing is the shape of the response curve and the way that it might affect the FOM. The baseline shape of the response curve varies slightly between different batches of sensors. It is possible that a change in the response curve of one sensor registers as a larger change in the FOM because of the inherent shape of the curve. This characteristic of the FOM has not yet been examined.

The fact that some sensors age faster than others, however, is not important in the context of regenerating sensors with the UV treatment. The rate at which the sensors are regenerated is much faster than the rate at which they age. Thus a continuous UV treatment may negate the ageing of the sensors and stabilize their response curves.

The accumulation of organic contaminants on the surface of the sensors [86,87] has been observed to reduce the hydrophilicity of titanium dioxide films and could be one factor that contributes to sensor ageing. Titanium oxide is well known as a photocatalyst [27]. Treatment with UV light might regenerate the sensors by photocatalyzing the decomposition of the organic contaminants on the surface.

Alternatively, a reduction or restructuring of the surface hydroxylation of the sensors could also be responsible for the ageing. Formation of stable hydroxyl groups bound to oxygen vacancies at the surface is thought to cause a drift in the conductivity of humidity sensors [20]. This can also be thought of hydration of bridging oxygen atoms on the surface, caused by long-term exposure to water [20]. This is reflected in the ageing of sensors after extreme humidity cycling [28]. Sensor conductivity is due to protons hopping between the hydroxyl ions of the initial layer of chemisorbed water at low humidities and between physisorbed water molecules according to the Grotthuss chain reaction mechanism at higher humidities [88] (see Chapter 2). The decreased conductivity of aged sensors is caused by the increase in the stable chemisorbed hydroxyl groups on the surface, instead of the presence of chemisorbed water molecules [89].

A model for the surface restructuring is described by Sakai *et al.* [84]. In this model under dark conditions oxygen vacancy (defect) sites are occupied by hydroxyl groups that are bound to two Ti atoms. UV illumination recreates the oxygen defect at which water is chemisorbed and dissociates. This results in two hydroxyl groups at the site that are each bound to a single Ti atom. This process increases the total number of hydroxyl groups and reduces the number of hydroxyl groups bound to oxygen vacancies. In the dark, the surface reverts back to doubly-bound hydroxyl groups at oxygen vacancies. This change and restructuring of surface hydroxylation changes the hydrophilicity of the films and could be related to the shift in the steep region of the sensor response between higher and lower humidities.

Annealing titanium oxide films at high temperatures also creates oxygen defects and it was found that after annealing at 900 K the surface defect coverage reaches a maximum of about 10 % [90]. Unfortunately, such treatments will degrade the GLAD structure which is responsible for the rapid response times our sensors are capable of. It is possible that the saturation observed in the FOM is related to a maximum density of hydroxyl groups or defect states on the sensor surface. After delivering a certain amount of energy to the films, the hydroxyl density might become saturated. Further characterization of sensor regeneration across a range of UV LEDs will determine the longest wavelength and minimum power required to regenerate and stabilize a sensor.

5.5 Conclusion

In this chapter, the wavelength dependency of the UV treatment of GLAD RH sensors was characterized using a filtered mercury vapour lamp. A figure of merit (FOM) was defined and used to quantify the change in sensor response after treatment and ageing relative to the original sensor response. A general trend of a larger FOM with shorter UV treatment wavelength was observed. However when the delivered power was taken into consideration, high-power and low-power treatment regimes became evident. The effect of the treatment in the high-power regime seemed to saturate regardless of the wavelength. The low-power regime still followed the trend of greater FOM with shorter wavelength. Treated sensors were observed to age on average at a constant rate of -0.43 FOM/day independent of the treatment wavelength, much slower than the sensor regeneration of up to 61 FOM over 48 hours. It is thought that these effects are the result of changes in the hydroxylation of the surface of the sensors.

The next chapter describes work on characterization of the sensor treatment and long-term stabilization of sensors with a range of UV LED wavelengths.

Chapter 6

UV LED Treatment and Stability [1]

6.1 Introduction

The treatment and stabilization of GLAD relative humidity sensors using UV LEDs are described in this chapter. The UV LED technology discussed in Chapter 4 offers a small, low power alternative to mercury vapour lamps and does not have environmental hazard disposal issues. This chapter first discusses the treatment and performance enhancement of sensors using a range of UV LED wavelengths in order to identify an optimum wavelength. The repeatability of the UV LED treatment is then presented. Next the long-term stability of sensors under continuous illumination with the UV LEDs is investigated. Lastly, the performance of these UV-stabilized sensors is compared to commercial humidity sensors.

6.2 Experimental Setup

Two sets of relative humidity sensors were fabricated using the procedures described in Chapter 5. Porous films 1.5 μm thick were deposited on top of countersunk IDEs (Micralyne, Inc.) with a deposition angle $\alpha = 80°$ using GLAD. Rutile TiO_2 (99.9%, Cerac, Inc.) was used as a source material to deposit TiO_2 vertical columns 1.5 μm thick at a vapour flux deposition rate of 0.5 nm s^{-1} between pressures of 6×10^{-5} Torr and 8×10^{-5} Torr. Oxygen was added during deposition at a flow rate of 2.2 sccm - 3.6 sccm to grow a more stoichiometric film.

[1] Part of this chapter has been submitted for publication in: D.P. Smetaniuk, M.T. Taschuk, and M.J. Brett, ”Performance and lifetime enhancement of TiO_2 capacitive nanostructured relative humidity sensors using ultraviolet LEDs,” Under revision August 14, 2011 for IEEE Sensors Journal.

The samples were thermally oxidized at 100°C in atmosphere for 24 hours after deposition [16,28,40].

One set of sensors was constantly irradiated with an UV LED (UVTOP, Thorlabs) for each sensor. Typical UV LED specifications are summarized in Table 6.1. Note that the costs are approximate $CAD and for a small volume purchase. The UV LED spectra were measured using an Ocean Optics HR2000 UV-VIS CCD spectrometer. LED260W, LED285W, LED315W, and LED341W come in TO-39 packages while LED370E and LED405E come in T-1 3/4 packages. All LEDs were operated with a constant current of 20 mA. Each sensor and LED was mounted inside a small aluminum box with the sensor on one side of the box, and the LED on the other, positioned so that the maximum irradiance was delivered to the sensor IDE (Figure 6.1). Because of differences in the beam divergence angle, LEDs in TO-39 packages were positioned approximately 5 mm from the sensor and LEDs in T-1 3/4 packages were positioned approximately 2 cm from the sensor so that the beam spot covers the entire IDE. The LEDs were left on for the entire duration of the experiment, including during sensor testing. The other set of sensors was tested before and after a 48 hour treatment under a 370 nm UV LED to test the treatment variability from device to device.

For this chapter a new figure of merit different than that in Chapter 5 was defined (Figure 6.2):

$$FOM^{*} = \int_{RH_{min}}^{RH_{max}} \left[\log_{10} C_s(RH) - \log_{10} C_g\right] dRH \qquad (6.1)$$

where RH_{min} and RH_{max} are the minimum and maximum relative humidities achieved during testing, C_s is the capacitive response of the sensor to relative humidity, and C_g is the geometric capacitance of the IDE. FOM* uses the geometric capacitance instead of the initial response of the sensor. Because of this FOM* is able to track the overall performance change in the sensor, rather than the relative change and is not susceptible to variations in the initial sensor response. FOM* is always positive, and a value of zero or near zero effectively means the sensor is dead. The new FOM* values may be approximately compared to the values in Chapter 5 by adding 80 to the old FOM.

Table 6.1: UV LED specifications

Part #	Center Wavelength	Spectral FWHM	Optical Power @ 20 mA	Cost ($CAD)
LED260W	260 ± 5 nm	12 nm	0.3 mW	$289
LED285W	285 ± 5 nm	12 nm	0.8 mW	$182
LED315W	315 ± 5 nm	10 nm	0.6 mW	$142
LED341W	340 ± 10 nm	15 nm	0.35 mW (Max)	$175
LED370E	375 ± 10 nm	10 nm	2.5 mW	$4
LED405E	405 ± 10 nm	15 nm	6 mW	$14

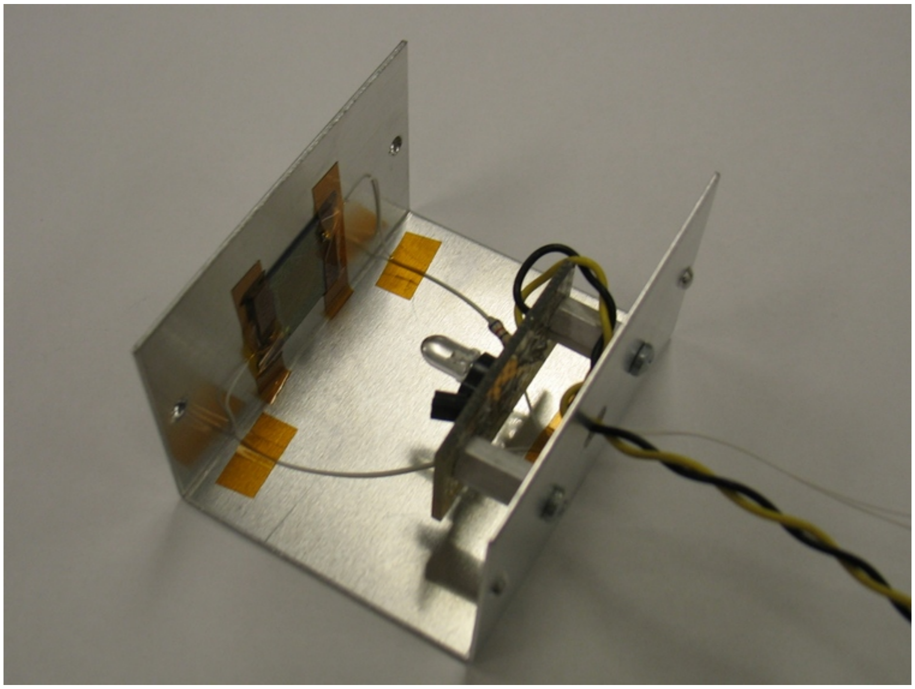


Figure 6.1: Aluminum box (without cover) containing a sensor and UV LED.

The geometric capacitance may be calculated from a model that takes into account the IDE geometry, substrate, and deposited film electrical properties [58]. The values $RH_{max} = 80$ %RH, $RH_{min} = 5$ %RH, and $C_g = 1.6$ nF were used when calculating FOM* values. C_g was obtained by a capacitive measurement at 1 MHz at a low humidity where the contributions of adsorbed water may be neglected.

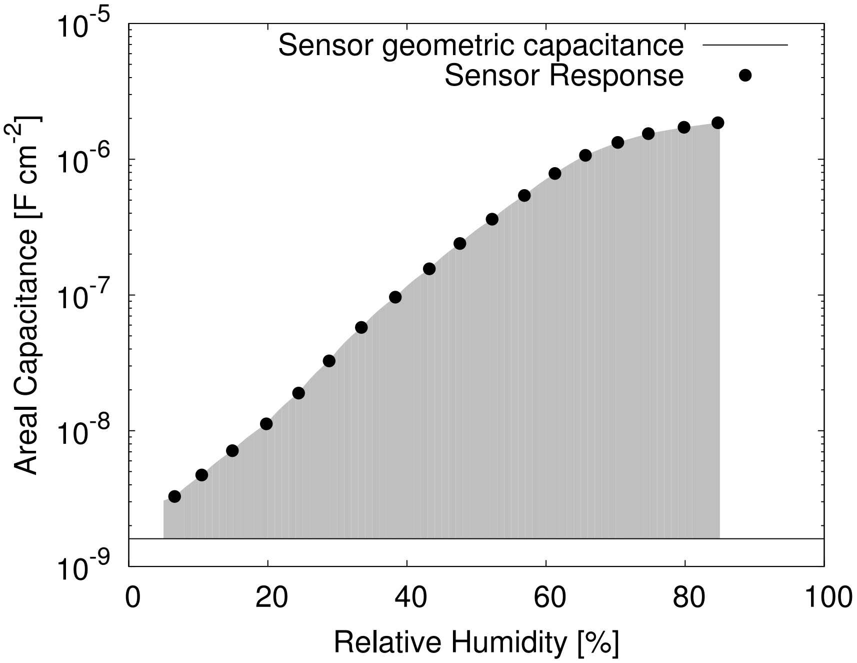


Figure 6.2: Graphical representation of the new figure of merit (FOM*), defined as the area between the sensor response curve and the geometric capacitance of the IDE. The FOM* is always greater than or equal to zero. A bare IDE without a film will have a FOM* of zero.

6.3 Results

UV LED wavelengths up to 370 nm effectively enhanced sensor performance after 15-21 days of UV treatment (Fig. 6.3). A typical response for a new, untreated sensor is shown by the solid line. The response curves of all regenerated sensors are very similar in shape. The sensor treated with the 405 nm LED showed some ageing. 370 nm was identified as the longest wavelength and least costly LED that effectively regenerates the sensors.

Figure 6.4 shows the hysteresis in the response of the sensor treated with the 370 nm LED over time. In a new sensor, hysteresis is prevalent at higher humidities and capacitance. As the sensor is treated over time the hysteresis loop tends to shift to the left towards lower humidities and gets smaller relative to the response. During sensor ageing without UV treatment the hysteresis loop opens up and shifts right towards higher humidities [28].

Sensor sensitivity is defined as the change in areal capacitance with a change

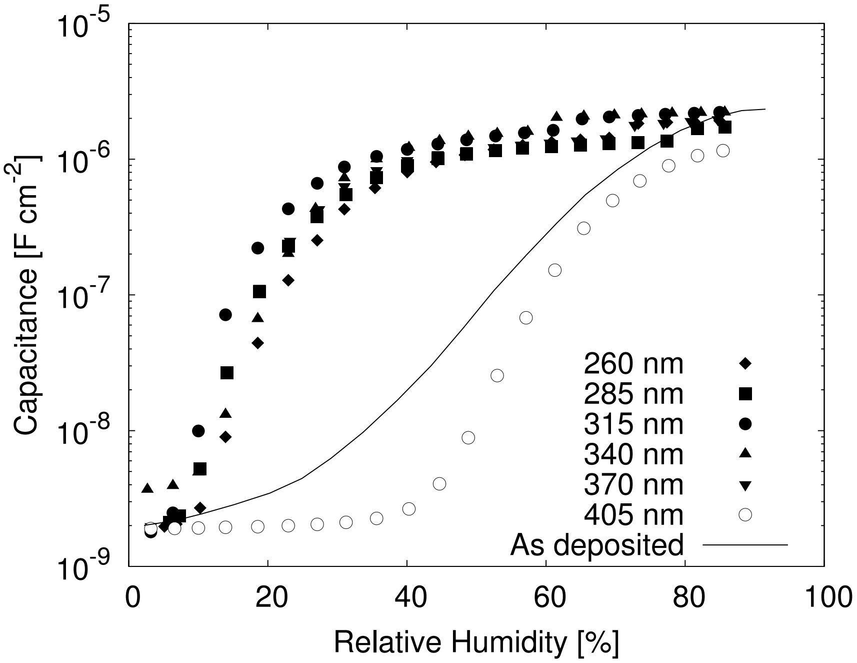


Figure 6.3: Sensor response after 15-21 days of treatment with UV LEDs. The solid line is the response of an untreated, unaged sensor.

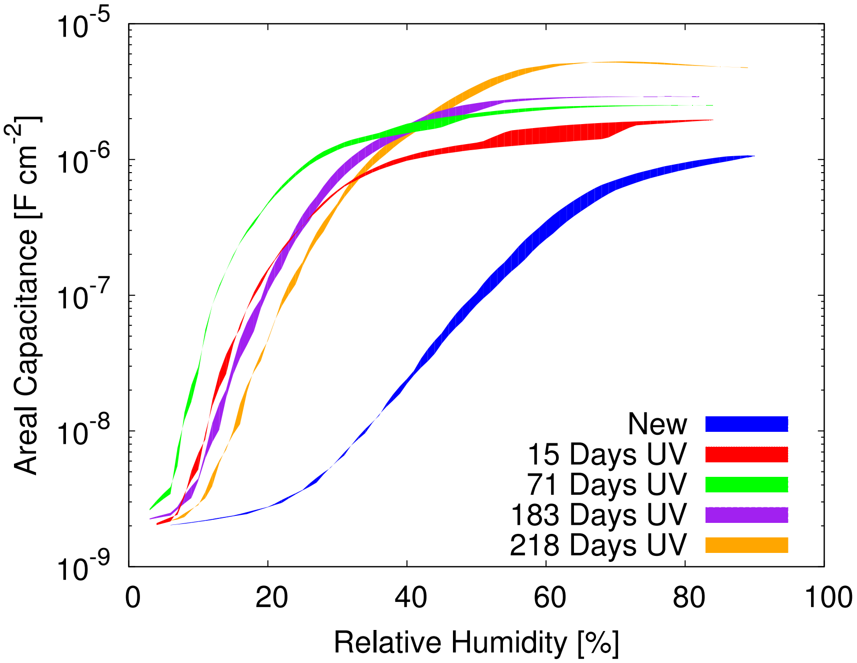


Figure 6.4: Sensor hysteresis loops for treatment with the 370 nm LED. The hysteresis loop tends to shift towards the left and closes as the sensor is treated.

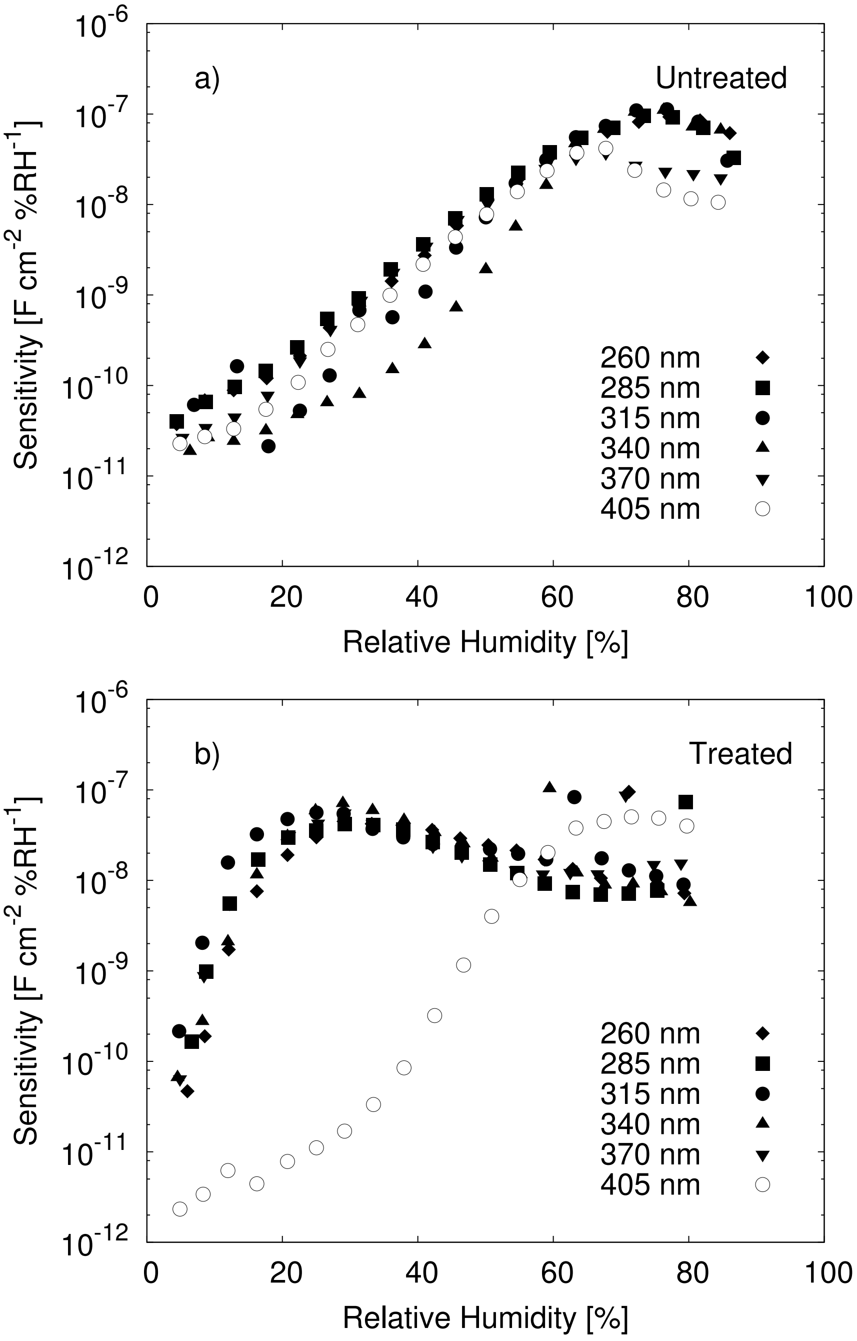


Figure 6.5: Sensor sensitivity before (a) and after (b) UV treatment. All LEDs except for 405 nm result in extension of the high sensitivity range to lower humidities.

in relative humidity and has units of nF cm^{-2} %RH^{-1}. Figure 6.5a shows the sensitivity of new sensors before treatment. The sensors show low sensitivity at low humidities, in the range of 30 pF cm^{-2} %RH^{-1}. The sensitivity increases with increasing humidity and reaches a maximum between 50 - 100 nF cm^{-2} %RH^{-1} around approximately 70 %RH. Figure 6.5b shows sensor sensitivity after 15-21 days of treatment with a UV LED. The sensor treated with the 405 nm LED shows an overall drop in sensitivity, down to around 3 pF cm^{-2} %RH^{-1} at low humidities. Sensors treated with UV wavelengths below 405 nm show an extension of the high sensitivity region. The sensitivity starts off in the same range at low humidities, but quickly increases to over 10 nF cm^{-2} %RH^{-1} above 15 %RH.

The FOM* was calculated for the sensors treated with UV LEDs at several times during treatment. The behavior of the FOM* values was modelled by fitting the following function:

$$FOM^{*}(t) = A\left[1 - exp\left(\frac{c-t}{\tau}\right) - mt\right] \tag{6.2}$$

where t is the time in days, τ is the time constant, and A, c, and m are constants with A being the maximum value of the FOM*. The exponential term fits the increasing behavior of the FOM*. The linear term with a slope $-m$ models the approximately linear rate at which the FOM* decreases as the sensor ages, as seen in Chapter 5. The FOM* values and trend lines are shown in Figure 6.6, where initial FOM* values for new, untreated sensors range from 71 to 105. After 15-21 days of treatment, the FOM* increased significantly, ranging from 175 to 195, for sensors treated with LEDs with wavelengths of 370 nm and lower. Up to approximately 75 days the FOM* continues to increase, but only slightly compared to the first treatment interval. The 315 nm and 370 nm sensors both remain fairly stable after this period, however the FOM* for 285 nm and 260 nm starts to decline and drop back towards the initial value. The FOM* value for the sensor treated with the 405 nm LED decreased slowly from 101 to 53 over 246 days of treatment.

The time constant τ and LED wavelength are compared on the left axis in Figure 6.7a. There is a correlation between FOM* time constant and LED wavelength, specifically, the greater the LED wavelength, the FOM* increases with a smaller

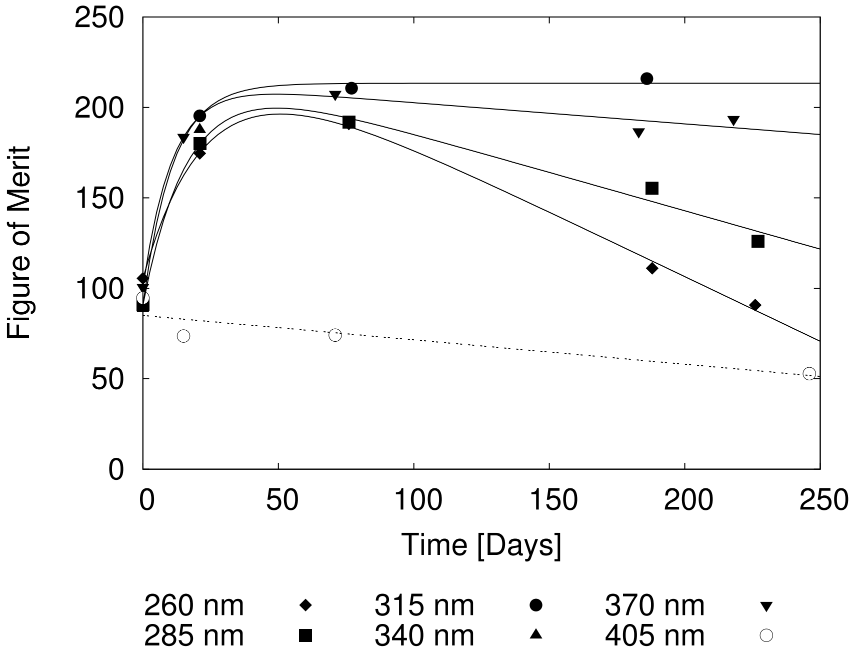


Figure 6.6: Figure of merit (FOM*) values over time with fitted curves.

time constant. However the right axis in Figure 6.7a shows that the LED output power generally increases with increasing LED wavelength. The time constant is compared to the LED power in Figure 6.7b; the solid line is a linear fit to the data. There is a general inverse correlation between the time constant and the LED power, i.e., the greater the LED power, the smaller the time constant. The 370 nm LED had the smallest time constant of 10 days.

In order to examine variability in the UV treatment from device to device, FOM* values were calculated for the second set of sensors before and after treatment with the 370 nm LED. The FOM* results of the repeatability experiment are summarized in Table 6.2. Before treatment the mean FOM* was 132.1 with a standard deviation of 2.4. After treatment the mean FOM* was 169.6 with a standard deviation of 0.9, demonstrating in both cases that device to device performance is consistent.

The performance of GLAD TiO_2 relative humidity sensors is compared with the performance of several commercial sensors in Table 6.3. The response time of less than 250 ms for GLAD sensors is much faster than the commercial sensors shown, and the sensitivity is around 10 nF/%RH over 15% RH, over four orders of magnitude greater than any of the commercial sensors. The ±2.5% RH hysteresis

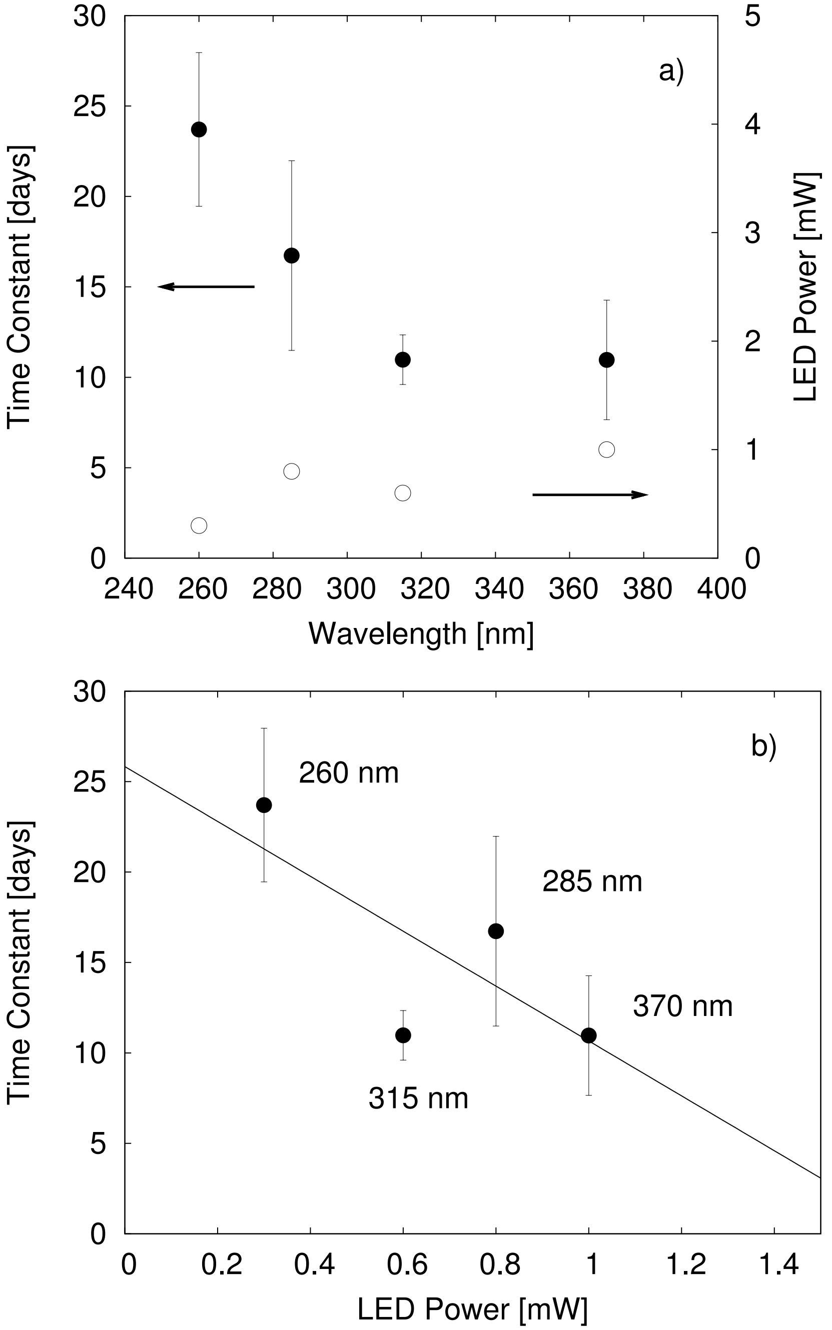


Figure 6.7: a) The time constant from the curve fitted to the FOM* values versus the LED wavelength is shown with the closed symbols on the left axis. The LED power versus the LED wavelength is shown with the open symbols on the right axis. b) FOM* time constant versus LED power. The solid line is a linear best fit to the data.

Table 6.2: 370 nm LED treatment repeatability

Sample #	FOM* before treatment	FOM* after treatment	ΔFOM*
1	128.98	170.53	41.55
2	131.56	169.27	37.70
3	132.99	168.90	35.91
4	134.73	-	-
mean	132.07	169.57	38.39
standard deviation	2.428	0.858	2.882

is similar to that of most sensors shown, which range from ≈ 0% RH to ±2% RH. Below approximately 50% RH the stability of GLAD sensors is ±3.5% RH over four months. Above approximately 50% RH or the flat region of the sensor response the stability is worse with a drift of ±12% RH or greater.

6.4 Discussion

The goal of the UV treatment is to enhance sensor performance, improve stability and prevent ageing. Since the bandgap of anatase titanium dioxide is 3.2 eV, corresponding to a wavelength of 387 nm, electron-hole pair generation will occur for UV treatment wavelengths below 387 nm. Photogenerated holes trapped at the surface create oxygen defects, which act as sites for catalysis [29, 84]. This supports the enhanced sensor performance observed with UV LED treatments up to 370 nm wavelengths. The sensor treated with the 405 nm LED exhibited ageing, however the rate of ageing with respect to the FOM* was slower than was seen with other untreated sensors in Chapter 5. This is likely because approximately 25 μW or 0.4 % of the spectral power of the 405 nm LED is below 387 nm, based on spectral measurements. This may be enough UV illumination to slow the rate of ageing, but not enough to enhance the performance.

Overall, sensor performance is enhanced by the tightening and shifting of the hysteresis loop to lower humidities (see Figure 6.4) as well as extension of the high sensitivity range to lower humidities (see Figure 6.5). During sensor ageing, the

Table 6.3: Comparison of GLAD humidity sensors with commercial humidity sensing technologies

Description	Humidity Range	Sensitivity	Accuracy	Hysteresis	Stability	Response sTime
CHS-U [21]	5 ∼ 95% RH	-	±3% RH @ 5 ∼ 95% RH	≈ 0	-	60 s (30 ∼ 85% RH)
HS1100 [22]	1 ∼ 99% RH	0.34 pF/%RH	±2% RH @ 10 ∼ 90% RH	±1.5% RH	0.5% RH/yr	5 s (33 ∼ 76% RH)
HIH-5030 [23]	0 ∼ 100% RH	-	±3% RH @ 11 ∼ 89% RH	2% RH	±1.2% RH/5 yrs	5 s
HCH-1000 [24]	0 ∼ 100% RH	0.6 pF/%RH	±2% RH	±2% RH	0.2 %RH/yr	15 s (20 ∼ 90% RH)
HMP110 [25]	0 ∼ 100% RH	-	±1.7% RH @ 0 ∼ 90% RH ±2.5% RH @ 90 ∼ 100% RH	-	±2% RH/2 yrs	60 s (90% RH)
GLAD TiO_2	2 ∼ 92% RH[1]	∼10 nF/%RH[2]	±2% RH	±2.5% RH	7% RH/4 mths[3] 25% RH/4 mths[4]	< 250 ms

[1] Limited by humidity chamber capabilities.

[2] Above ∼15% RH. See Figure 6.5.

[3] Below ∼50% RH.

[4] Above ∼50% RH.

opposite is observed: the hysteresis loops open up and shift to the right, while the high sensitivity range shrinks to higher humidities [28].

In Figure 5.13 the initial increasing behavior of the FOM* follows an exponential trend and saturates at approximately 200, depending on the UV wavelength. Chapter 5 suggests that the change in the sensor response is related to hydroxylation of the sensor surface [29]. At low humidities sensor conductivity is due to proton hopping between the hydroxyl ions in the first chemisorbed layer of water. As the humidity is increased, sensor conductivity and sensitivity increase drastically as subsequent layers of water are physisorbed and proton-hopping occurs between freely-bound water molecules via the Grotthuss mechanism. The number of hydroxyl groups bound to a single Ti atom is increased under UV illumination, increasing the conductivity at lower humidities. We expect that the UV LED-induced change in hydroxyl density is responsible for the reported shift in the steep region of the response and sensitivity curves after UV treatment. The inverse exponential decay behaviour of the FOM* supports the idea that there are a limited number of sites available for hydroxylation and that the majority of sites have been hydroxylated when the FOM* begins to saturate. At this point the rates of the sensor ageing and regeneration processes have reached equilibrium.

Once treated, the ideal case is that the sensor response remains stable and the FOM* is constant. However we observe a decreasing FOM* for all sensors except for the one treated with 315 nm. This slow decrease in FOM* may be due to decreasing UV output power from the LEDs over time. Meneghini *et al.* found an output power decrease of 13% in UV LEDs after 100 hours of operation at 20 mA [91]. UV LEDs manufactured by Nichia, which use similar technology as the UV LEDs used in this study, typically have lifetimes of 500 - 1000 hours [92]. The 370 nm and 405 nm sensors show some stability, with only a modest decrease in the FOM* with time. However the 405 nm sensor is not an attractive choice for a sensor because of the poor sensitivity range (see Figure 6.5) and because its wavelength is above the bandgap wavelength of TiO_2.

To counteract the ageing of the LEDs, we recommend increasing the number of LEDs and decreasing the average current per LED. This will extend the lifetime of the LEDs, and thereby the lifetime of the combined sensor package. From a cost and performance perspective, we recommend an array of 370 nm LEDs. A close hexagonal packed array of seven LEDs would allow three LEDs to illuminate the sensor, increasing lifetime well beyond the 200 days observed here. As these LEDs fail, the additional LEDs could be switched on autonomously to replace them, thereby extending the combined sensor package lifetime significantly.

The time constant for the increase of the FOM*, τ, is generally smaller with longer LED wavelengths (Figure 6.7a). However this apparent correlation is deceptive since the LED power tends to increase with increasing wavelength. There is a stronger inverse linear correlation in Figure 6.7b between LED power and τ, suggesting that the power of the LED is more important than the wavelength. Thus initially treating sensors with greater UV power will more quickly enhance the performance. From a manufacturing standpoint, treatment with a high power Hg lamp after sensor fabrication would eliminate the initial slow rise in FOM observed here. The treated sensors could then be coupled to the LED array described above, and deployed in the field.

6.5 Conclusion

In this chapter we reported the regeneration and stabilization of GLAD relative humidity sensors using commercial UV LEDs. We have identified 370 nm as the best LED to use for sensor regeneration and stabilization. All LED wavelengths tested below 387 nm enhance sensor performance, but 370 nm is significantly less expensive than all of the others in addition to having the smallest time constant for the increase of the FOM* and the greatest optical output power. The repeatability test for treatment with 370 nm showed a large reduction in the variability of the FOM* after treatment. The decrease in ΔFOM* between each sample could be because the LED power was decreasing, since the same LED was used serially to treat the sensors.

The next chapter discusses the design and testing of a custom 8-channel impedance analyzer for future UV treatment experiments beyond the scope of this thesis.

Chapter 7

Impedance Measurement Electronics: Design and Testing

This chapter discusses the design and testing of a custom impedance analyzer with 8 channels, capable of independent impedance measurements. The objective of this work is to allow for future experiments beyond the scope of this thesis that require the testing of several sensors in parallel. These experiments will allow for better sensor design and are suggested in the Proposed Future Work section of Chapter 8. This chapter first introduces impedance spectroscopy, discusses the theory behind impedance measurement, describes the implementation of the impedance analyzer, and then provides initial testing results.

7.1 Introduction

The impedance of a device can generally be defined as the degree to which the device opposes the flow of an alternating current (AC) at a frequency f. Impedance, Z, is described as a complex quantity:

$$Z = R + jX = |Z|\angle\theta \tag{7.1}$$

where the real component R is the resistance and the imaginary component X is the reactance when described in rectangular-coordinate form, and $|Z|$ is the impedance magnitude and θ is the phase angle when described in polar form. The reactance may either be inductive (X_L) or capacitive (X_C) and may be determined by the following relationships:

$$X_L = 2\pi f L = 2\omega L \tag{7.2}$$

$$X_C = \frac{-1}{2\pi f C} = \frac{-1}{\omega C} \tag{7.3}$$

where L is the component inductance, C is the component capacitance, and $\omega = 2\pi f$ is the angular frequency.

Impedance spectroscopy is the analysis of the electrical response of a system to perturbation by a signal over a range of frequencies in order to gain information about the properties or characteristics of the system. Impedance spectroscopy could assist in gaining insight into the physical mechanisms behind humidity sensing in GLAD devices. It may be useful in determining when ageing has occurred in a sensor, and provide feedback for modulating UV LED power to stabilize the device. An equivalent circuit model for titania thick film humidity sensors has been proposed by Faia *et al.* [93, 94]. The model contains discrete elements and a physical justification for each element has been provided. Applying the model to GLAD RH sensors may provide valuable information and insight and aide in better sensor design, but is beyond the scope of this thesis.

Work on GLAD RH sensors has so far primarily focused on capacitive changes in the sensors. However a large amount of information may additionally be gained by examining changes in overall impedance of the device. Figure 7.1 shows an example of the impedance magnitude, phase, and capacitive response of a sensor measured at 1 kHz using the Quadtech 1920 Precision LCR meter. Complex changes also occur in all three parameters across a wide frequency range. Figures 7.2, 7.3, and 7.4 show examples of frequency-resolved curves for impedance, capacitance, and phase respectively for the same sensor across a frequency range from 20 Hz to 1 MHz measured with the Quadtech 1920 Precision LCR meter.

The Quadtech 1920 Precision LCR meter currently used for impedance measurements only has one channel and is only able to measure a single device at a time. Ideally we would like to be able to test several sensors simultaneously. This would allow for several devices with differing UV irradiation power or duty cycle

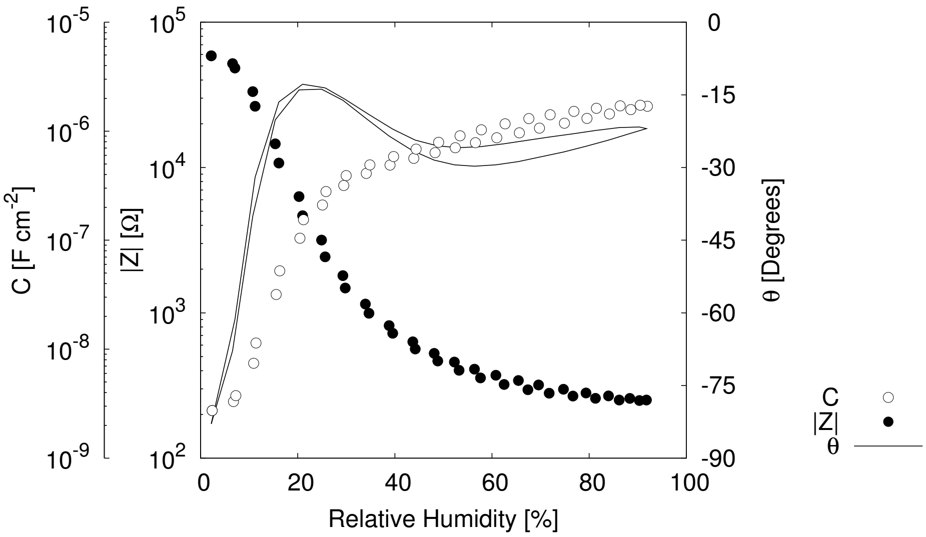


Figure 7.1: Example of sensor impedance, capacitance, and phase response measured at 1 kHz with the Quadtech 1920 Precision LCR meter.

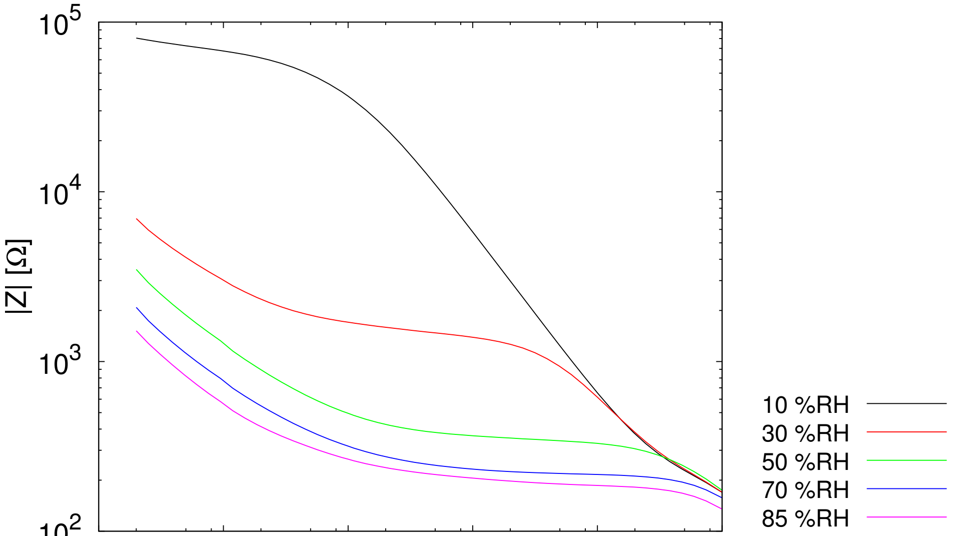


Figure 7.2: Frequency-resolved impedance $|Z|$ of a sensor measured with the Quadtech 1920 Precision LCR meter.

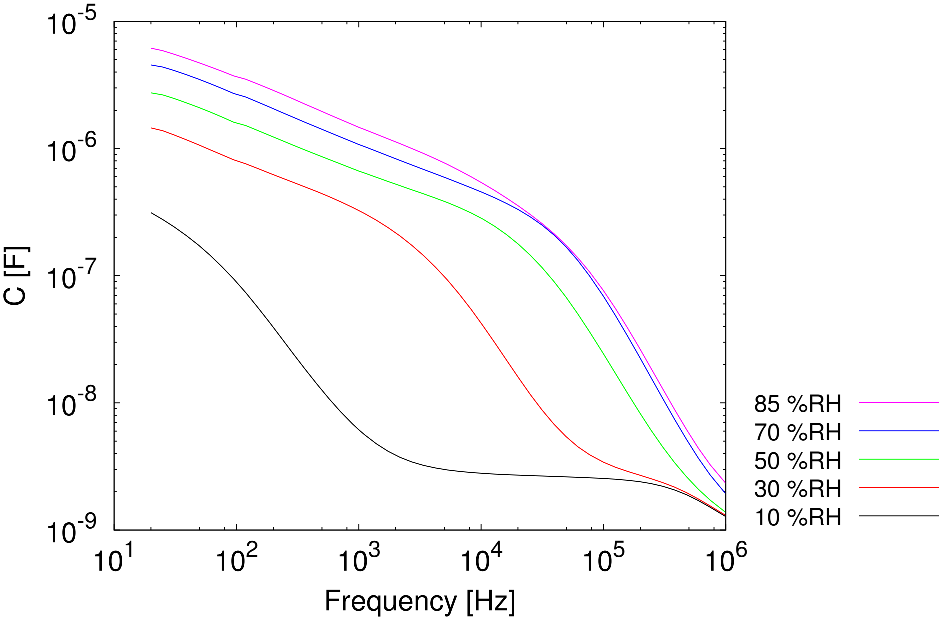


Figure 7.3: Frequency-resolved capacitance C of a sensor measured with the Quadtech 1920 Precision LCR meter.

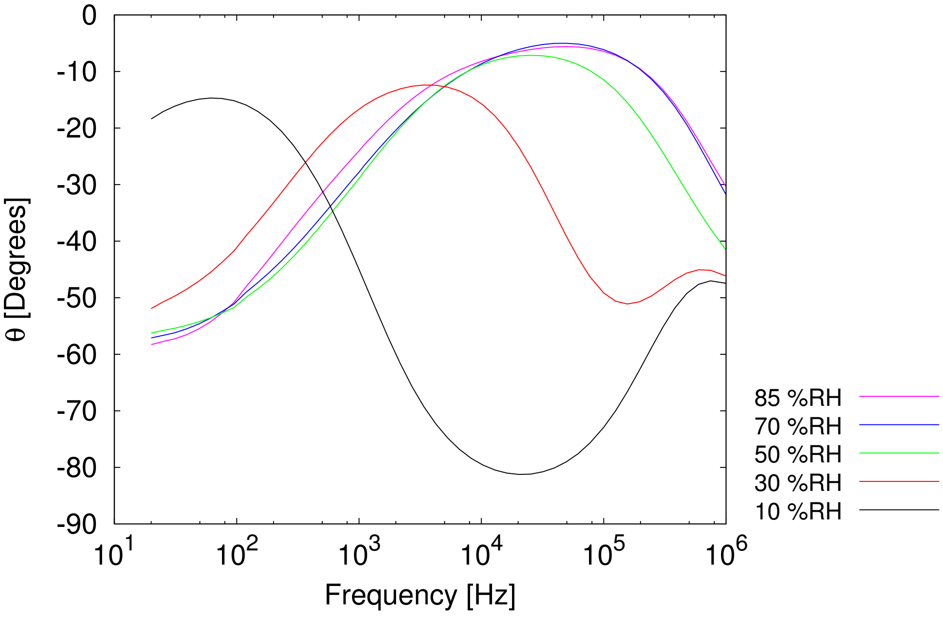


Figure 7.4: Frequency-resolved phase θ of a sensor measured with the Quadtech 1920 Precision LCR meter.

to be tested together in the humidity chamber under the same humidity cycling conditions. Also, time-resolved ageing or regeneration changes in the sensors could be directly compared. Using relays to switch measurement between sensors is undesirable because at lower impedances the relays would affect the measurements, introducing significant errors. Ideally, each sensor will have its own dedicated analog measurement electronics. Given the cost of hundreds to thousands of dollars for LCR meters, a custom impedance analyzer capable of measurements on up to eight separate channels was developed. It is capable of measuring the impedance magnitude and phase with a frequency range of 0.1 Hz to 100 kHz.

The impedance analyzer consists of a main motherboard and eight daughterboards. The motherboard contains a DLP-2322PB-G microcontroller module and interfaces with a computer via USB connection. Each daughterboard contains the analog impedance measurement circuitry and interfaces digitally with the motherboard by plugging into one of eight ports. The rest of this chapter describes the theory behind the impedance measurement and the hardware and software implementation.

7.2 Theory

There are several different techniques for measuring impedance: bridge method, resonant method, I-V method, RF I-V method, network analysis method, and auto-balancing bridge method. The auto-balancing bridge method is generally the best for lower frequencies under 100 MHz and allows for measurements over a large frequency range and high accuracy over a large impedance range [95] and will be the only method discussed below.

The general principle behind measurements with the auto-balancing bridge method is to apply a test signal to the device under test (DUT) and to measure the voltage of the test signal and the current flowing through the DUT (Figure 7.5). The impedance Z may simply be determined by the following:

$$Z = \frac{V}{I} \tag{7.4}$$

where V is the measured voltage, and I is the measured current through the DUT.

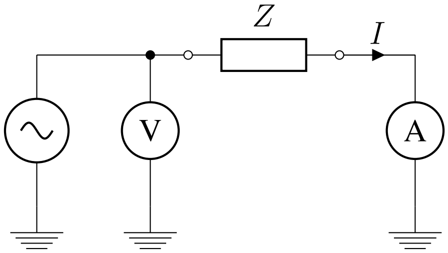


Figure 7.5: The principle behind impedance measurements using an autobalancing bridge.

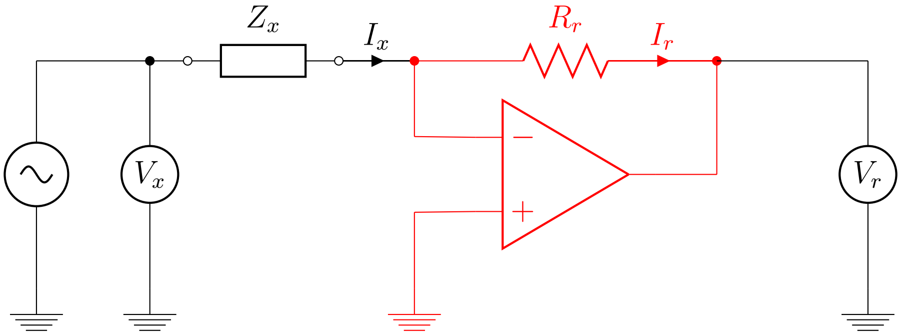


Figure 7.6: Simple auto-balancing bridge requiring two voltage measurements. The circuit in red is the op-amp I-V converter.

In practice, the current through the DUT is converted to a voltage, using an I-V converter circuit, and then the voltage instead of current is measured. Figure 7.6 shows the auto-balancing bridge with the I-V converter inserted (shown in red). The I-V converter is a simple circuit that uses an operational amplifier (op-amp) with a resistor in the negative feedback loop. The non-inverting input of the op-amp is connected to ground, so the inverting input acts as a virtual ground. The current I_x flows through the feedback resistor R_r, creating a voltage drop which is measured at the op-amp output. Because the voltage drop is with respect to the virtual ground, the voltage at the output V_R is negative and is determined by the following:

$$V_r = -I_r R_r \tag{7.5}$$

Since $I_r = I_x$, rearranging we get

$$I_x = -\frac{V_r}{R_r} \tag{7.6}$$

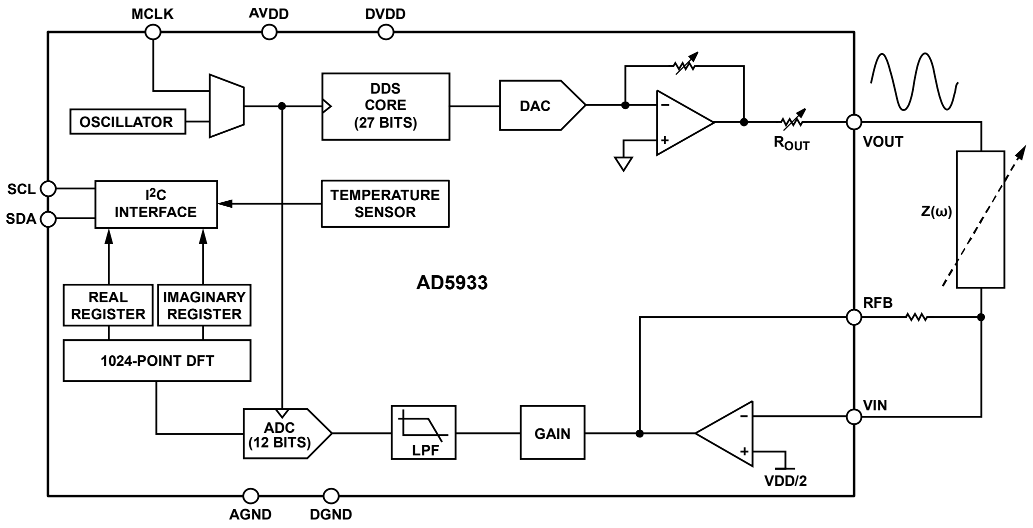


Figure 7.7: Functional block diagram of the AD5933 with the suggested configuration. Image reproduced from [96] with permission. Copyright Analog Devices, Inc.

Substituting into Equation 7.4 we derive the equation for the impedance Z_x of the DUT:

$$Z_x = \frac{V_x}{I_x} = -R_r \frac{V_x}{V_r} \tag{7.7}$$

The value of the resistor R_r is important and is known as the range resistor because it determines what range of impedances can be measured. If the value is too high, then the op-amp output will saturate, or the voltage will be too high to measure. Conversely, if the value is too low, the signal becomes more vulnerable to noise and the measurement accuracy will be lower.

Although in principle the auto-balancing bridge is simple, in practice implementing it requires numerous modules which add to the system complexity, part count, cost, and required layout area. However, recently Analog Devices has made available the AD5933, a system-on-a-chip (SoC) that contains an impedance converter network analyzer (Figure 7.7). It internally contains a frequency generator capable of synthesizing a wide range of frequencies up to 100 kHz with 27 bit resolution (0.1 Hz) using the direct digital synthesis (DDS) method. The output stage consists of the DDS core, a digital to analog converter (DAC), and a programmable gain amplifier (PGA) with an output resistance R_{out}. The generated sinusoid is applied as a test signal to the unknown impedance and the resulting response signal is passed to the input stage which consists of a current-to-voltage amplifier, a low-pass filter (LPF) and a 12-bit analog-to-digital converter (ADC). A 1024-point discrete Fourier transform (DFT) is then applied to the quantized values from the ADC to produce 16-bit signed values for the real and imaginary components of the DFT.

The configuration suggested by Analog Devices [96] has several drawbacks. The first drawback is that two measurement cycles must be conducted, one with a calibration resistance connected instead of the impedance, the second with the actual impedance connected. The calibration resistance allows for the voltage applied to the impedance to be estimated before measuring the current through the impedance in the second cycle. In order to minimize errors, the calibration resistance should be close in value to the impedance being measured. For the large impedance range of GLAD RH sensors, this two-cycle process would be very cumbersome and impractical to automate. The second drawback is the large error when measuring at lower resistances. The series output resistance R_{out} of the output stage typically ranges from around 200 Ω - 2.4 kΩ, depending on the selected output excitation voltage of the stage. If this value is not accounted for in the system calibration, then there will be large errors introduced when the measured impedance is in a similar range. Additional error at lower impedances comes from the higher current that flows through the device, requiring the I-V amplifier on the V_{in} pin to sink and source greater currents, potentially putting the amplifier in a nonlinear region of operation.

A configuration using two AD5933 SoCs that addresses the drawbacks of the configuration described above has been designed by Hoja *et al.* [97]. A block diagram of the design is shown in Figure 7.8. One SoC generates the test signal. The purpose of capacitor C, when combined with the series output resistance R_{out}, is to low-pass filter the generated signal after it passes through the DAC, removing higher order harmonics. The resulting signal after filtering, u_o, is the test signal applied to the rest of the circuit and causes a current to flow through the range resistor R_R and the unknown impedance Z_x. Ideally no current flows into the input of the operational amplifiers, so i_x = i_R. The voltage-follower op amp configuration, shown in red in Figure 7.8, is used to measure the voltage across the unknown impedance u_x. Because of the feedback loop from the output to the inverting input

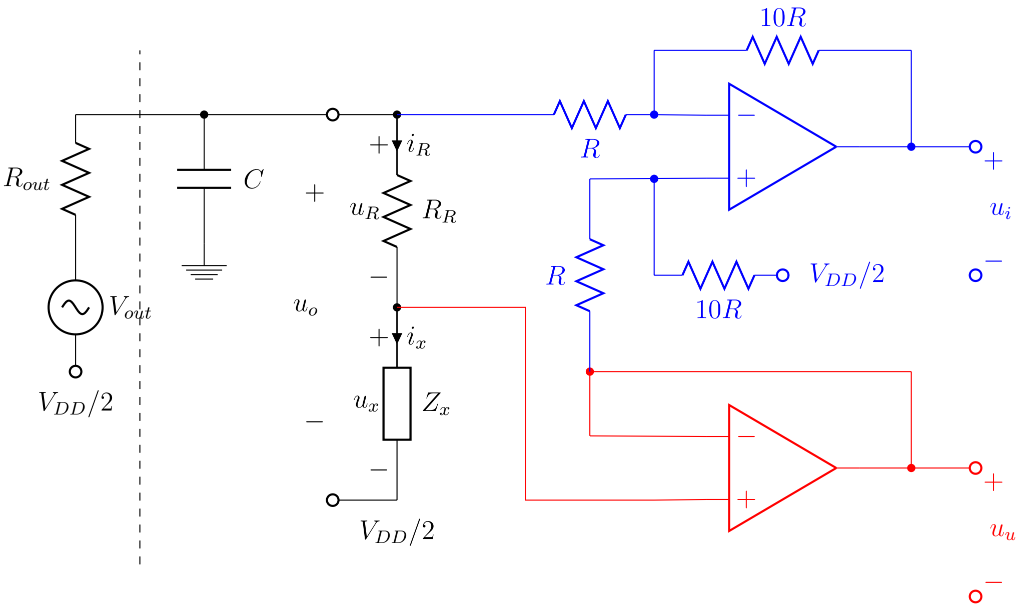


Figure 7.8: Block diagram of circuit configuration suggested by Hoja *et al.*

the voltage on the output mirrors the voltage on the input, and $u_u = u_x$. This circuit buffers the voltage because of the high input impedance and low output impedance of the op amp and eliminates loading effects. The second op amp configuration, shown in blue in Figure 7.8, is a differential configuration which is used to measure the current i_R through resistor R_R by measuring the voltage drop across the resistor. The differential amplifier outputs the difference between two voltages multiplied by a gain factor determined by the resistors. The values of R and $10R$ for the resistances of the differential amplifier sets a gain factor of 10, which will be explained later. The general output of the amplifier is:

$$V_{out} = 10(V_2 - V_1) \tag{7.8}$$

From the circuit diagram we get

$$u_i = 10(u_o - u_x) = 10u_R = 10R_R i_x \tag{7.9}$$

Rearranging and substituting we can get an equation to determine the unknown impedance:

$$Z_x = \frac{u_x}{i_x} = 10R_R \frac{u_u}{u_i} \tag{7.10}$$

Signals u_u and u_i are sampled by the two SoCs and then a DFT is applied to them. The result of the transform are the complex values U_u and U_i which represent the amplitude and phase of the measured sinusoid signal. The real and imaginary components of these values are stored separately in the AD5933 as 16-bit signed numbers. Thus we can modify Equation 7.10 to get the following equation for the magnitude of the unknown impedance:

$$|Z_x| = 10R_R\sqrt{\frac{(Re\{U_u\})^2 + (Im\{U_u\})^2}{(Re\{U_i\})^2 + (Im\{U_i\})^2}} \tag{7.11}$$

The impedance phase is:

$$\theta_x = \arctan\frac{Im\{U_u\}}{Re\{U_u\}} - \arctan\frac{Im\{U_i\}}{Re\{U_i\}} \tag{7.12}$$

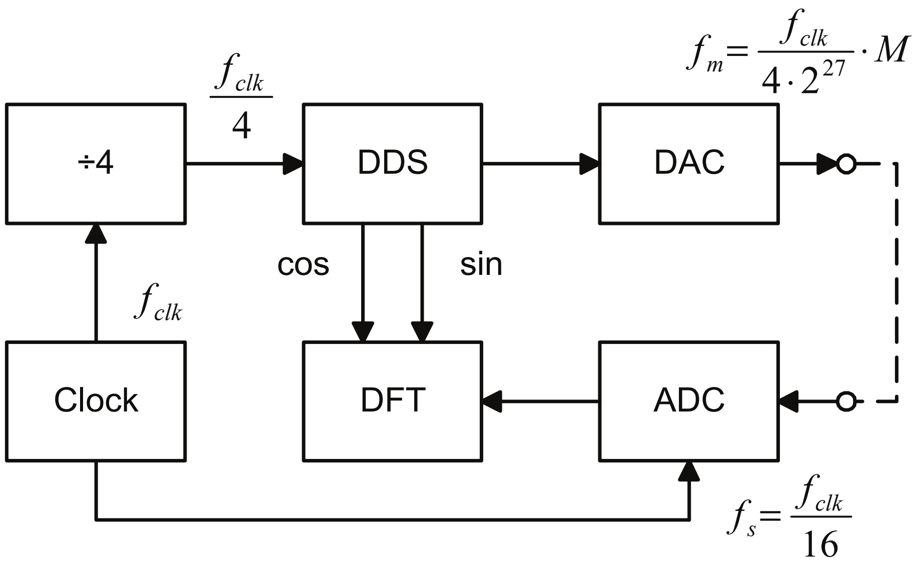
7.3 Implementation

In order to implement the circuit described above, the hardware needed to be designed carefully and there were many considerations that were made when selecting components, especially for the analog circuitry.

To minimize the error when measuring a large impedance range, multiple range resistor values must be used. By choosing a value for R_R that is an order lower than the magnitude of the unknown impedance $|Z_x|$ the noise on the unknown impedance is reduced. Amplifying the signal u_R by a gain factor of 10 ensures that the amplitude of signals u_u and u_i are a similar range. The resistors are switched in decades according to the following criteria, as suggested by Hoja *et al.*:

$$0.01|Z_x| < R_R \leq 0.1|Z_x| \tag{7.13}$$

The system is designed to be able to measure impedances from 100 Ω to 1 GΩ if the proper range resistors are used. Currently five 1% precision range resistors that span four decades from 10 Ω to 100 kΩ are used, allowing a measurement range from 100 Ω to 1 MΩ. Miniature reed relays are used to switch the resistances. The advantages that reed relays have over other types is that they have very fast switching speeds, require little power, and can be miniaturized. MEDER SIL05-1A72-71L relay switches were used and have a contact resistance $<$ 100 mΩ, contact capacitance of 0.3 pF and maximum switching time of 0.7 ms. The



$\frac{f_{clk}}{4}$
$f_m = \frac{f_{clk}}{4 \cdot 2^{27}} \cdot M$
÷4
DDS
DAC
cos
sin
f_{clk}
Clock
DFT
ADC
$f_s = \frac{f_{clk}}{16}$

Figure 7.9: Block diagram of clock signals in the AD5933. Image reproduced from [98] with permission.

contact resistance is added in series with the range resistors, so it is very important to have a small contact resistance for the 10 Ω resistor. A 74AC138 1-of-8 decoder/demultiplexer is used to switch the relays.

7.3.1 Frequency Synthesis

The AD5933 uses the direct digital synthesis (DSS) method to generate the output sinusoidal signal, which is also used as input for the discrete Fourier transform (DFT) calculation. DDS works by generating the phase of a signal and then using a look-up table and digital-to-analog converter (DAC) to convert the phase into the desired signal. The phase of any periodic waveform such as a sinusoid is a sawtooth function, which is easy to implement by continuously incrementing a counter, known as the phase accumulator, by a constant value stored in the phase increment register (PIR). The phase accumulator in the AD5933 is 27 bits, and the frequency resolution is controlled by a 24-bit word. The output frequency f_m is determined as follows:

$$f_m = \frac{\text{PIR}}{2^{27}} \cdot \frac{f_{clk}}{4} \tag{7.14}$$

where f_{clk} is the system clock frequency and PIR is the value stored in the phase increment register. The maximum system clock frequency that can by used is 16.776

MHz. Figure 7.9 shows the clock signals in the AD5933. With this method a large range of arbitrary frequencies may be generated with the frequency resolution determined by the size of the phase accumulator. Higher frequencies requiring large phase increases have correspondingly larger steps in the staircase approximation of the output waveform, resulting in larger harmonics and a higher signal noise floor. Smaller phase increases are more desirable because they produce a better staircase approximation which is more easily smoothed by a low-pass filter.

7.3.2 Discrete Fourier Transform

On the receive side of the AD5933, a discrete Fourier transform (DFT) is processed by an on-board digital signal processor (DSP). The DFT algorithm takes 1024 samples and using the sinusoid generated by the DDS core produces a real and imaginary value for that frequency. The window sampled by the DFT is assumed to be periodic, so it is important that the sampled signal contains an integer number of periods L. If it does not, then there will be spectrum leakage, which means that some of the power of the signal will 'leak' to other frequencies during computation of the DFT. In order to reduce spectrum leakage and improve accuracy we require that the sampling period contains an integer number of periods of the measured signal:

$$\frac{\mathrm{N}}{f_s} = \frac{\mathrm{L}}{f_m} \tag{7.15}$$

where L is an integer number of periods, $f_s = f_{clk}/16$ is the ADC sampling frequency, N = 1024 is the number of samples, and f_m is the measuring signal frequency. Substituting and simplifying, we get the following relation:

$$f_m = \frac{\mathrm{L} f_{clk}}{16384} \tag{7.16}$$

As described above, it is desirable that f_m and therefore L are as small as possible. Based on the requirements established, multiple system clock frequencies are necessary to cover a large frequency range. The solution to this proposed by Hoja *et al.* is to scale the system clock frequency supplied to the AD5933 [97, 98]. Each decade of generated frequencies has a corresponding f_{clk} (Table 7.1). The measured frequencies in each decade are spaced evenly by Δf_m. The time required

Table 7.1: Probe signal measurement parameters. Reproduced from [97]

f_{clk} [Hz]	f_m [Hz]	Δf_m [Hz]	L	T_{meas} [s]
4.096 M	10-100 k	10 k	40-400	4 m
819.2 k	1-9 k	1 k	20-180	20 m
	100-900	100	2-18	
81.92 k	10-90	10	2-18	0.2
8.192 k	1-9	1	2-18	2
819.2	0.1-0.9	0.1	2-18	20

for the measurement T_{meas} depends on the sampling frequency (and thus system clock frequency) and the number of samples N. It is determined by the following equation:

$$T_{meas} = \frac{16384}{f_{clk}} = \frac{\mathrm{L}}{f_m} \tag{7.17}$$

The master clock for both AD5933 systems was generated using the pulse width modulation (PWM) output on the microcontroller on the DLP-2232PB-G module. The 20 MHz crystal oscillator on the DLP-2232PB-G module was replaced with a 16.384 MHz crystal oscillator so that the frequencies in Table 7.1 could be generated. With this clock frequency 4.096 MHz is the maximum frequency that may be generated by the PWM output.

It is important that both AD5933 SoCs are synchronized when taking measurements, since any phase difference in the generated sinusoids will affect the measurements. Even though the output signal is only used from one AD5933, the DFT algorithm internally uses the generated sinusoid to calculate the resulting real and imaginary values. The Inter-Integrated Circuit (I^{2}C) bus protocol is used to communicate with the AD5933 systems and requires two bidirectional lines: a serial data line (SDA) and a serial clock (SCL). Multiple devices with different addresses can share the two-wire interface without interference, however the address of the AD5933 is fixed and cannot be changed. The daughter-boards have one SCL line that both AD5933's use and two SDA lines, one for each AD5933. Both SDA lines are used to send the same command string when starting a measurement sequence so that both AD5933's are synchronized. The results are read by disabling one of the SDA lines and reading from one AD5933 at a time. On the mother-board the

SDA lines are multiplexed using two 74HC4051D 8-channel analog multiplexor/demultiplexors. The SDA line from the microcontroller is split into two lines, each going to its own multiplexor. The multiplexors are used to select the channel that the measurements are being done on. When sending commands to both AD5933s, both multiplexors are enabled. When communicating with one AD5933 on a channel, only one multiplexor is enabled while the other is disabled. This allows the two SoCs to be synchronized, while allowing each to be addressed individually.

7.4 Firmware

Firmware for the PIC16F877A microcontroller on the DLP-2232PB-G module was written in assembly and programmed over USB using the flash program utility supplied by DLP Design. The FT2232D Dual UART/FIFO IC on-board the DLP-2232PB-G module is used to convert communication from the USB to two serial/parallel channels, which are interfaced with the microcontroller. One channel is used solely for on-board flash programming of the PIC microcontroller, while the other channel is configured as a parallel asynchronous FIFO (first-in-first-out) interface and used for sending data in binary format to the computer. This channel may be accessed with any terminal program as a virtual COM port using the VCP driver, or may be incorporated into software using the DLL software interface with the D2XX driver. Currently the VCP is used with simple terminal software, but may be expanded in the future.

7.5 Testing Results

The performance of the impedance analyzer was tested by measuring the values of 10 resistors logarithmically spaced from 100 Ω to 3.3 MΩ and 9 capacitors logarithmically spaced from 1 nF to 2.2 μF, with frequencies from 0.1 Hz to 10 kHz. These values were compared to those measured with the Quadtech 1920 Precision LCR meter. The resistance and capacitance measured by the impedance analyzer are denoted by R_{meas} and C_{meas} respectively. The values measured by the Quadtech meter are assumed to be the nominal values and are denoted by R_{nom} and C_{nom}.

The meter has a measurement error of 2.4 % (see Chapter 5).

Figures 7.10 and 7.11 show plots of R_{meas} vs. R_{nom} at 100 Hz and 10 kHz respectively, and Figure 7.13 shows a plot of C_{meas} vs. R_{nom} at 100 Hz. The values in these plots span a wide range. A least squares regression technique for fitting a slope to the data will be dominated by the large data points. A fitting technique derived by Steele [40] based on fitting the logarithm was used to fit a linear slope to the data:

$$y = mx \tag{7.18}$$

To prevent large data values from dominating the fit, Equation 7.18 was converted to logarithmic space:

$$log(y) = log(m) + log(x) \tag{7.19}$$

This equation was written in linear form by applying the transformation $Y = log(y)$, $M = log(m)$, and $X = log(x)$:

$$Y = M + X \tag{7.20}$$

The analytical expression for the slope parameter is:

$$M = \frac{\Sigma Y_i - \Sigma X_i}{N} \tag{7.21}$$

where i is the element index in the data set and N is the total number of elements in the data set. From this the slope m may be found:

$$m = 10^M \tag{7.22}$$

Error propagation was applied to Equation 7.21 to determine the error of the slope parameter, M:

$$\delta_M = \sqrt{\frac{1}{N(N-2)} \sum_i^N (Y_i - M - X_i)^2} \tag{7.23}$$

The error of the slope m is found by:

$$\delta_m = ln(10) m \delta_M \tag{7.24}$$

The relative slope error, δ_m/m, can be used to estimate the accuracy. It represents the range of nominal values, x, that correspond to the measured value, y.

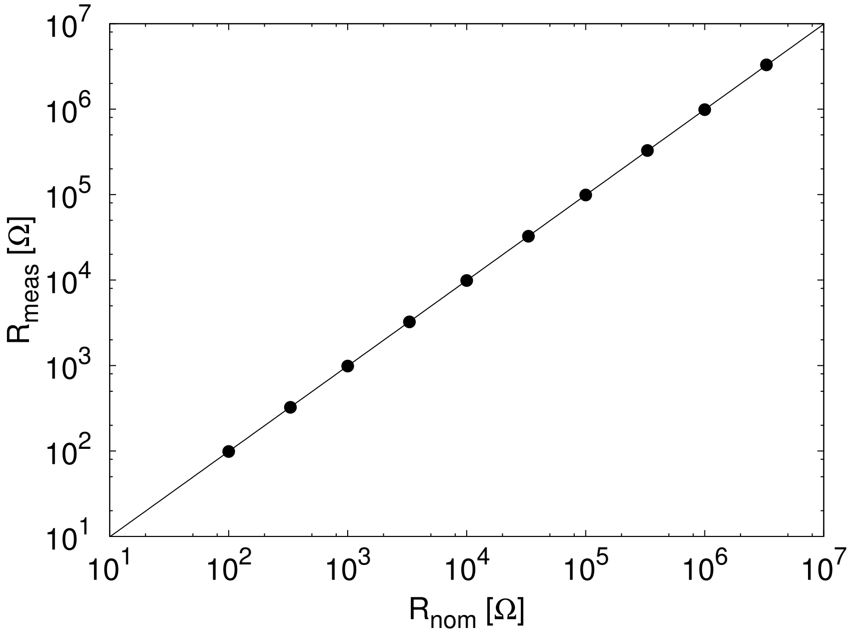


Figure 7.10: Nominal versus measured resistance at 100 Hz.

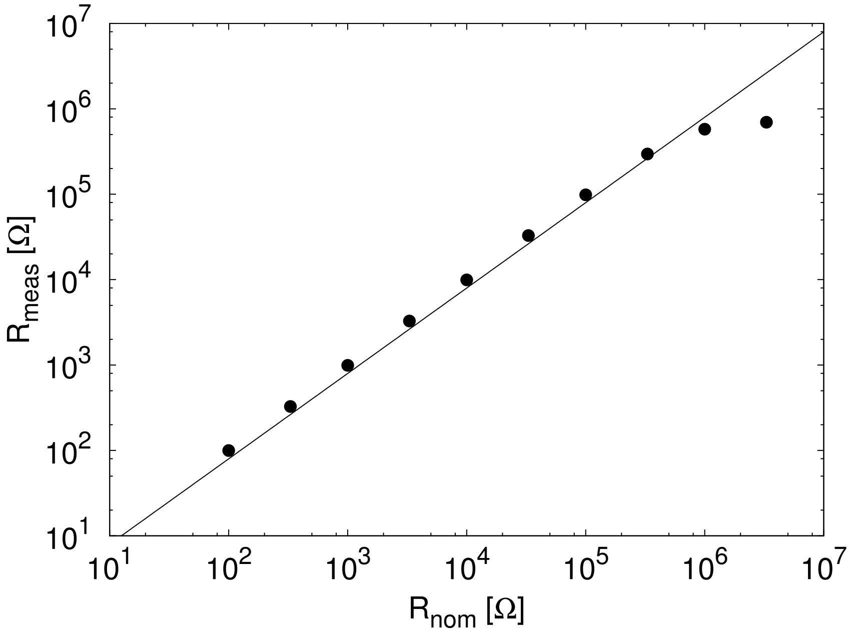


Figure 7.11: Nominal versus measured resistance at 10 kHz.

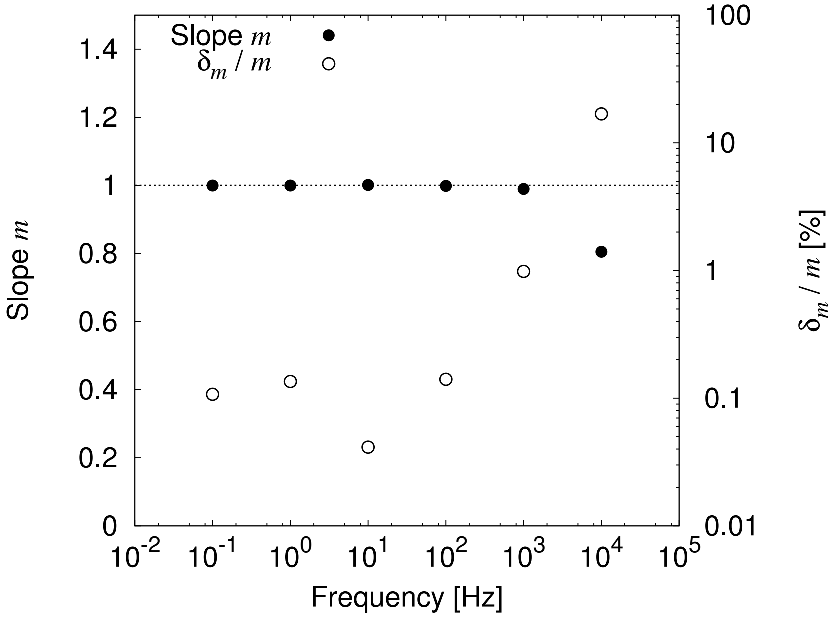


Figure 7.12: The fit slope, m, and relative slope error, δ_m/m, of resistances measured across a range of frequencies. The dotted line corresponds to a slope of 1.

Figure 7.12 compares the fit slopes and relative slope errors across a range of frequencies for resistive measurements. For measurements below 10 kHz, the slope m ranged from 0.9896 to 1.0013 and the relative slope error, δ_m/m which is considered to be the precision, ranged from 0.04 % to 0.98 %. At 10 kHz $m = 0.8050$ and $\delta_m/m = 16.8$ %. The reason for the inaccurate slope at 10 kHz is the deviation of the measurements from the nominal value at higher resistance (Figure 7.11). At low resistance, the slope of the values is quite accurate. The inaccuracies present at 10 kHz could be because there is a low number of steps in the staircase approximation when generating the sinusoid waveform (see Section 7.3.1).

Figure 7.14 compares the fit slopes and relative slope errors across a range of frequencies for capacitive measurements. Note that capacitive measurements at frequencies that resulted in measured impedances outside the designed range of 100 Ω to 1 MΩ were omitted. For example, a 1 nF capacitor has an impedance of 1.6 GΩ at 0.1 Hz, much higher than the acceptable measurement range. The fit slopes, m, ranged from 0.9601 to 1.0256 and the relative slope error, δ_m/m which is considered to be the precision, ranged from 0.86 % to 4.86 %. The higher error at low frequencies could be because the impedance of capacitors becomes very

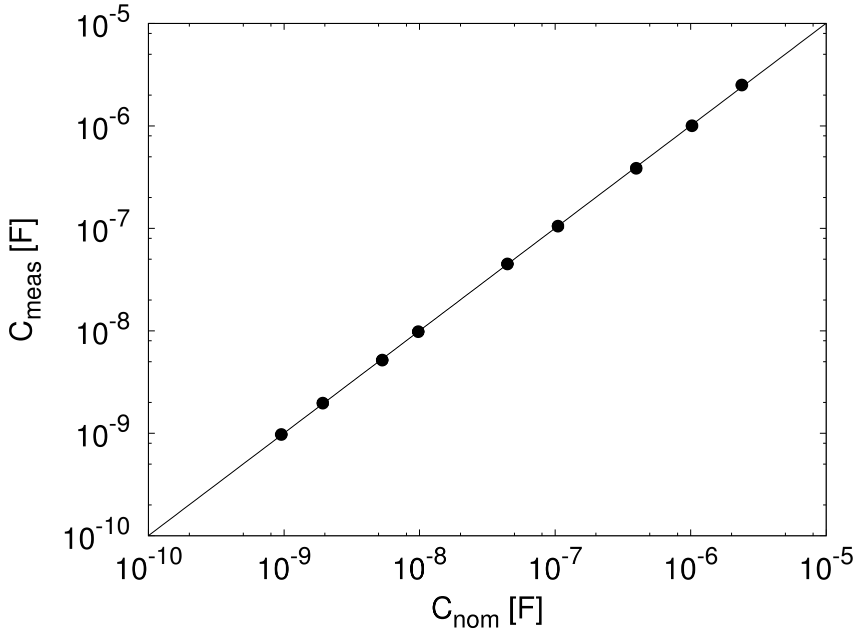


Figure 7.13: Nominal versus measured capacitance at 100 Hz.

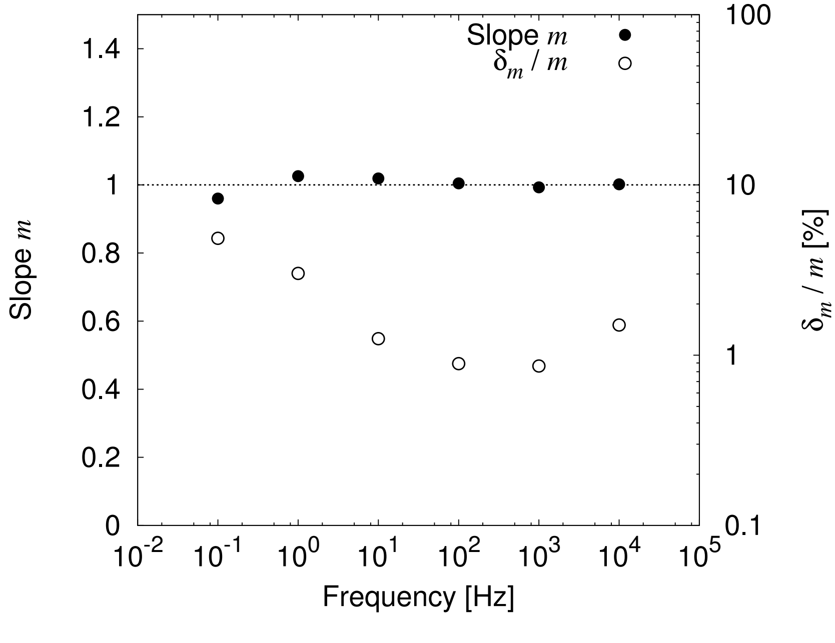


Figure 7.14: The fit slope, m, and relative slope error, δ_m/m, of capacitances measured across a range of frequencies. The dotted line corresponds to a slope of 1.

large at small frequencies, and several measurements had to be omitted at the lower frequencies.

7.6 Conclusion

The custom-built impedance analyzer described in this chapter used two AD5933 SoC's in a circuit configuration that measures the current and voltage across the DUT to calculate the impedance. The impedance analyzer is capable of generating test frequencies from 0.1 Hz to 100 kHz and measuring impedances from 100 Ω to 1 MΩ, expandable to 1 GΩ. Initial testing showed that the accuracy of the impedance analyzer is usually on the order of 0.1 - 1 %, depending on the frequency and type of measurement. The 8 separate channels of the device allow for parallel testing of several sensors and makes possible new future experiments that were not possible with the single-channel Quadtech LCR meter.

Chapter 8

Conclusions and Future Work

8.1 Summary of Results

The research presented in this thesis addresses the current lack of humidity sensors with sub-second response times and high sensitivities required for some applications. The motivation was to investigate the performance enhancement and stabilization of high-speed, high-sensitivity GLAD RH sensors using UV treatments. The goal was to lay the groundwork for future development of a sensing platform that incorporates a GLAD RH sensor and stabilizing UV LEDs in a single deployable device.

Capacitive relative humidity sensors were fabricated by depositing porous TiO_2 GLAD films on planar IDEs. The films were vertical posts deposited at an angle α = 80° and grown 1.5 μm thick. Sensor response times less than 250 ms are attainable, faster than any commercially available sensor. These sensors exhibit a capacitive response that spans over 3 orders of magnitude from 1 nF cm^{-2} to over 1 μF cm^{-2}. The sensitivity of these sensors is orders of magnitude larger than high-speed commercial sensors. The largest issue with GLAD RH sensors is ageing, which may be reversed with a UV treatment.

Initial UV treatment characterization used a mercury vapour lamp and optical filters with differing UV wavelengths to treat a set of sensors. A figure of merit (FOM) was defined to quantify the change in sensor response curves. In general, short UV wavelengths were observed to have larger FOMs. However there were significant differences in the power at each wavelength. At high power the effect saturates independent of wavelength. At lower powers there was a general trend of larger FOM with shorter wavelength.

The ageing study showed that on average UV treated sensors aged at a rate of -0.43 FOM/day. This was slower than untreated sensors and much slower than the rate of sensor regeneration, which was up to 30.5 FOM/day. Frequent humidity cycling had the effect of ageing the sensors more quickly.

Testing of treatment with UV LEDs showed that all LEDs with wavelengths below 387 nm, the bandgap of TiO_2, significantly enhanced the sensor performance. Of the UV LEDs tested, 370 nm was identified as the optimum wavelength to use for sensor regeneration and stabilization because it is significantly less expensive than the other LEDs, has the greatest optical output power, and enhanced sensors the quickest. This is important because it establishes that the 370 nm LED should be used for future experiments and sensor designs that incorporate a UV LED for stabilization.

Repeatability tests of treatment with 370 nm for 48 hours showed a large decrease in the variability of the FOM after the sensors were treated. The shape of the sensor response curve after treatment was duplicated with several treatments.

Sensor lifetime and stability were greatly improved using long-term UV LED illumination, which was the main goal of the research. The performance of GLAD sensors stabilized with UV LEDs was compared with the performance of commercially available sensors. The response time of less than 250 ms for GLAD sensors is much faster than that of commercial sensors, which are typically on the order of several seconds to 60 seconds. The sensitivity of 10 nF/%RH over 15 %RH for GLAD sensors is over four orders of magnitude larger than any commercial sensors compared. Most commercial sensors have a hysteresis of up to ±2% RH, which is comparable to the ±2.5% RH hysteresis of GLAD sensors. When stabilized with UV LEDs, the stability of GLAD sensors over 4 months was ±3.5% RH below approximately 50% RH. Above 50% RH the stability was worse with a drift of ±12 % RH or greater.

A custom 8-channel impedance analyzer was designed and built. It is capable of measuring impedance magnitude and phase with test frequencies ranging from

0.1 Hz to 100 kHz. It can measure impedances from 100 Ω up to 1 MΩ, expandable up to 1 GΩ with the addition of larger range resistors. The analyzer will allow for multiple sensors to be tested in parallel under the same humidity cycling conditions and is necessary for some of the future work suggested below.

8.2 Proposed Future Work

8.2.1 Morphology vs Response Characteristics

Currently a major issue with GLAD relative humidity sensors is the long-term stability at higher humidities. This is because the response is relatively flat at higher humidities, so a smaller drift in capacitance translates into a large change in the humidity reading. Despite the high sensitivity of 10 nF/%RH compared to commercial sensors, it is small compared to the absolute capacitance at higher humidities, which is over 1 μF. It may be possible to obtain sensor response curves that have more desirable stability, sensitivity, or accuracy in certain ranges by modifying the morphology of the film. An extensive study investigating the relationship between morphology and the response characteristics may determine the morphology that obtains the best response characteristics.

8.2.2 UV Treatment Standardization

The UV treatment of sensors serves two purposes: enhancement of the response curve, and stabilization of the response. The initial enhancement in sensor response occurs over a number of days, depending on the time constant. After the sensor response is enhanced, the UV treatment serves to stabilize the sensor and counter ageing. It is possible that a mercury vapour lamp could be used as part of a standard sensor fabrication step to enhance the sensor response to a desired curve after which point UV LEDs integrated with the sensor stabilize it. Much less power is required to stabilize the response rather than enhance it, so lower current through the UV LEDs may be used, greatly extending the lifetime of the LEDs, and thus the lifetime of the sensors. Research needs to be done on finding a suitable treatment duration and power with the mercury vapour lamp to enhance sensor responses to get desired response characteristics. This may be coupled with the morphology research described above.

8.2.3 Time-Resolved Testing

Testing several sensors in parallel under different UV powers will be useful for examining the the time-resolved effects of treatment and ageing on the sensors. The 8-channel impedance analyzer that was constructed will allow for impedance data to be collected for multiple sensors while testing them under the same conditions. Time-resolved testing will show how the sensor response changes over a small time-scale under different conditions and will be useful for optimizing the required UV power for regeneration and stabilization.

8.2.4 UV LED Ageing

Extending the operational lifetime of the UV LEDs will will allow for the sensors to be stabilized for longer. The issue of LED ageing could be alleviated by using lower currents, or operating the LEDs in pulsed mode at higher currents. Using an array of LEDs would allow for lower currents to be used in each while still providing the same total output power. A photodiode could be used to monitor the optical power and maintain the same level by regulating the LED current.

8.2.5 Modelling

Modelling of the sensors could provide valuable insight into the physics behind the sensing mechanism. Existing equivalent circuit models for titania RH sensors may be applied to or modified for GLAD RH sensors and can be combined with impedance spectroscopy techniques. This type of work could lead to better sensor designs.

8.2.6 Sensing of Other Analytes

Research on the sensitivity and selectivity of GLAD titania sensors to analytes other than water vapour is extremely important work that needs to be done. There are many other analytes for which high sensitivity and high speed sensors are critical, such as H_2, CO, and H_2S [99]. This work would expand the potential applications of the research so far. It is also important to establish the selectivity of GLAD RH sensors since in the field there are often analytes present other than water vapour which could potentially affect the sensor performance.

8.2.7 Frequency Selectivity

Better performance characteristics such as sensitivity or stability in certain humidity ranges could potentially be achieved by measuring at different frequencies. It is also possible that different analytes will have different frequency responses, so improved selectivity between analytes could be achieved by probing at different frequencies. This will require detailed analysis of the frequency response of different analytes.

References

[1] Pieter R. Wiederhold. *Water Vapor Measurement: Methods and Instrumentation.* Marcel Dekker, New York, NY, 1997.

[2] P.L. Kebabian, C.E. Kolb, and A. Freedman. Spectroscopic water vapor sensor for rapid response measurements of humidity in the troposphere. *Journal of Geophysical Research: Atmospheres*, 107(D23):4670, 2002.

[3] C. Laville and C. Pellet. Comparison of three humidity sensors for a pulmonary function diagnosis microsystem. *IEEE Sensors Journal*, 2(2):96 –101, April 2002.

[4] C. Laville and C. Pellet. Interdigitated humidity sensors for a portable clinical microsystem. *IEEE Transactions on Biomedical Engineering*, 49(10):1162–1167, October 2002.

[5] A.F.P. van Putten, M.J.A.M. van Putten, M.H.P.M. van Putten, and P.F.A.M. van Putten. Multisensor microsystem for pulmonary function diagnostics. *IEEE Sensors Journal*, 2(6):636 – 643, December 2002.

[6] A. Tetelin, C. Pellet, C. Laville, and G. N'Kaoua. Fast response humidity sensors for a medical microsystem. *Sensors and Actuators B: Chemical*, 91(1-3):211–218, June 2003.

[7] T. Tatara and K. Tsuzaki. An apnea monitor using a rapid-response hygrometer. *Journal of Clinical Monitoring*, 13(1):5–9, January 1997.

[8] N. Andre, S. Druart, P. Gerard, R. Pampin, L. Moreno-Hagelsieb, T. Kezai, L.A. Francis, D. Flandre, and J.P. Raskin. Miniaturized wireless sensing system for real-time breath activity recording. *IEEE Sensors Journal*, 10(1):178–184, January 2010.

[9] H.B. Valman, B.M. Wright, and C. Lawrence. Measurement of respiratory rate in the newborn. *British Medical Journal*, 286(6380):1783–1784, 1983.

[10] Matthew M. Hawkeye and Michael J. Brett. Glancing angle deposition: Fabrication, properties, and applications of micro- and nanostructured thin films. *Journal of Vacuum Science & Technology A*, 25(5):1317–1335, 2007.

[11] M. T. Taschuk, K. M. Krause, J. J. Steele, M. A. Summers, and M. J. Brett. Growth scaling of metal oxide columnar thin films deposited by glancing angle depositions. *Journal of Vacuum Science & Technology B*, 27(5):2106–2111, 2009.

[12] John J. Steele and Michael J. Brett. Nanostructure engineering in porous columnar thin films: Recent advances. *Journal of Materials Science: Materials in Electronics*, 18(4):367–379, 2007.

[13] Kevin Robbie and Michael J. Brett. Method of depositing shadow sculpted thin films. U.S. Patent 5 866 204, February 2, 1999.

[14] M. T. Taschuk, M. M. Hawkeye, and M. J. Brett. Glancing angle deposition. In Peter Martin, editor, *Handbook of Deposition Technologies for Films and Coatings: Science, Applications and Technology*. Elsevier, Oxford, UK, 3rd edition, 2010.

[15] J.J. Steele, J.P. Gospodyn, J.C. Sit, and M.J. Brett. Impact of morphology on high-speed humidity sensor performance. *IEEE Sensors Journal*, 6(1):24 – 27, 2006.

[16] J.J. Steele, M.T. Taschuk, and M.J. Brett. Nanostructured metal oxide thin films for humidity sensors. *IEEE Sensors Journal*, 8(8):1422 –1429, 2008.

[17] J.J. Steele, G.A. Fitzpatrick, and M.J. Brett. Capacitive humidity sensors with high sensitivity and subsecond response times. *IEEE Sensors Journal*, 7(6):955 –956, 2007.

[18] J. J. Steele, M. T. Taschuk, and M. J. Brett. Response time of nanostructured relative humidity sensors. *Sensors and Actuators B: Chemical*, 140(2):610–615, 2009.

[19] Martin R. Kupsta, Michael T. Taschuk, Michael J. Brett, and Jeremy C. Sit. Reactive ion etching of columnar nanostructured TiO_2 thin films for modified relative humidity sensor response time. *IEEE Sensors Journal*, 9(12):1979–1986, 2009.

[20] G. Korotcenkov. Metal oxides for solid-state gas sensors: What determines our choice? *Materials Science and Engineering: B*, 139(1):1 – 23, 2007.

[21] TDK. *Humidity Sensor Units*, 2010.

[22] Humirel. *Relative Humidity Sensor HS1100/HS1101*, 2002.

[23] Honeywell. *HIH-5030/5031 Series Low Voltage Humidity Sensors*, 2010.

[24] Honeywell. *HCH-1000 Series Capacitive Humidity Sensor*, 2007.

[25] Vaisala. *Vaisala HUMICAP Humidity and Temperature Probe HMP110*, 2010.

[26] O. Carp, C.L. Huisman, and A. Reller. Photoinduced reactivity of titanium dioxide. *Progress in Solid State Chemistry*, 32(1-2):33–177, 2004.

[27] Akira Fujishima, Xintong Zhang, and Donald A. Tryk. TiO_2 photocatalysis and related surface phenomena. *Surface Science Reports*, 63(12):515–582, 2008.

[28] Michael T. Taschuk, John J. Steele, Andy C. van Popta, and Michael J. Brett. Photocatalytic regeneration of interdigitated capacitor relative humidity sensors fabricated by glancing angle deposition. *Sensors and Actuators B: Chemical*, 134(2):666 – 671, 2008.

[29] D. P. Smetaniuk, M. T. Taschuk, and M. J. Brett. Photocatalytic titanium dioxide nanostructures for self-regenerating relative humidity sensors. *IEEE Sensors Journal*, 11(8):1713 –1719, August 2011.

[30] Z.M. Rittersma. Recent achievements in miniaturised humidity sensors – A review of transduction techniques. *Sensors and Actuators A: Physical*, 96(2-3):196–210, 2002.

[31] N.T.T. Ha, D.K. An, P.V. Phong, P.T.M. Hoa, and L.H. Mai. Study and performance of humidity sensor based on the mechanical-optoelectronic principle for the measurement and control of humidity in storehouses. *Sensors and Actuators B: Chemical*, 66(1-3):200–202, July 2000.

[32] G. Gerlach and K. Sager. A piezoresistive humidity sensor. *Sensors and Actuators A: Physical*, 43(1-3):181–184, May 1994.

[33] S. Singamaneni, M.E. McConney, M.C. LeMieux, H. Jiang, J.O. Enlow, T.J. Bunning, R.R. Naik, and V.V. Tsukruk. Polymer-silicon flexible structures for fast chemical vapor detection. *Advanced Materials*, 19(23):4248 – 4255, December 2007.

[34] B. Schirmer, H. Venzke, A. Melling, C.S. Edwards, G.P. Barwood, P. Gill, M. Stevens, R. Benyon, and P. Mackrodt. High precision trace humidity measurements with a fibre-coupled diode laser absorption spectrometer at atmospheric pressure. *Measurement Science & Technology*, 11(4):382–391, 2000.

[35] F. Mitschke. Fiber-optic sensor for humidity. *Optics Letters*, 14(17):967–969, 1989.

[36] John J. Steele, Andy C. van Popta, Matthew M. Hawkeye, Jeremy C. Sit, and Michael J. Brett. Nanostructured gradient index optical filter for high-speed humidity sensing. *Sensors and Actuators B: Chemical*, 120(1):213–219, 2006.

[37] T.L. Yeo, T. Sun, and K.T.V. Grattan. Fibre-optic sensor technologies for humidity and moisture measurement. *Sensors and Actuators A: Physical*, 144(2):280–295, 2008.

[38] E. Traversa. Ceramic sensors for humidity detection: the state-of-the-art and future developments. *Sensors and Actuators B: Chemical*, 23(2-3):135–156, 1995.

[39] Z. Chen and C. Lu. Humidity sensors: A review of materials and mechanisms. *Sensor Letters*, 3(4):274–295, 2005.

[40] John J. Steele. *Nanostructured thin films for humidity sensing*. PhD thesis, University of Alberta, Canada, 2007.

[41] J.G. Korvink, L. Chandran, T. Bolthauser, and H. Baltes. Accurate 3D capacitance evaluation in integrated capacitive humidity sensors. *Sensors and Materials*, 4(6):323–335, 1993.

[42] N. Yamazoe and Y. Shimizu. Humidity sensors: Principles and applications. *Sensors and Actuators*, 10(3-4):379–398, 1986.

[43] A.C. Salaun, H.M. Kotb, T. Mohammed-Brahim, F. Le Bihan, H. Lhermite, and F. Bendriaa. Suspended-gate thin film transistor as highly sensitive humidity sensor. volume 5836 of *Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE)*, pages 231–238, 2005.

[44] Sung Pil Lee. Fabrication and sensing properties of a micro-humidity sensor system using CMOS technology. *Electronic Materials Letters*, 6(1):7–12, March 2010.

[45] Michael A. Henderson. The interaction of water with solid surfaces: Fundamental aspects revisited. *Surface Science Reports*, 46(1-8):1 – 308, 2002.

[46] T. Morimoto, M. Nagao, and F. Tokuda. Relation between amounts of chemisorbed and physisorbed water on metal oxides. *Journal of Physical Chemistry*, 73(1):243–248, 1969.

[47] McCafferty E. and Zettlemoyer A.C. Adsorption of water vapor on α-Fe_2O_3. *Discussions of the Faraday Society*, (52):239–254, 1971.

[48] C.J.T de Grotthuss. Sur la décomposition de l'eau et des corps qu'elle tient en dissolution à l'aide de l'électricité galvanique. *Annales Des Chimie Et Des Physique*, 58:54–73, 1806.

[49] Samuel Cukierman. Et tu, Grotthuss! and other unfinished stories. *Biochimica et Biophysica Acta-Bioenergetics*, 1757(8):876–885, 2006.

[50] Y. Shimizu, H. Arai, and T. Seiyama. Theoretical studies on the impedance-humidity characteristics of ceramic humidity sensors. *Sensors and Actuators*, 7(1):11 – 22, 1985.

[51] J.H. de Boer. *The Shapes of Capillaries, The Structure and Properties of Porous Materials*, pages 68 – 94. Butterworth, London, 1958.

[52] J.A. Thornton. High-rate thick-film growth. *Annual Review of Materials Science*, 7:239–260, 1977.

[53] K.M. Krause, M.T. Taschuk, K.D. Harris, D.A. Rider, N.G. Wakefield, J.C. Sit, J.M. Buriak, M. Thommes, and M.J. Brett. Surface area characterization of obliquely deposited metal oxide nanostructured thin films. *Langmuir*, 26(6):4368–4376, 2010.

[54] K.D. Harris, J.R. McBride, K.E. Nietering, and M.J. Brett. Fabrication of porous platinum thin films for hydrocarbon sensor applications. *Sensors and Materials*, 13(4):225–234, 2001.

[55] A.T. Wu, M. Seto, and M.J. Brett. Capacitive SiO humidity sensors with novel microstructures. *Sensors and Materials*, 11(8):493–505, 1999.

[56] A.T. Wu and M.J. Brett. Sensing humidity using nanostructured SiO posts: Mechanism and optimization. *Sensors and Materials*, 13(7):399–431, 2001.

[57] K.D. Harris, A. Huizinga, and M.J. Brett. High-speed porous thin film humidity sensors. *Electrochemical and Solid State Letters*, 5(11):H27–H29, 2002.

[58] R. Igreja and C.J. Dias. Analytical evaluation of the interdigital electrodes capacitance for a multi-layered structure. *Sensors and Actuators A: Physical*, 112(2-3):291–301, 2004.

[59] K.M. Krause, M. Thommes, and M.J. Brett. Pore analysis of obliquely deposited nanostructures by krypton gas adsorption at 87 K. *Microporous and Mesoporous Materials*, 143(1):166–173, August 2011.

[60] M. Ohring. *Materials Science of Thin Films*, chapter 2.2, pages 58–65. Academic Press, San Diego, CA, 2nd edition, 2002.

[61] W. Smith. *Foundations of Materials Science and Engineering*, chapter 5.2, pages 177–184. McGraw-Hill, Boston, MA, 4th edition, 2006.

[62] V. Adivarahan, J.P. Zhang, A. Chitnis, W. Shuai, J. Sun, R. Pachipulusu, M. Shatalov, and M.A. Khan. Sub-milliwatt power III-N light emitting diodes at 285 nm. *Japanese Journal of Applied Physics*, 41(4B):L435–L436, 2002.

[63] A. Yasan, R. McClintock, K. Mayes, D. Shiell, L. Gautero, S. R. Darvish, P. Kung, and M. Razeghi. 4.5 mW operation of AlGaN-based 267 nm deep-ultraviolet light-emitting diodes. *Applied Physics Letters*, 83(23):4701–4703, 2003.

[64] H. Amano, N. Sawaki, I. Akasaki, and Y. Toyoda. Metalorganic vapor phase epitaxial growth of a high quality GaN film using an AlN buffer layer. *Applied Physics Letters*, 48(5):353–355, 1986.

[65] J.P. Zhang, M.A. Khan, W.H. Sun, H.M. Wang, C.Q. Chen, Q. Fareed, E. Kuokstis, and J.W. Yang. Pulsed atomic-layer epitaxy of ultrahigh-quality $\mathrm{Al}_x\mathrm{Ga}_{1-x}\mathrm{N}$ structures for deep ultraviolet emissions below 230 nm. *Applied Physics Letters*, 81(23):4392–4394, December 2002.

[66] Bahaa E.A. Saleh. *Fundamentals of Photonics*. New York: Wiley, 1991.

[67] Sendill Gnanaeswaran. Study of energy band diagrams of Group III-Nitride heterostructures. Master's thesis, Texas Tech University, 2003.

[68] P. Kung and M. Razeghi. III-Nitride wide bandgap semiconductors: a survey of the current status and future trends of the material and device technology. *Opto-Electronics Review*, 8(3):201–239, September 2000.

[69] Yen-Kuang Kuo and Wen-Wei Lin. Band-gap bowing parameter of the $\mathrm{Al}_x\mathrm{Ga}_{1-x}\mathrm{N}$ derived from theoretical simulation. *Japanese Journal of Applied Physics*, 41(Part 1, No. 1):73–74, 2002.

[70] Toshio Nishida, Hisao Saito, and Naoki Kobayashi. Efficient and high-power AlGaN-based ultraviolet light-emitting diode grown on bulk GaN. *Applied Physics Letters*, 79(6):711–712, 2001.

[71] B. Monemar and G. Pozina. Group III-Nitride based hetero and quantum structures. *Progress in Quantum Electronics*, 24(6):239 – 290, 2000.

[72] M. Razeghi and R. McClintock. A review of III-Nitride research at the Center for Quantum Devices. *Journal of Crystal Growth*, 311(10):3067 – 3074, 2009.

[73] J. Zhang, X. Hu, A. Luneva, J. Deng, Y. Bilenko, T.M. Katona, M.S. Shur, R. Gaska, and M.A. Khan. AlGaN deep-ultraviolet light-emitting diodes. *Japanese Journal of Applied Physics*, 44(10):7250–7253, 2005.

[74] M. Razeghi, A. Yasan, R. McClintock, K. Mayes, D. Shiell, S. Ramezani Darvish, and P. Kung. Review of III-Nitride optoelectronic materials for light emission and detection. *physica status solidi (c)*, 1(2):141–148, 2004.

[75] M.H. Crawford, A.A. Allerman, A.J. Fischer, K.H.A. Bogart, S.R. Lee, R.J. Kaplar, W.W. Chow, and D.M. Follstaedt. Optimization and performance of AlGaN-based multi-quantum-well deep-UV LEDs. volume 5366, pages 75–84. SPIE Proceedings, 2004.

[76] H. Hirayama, T. Yatabe, N. Noguchi, and N. Kamata. Development of 230-270 nm AlGaN-based deep-UV LEDs. *Electronics and Communications in Japan*, 93(3):24–33, 2010.

[77] M. Suzuki, T. Ito, and Y. Taga. Photocatalysis of sculptured thin films of TiO_2. *Applied Physics Letters*, 78(25):3968–3970, 2001.

[78] Y. P. He, Z. Y. Zhang, and Y. P. Zhao. Optical and photocatalytic properties of oblique angle deposited TiO_2 nanorod array. *Journal of Vacuum Science & Technology B*, 26(4):1350–1358, 2008.

[79] R. Wang, K. Hashimoto, A. Fujishima, M. Chikuni, E. Kojima, A. Kitamura, M. Shimohigoshi, and T. Watanabe. Light-induced amphiphilic surfaces. *Nature*, 388(6641):431–432, 1997.

[80] Masatoshi Nakamura, Koumei Makino, Lucel Sirghi, Toru Aoki, and Yoshinori Hatanaka. Hydrophilic properties of hydro-oxygenated TiO_x films prepared by plasma enhanced chemical vapor deposition. *Surface and Coatings Technology*, 169-170:699 – 702, 2003.

[81] L. Greenspan. Humidity fixed points of binary saturated aqueous solutions. *Journal of Research of the National Bureau of Standards – A. Physics and Chemistry*, 81(1):89–96, 1977.

[82] J. Grönblad. Easy and reliable calibration with the HMK15 humidity calibrator. *Vaisala News Magazine*, 148:18–20, 1998.

[83] N. Sakai, K. Fukuda, T. Shibata, Y. Ebina, K. Takada, and T. Sasaki. Photoinduced hydrophilic conversion properties of titania nanosheets. *Journal of Physical Chemistry B*, 110(12):6198–6203, 2006.

[84] N. Sakai, A. Fujishima, T. Watanabe, and K. Hashimoto. Quantitative evaluation of the photoinduced hydrophilic conversion properties of TiO_2 thin film surfaces by the reciprocal of contact angle. *Journal of Physical Chemistry B*, 107(4):1028–1035, 2003.

[85] Z. Lin, K. Liu, Y.C. Zhang, X.J. Yue, G.Q. Song, and D.C. Ba. The microstructure and wettability of the TiO_x films synthesized by reactive DC magnetron sputtering. *Materials Science and Engineering B*, 156(1-3):79–83, 2009.

[86] T. Zubkov, D. Stahl, T.L. Thompson, D. Panayotov, O. Diwald, and J.T. Yates. Ultraviolet light-induced hydrophilicity effect on TiO_2(110)(1x1). Dominant role of the photooxidation of adsorbed hydrocarbons causing wetting by water droplets. *Journal of Physical Chemistry B*, 109(32):15454–15462, 2005.

[87] J.M. White, J. Szanyi, and M.A. Henderson. The photon-driven hydrophilicity of titania: A model study using TiO_2(110) and adsorbed trimethyl acetate. *Journal of Physical Chemistry B*, 107(34):9029–9033, 2003.

[88] B.M. Kulwicki. Ceramic sensors and transducers. *Journal of Physics and Chemistry of Solids*, 45(10):1015–1031, 1984.

[89] E. Traversa, G. Gnappi, A. Montenero, and G. Gusmano. Ceramic thin films by sol-gel processing as novel materials for integrated humidity sensors. *Sensors and Actuators B: Chemical*, 31(1-2):59–70, 1996.

[90] Tracy L. Thompson and John T. Yates, Jr. Surface science studies of the photoactivation of TiO_2 – New photochemical processes. *Chemical Reviews*, 106(10):4428–4453, 2006.

[91] M. Meneghini, L.-R. Trevisanello, G. Meneghesso, and E. Zanoni. A Review on the reliability of GaN-based LEDs. *IEEE Transactions on Device and Materials Reliability*, 8(2):323 –331, 2008.

[92] Nichia. *Specifications for Nichia UV LED*.

[93] P.M. Faia, C.S. Furtado, and A.J. Ferreira. AC impedance spectroscopy: A new equivalent circuit for titania thick film humidity sensors. *Sensors and Actuators B: Chemical*, 107(1):353 – 359, 2005.

[94] P.M. Faia, A.J. Ferreira, and C.S. Furtado. Establishing and interpreting an electrical circuit representing a TiO_2-WO_3 series of humidity thick film sensors. *Sensors and Actuators B: Chemical*, 140(1):128–133, 2009.

[95] Agilent. *Agilent Impedance Measurment Handbook: A Guide to measurement technology and techniques*, 4th edition, March 2009.

[96] Analog Devices. *AD5933 1 MSPS, 12-Bit Impedance Converter, Network Analyzer, Rev. C*, 2010.

[97] Jerzy Hoja and Grzegorz Lentka. Interface circuit for impedance sensors using two specialized single-chip microsystems. *Sensors and Actuators A: Physical*, 163(1):191–197, September 2010.

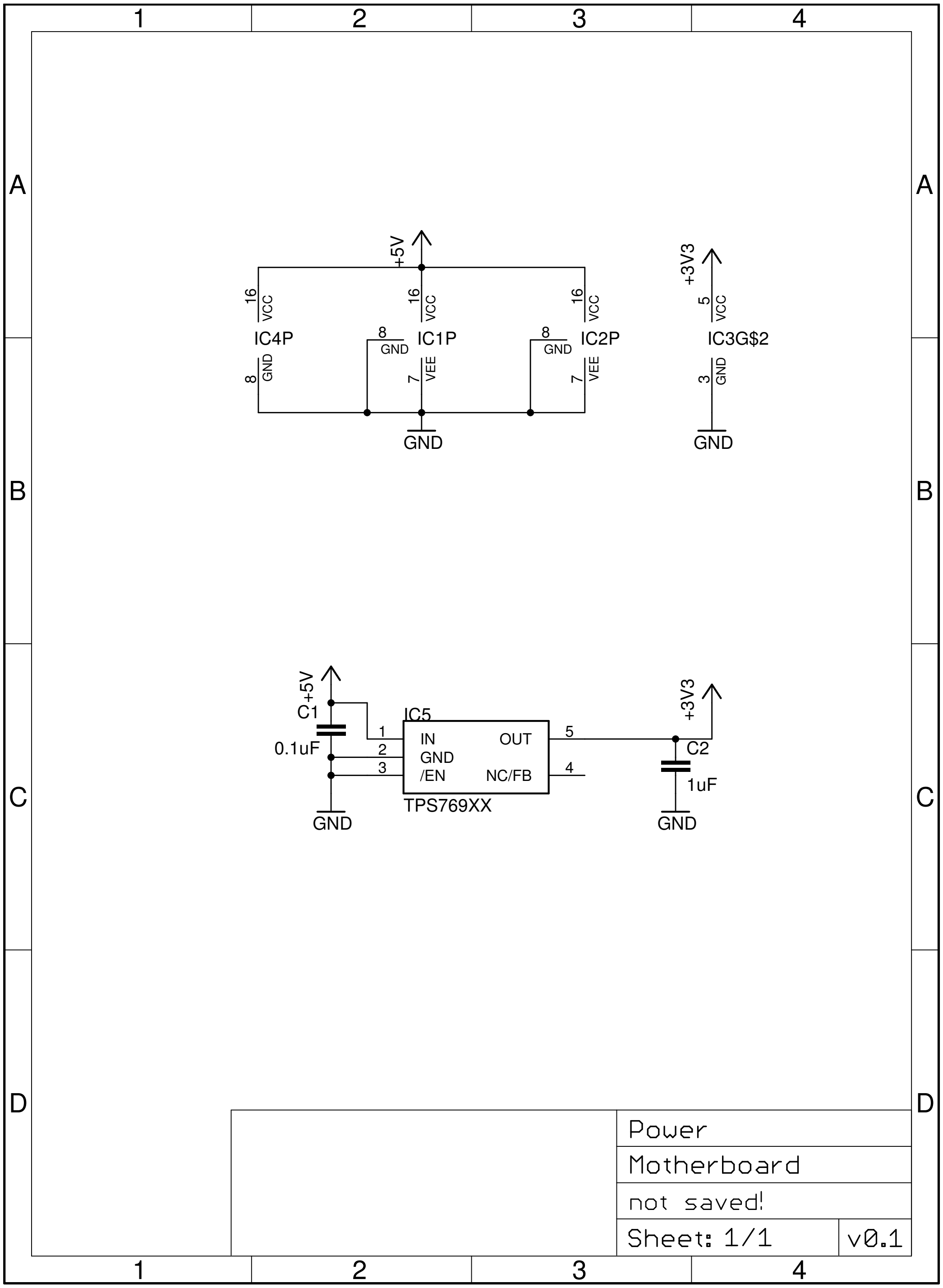
[98] Jerzy Hoja and Grzegorz Lentka. Portable analyzer for impedance spectroscopy. In *XIX IMEKO World Congress: Fundamental and Applied Metrology, Proceedings*, pages 497–502, 2009.

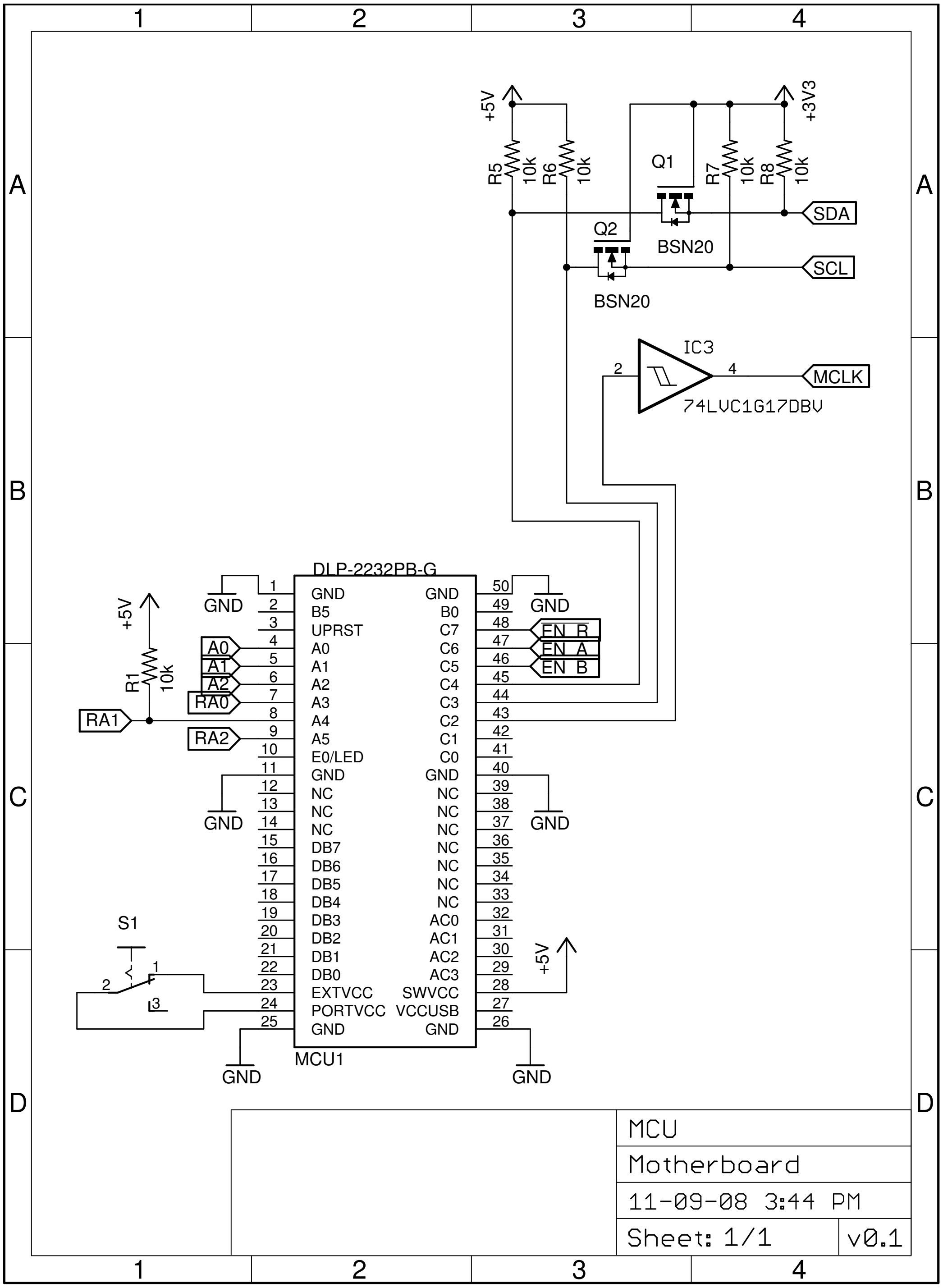
[99] Chengxiang Wang, Longwei Yin, Luyuan Zhang, Dong Xiang, and Rui Gao. Metal oxide gas sensors: Sensitivity and influencing factors. *Sensors*, 10(3):2088–2106, 2010.

Appendix A

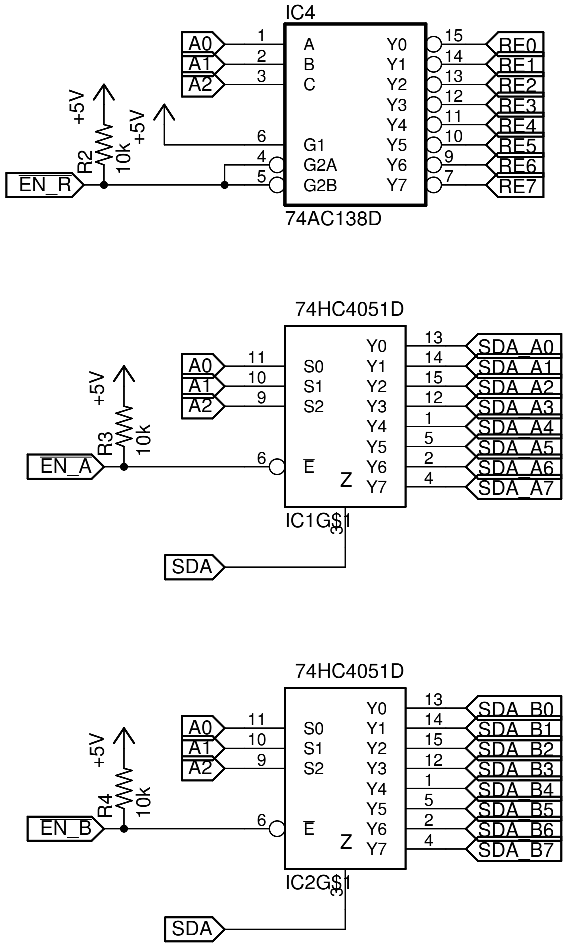
Impedance Analyzer Schematics

In this appendix schematic diagrams for the impedance analyzer electronics described in Chapter 7 are presented. Four schematics for the motherboard show the power connections, connections to the DLP-2232PB-G module containing the PIC microcontroller, the signal multiplexors, and the eight port headers on the board. Two schematics are shown for the daughterboards: one for the power connections, and one for the analog measurement electronics.


1
2
3
4
A
B
C
D
+5V
16
VCC
IC4P
8
GND
16
VCC
8
GND
IC1P
7
VEE
16
VCC
8
GND
IC2P
7
VEE
GND
+3V3
5
VCC
IC3G$2
3
GND
GND
+5V
C1
0.1uF
IC5
1
IN
2
GND
3
/EN
OUT
5
NC/FB
4
TPS769XX
GND
+3V3
C2
1uF
GND
Power
Motherboard
not saved!
Sheet: 1/1
v0.1



1
2
3
4
A
B
C
D
+5V
+3V3
R5
10k
R6
10k
R7
10k
R8
10k
Q1
BSN20
Q2
BSN20
SDA
SCL
IC3
2
4
MCLK
74LVC1G17DBV
DLP-2232PB-G
MCU1
+5V
R1
10k
A0
A1
A2
RA0
RA1
RA2
EN_R
EN_A
EN_B
GND
S1
1
2
3
1 GND
2 B5
3 UPRST
4 A0
5 A1
6 A2
7 A3
8 A4
9 A5
10 E0/LED
11 GND
12 NC
13 NC
14 NC
15 DB7
16 DB6
17 DB5
18 DB4
19 DB3
20 DB2
21 DB1
22 DB0
23 EXTVCC
24 PORTVCC
25 GND
50 GND
49 B0
48 C7
47 C6
46 C5
45 C4
44 C3
43 C2
42 C1
41 C0
40 GND
39 NC
38 NC
37 NC
36 NC
35 NC
34 NC
33 NC
32 AC0
31 AC1
30 AC2
29 AC3
28 SWVCC
27 VCCUSB
26 GND
MCU
Motherboard
11-09-08 3:44 PM
Sheet: 1/1
v0.1

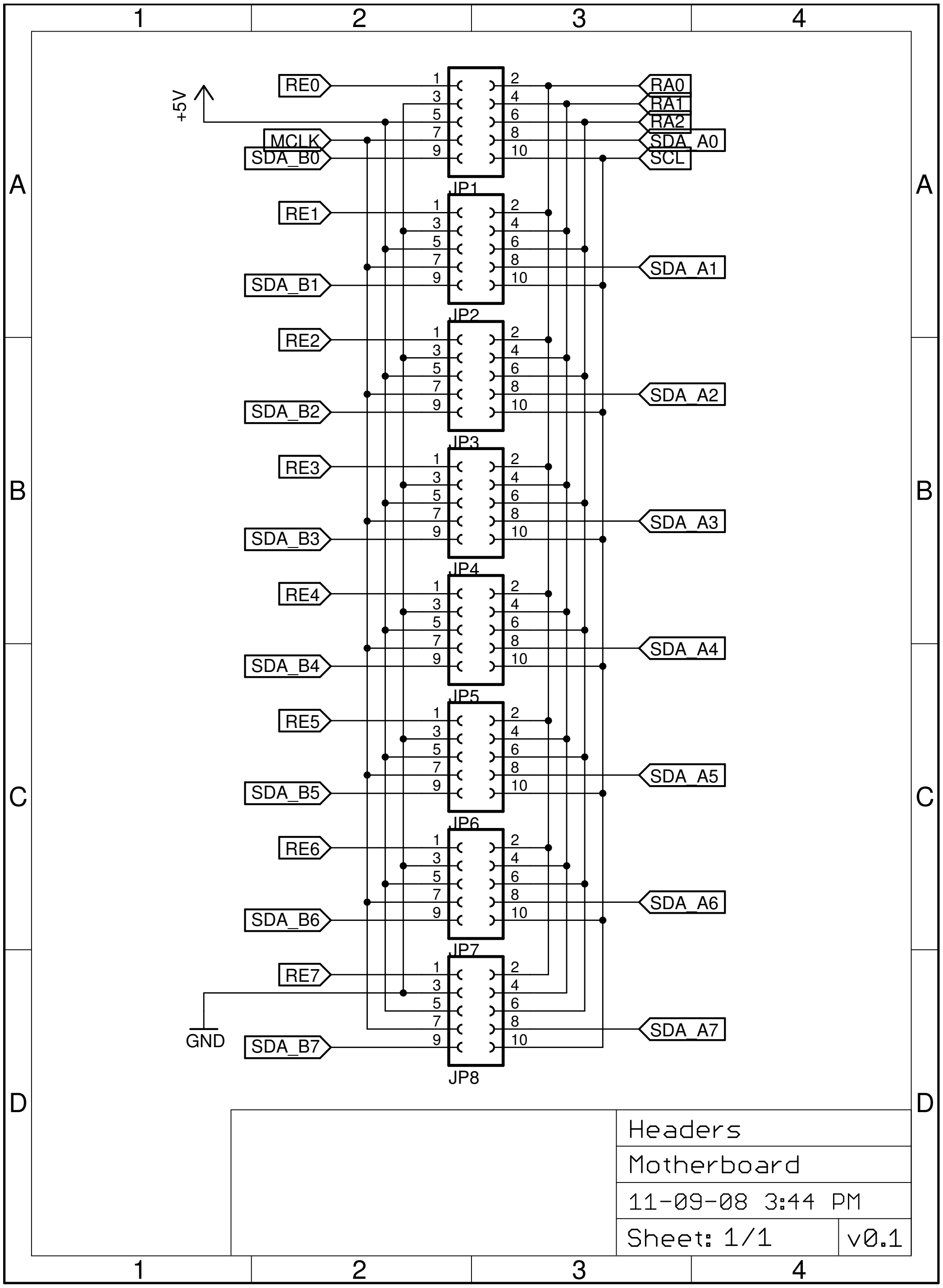


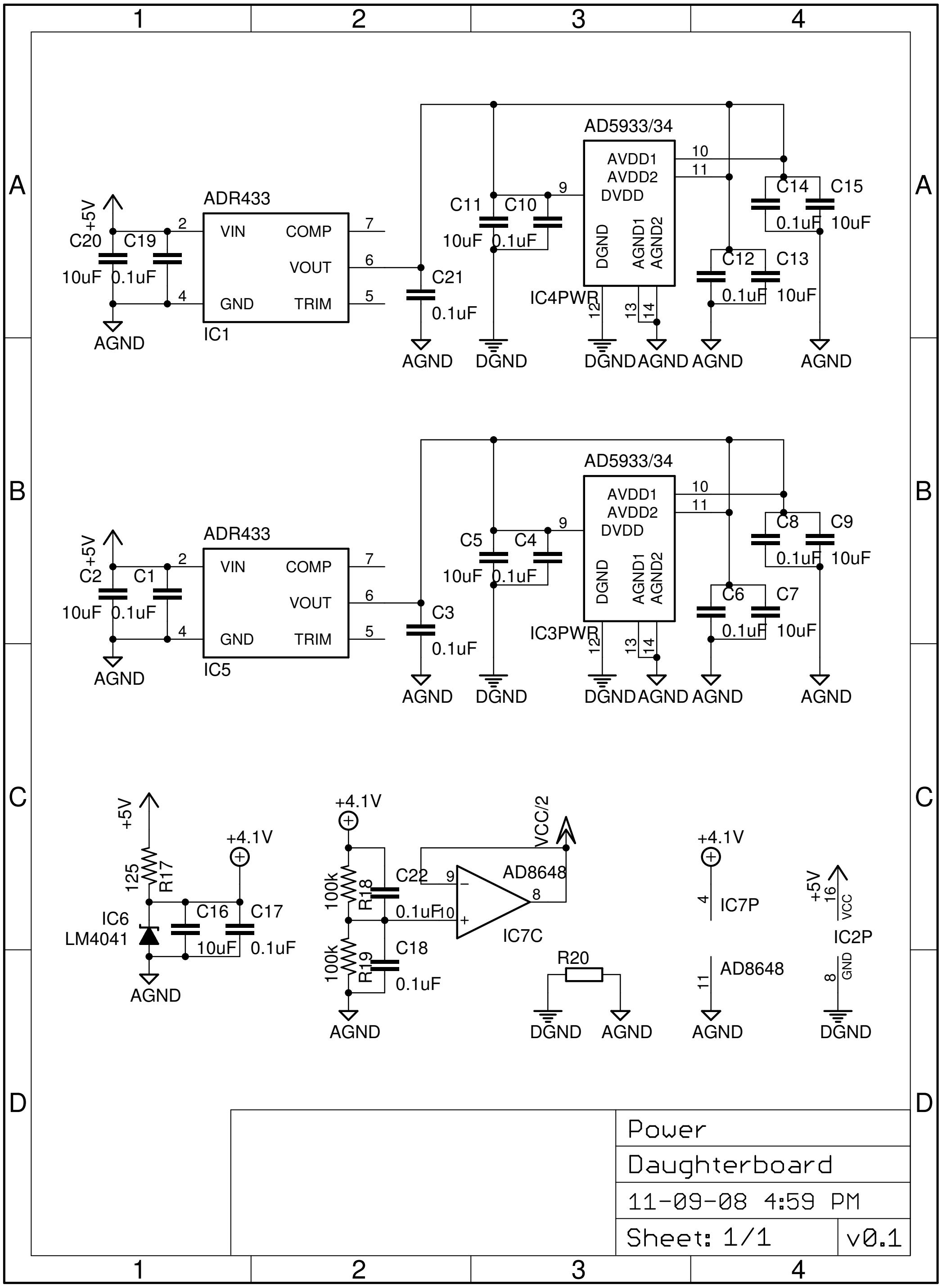
Multiplexors

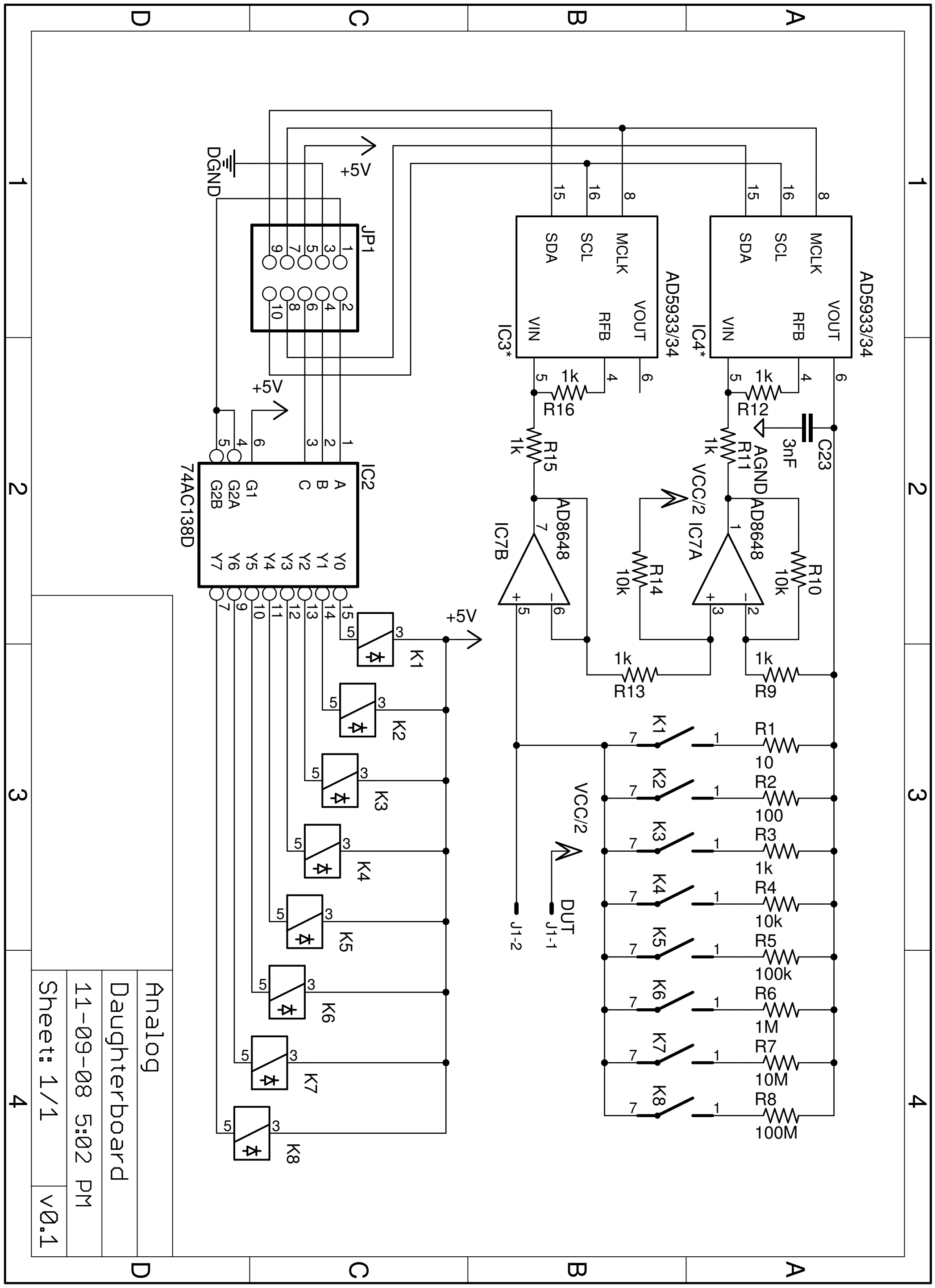
Motherboard

11-09-08 3:44 PM

Sheet: 1/1 v0.1


+5V
RE0
MCLK
SDA_B0
RA0
RA1
RA2
SDA_A0
SCL
JP1
RE1
SDA_B1
SDA_A1
JP2
RE2
SDA_B2
SDA_A2
JP3
RE3
SDA_B3
SDA_A3
JP4
RE4
SDA_B4
SDA_A4
JP5
RE5
SDA_B5
SDA_A5
JP6
RE6
SDA_B6
SDA_A6
JP7
RE7
GND
SDA_B7
SDA_A7
JP8
1
3
5
7
9
2
4
6
8
10
A
B
C
D
1
2
3
4
Headers
Motherboard
11-09-08 3:44 PM
Sheet: 1/1
v0.1


1
2
3
4
A
B
C
D
ADR433
VIN
COMP
VOUT
GND
TRIM
IC1
+5V
C20
C19
10uF
0.1uF
2
4
7
6
5
C21
0.1uF
AGND
AD5933/34
AVDD1
AVDD2
DVDD
DGND
AGND1
AGND2
10
11
9
12
13
14
IC4PWR
C11
C10
10uF
0.1uF
C14
C15
0.1uF
10uF
C12
C13
0.1uF
10uF
DGND
ADR433
IC5
C2
C1
10uF
0.1uF
C3
0.1uF
AD5933/34
IC3PWR
C5
C4
10uF
0.1uF
C8
C9
0.1uF
10uF
C6
C7
0.1uF
10uF
+5V
125
R17
IC6
LM4041
C16
C17
10uF
0.1uF
+4.1V
100k
R18
100k
R19
C22
0.1uF
C18
0.1uF
VCC/2
AD8648
IC7C
8
9
10
R20
IC7P
4
11
AD8648
+5V
16
VCC
IC2P
8
GND
Power
Daughterboard
11-09-08 4:59 PM
Sheet: 1/1
v0.1


Analog
Daughterboard
11-09-08 5:02 PM
Sheet: 1/1
v0.1
A
B
C
D
1
2
3
4
AD5933/34
IC4*
MCLK
SCL
SDA
VOUT
RFB
VIN
AD5933/34
IC3*
MCLK
SCL
SDA
VOUT
RFB
VIN
15
16
8
6
4
5
R12
1k
R16
1k
C23
3nF
AGND
R11
1k
R15
1k
AD8648
IC7A
AD8648
IC7B
VCC/2
R10
10k
R14
10k
R9
1k
R13
1k
R1
10
R2
100
R3
1k
R4
10k
R5
100k
R6
1M
R7
10M
R8
100M
K1
K2
K3
K4
K5
K6
K7
K8
VCC/2
DUT
J1-1
J1-2
JP1
+5V
DGND
IC2
74AC138D
A
B
C
G1
G2A
G2B
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7

Appendix B

Impedance Analyzer Firmware Assembly Code

```
;;; **************************************************************
;;; File Name:  ad5933.h
;;; Purpose:    Register definitions for the AD5933 chip
;;; Revision:   0.0
;;; Date:       17 January 2011
;;; Author:     D. Smetaniuk
;;; **************************************************************
;;; Revision History:
;;;     0.0 - 17 January 2011
;;;             - Creation
;;; **************************************************************

ADDR5933        EQU     0x0D

;; Register map

CTRLH           EQU     0x80
CTRLL           EQU     0x81
SFREQ2          EQU     0x82
SFREQ1          EQU     0x83
SFREQ0          EQU     0x84
FINC2           EQU     0x85
FINC1           EQU     0x86
FINC0           EQU     0x87
NINCH           EQU     0x88
NINCL           EQU     0x89
NSETLH          EQU     0x8A
NSETLL          EQU     0x8B
STAT            EQU     0x8F
TEMPH           EQU     0x92
TEMPL           EQU     0x93
REALH           EQU     0x94
REALL           EQU     0x95
IMAGH           EQU     0x96
IMAGL           EQU     0x97
```

```
;; Command Codes

BWRITE          EQU     0xA0
BREAD           EQU     0xA1
ADRPTR          EQU     0xB0
```

```
;;; ***********************************************************
;;; File Name:   branches.h
;;; Purpose:     Macros for conditional branching based on
;;;              unsigned values.
;;; Revision:    0.0
;;; Date:        27 September 2003
;;; Author:      L. Wyard-Scott
;;; Copyright:   (c) 2003
;;; Device:      PIC16F873,  PIC16F874,  PIC16F876,  PIC16F877.
;;;              PIC16F873A, PIC16F874A, PIC16F876A, PIC16F877A.
;;; ***********************************************************
;;; Revision History:
;;;     0.0 - 27 September 2003
;;;             - Creation
;;; ***********************************************************
;;; Ideas for future work:
;;; - make signed equivalents of these routines (a lot more work).
;;; ***********************************************************



;;; -----------------------------------------------------------
;;; Macros.
;;; -----------------------------------------------------------
;;; -----------------------------------------------------------
;;; The following 2 macros can be used to to determine if W is
;;; equal or not equal to a constant.  These values may be
;;; considered signed or unsigned.
;;; -----------------------------------------------------------
;;; ***********************************************************
;;; Macro Name:          brweqk
;;; Description:         Branch if W is equal to a specified byte
;;;                      constant.  Page selection bits must be
;;;                      established appropriately in order to
;;;                      successfully branch to the destination.
;;; Requires:            W contains one of the values being
;;;                      compared to.
;;;                      Two additional parameters are:
;;;                              value - the other value for the
;;;                              comparison.
;;;                              destination - the destination
;;;                              address.
;;; Returns:             Nothing.  Program execution goes to
;;;                      "destination" if W == the unsigned value.
;;; Locations Affected: W is destroyed.
;;; ***********************************************************
brweqk: MACRO   value, destination
        sublw   value
        btfsc   STATUS,Z
        goto    destination
        ENDM

;;; ***********************************************************
;;; Macro Name:          brwneqk
;;; Description:          Branch if W is not equal to a specified
;;;                       byte constant.  Page selection bits must
;;;                       be established appropriately in order to
;;;                       successfully branchto the destination.
;;; Requires:             W contains one of the values being
;;;                       compared to.
;;;                       Two additional parameters are:
;;;                               value - the other value for the
;;;                               comparison.
;;;                               destination - the destination
;;;                               address.
;;; Returns:              Nothing.  Program execution goes to
;;;                       "destination" if W != the unsigned value.
;;; Locations Affected: W is destroyed.
;;; **************************************************************
brwneqk: MACRO  value, destination
        sublw   value
        btfss   STATUS,Z
        goto    destination
        ENDM

;;; --------------------------------------------------------------
;;; The following 4 macros are used for comparing unsigned literal
;;; values with the contents of the W register.
;;; --------------------------------------------------------------
;;; **************************************************************
;;; Macro Name:           brwlok
;;; Description:          Branch if W is lower than a specified byte
;;;                       constant.  Page selection bits must be
;;;                       established appropriately in order to
;;;                       successfully branch to the destination.
;;; Requires:             W contains one of the values being
;;;                       compared to.
;;;                       Two additional parameters are:
;;;                               value - the other value for the
;;;                               comparison. This value must be
;;;                               non-zero.
;;;                               destination - the destination
;;;                               address.
;;; Returns:              Nothing.  Program execution goes to
;;;                       "destination" if W < the unsigned value.
;;; Locations Affected: W is destroyed.
;;; **************************************************************
brwlok: MACRO   value, destination
        addlw   (0x100-value)
        btfss   STATUS,C
        goto    destination
        ENDM

;;; **************************************************************
;;; Macro Name:           brwlsk
;;; Description:          Branch if W is lower than or the same as
;;;                       the specified byte constant.  Page
;;;                       selection bits must be established
;;;                     appropriately in order to successfully
;;;                     branch to the destination.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             value - the other value for the
;;;                             comparison.
;;;                             destination - the destination
;;;                             address.
;;; Returns:            Nothing.  Program execution goes to
;;;                     ”destination” if W <= the unsigned value.
;;; Locations Affected: W is destroyed.
;;; ***********************************************************
brwlsk: MACRO   value, destination
        addlw   (0xFF-value)
        btfss   STATUS,C
        goto    destination
        ENDM

;;; ***********************************************************
;;; Macro Name:         brwhik
;;; Description:        Branch if W is higher than a specified
;;;                     byte constant.  Page selection bits must
;;;                     be established appropriately in order to
;;;                     successfully branch to the destination.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             value - the other value for the
;;;                             comparison.
;;;                             destination - the destination
;;;                             address.
;;; Returns:            Nothing.  Program execution goes to
;;;                     ”destination” if W > value.
;;; Locations Affected: W is destroyed.
;;; ***********************************************************
brwhik: MACRO   value, destination
        addlw   (0xFF-value)
        btfsc   STATUS,C
        goto    destination
        ENDM

;;; ***********************************************************
;;; Macro Name:         brwhsk
;;; Description:        Branch if W is higher than or the same as
;;;                     the specified byte constant.  Page
;;;                     selection bits must be established
;;;                     appropriately in order to successfully
;;;                     branch to the destination.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             value - the other value for the
;;;                             comparison. This value must be
;;;                             non-zero. Use other techniques in
;;;                             the case where you'd like to see
;;;                             if W is greater than 0.
;;;                             destination - the destination
;;;                             address.
;;; Returns:                Nothing.  Program execution goes to
;;;                         "destination" if W >= the unsigned value.
;;; Locations Affected: W is destroyed.
;;; ***********************************************************
brwhsk: MACRO   value, destination
        addlw   (0x100-value)
        btfsc   STATUS,C
        goto    destination
        ENDM

;;; -----------------------------------------------------------
;;; The following 2 macros can be used to to determine if W is
;;; equal or not equal to the contents of a file register. These
;;; values may be considered signed or unsigned.
;;; -----------------------------------------------------------
;;; ***********************************************************
;;; Macro Name:             brweqf
;;; Description:            Branch if W is equal to the contents of
;;;                         the specified file register.  Page
;;;                         selection bits must be established
;;;                         appropriately in order to successfully
;;;                         branch to the destination. Bank selection
;;;                         bits must be established in order to
;;;                         access the file register.
;;; Requires:               W contains one of the values being
;;;                         compared to.
;;;                         Two additional parameters are:
;;;                             fileregister - the location of
;;;                             the other value.
;;;                             destination - the destination
;;;                             address.
;;; Returns:                Nothing.  Program execution goes to
;;;                         "destination" if W == the file register
;;;                         contents.
;;; Locations Affected: W is destroyed.
;;; ***********************************************************
brweqf: MACRO   fileregister, destination
        subwf   fileregister,W
        btfsc   STATUS,Z
        goto    destination
        ENDM

;;; ***********************************************************
;;; Macro Name:             brwneqf
;;; Description:            Branch if W is not equal to the contents
;;;                         of the specified file register.  Page
;;;                         selection bits must be established
;;;                         appropriately in order to successfully
;;;                         branch to the destination. Bank selection
;;;                     bits must be established in order to
;;;                     access the file register.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             fileregister - the location of the
;;;                             other value.
;;;                             destination - the destination
;;;                             address.
;;; Returns:            Nothing.  Program execution goes to
;;;                     "destination" if W != the file register
;;;                     contents.
;;; Locations Affected: W is destroyed.
;;; ************************************************************
brwneqf: MACRO  fileregister, destination
        subwf   fileregister,W
        btfss   STATUS,Z
        goto    destination
        ENDM

;;; ------------------------------------------------------------
;;; The following 4 macros are used for comparing unsigned values
;;; located in specified file registers with the contents of the
;;; W register.
;;; ------------------------------------------------------------
;;; ************************************************************
;;; Macro Name:         brwlof
;;; Description:        Branch if W is lower than the contents of
;;;                     the specified file register.  Page
;;;                     selection bits must be established
;;;                     appropriately in order to successfully
;;;                     branch to the destination. Bank selection
;;;                     bits must be established in order to
;;;                     access the file register.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             fileregister - the location of the
;;;                             other value.
;;;                             destination - the destination
;;;                             address.
;;; Returns:            Nothing.  Program execution goes to
;;;                     "destination" if W < the file register
;;;                     contents.
;;; Locations Affected: W is destroyed.
;;; ************************************************************
brwlof: MACRO   fileregister, destination
        subwf   fileregister,W  ; F-W -> W
        ;; Branches iff Z=0, C=1
        btfsc   STATUS,Z
        goto    $ + 3
        btfsc   STATUS,C
        goto    destination
        ENDM
```

```
;;; *************************************************************
;;; Macro Name:         brwlsf
;;; Description:        Branch if W is lower than or the same as
;;;                     the  contents of the specified file
;;;                     register. Page selection bits must be
;;;                     established appropriately in order to
;;;                     successfully branch to the destination.
;;;                     Bank selection bits must be established
;;;                     in order to access the file register.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             fileregister - the location of the
;;;                             other value.
;;;                             destination - the destination
;;;                             address.
;;; Returns:            Nothing.  Program execution goes to
;;;                     "destination" if W <= the file register
;;;                     contents.
;;; Locations Affected: W is destroyed.
;;; *************************************************************
brwlsf: MACRO   fileregister, destination
        subwf   fileregister,W
        ;; Branches iff Z=1, C=1, or Z=0, C=1 (resulting in C=1).
        btfsc   STATUS,C
        goto    destination
        ENDM

;;; *************************************************************
;;; Macro Name:         brwhif
;;; Description:        Branch if W is higher than the contents of
;;;                     the specified file register.  Page
;;;                     selection bits must be established
;;;                     appropriately in order to successfully
;;;                     branch to the destination. Bank selection
;;;                     bits must be established in order to
;;;                     access the file register.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             fileregister - the location of the
;;;                             other value.
;;;                             destination - the destination
;;;                             address.
;;; Returns:            Nothing.  Program execution goes to
;;;                     "destination" if W > the file register
;;;                     contents.
;;; Locations Affected: W is destroyed.
;;; *************************************************************
brwhif: MACRO   fileregister, destination
        subwf   fileregister,W
        ;; Branches iff Z=0, C=0
        btfsc   STATUS,Z
        goto    $ + 3
        btfss   STATUS,C
        goto    destination
        ENDM

;;; ************************************************************
;;; Macro Name:         brwlsf
;;; Description:        Branch if W is lower than or the same as
;;;                     the contents of the specified file
;;;                     register. Page selection bits must be
;;;                     established appropriately in order to
;;;                     successfully branch to the destination.
;;;                     Bank selection bits must be established
;;;                     in order to access the file register.
;;; Requires:           W contains one of the values being
;;;                     compared to.
;;;                     Two additional parameters are:
;;;                             fileregister - the location of the
;;;                             other value.
;;;                             destination - the destination
;;;                             address.
;;; Returns:            Nothing.  Program execution goes to
;;;                     "destination" if W <= the file register
;;;                     contents.
;;; Locations Affected: W is destroyed.
;;; ************************************************************
brwhsf: MACRO   fileregister , destination
        subwf   fileregister ,W
        ;; Branches iff Z=1, C=1, or Z=0, C=0 (resulting in Z=C).
        btfss   STATUS,Z
        goto    $ + 4
        btfsc   STATUS,C
        goto    destination
        goto    $ + 3
        btfss   STATUS,C
        goto    destination
        ENDM
```

```
;;; ************************************************************
;;; Filename:    dbus_lib.h
;;; Purpose:     FIFO databus library routine headers and other
;;;              library equates.
;;; Revision:    0.5
;;; Date:        7 January 2011
;;; Author:      L. Wyard-Scott, D. Smetaniuk
;;; Copyright:   Public Domain
;;; ************************************************************
;;; Revision History:
;;;     0.5      7 January 2011 - DPS
;;;              - Adapted from UART to databus FIFO. Changed
;;;                "UART" names to "DBUS". Modified "DBUS_RxByte"
;;;                and "DBUS_TxByte" to work with databus FIFO
;;;     0.4      No update.
;;;     0.3      No update.
;;;     0.2      24 February 2005 - LWS
;;;              - Updated to specify baudrates for various
;;;                crystals. Now an "FXTAL" define is required on
;;;                the command line.
;;;     0.1      No update.
;;;     0.0      20 October 2002 - LWS
;;;              Creation.
;;; ************************************************************
;;; ************************************************************


;;; ------------------------------------------------------------
;;; Macros. (File-specific)
;;; ------------------------------------------------------------

;;; ------------------------------------------------------------
;;; Externally defined variables.
;;; ------------------------------------------------------------
        extern  PTRHIGH          ; Pointer passing for
                                 ; UART_TxStringKx
        extern  PTRLOW           ; string constant transmission.

;;; ------------------------------------------------------------
;;; Externally defined subroutines.
;;; ------------------------------------------------------------
        extern  DBUS_Init        ; Initialize the DBUS FIFO
        extern  DBUS_RxByte      ; Receive a byte through the DBUS.
        extern  DBUS_KbHit       ; Determine if byte received.
        extern  DBUS_TxByte      ; Transmit byte through the DBUS.
        extern  DBUS_TxStringK   ; Transmit a NULL-terminated
                                 ; string constant.
        extern  DBUS_TxStringKN  ; Transmit N bytes of a string
                                 ; (or data). Will transmit a NULL,
                                 ; if encountered.
        extern  DBUS_TxString    ; Transmit a NULL-terminated
                                 ; string in RAM.
        extern  DBUS_TxStringN   ; Transmit N characters of a
                                 ; string out through the DBUS.
```

```
                          ; Will transmit NULLS, if
                          ; encountered.
extern  DBUS_RxString     ; Receive a NULL-terminate a
                          ; string.
```

```
;;; **********************************************************
;;; Filename:   del_lib.h
;;; Purpose:    Delay library routine headers and other equates.
;;; Revision:   0.1
;;; Date:       16 March 2004
;;; Author:     L. Wyard-Scott
;;; Copyright:  Public Domain
;;; **********************************************************
;;; Revision History:
;;;     0.1     16 March 2004 - LWS
;;;             No changes to header file -- renumbered to be
;;;             consistent with del_lib.c
;;;     0.0     20 October 2002 - LWS
;;;             Creation.
;;; **********************************************************

;;; ----------------------------------------------------------
;;; Externally defined subroutines.
;;; ----------------------------------------------------------
        extern  DELAY_Wx1ms     ; Delay a multiple of 1ms.
        extern  DELAY_Wx10ms    ; Delay a multiple of 10ms.
```

```
;;; **********************************************************
;;; Filename:   i2c_lib.h
;;; Purpose:    Header file for I2C library routines.
;;; Revision:   0.0
;;; Date:       11 January 2011.
;;; Author:     D. Smetaniuk
;;; Copyright:  Public domain.
;;; **********************************************************
;;; Revision History:
;;;     0.0 - 11 January 2011
;;;             Creation.
;;; **********************************************************

;;; ----------------------------------------------------------
;;; Externally defined variables.
;;; ----------------------------------------------------------
        extern  I2C_ADDR
        extern  I2C_REG
        extern  I2C_DAT

        extern  I2C_PTRHIGH     ; Pointer passing for
                                ; UART_TxStringKx
        extern  I2C_PTRLOW      ; string constant transmission.

;;; ----------------------------------------------------------
;;; Externally defined subroutines.
;;; ----------------------------------------------------------
        extern  I2C_Init
        extern  I2C_Write
        extern  I2C_BlockWrite
        extern  I2C_Read
        extern  I2C_BlockRead
```

```
;;; ***********************************************************
;;; Filename:    math_lib.h
;;; Purpose:     Math library routine headers and other library
;;;              variable definitions.
;;; Revision:    0.0
;;; Date:        19 January 2011
;;; Author:      D. Smetaniuk
;;; Copyright:   Public Domain
;;; ***********************************************************
;;; Revision History:
;;;              0.0      19 January 2011 - DPS
;;;              Creation.
;;; ***********************************************************
;;; ***********************************************************


;;; -----------------------------------------------------------
;;; Macros.  (File-specific)
;;; -----------------------------------------------------------

;;; -----------------------------------------------------------
;;; Externally defined variables.
;;; -----------------------------------------------------------
        extern  ACCa,ACCa1,ACCa2,ACCa3,ACCa4
        extern  ACCb,ACCb1,ACCb2,ACCb3,ACCb4
        extern  ACCc,ACCc1,ACCc2,ACCc3,ACCc4


;;; -----------------------------------------------------------
;;; Externally defined subroutines.
;;; -----------------------------------------------------------
        extern  ABS16
        extern  ADD32U
        extern  MUL16U
        extern  DIV32U
```

```
;;; ****************************************************************
;;; Library Name:       dbus_lib.asm
;;; Purpose:            Routines for using PORTD for parallel
;;;                     communication on the DLP-2232PB-G module
;;; Revision:           0.0
;;; Date:               12 February 2011
;;; Author:             L. Wyard-Scott, D. Smetnaiuk
;;; Copyright:          Public Domain
;;; Device:             PIC16F873, PIC16F874, PIC16F873A,
;;;                     PIC16F874A, PIC16F876, PIC16F877,
;;;                     PIC16F876A, PIC16F877A.
;;; ****************************************************************
;;; Special Directions:
;;;   - To use these routines, include "dbus_lib.h" in the source
;;;     code calling the routines.
;;;
;;; ****************************************************************
;;; Revision History:
;;;     0.0 - 12 February 2011 - DPS
;;;             - Creation. Adapted from code by L. Wyard-Scott
;;; ****************************************************************
;;;
;;; ****************************************************************
;;; NOTE:  An error will be generated by the assembler if any
;;;        labels defined as global in this file are the same as
;;;        those in the main file (or other libraries).

        list            ; Turn on list output.  Note that device
                        ; type needs to be specified to the
                        ; assembler (on the command line, or
                        ; through the MPLAB IDE).

        ;; Include register, bit, and other info specific to
        ;; the specified device.
        ifdef   __16F873
        #include <p16f873.inc>
        messg "Assembling␣dbus_lib.asm␣for␣PIC16F873."
        endif
        ifdef   __16F874
        #include <p16f874.inc>
        messg "Assembling␣dbus_lib.asm␣for␣PIC16F874."
        endif
        ifdef   __16F876
        #include <p16f876.inc>
        messg "Assembling␣dbus_lib.asm␣for␣PIC16F876."
        endif
        ifdef   __16F877
        #include <p16f877.inc>
        messg "Assembling␣dbus_lib.asm␣for␣PIC16F877."
        endif
        ifdef   __16F873A
        #define A_Device
        #include <p16f873a.inc>
        messg "Assembling␣dbus_lib.asm␣for␣PIC16F873A."
        endif
        ifdef   __16F874A
        #define A_Device
        #include <p16f874a.inc>
        messg ”Assembling␣dbus_lib.asm␣for␣PIC16F874A.”
        endif
        ifdef   __16F876A
        #define A_Device
        #include <p16f876a.inc>
        messg ”Assembling␣dbus_lib.asm␣for␣PIC16F876A.”
        endif
        ifdef   __16F877A
        #define A_Device
        #include <p16f877a.inc>
        messg ”Assembling␣dbus_lib.asm␣for␣PIC16F877A.”
        endif

;;; ---------------------------------------------------------------
;;; Assembler Equates Section. Define assembly-time constants here
;;;                            using the EQU assembler directive.
;;; ---------------------------------------------------------------

;; PORT B
PIN_RD                  EQU 1
PIN_WR                  EQU 2
PIN_RXF                 EQU 4
;; PORT E
PIN_TXE                 EQU 2

#define DBUS            PORTD

#define bitRD           PORTB,PIN_RD
#define bitWR           PORTB,PIN_WR
#define bitRXF          PORTB,PIN_RXF
#define bitTXE          PORTE,PIN_TXE

;;; ---------------------------------------------------------------
;;; Variable Address Assignments.
;;; ---------------------------------------------------------------
        UDATA                   ; Start of the uninitialized data
                                ; section. The following
                                ; statements reserve memory, the
                                ; address of which is allocated by
                                ; the linker.

        ;; For DBUS_TxStringK.

PTRHIGH:        res 1           ; High byte of the saved pointer.
PTRLOW:         res 1           ; Low byte of the saved pointer.

        global  PTRHIGH, PTRLOW


TemporaryVars:  UDATA_OVR
```

```
        ;; For DBUS_TxStringN
NUMBYTESTOSEND  res 1           ; Number of bytes left to send.

        ;; For DBUS_RxString.
PTRSAVEHIGH:    res 1           ; A place to save the pointer.
PTRSAVELOW:     res 1
NUMCHARS:       res 1           ; The # of characters received.
MAXCHARS:       res 1           ; The maximum number of characters
                                ; that will be accepted.
RXDCHAR:        res 1           ; Individual received character.



;;; ------------------------------------------------------------------
;;; Subroutines.
;;; ------------------------------------------------------------------
        CODE                    ; Start the code section.

;;; ***********************************************************
;;; Subroutine Name:    DBUS_Init
;;; Description:        Initializes the FIFO by emptying out the
;;;                     Receive buffer.
;;; Requires:           Nothing.
;;; Returns:            NULL
;;; Locations Affected: Destroys W
;;; ***********************************************************
DBUS_Init:
        global  DBUS_Init       ; Make this subroutine callable
                                ; outside the module
ChkFIFO:
        call    DBUS_KbHit
        skpc
         goto   Done
        call    DBUS_RxByte
        bcf     STATUS,C
        goto    ChkFIFO
Done:
        return

;;; ***********************************************************
;;; Subroutine Name:    DBUS_RxByte
;;; Description:        Receives a byte through the FIFO databus
;;;                     and returns it in W.  This routine blocks
;;;                     until a character is received.
;;; Requires:           Nothing.
;;; Returns:            The character in W.
;;; Locations Affected: None.
;;; ***********************************************************
DBUS_RxByte:
        global  DBUS_RxByte     ; Make this subroutine callable
                                ; outside  the module.
        banksel PORTD           ; Set to bank0

        ;; Loop until a character is received.
```

```
DBUS_RxByteLoop:
        btfsc   bitRXF          ; Poll the receive flag.
         goto   DBUS_RxByteLoop
        bcf     bitRD
        nop
        movf    DBUS, W         ; Copy character from databus to W
        bsf     bitRD

        return


;;; ************************************************************
;;; Subroutine Name:    DBUS_KbHit
;;; Description:        Determines if a character has been
;;;                     received by the FIFO (but does not return
;;;                     the character itself).
;;; Requires:           Nothing.
;;; Returns:            STATUS, C = 1 (true) if a character has
;;;                                 been received.
;;;                     STATUS, C = 0 (false) if no character is
;;;                                 waiting.
;;; Locations Affected: None.
;;; ************************************************************
DBUS_KbHit:
        global  DBUS_KbHit      ; Make this subroutine callable
                                ; outside the module.
        banksel PORTB           ; Set to bank0

        btfsc   bitRXF          ; Check the receive flag.
         goto   DBUS_KbHitFalse

DBUS_KbHitTrue:
        ;; A character has been received.  Return true in the
        ;; STATUS register carry bit.
        bsf     STATUS,C
        return

DBUS_KbHitFalse:
        ;; No character has been received.  Return false.
        bcf     STATUS,C
        return


;;; ************************************************************
;;; Subroutine Name:    DBUS_TxByte
;;; Description:        Sends the byte in W out through the FIFO
;;;                     databus. This routine will block until the
;;;                     Transmit Buffer is available.
;;; Requires:           The byte to send in W.
;;; Returns:            Nothing.
;;; Locations Affected: None.
;;; ************************************************************
DBUS_TxByte:
        global  DBUS_TxByte     ; Make this subroutine callable
```

```
                                ; outside the module.
        banksel PORTD           ; Set to bank0.

DBUS_TxByteWait
        btfsc   bitTXE          ; check that the Transmit buffer
                                ; is available
         goto   DBUS_TxByteWait
        bsf     STATUS, RP0     ; Select Bank 1
        clrf    TRISD           ; DBUS output
        bcf     STATUS, RP0     ; Select Bank 0

        movwf   DBUS            ; Send data
        bsf     bitWR
        nop
        bcf     bitWR

        bsf     STATUS, RP0     ; Select Bank 1
        movlw   0xFF
        movwf   TRISD           ; DBUS input
        bcf     STATUS, RP0     ; Select Bank 0

        return

;;; **********************************************************
;;; Subroutine Name:    DBUS_TxStringK
;;; Description:        Uses the DBUS to send a NULL-terminated
;;;                     string constant (a string that resides
;;;                     in the code space).
;;;                     This routine calls another (specialized)
;;;                     subroutine, DBUS_GetStringKChar
;;;                     in order to overcome the
;;;                     PIC CPU limitation not allowing pointers
;;;                     to constants.
;;;                     To send a string in RAM, see DBUS_TxString
;;; Requires:           PTRHigh - high byte of address of string
;;;                               constant.
;;;                     PTRLow -  low byte of address of string
;;;                               constant.
;;; Returns:            Nothing.
;;; Locations Affected: PTRHigh, PTRLow will point at the NULL.
;;;                     W will contain the NULL.
;;; **********************************************************
DBUS_TxStringK:
        global  DBUS_TxStringK  ; Make this subroutine callable
                                ; outside the module.

        ;; This routine uses a ”trick” to allow specification of
        ;; and arbitrary string constant by PTRHigh:PTRLow.
        ;; The trick is to make a subroutine call to an
        ;; intermediate piece of code which then uses a ”computed
        ;; goto” to jump to a string character.
        ;; When making a subroutine call, the return address
        ;; gets pushed onto the stack. The intermediate
        ;; subroutine itself does not pull the value off the
        ;; stack, but instead passes control to the table.
        ;; This is slightly different than the suggested method
        ;; by Microchip in AN556 (Implementing a Table Read)
        ;; wherein the table itself includes an instruction to
        ;; complete a computed goto statement.


DBUS_TXStringKLoop:
        banksel PTRLOW
        ;; Get a character.
        call    DBUS_GetStringKChar

        ;; Return to this page if the string constant was
        ;; located in another page.
        pagesel DBUS_TXStringKLoop

        ;; Check to see if it is a NULL.
        andlw   0xFF            ; Does nothing but set/clear Z in
                                ; STATUS.
        btfsc   STATUS,Z
        return                  ; String transmission complete.

        ;; Send the character.
        call    DBUS_TxByte

        banksel PTRLOW
        ;; Increment the pointer.
        incf    PTRLOW,F        ; Low byte.
        btfsc   STATUS,Z
        incf    PTRHIGH,F       ; High byte, if there is a carry
                                ; from the low byte.
        ;; Process another character.
        goto    DBUS_TXStringKLoop


DBUS_GetStringKChar:
        ;; Address of string is PTRHigh:PTRLow.
        ;; Use a computed GOTO to get the character to send.
        movf    PTRHIGH,W
        movwf   PCLATH          ; High byte of address.
        movf    PTRLOW,W
        movwf   PCL             ; Low Byte of address.
        ;; Execution should never past here.  In a way,
        ;; this is a subroutine without a return statement.
        ;; The actual return is contained along with the
        ;; string 'table' data.


;;; *********************************************************
;;; Subroutine Name:    DBUS_TxStringKN
;;; Description:        Uses the DBUS to send out N bytes of a
;;;                     NULL-terminated string constant (a string
;;;                     that resides in the code space).
;;;                     This routine calls another (specialized)
;;;                     subroutine, DBUS_GetStringKChar
;;;                     in order to overcome the
;;;                     PIC CPU limitation not allowing pointers
;;;                     to constants.
;;;                     To send a string in RAM, see DBUS_TxString
;;; Requires:           PTRHigh - high byte of address of string
;;;                               constant.
;;;                     PTRLow -  low byte of address of string
;;;                               constant.
;;; Returns:            Nothing.
;;; Locations Affected: PTRHigh, PTRLow will point at the NULL.
;;;                     W will contain the NULL.
;;; ***********************************************************
DBUS_TxStringKN:
        global  DBUS_TxStringKN ; Make this subroutine callable
                                ; outside the module.

        ;; This routine uses a "trick" to allow specification of
        ;; and arbitrary string constant by PTRHigh:PTRLow.
        ;; The trick is to make a subroutine call to an
        ;; intermediate piece of code which then uses a "computed
        ;; goto" to jump to a string character.
        ;; When making a subroutine call, the return address
        ;; gets pushed onto the stack. The intermediate
        ;; subroutine itself does not pull the value off the
        ;; stack, but instead passes control to the table.
        ;; This is slightly different than the suggested method
        ;; by Microchip in AN556 (Implementing a Table Read)
        ;; wherein the table itself includes an instruction to
        ;; complete a computed goto statement.

        banksel NUMBYTESTOSEND
        movwf   NUMBYTESTOSEND

DBUS_TXStringKNLoop:
        banksel PTRLOW
        ;; Get a character.
        call    DBUS_GetStringKChar

        ;; Return to this page if the string constant was
        ;; located in another page.
        pagesel DBUS_TXStringKNLoop

        ;; Send the character.
        call    DBUS_TxByte

        ;; Check if there are more bytes to send.
        banksel NUMBYTESTOSEND
        decfsz  NUMBYTESTOSEND,F
        goto    DBUS_TXStringKNIncPointer
        return

DBUS_TXStringKNIncPointer
        banksel PTRLOW
        ;; Increment the pointer.
        incf    PTRLOW,F        ; Low byte.
        btfsc   STATUS,Z
        incf    PTRHIGH,F       ; High byte, if there is a carry
                                ; from the low byte.
        ;; Process another character.
        goto    DBUS_TXStringKNLoop

;;; ************************************************************
;;; Subroutine Name:    DBUS_TxString
;;; Description:        Sends a NULL-terminated string in RAM
;;;                     through the DBUS.
;;;                     To send a string constant in ROM, see
;;;                     subroutine DBUS_TxStringK.
;;; Requires:           IRP (STATUS:7) and FSR point to the string
;;;                     in RAM. (See PIC16F87X data sheet section
;;;                     2.5 for more information on indirect
;;;                     addressing.)
;;; Returns:            Nothing.
;;; Locations Affected: W will contain 0 (NULL). IRP, FSR will
;;;                     point to the NULL of the string.
;;; ************************************************************
DBUS_TxString:
        global  DBUS_TxString   ; Make this subroutine callable
                                ; outside the module.

        ;; This routine requires no RAM bank switching since
        ;; INDF, StATUS, and FSR all appear in every bank.

        movf    INDF,W          ; Get the character pointed to by
                                ; indirect addressing.
        btfsc   STATUS,Z        ; Exit if the NULL is found.
        return

        call    DBUS_TxByte     ; Send the character out.

        incf    FSR,F           ; Increment the LSByte of pointer.
        btfsc   STATUS,C        ; Check if there was a carry.
        bsf     STATUS,IRP      ; If so, ensure that a wrap to an
                                ; upper bank occurs.
        goto    DBUS_TxString


;;; ************************************************************
;;; Subroutine Name:    DBUS_TxStringN
;;; Description:        Sends N characters of a string out through
;;;                     the DBUS. This routine will transfer
;;;                     characters past any terminating NULL, if
;;;                     the number of specified characters is
;;;                     beyond the NULL. To send a NULL-terminated
;;;                     string in RAM, see subroutine
;;;                     DBUS_TxString. To send a string constant
;;;                     in ROM, see subroutine DBUS_TxStringK.
;;; Requires:           W contains the number of characters to
;;;                     transmit. IRP (STATUS:7) and FSR point to
;;;                     the string in RAM. (See PIC16F87X data
;;;                     sheet section 2.5 for more information on
;;;                     indirect addressing.)
;;; Returns:            Nothing.
;;; Locations Affected: W will contain 0.  IRP, FSR will point
;;;                     to the last transmitted byte.
;;; *************************************************************
DBUS_TxStringN:
        global  DBUS_TxStringN   ; Make this subroutine callable
                                 ; outside the module.

        ;; Save the character count.
        banksel NUMBYTESTOSEND
        movwf   NUMBYTESTOSEND

DBUS_TxStringNLoop:
        movf    INDF,W           ; Get the character pointed to by
                                 ; indirect addressing.

        call    DBUS_TxByte      ; Send the character out.

        ;; Determine if all desired characters are sent out.
        banksel NUMBYTESTOSEND
        decf    NUMBYTESTOSEND,F
        btfsc   STATUS,Z         ; If zero then the routine is done
        return


        incf    FSR,F            ; Increment LSByte of the pointer.
        btfsc   STATUS,C         ; Check  if there has been a carry
        bsf     STATUS,IRP       ; If so, ensure that a wrap to an
                                 ; upper bank occurs.
        goto    DBUS_TxStringNLoop


;;; *************************************************************
;;; Subroutine Name:    DBUS_RxString
;;; Description:        Receives a string from the user,
;;;                     terminated by a new-line character.
;;; Requires:           IRP (STATUS:7) and FSR point to the start
;;;                     of an initialized string structure in RAM.
;;;
;;;                     On call:
;;;     Position:       byte0  byte1  byte2  byte3  ....  byte n
;;;     Value:          MaxChars
;;;
;;;                     MaxChars is the maximum number of
;;;                     characters that will be acccepted by this
;;;                     routine, excluding the NULL. Thus, if the
;;;                     buffer has size (n+1), as shown, then it
;;;                     can hold (n-1) characters plus the NULL.
;;;                     The maximum value that can be placed in
;;;                     the MaxChars position for a buffer of this
;;;                     size is (n-1). A lower number of
;;;                     characters, of course, is permitted.
;;;
;;;                     From the perspective of buffersize, the
;;;                     memory allocated to this structure must be
;;;                     at least the maximum string length plus 3.
;;;
;;;                     This routine can handle strings of up to
;;;                     125 characters.  Any more will require a
;;;                     rewrite of some of the signed pointer
;;;                     arithmetic.
;;;
;;; Returns:            The NULL-terminated string and the number
;;;                     of non-NULL characters in it as part of
;;;                     the structure.
;;;
;;;     Position:       byte0   byte1   byte2   byte3    ....   byte n
;;;     Value:          MaxChars NumUsed  String data........NULL
;;;
;;;                     Where NumUsed is the number of characters
;;;                     that were received, excluding the appended
;;;                     NULL. The string data is NULL-terminated.
;;;                     The new-line character is stripped.
;;;
;;;                    W contains the number of characters
;;;                     received (a duplicate of the value in
;;;                     ”byte1”.
;;;
;;; Locations Affected: None.  (IRP:FSR will still point to the
;;;                                 start of the structure.)
;;; **********************************************************
DBUS_RxString:
        global  DBUS_RxString    ; Make this subroutine callable
                                 ; outside the module.
        ;; Move the max number of chars to variable MAXCHARS.
        banksel NUMCHARS
        movf    INDF,W           ; *IRP:FSR->W
        movwf   MAXCHARS         ; W->MAXCHARS.

        clrf    NUMCHARS         ; 0->NUMCHARS

        ;; Save the IRP:FSR pointer.
        movf    FSR,W            ; Save the lower byte.
        movwf   PTRSAVELOW
        movf    STATUS,W         ; Isolate and save the IRP bit.
        andlw   0x80
        movwf   PTRSAVEHIGH

        ;; Make IRP:FSR point to the string data by incrementing
        ;; it by 2.
        incf    FSR,F
        incf    FSR,F

DBUS_RxStringGetChar:
        ;; Get a character and save it.
        call    DBUS_RxByte
        banksel RXDCHAR
        movwf   RXDCHAR

        ;; Is the character a backspace?
        sublw   '\b'            ; Compare with the backspace.
        btfsc   STATUS,Z
        goto    DBUS_RxStringIsBackspace

        ;; Restore the character.
        movf    RXDCHAR,W

        ;; Is the character a newline?
        sublw   '\r'
        btfsc   STATUS,Z
        goto    DBUS_RxStringIsNewline

        ;; Falling through to here means the character is neither
        ;; a backspace nor a return character.
        ;; Ensure that there is room for the character.
        ;; Get the maximum number of allowed characters into W.
        movf    MAXCHARS,W

        ;; Compare this with the number already received.
        subwf   NUMCHARS,W
        btfsc   STATUS,Z        ; If they are equal, then
                                ; there is no room for the
                                ; character.
        goto    DBUS_RxStringSendBell

        ;; There is room for the character.
        ;; Add one to the NUMCHARS variable to indicate that the
        ;; byte has been received.
        incf    NUMCHARS,F

        ;; Place the character in the location pointed to.
        movf    RXDCHAR,W
        movwf   INDF

        ;; Echo the character back to the user.
        call    DBUS_TxByte

        ;; Make the pointer point to where the next character is
        ;; to be inserted.
        incf    FSR,F

        ;; Go get another character.
        goto    DBUS_RxStringGetChar

DBUS_RxStringSendBell:
        movlw   '\a'            ; Alarm character.
        call    DBUS_TxByte
        goto    DBUS_RxStringGetChar
```

```
DBUS_RxStringIsBackspace:
        ;; To get here, the backspace key has been pressed.
        ;; Determine if there is room to backspace.
        movf    NUMCHARS,W
        btfsc   STATUS,Z        ; If not, let the user know.
        goto    DBUS_RxStringSendBell

        ;; There is room to backspace.  Decrement number of chars.
        decf    NUMCHARS, F

        ;; Send a backspace, a space, and a backspace to
        ;; remove the previous character and move back.
        ;; This destroys the pointer to the buffer.
        movlw   high DBUS_RxStringErase
        movwf   PTRHIGH
        movlw   low DBUS_RxStringErase
        movwf   PTRLOW
        call    DBUS_TxStringK

        ;; Move to the now empty location by decrementing it.
        movlw   -1
        addwf   FSR, F

        ;; Go and get another character.
        goto    DBUS_RxStringGetChar

DBUS_RxStringIsNewline:
        ;; The only way that this subroutine terminates is
        ;; if the user presses the <enter> key.
        ;; Append a NULL.
        clrf    INDF

        ;; Restore the pointer.
        movf    STATUS,W        ; Restore the IRP bit.
        iorwf   PTRSAVEHIGH,W
        movwf   STATUS
        movf    PTRSAVELOW,W    ; Restore the lower byte.
        movwf   FSR

        ;; Move the pointer over to the byte that contains
        ;; the number of characters entered.
        incf    FSR,F


        ;; Set the number of entered characters.
        movf    NUMCHARS,W
        movwf   INDF

        ;; Reset the pointer (again).
        movlw   -1
        addwf   FSR,F

        return
        ;; Define the string constant that backs up the cursor,
        ;; over-writes the last character, and then backs up to
        ;; the empty spot.
DBUS_RxStringErase:     dt      ”\b␣\b”,0

;;; ----------------------------------------------------------------------
;;; Constant Data (if not placed along with the subroutines).
;;; ----------------------------------------------------------------------

        END
```

```
;;; ************************************************************
;;; Library Name:       del_lib.asm
;;; Purpose:            Routines for (blocking) software delays.
;;; Revision:           0.2.1
;;; Date:               07 March 2011
;;; Author:             L. Wyard-Scott, M. Cumming, D. Smetaniuk
;;; Copyright:          Public Domain
;;; Device:             PIC16F873,  PIC16F874,  PIC16F876,
;;;                     PIC16F877, PIC16F873A, PIC16F874A,
;;;                     PIC16F876A, PIC16F877A.
;;; ************************************************************
;;; Special Directions:
;;;     To use these routines, include "del_lib.h" in the source
;;;     code calling the routines.
;;;     The crystal frequency needs to be specified on the command
;;;     line:
;;;             FXTAL=xx
;;;     Where valid choices of FXTAL are integer representations
;;;     of the crystal frequency.
;;;             3  --> 3.57945 MHz
;;;             4  --> 4.00 MHz
;;;             5  --> 5.0688 MHz
;;;             7  --> 7.15909 MHz
;;;         10  --> 10.00 MHz
;;;             12  --> 12.00 MHz
;;;         16  --> 16.384 MHz
;;;         20  --> 20.00 MHz

;;; ************************************************************
;;; Revision History:
;;;     0.2.1 - 07 March 2011 - DPS
;;;             - changed 16.00 MHz to 16.384 MHz
;;;     0.2 - 03 October 2005
;;;             - Added support for 16F87XA devices.
;;;     0.1 - 16 March 2004
;;;             - Added conditional assembly directives to remove
;;;               dependency on a particular frequency.  Possible
;;;               frequencies are specified above.
;;;     0.0 - 13 October 2002
;;;             - Created from routines in debug873.asm (r3.1).
;;; ************************************************************

        list                    ; Turn on list output.  Note that
                                ; device type needs to be
                                ; specified to the assembler (on
                                ; the command line, or through the
                                ; MPLAB IDE).

        ;; Include register, bit, and other info specific to
        ;; the specified device.
        ifdef   __16F873
        #include <p16f873.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F873."
        endif
        ifdef   __16F874
        #include <p16f874.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F874."
        endif
        ifdef   __16F876
        #include <p16f876.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F876."
        endif
        ifdef   __16F877
        #include <p16f877.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F877."
        endif
        ifdef   __16F873A
        #define A_Device
        #include <p16f873a.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F873A."
        endif
        ifdef   __16F874A
        #define A_Device
        #include <p16f874a.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F874A."
        endif
        ifdef   __16F876A
        #define A_Device
        #include <p16f876a.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F876A."
        endif
        ifdef   __16F877A
        #define A_Device
        #include <p16f877a.inc>
        messg "Assembling␣del_lib.asm␣for␣PIC16F877A."
        endif


;;; Ensure that the crystal frequency has been specified on the
;;; assembler command line.
        ifndef  FXTAL
        error "Timing␣element␣frequency␣not␣specified␣on␣command␣
           line."
        error "Possible␣values␣for␣FXTAL␣are:"
        error "3␣␣-->␣3.57945␣MHz"
        error "4␣␣-->␣4.00␣MHz"
        error "5␣␣-->␣5.0688␣MHz"
        error "7␣␣-->␣7.15909␣MHz"
        error "10␣␣-->␣10.00␣MHz"
        error "12␣␣-->␣12.00␣MHz"
        error "16␣␣-->␣16.384␣MHz"
        error "20␣␣-->␣20.00␣MHz"
        endif

;;; Establish the loop delay values based on the crystal
;;; frequency. Both routines (_Wx1ms and _Wx10ms) use the same
;;; code structure. The code is quite simple, but the counting of
;;; cycles to get an accurate delay is not simple. This is what
;;; I get:
;;;
;;; TOTALDELAY = W * (OUTER*(3*INNER+4)+4) + 6
;;;
;;; Since W is a parameter, the (OUTER*(3*INNER+4)+4) term should
;;; be the unit of the delay.  The 6 cycles is overhead and there
;;; is no simple way of accommodating it without complicating
;;; matters further.
;;;
;;; Terms OUTER and INNER can be any value above zero and below
;;; 256 because they are byte values.
;;; The following values were created using a little program to
;;; search through all possible combinations and indicate the
;;; closest combinations.
;;;
;;; FOR A 1 ms DELAY
;;; fXTAL (MHz)  tcyc (ns)       ncyc/ms  INNERCOUNT   OUTERCOUNT
;;; -----------------------------------------------------------------
;;; 3.57945      1117.49         894.86   58           5 (894~)
;;; 4.00         1000.00         1000     54           6
;;; 5.0688       789.14          1267.2   139          3
;;; 7.15909      558.73          1789.77  30           19
;;; 10.00        400.00          2500     68           12
;;; 12.00        333.33          3000     70           14
;;; 16.00        250.00          4000     11           108
;;;     16.384   244.14          4096     40           33
;;; 20.00        200.00          5000     11           135 (4999~)
;;;
;;; FOR A 10ms DELAY
;;; fXTAL (MHz)  tcyc (ns)     ncyc/10ms  INNERCOUNT   OUTERCOUNT
;;; -----------------------------------------------------------------
;;; 3.57945      1117.49         8948.6   13           208 (8948~)
;;; 4.00         1000.00         10000    15           204
;;; 5.0688       789.14          12672    33           123 (12673~)
;;; 7.15909      558.73          17897.7  46           126 (17896~)
;;; 10.00        400.00          25000    46           176 (24996~)
;;; 12.00        333.33          30000    93           106 (30002~)
;;; 16.00        250.00          40000    66           198
;;; 16.384       244.14          40960    61           219 (40957~)
;;; 20.00        200.00          50000    167          99 (49999~)
;;;

        if (FXTAL == 3)
        messg "3.579545␣MHz␣timing␣element␣specified."
INNERCOUNT1ms   EQU     58
OUTERCOUNT1ms   EQU     5
INNERCOUNT10ms  EQU     13
OUTERCOUNT10ms  EQU     208
        endif

        if (FXTAL == 4)
        messg "4␣MHz␣timing␣element␣specified."
INNERCOUNT1ms   EQU     54
OUTERCOUNT1ms   EQU     6
```

```
INNERCOUNT10ms   EQU      15
OUTERCOUNT10ms   EQU      204
        endif

        if (FXTAL == 5)
        messg "5.0688␣MHz␣timing␣element␣specified."
INNERCOUNT1ms    EQU      139
OUTERCOUNT1ms    EQU      3
INNERCOUNT10ms   EQU      33
OUTERCOUNT10ms   EQU      123
        endif

        if (FXTAL == 7)
        messg "7.15909␣MHz␣timing␣element␣specified."
INNERCOUNT1ms    EQU      30
OUTERCOUNT1ms    EQU      19
INNERCOUNT10ms   EQU      46
OUTERCOUNT10ms   EQU      126
        endif

        if (FXTAL == 10)
        messg "10␣MHz␣timing␣element␣specified."
INNERCOUNT1ms    EQU      68
OUTERCOUNT1ms    EQU      12
INNERCOUNT10ms   EQU      46
OUTERCOUNT10ms   EQU      176
        endif

        if (FXTAL == 12)
        messg "12␣MHz␣timing␣element␣specified."
INNERCOUNT1ms    EQU      70
OUTERCOUNT1ms    EQU      14
INNERCOUNT10ms   EQU      93
OUTERCOUNT10ms   EQU      106
        endif

        if (FXTAL == 16)
        messg "16␣MHz␣timing␣element␣specified."
INNERCOUNT1ms    EQU      40              ;11
OUTERCOUNT1ms    EQU      33              ;108
INNERCOUNT10ms   EQU      61              ;66
OUTERCOUNT10ms   EQU      219             ;198
        endif

        if (FXTAL == 20)
        messg "20␣MHz␣timing␣element␣specified."
INNERCOUNT1ms    EQU      11
OUTERCOUNT1ms    EQU      135
INNERCOUNT10ms   EQU      167
OUTERCOUNT10ms   EQU      99
        endif

        ;; Make sure that a *valid* crystal frequency was
        ;; specified.
        ifndef  INNERCOUNT1ms
        error "Invalid␣frequency␣specified:␣must␣be␣3,4,5,7,10,16,
          ␣or␣20."
        endif

;;; ------------------------------------------------------------------
;;; Variable Address Assignments.
;;; ------------------------------------------------------------------
        UDATA                       ; Start of the uninitialized data
                                    ; section. The following
                                    ; statements reserve memory, the
                                    ; address of which is allocated by
                                    ; the linker.

DCOUNT1:        res 1               ; Variables used to count the
DCOUNT2:        res 1               ; number of instruction cycles.
DCOUNT3:        res 1
DCOUNT4:        res 1
        ;; None of these variables need to be declared as global.
        ;; None of them should be "overlay" (UDATA_OVR), as
        ;; several libraries make use of these routines and a
        ;; collision could occur.
;;; ------------------------------------------------------------------
;;; Assembler Equates Section. Define assembly-time constants here
;;;                             using the EQU assembler directive.
;;; ------------------------------------------------------------------
;;;
;;; ------------------------------------------------------------------
;;; Macros.  (File-specific)
;;; ------------------------------------------------------------------
;;;
;;; ------------------------------------------------------------------
;;; Subroutines.
;;; ------------------------------------------------------------------
        CODE                        ; Start the code section.

;;; **************************************************************
;;; Subroutine Name:    DELAY_Wx1ms
;;; Description:        Delays (blocks) a multiple of 1ms
;;;                     increments. See above for development of
;;;                     the timing values.
;;; Requires:           The number of increments (ms) in W.
;;; Returns:            None.
;;; Locations Affected: Variables DCOUNT1,2,3,4
;;; **************************************************************
DELAY_Wx1ms:
        global  DELAY_Wx1ms         ; Make this subroutine callable
                                    ; outside the module.

        banksel DCOUNT1             ; [2~]  ([1~] if using MPLAB and
                                    ;        16F873)
        movwf   DCOUNT4             ; [1~]  Save a copy of w.

        movwf   DCOUNT1             ; [1~]  DCOUNT1 = W
```

```
Outercount1ms:
        movlw   OUTERCOUNT1ms       ; [1~]
        movwf   DCOUNT2             ; [1~]
Innercount1ms:
        movlw   INNERCOUNT1ms       ; [1~]
        movwf   DCOUNT3             ; [1~]
        decfsz  DCOUNT3,F           ; [1~,2~ on skip]
        goto    $ - 1               ; [2~]
        decfsz  DCOUNT2,F           ; [1~,2~ on skip]
        goto    $ - 5               ; [2~]
        decfsz  DCOUNT1,F           ; [1~,2~ on skip]
        goto    $ - 9               ; [2~]

        movf    DCOUNT4,W           ; [1~] Restore w.

        return                      ; [2~]


;;; *********************************************************
;;; Subroutine Name:     DELAY_Wx10ms
;;; Description:         Delays (blocks) a multiple of 10ms
;;;                      increments.
;;; Requires:            The number of increments (10 milliseconds)
;;;                      in W.
;;; Returns:             None.
;;; Locations Affected: Variables DCOUNT1,2,3,4
;;; *********************************************************
DELAY_Wx10ms:
        global  DELAY_Wx10ms        ; Make this subroutine callable
                                    ; outside the module.
        banksel DCOUNT1
        movwf   DCOUNT4             ; Save a copy of w.

        movwf   DCOUNT1
        movlw   OUTERCOUNT10ms
        movwf   DCOUNT2
        movlw   INNERCOUNT10ms
        movwf   DCOUNT3
        decfsz  DCOUNT3,F
        goto    $ - 1
        decfsz  DCOUNT2,F
        goto    $ - 5
        decfsz  DCOUNT1,F
        goto    $ - 9


        movf    DCOUNT4,W           ; Restore w.
        return


        END
```

```
;;; **************************************************************
;;; Library Name:       i2c_lib.asm
;;; Purpose:            Routines for reading and writing over the
;;;                     i2c bus. Modularized from code for AN732.
;;; Revision:           0.1
;;; Date:               16 February 2001
;;; Author:             See below.  Modularized by L. Wyard-Scott.
;;;                     Adapted for I2C by D. Smetaniuk
;;; Copyright:          See below.
;;; Device:             PIC16F873,  PIC16F874,  PIC16F876,
;;;                     PIC16F877,  PIC16F873A, PIC16F874A,
;;;                     PIC16F876A, PIC16F877A.
;;; **************************************************************

;;; (start copyright)
;;;=====================================================================
;;; Software License Agreement
;;;
;;; The software supplied herewith by Microchip Technology
;;; Incorporated (the ”Company”) for its PICmicro Microcontroller
;;; is intended and supplied to you, the Company's customer, for
;;; use solely and exclusively on Microchip PICmicro
;;; Microcontroller products. The software is owned by the Company
;;; and/or its supplier, and is protected under applicable
;;; copyright laws. All rights are reserved. Any use in violation
;;; of the foregoing restrictions may subject the user to criminal
;;; sanctions under applicable laws, as well as to civil liability
;;; for the breach of the terms and conditions of this license.
;;;
;;; THIS SOFTWARE IS PROVIDED IN AN ”AS IS” CONDITION. NO
;;; WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING,
;;; BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND
;;; FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE
;;; COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR
;;; SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON
;;; WHATSOEVER.
;;;
;;;=====================================================================
;;;     Filename:       boot877.asm
;;;=====================================================================
;;;     Author:         Mike Garbutt
;;;     Company:        Microchip Technology Inc.
;;;     Revision:       1.00
;;;     Date:           26 June 2000
;;;     Assembled using MPASM V2.40
;;;=====================================================================
;;; (end copyright)

;;; **************************************************************
;;; Special Directions:
;;;     To use these routines, include ”i2c_lib.h” in the source
;;;     code calling the routines.  This file contains information
;;;     about the routines and other equates that the calling
;;;     code requires.
;;; ************************************************************
;;; Revision History:
;;;     0.2 - 16 February 2011
;;;             - Adapted for I2C
;;;     0.1 - 03 October 2005
;;;             - Added support for 16F87XA devices.
;;;     0.0 - 13 November 2002
;;;             - Creation.  Made modular.
;;; ************************************************************
;;; Ideas for future work:
;;;
;;; ************************************************************

        list                    ; Turn on list output.  Note that
                                ; device type needs to be
                                ; specified to the assembler (on
                                ; the command line, or through the
                                ; MPLAB IDE).

        ;; Include register, bit, and other info specific to
        ;; the specified device.
        ifdef   __16F873
        #include <p16f873.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F873."
        endif
        ifdef   __16F874
        #include <p16f874.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F874."
        endif
        ifdef   __16F876
        #include <p16f876.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F876."
        endif
        ifdef   __16F877
        #include <p16f877.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F877."
        endif
        ifdef   __16F873A
        #define A_Device
        #include <p16f873a.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F873A."
        endif
        ifdef   __16F874A
        #define A_Device
        #include <p16f874a.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F874A."
        endif
        ifdef   __16F876A
        #define A_Device
        #include <p16f876a.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F876A."
        endif
        ifdef   __16F877A
        #define A_Device
        #include <p16f877a.inc>
        messg "Assembling␣flsh_lib.asm␣for␣PIC16F877A."
        endif


;;; Ensure that the crystal frequency has been specified on the
;;; assembler command line.
        ifndef  FXTAL
        error "Timing␣element␣frequency␣not␣specified␣on␣command␣
            line."
        error "Possible␣values␣for␣FXTAL␣are:"
        error "3␣␣—>␣3.57945␣MHz"
        error "4␣␣—>␣4.00␣MHz"
        error "5␣␣—>␣5.0688␣MHz"
        error "7␣␣—>␣7.15909␣MHz"
        error "10␣␣—>␣10.00␣MHz"
        error "12␣␣—>␣12.00␣MHz"
        error "16␣␣—>␣16.00␣MHz"
        error "20␣␣—>␣20.00␣MHz"
        endif

        if (FXTAL == 3)
        messg "3.579545␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'3579545'
        endif

        if (FXTAL == 4)
        messg "4␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'4000000'
        endif

        if (FXTAL == 5)
        messg "5.0688␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'5068800'
        endif

        if (FXTAL == 7)
        messg "7.15909␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'7159090'
        endif

        if (FXTAL == 10)
        messg "10␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'10000000'
        endif

        if (FXTAL == 12)
        messg "12␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'12000000'
        endif

        if (FXTAL == 16)
        messg "16␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'16000000'
        endif

        if (FXTAL == 20)
        messg "20␣MHz␣timing␣element␣specified."
        FOSC    EQU             D'20000000'
        endif

        ;; Make sure that a *valid* crystal frequency was
            specified.
        ifndef  FOSC
        error "Invalid␣frequency␣specified:␣must␣be␣3,4,5,7,10,16,
            ␣or␣20."
        endif


;;; ------------------------------------------------------------------------
;;; Assembler Equates Section. Define assembly-time constants here
;;;                              using the EQU assembler directive.
;;; ------------------------------------------------------------------------

;;; ------------------------------------------------------------------------
;;; Variable Address Assignments.
;;; ------------------------------------------------------------------------
        UDATA                   ; Start of the uninitialized data
                                ; section. The following
                                ; statements reserve memory, the
                                ; address of which is allocated by
                                ; the linker.

I2C_ADDR:               res 1
I2C_REG:                res 1
I2C_DAT:                res 1

        global  I2C_ADDR, I2C_REG, I2C_DAT


I2C_PTRHIGH:    res 1           ; High byte of the saved pointer.
        global  I2C_PTRHIGH
I2C_PTRLOW:     res 1           ; Low byte of the saved pointer.
        global  I2C_PTRLOW



TemporaryVars:  UDATA_OVR
        ;; For DBUS_TxStringN
NUMBYTES:       res 1           ; Number of bytes left to send.

;;; ------------------------------------------------------------------------
;;; Macros.  (File-specific)
;;; ------------------------------------------------------------------------
;;; ------------------------------------------------------------------------
;;; Subroutines.
;;; ------------------------------------------------------------------------
        CODE                    ; Start the code section.
```

```
;;; ************************************************************
;;; Subroutine Name:     I2C_Init
;;; Description:         Initalizes I2C peripheral
;;; Requires:            None
;;; Returns:             None
;;; Locations Affected: W is destroyed
;;; ************************************************************
I2C_Init:
        global I2C_Init

        banksel PORTC                   ; Select Bank 0
        bsf     PORTC,3                 ; SCL = HIGH
        bsf     PORTC,4                 ; SDA = HIGH
        banksel TRISC                   ; Select Bank 1
        bsf     TRISC,3                 ; SCL = Input
        bsf     TRISC,4                 ; SDA = Input

        movlw   .49                     ; 100kHz I2C clock rate
                                        ; at 20MHz
        movwf   SSPADD                  ; SSPADD =
                                        ; (20MHZ/100KHz)/4 - 1
        movlw   b'10000000'
        movwf   SSPSTAT                 ; disable slew rate
                                        ; control
        movlw   b'00000000'
        movwf   SSPCON2                 ; Idle

        bcf     STATUS,RP0              ; Select Bank 0
        movlw   b'00111000'             ; SSP enable, Master mode
        movwf   SSPCON                  ; Clock rate =
                                        ; FOSC/(4*(SSPADD+1))

        return

;;; ************************************************************
;;; Subroutine Name:     I2C_Write
;;; Description:         Writes to slave device via I2C bus.
;;; Requires:            I2C slave device address in I2C_ADDR.
;;;                      Register address in I2C_REG.
;;;                      Data to write to register in I2C_DATA.
;;;
;;;                      I2C Timing is as follows:
;;;                      Start + I2C_ADDR + Write + Ack + I2C_REG +
;;;                            + Ack + I2C_DATA + Ack + Stop
;;;
;;; Returns:             Nothing.
;;; Locations Affected: W is destroyed.
;;; ************************************************************
I2C_Write:
        global I2C_Write

        call    StartI2C        ; Sets SSPCON2.SEN and waits for
                                ; SSPIF
        bcf     I2C_ADDR,0      ; Clear bit 0 for write
        movf    I2C_ADDR,W        ; Load slave device address
        call    Send_Byte         ; Writes to SSPBUF and waits for
                                  ; SSPIF
        movf    I2C_REG,W         ;
        call    Send_Byte         ; Set register address

        movf    I2C_DAT,W         ;
        call    Send_Byte         ; Send register value

        call    StopI2C           ; Send STOP condition to bus

        return

;;; ************************************************************
;;; Subroutine Name:    I2C_BlockWrite
;;; Description:        Writes a block to a slave device via I2C
;;;                     bus.
;;; Requires:           I2C slave device address in I2C_ADDR.
;;;                     Register address in I2C_REG.
;;;                     Data to write to register in I2C_DATA.
;;;
;;;                     I2C Timing is as follows:
;;;                     Start + I2C_ADDR + Write + Ack + I2C_REG +
;;;                             Ack + I2C_DATA + Ack + Stop
;;;
;;; Returns:            Nothing.
;;; Locations Affected: W is destroyed.
;;; ************************************************************


;;; ************************************************************
;;; Subroutine Name:    I2C_BlockWrite
;;; Description:        Uses the DBUS to send out a NULL-
;;;                     terminated string constant (a string that
;;;                     resides in the code space).
;;;                     This routine calls another (specialized)
;;;                     subroutine, DBUS_GetStringKChar
;;;                     in order to overcome the
;;;                     PIC CPU limitation not allowing pointers
;;;                     to constants.
;;;                     To send a string in RAM, see DBUS_TxString
;;; Requires:           PTRHigh - high byte of address of string
;;;                               constant.
;;;                     PTRLow -  low byte of address of string
;;;                               constant.
;;; Returns:            Nothing.
;;; Locations Affected: PTRHigh, PTRLow will point at the NULL.
;;;                     W will contain the NULL.
;;; ************************************************************
I2C_BlockWrite:
        global  I2C_BlockWrite    ; Make this subroutine callable
                                  ; outside the module.

        ;; This routine uses a "trick" to allow specification of
        ;; and arbitrary string constant by PTRHigh:PTRLow.
        ;; The trick is to make a subroutine call to an
        ;; intermediate piece of code which then uses a "computed
        ;; goto" to jump to a string character.
        ;; When making a subroutine call, the return address
        ;; gets pushed onto the stack.  The intermediate
        ;; subroutine itself does not pull the value off the
        ;; stack, but instead passes control to the table.
        ;; This is slightly different than the suggested method
        ;; by Microchip in AN556 (Implementing a Table Read)
        ;; wherein the table itself includes an instruction to
        ;; complete a computed goto statement.

        banksel I2C_PTRLOW
        ;; Get the first byte.
        call    I2C_GetStringKByte

        ;; Return to this page if the string constant was
        ;; located in another page.
        pagesel I2C_BlockWrite

        ;; First byte indicates the number of bytes to send
        banksel NUMBYTES
        movwf   NUMBYTES

        call    StartI2C          ; Sets SSPCON2.SEN and waits for
                                  ; SSPIF
        bcf     I2C_ADDR,0        ; Clear bit 0 for write
        movf    I2C_ADDR,W        ; Load slave device address
        call    Send_Byte         ; Writes to SSPBUF and waits for
                                  ; SSPIF
        movlw   0xA0              ;
        call    Send_Byte         ; Send the block write strobe

        movf    NUMBYTES,W        ; Send the number of bytes to
        call    Send_Byte         ; write


I2C_BlockWriteLoop:
        banksel I2C_PTRLOW
        ;; Increment the pointer.
        incf    I2C_PTRLOW,F      ; Low byte.
        btfsc   STATUS,Z
        incf    I2C_PTRHIGH,F     ; High byte, if there is a carry
                                  ; from the low byte.

                ;; Get a byte.
        call    I2C_GetStringKByte
        ;; Return to this page if the string constant was
        ;; located in another page.
        pagesel I2C_BlockWriteLoop
        call    Send_Byte         ; Send the number of bytes to
                                  ; write
        decfsz  NUMBYTES,F
        goto    I2C_BlockWriteLoop

        call    StopI2C         ; Send STOP condition to bus
        return


I2C_GetStringKByte:
        ;; Address of string is PTRHigh:PTRLow.
        ;; Use a computed GOTO to get the character to send.
        movf    I2C_PTRHIGH,W
        movwf   PCLATH          ; High byte of address.
        movf    I2C_PTRLOW,W
        movwf   PCL             ; Low Byte of address.
        ;; Execution should never past here.  In a way,
        ;; this is a subroutine without a return statement.
        ;; The actual return is contained along with the
        ;; string 'table' data.



;;; ***********************************************************
;;; Subroutine Name:    I2C_Read
;;; Description:        Read a slave device register via I2C bus.
;;; Requires:           I2C slave device address in I2C_ADDR.
;;;                     Register address in I2C_REG.
;;;
;;;                     I2C Timing is as follows:
;;;                     Start + I2C_ADDR + Write + Ack + I2C_REG +
;;;                             Ack + I2C_DATA + Ack + Stop
;;;
;;; Returns:            Register in I2C_DATA.
;;; Locations Affected: W is destroyed.
;;; ***********************************************************
I2C_Read:
        global  I2C_Read

        call    StartI2C        ; Sets SSPCON2.SEN and waits for
                                ; SSPIF
        bsf     I2C_ADDR,0      ; Set bit 0 for read
        movf    I2C_ADDR,W      ; Load slave device address
        call    Send_Byte       ; Writes to SSPBUF and waits for
                                ; SSPIF
        call    RecI2C          ; Receive byte

        banksel SSPBUF
        movf    SSPBUF,W        ; Save received byte
        movwf   I2C_DAT

        call    Non_Ack         ; Send NACK

        call    StopI2C         ; Send STOP condition to bus

        return
```

```
;;; ***********************************************************
;;; Subroutine Name:    I2C_BlockRead
;;; Description:        Read a slave device register via I2C bus.
;;; Requires:           I2C slave device address in I2C_ADDR.
;;;                     Number of bytes to be read in W.
;;;
;;;                     I2C Timing is as follows:
;;;                     Start + I2C_ADDR + Write + Ack + I2C_REG +
;;;                             Ack + I2C_DATA + Ack + Stop
;;;
;;; Returns:            Register in I2C_DATA.
;;; Locations Affected: W is destroyed.
;;; ***********************************************************
I2C_BlockRead:
        global  I2C_BlockRead

        movwf   NUMBYTES        ; The number of bytes to be read

        call    StartI2C        ; Sets SSPCON2.SEN and waits for
                                ; SSPIF
        bcf             I2C_ADDR,0      ; Clear bit 0 for write
        movf    I2C_ADDR,W      ; Load slave device address
        call    Send_Byte       ; Writes to SSPBUF and waits for
                                ; SSPIF
        movlw   0xA1
        call    Send_Byte       ; Send the block read strobe
        movf    NUMBYTES,W
        call    Send_Byte       ; Send the number of bytes to read

        call    RestartI2C      ; Sets SSPCON2.RSEN and waits for
                                ; SSPIF
        bsf     I2C_ADDR,0      ; Set bit 0 for read
        movf    I2C_ADDR,W      ; Load slave device address
        call    Send_Byte       ; Writes to SSPBUF and waits for
                                ; SSPIF
RxLoop:
        call    RecI2C          ; Receive byte

        banksel SSPBUF
        movf    SSPBUF,W        ; Save received byte
        movwf   INDF            ; Use indirect addressing
        incf    FSR,F           ; Increment file select register

        decfsz  NUMBYTES,F
        goto    TxACK
        goto    TxNACK

TxACK:
        call    An_Ack
        goto    RxLoop

TxNACK:
        call    Non_Ack             ; Send NACK to signal end of read
        call    StopI2C             ; Send STOP condition to bus

        return


;;;******   Send START condition to bus   *******
;;;
StartI2C:                           ; Initiate the I2C START condition
        banksel SSPCON2
        bsf     SSPCON2,SEN
        goto    I2C_Done
;;;
;;;******   Send STOP condition to bus   *******
;;;
StopI2C:
        banksel SSPCON2
        bsf     SSPCON2,PEN
        goto    I2C_Done
;;;
;;;******   Send RESTART condition to bus   *******
;;;
RestartI2C:
        banksel SSPCON2
        bsf     SSPCON2,RSEN
        goto    I2C_Done
;;;
;;;******   Send a Non-Acknowledge status to bus (ACK=1)   *******
;;;
Non_Ack:
        banksel SSPCON2
        bsf     SSPCON2,ACKDT     ; Set the ACK bit
        bsf     SSPCON2,ACKEN     ; Initiate the NACK sequence.
        goto    I2C_Done
;;;
;;;******   Send an Acknowledge status to bus (ACK=0)    *******
;;;
An_Ack:
        banksel SSPCON2
        bcf     SSPCON2,ACKDT     ; Clear the ACK bit
        bsf     SSPCON2,ACKEN     ; Initiate the NACK sequence.
        goto    I2C_Done
;;;
;;;******   Send data to I2C bus from Wreg.    *******
;;;
Send_Byte:
        banksel SSPBUF
        movwf   SSPBUF
        goto    I2C_Done
;;;
;;;******   Enable I2C Receive for Master Mode    *******
;;;
RecI2C:
        banksel SSPCON2
```

```
          bsf       SSPCON2,RCEN      ; Set the receive enable bit.
          goto      I2C_Done
;;;
;;;******  Check the I2C stage is completed   ********
;;;
I2C_Done:
          banksel   PIR1
          btfss     PIR1,SSPIF        ; Poll for SSPIF
          goto      $-1
          bcf       PIR1,SSPIF

          return

;;; ------------------------------------------------------------------
;;; Constant Data (if not placed along with the subroutines).
;;; ------------------------------------------------------------------

          END
```

```
;;; ************************************************************
;;; Library Name:          math_lib.asm
;;; Purpose:               Math routines
;;; Revision:              0.1
;;; Date:                  03 February 2011
;;; Author:                D. Smetaniuk
;;; Copyright:             Public Domain
;;; ************************************************************
;;; Special Directions:
;;;   - To use these routines, include ”math_lib.h” in the source
;;;     code calling the routines.
;;;
;;; ************************************************************
;;; Revision History:
;;;     0.1 - 3 February 2011 - DPS
;;;             - Creation.
;;; ************************************************************


;;; ------------------------------------------------------------
;;; Assembler Equates Section. Define assembly-time constants here
;;;                            using the EQU assembler directive.
;;; ------------------------------------------------------------

;;; ------------------------------------------------------------
;;; Variable Address Assignments.
;;; ------------------------------------------------------------
        UDATA                           ; Start of the uninitialized data
                                        ; section. The following
                                        ; statements reserve memory, the
                                        ; address of which is allocated by
                                        ; the linker.

ACCa:
ACCa4:          res     1               ; Three accumulators for
ACCa3:          res     1               ; intermediate results
ACCa2:          res     1
ACCa1:          res     1

ACCb:
ACCb4:          res     1
ACCb3:          res     1
ACCb2:          res     1
ACCb1:          res     1

ACCc:
ACCc4:          res     1
ACCc3:          res     1
ACCc2:          res     1
ACCc1:          res     1

bitcount        res 1

        ;; Make these variables identifiable to calling code.
        global  ACCa,ACCa1,ACCa2,ACCa3,ACCa4
        global  ACCb,ACCb1,ACCb2,ACCb3,ACCb4
        global  ACCc,ACCc1,ACCc2,ACCc3,ACCc4


;;; ----------------------------------------------------------------
;;; Subroutines.
;;; ----------------------------------------------------------------
        CODE                    ; Start the code section.

;;; **********************************************************
;;; Subroutine Name:    ABS16
;;; Description:        Computes the absolute value of a signed
;;;                     16-bit value.
;;;
;;;                     |ACCa| --> ACCa
;;;
;;; Requires:           Signed 16-bit value is in accumulator
;;;                     registers ACCa2 (high byte), and ACCa1
;;;                     (low byte)
;;; Returns:            Result in ACCa2 and ACCa1 registers
;;; Locations Affected: W is destroyed
;;; **********************************************************
ABS16
        global  ABS16

        btfss   ACCa2,7         ; Test MSB (Sign) bit
        goto    Positive
        ; 16 bit decrement
        movf    ACCa1,F         ; Test if zero
        skpnz
         decf   ACCa2,F
        decf    ACCa1,F

        btfss   ACCa2,7         ; Test MSB (Sign) bit again
                                ; This is for the 0x8000 (-32768)
                                ; case, which will return 0x7FFF
                                ; (+32767) which is the highest
                                ; possible value
        goto    Positive
        ; 16 bit 1's complement
        comf    ACCa1,F
        comf    ACCa2,F

Positive:
        return

;;; **********************************************************
;;; Subroutine Name:    ADD32U
;;; Description:        Adds two 32-bit unsigned values.
;;;
;;;                     ACCa + ACCb --> ACCb
;;;
;;;                     Note: Does not test for overflow.
;;;                          Subroutine by Regulus Berdin.
;;; Requires:                Unsigned 32-bit values in accumulator
;;;                          registers ACCa[4:1] and ACCb[4:1]
;;; Returns:                 Result in ACCb[4:1] registers
;;; Locations Affected: W is destroyed
;;; ***********************************************************
ADD32U:
        global  ADD32U

        movf    ACCa1,W
        addwf   ACCb1,F

        movf    ACCa2,W
        btfsc   STATUS,C
        incfsz  ACCa2,W
        addwf   ACCb2,F

        movf    ACCa3,W
        btfsc   STATUS,C
        incfsz  ACCa3,W
        addwf   ACCb3,F

        movf    ACCa4,W
        btfsc   STATUS,C
        incfsz  ACCa4,W
        addwf   ACCb4,F

        return


;;; ***********************************************************
;;; Subroutine Name:         MUL16U
;;; Description:             Multiplies two 16-bit unsigned numbers and
;;;                          Stores the result as a 32-bit unsigned
;;;                          Number.
;;;
;;;                          ACCa * ACCb --> ACCc
;;;
;;;                          Note: Does not test for overflow.
;;;                          Subroutine by malin@onspec.co.uk
;;; Requires:                Unsigned 16-bit values in accumulator
;;;                          registers ACCa[2:1] and ACCb[2:1]
;;; Returns:                 Result in ACCc[4:1] registers
;;; Locations Affected: W is destroyed
;;; ***********************************************************
;;;     malin@onspec.co.uk
;;;
;;;     Program length 32 line
;;;     time 129 to 228 cycles
;;;
;;;     This program looks at the LSB of a1 to decide whether to
;;;     add ACCb1 to ACCc2 and ACCb2 to ACCc3, with appropriate
;;;     carrys. It then looks at the LSB of ACCa2 to decide
;;;     whether to add ACCb1 to ACCc3 and ACCb2 to ACCc4, again
;;;     with appropriate carrys. The rotates then only have to be
;;;     done 8 times.
;;;
;;;     This is uses slightly more program but takes a little less
;;;     time than a routine that performs one 16 bit addition per
;;;     rotate and 16 rotates.
;;;
;;;     Multiple byte addition routine from Microchip AN617
;;;     Result registers used as loop counter from Bob Fehrenbach
;;;     and Scott Dattalo
;;;

MUL16U:
        global  MUL16U

        clrf    ACCc4
        clrf    ACCc3
        clrf    ACCc2
        movlw   0x80
        movwf   ACCc1

nextbit:
        rrf     ACCa2,F
        rrf     ACCa1,F

        btfss   STATUS, C
        goto    nobit_l
        movf    ACCb1,W
        addwf   ACCc2,F

        movf    ACCb2, W
        btfsc   STATUS, C
        incfsz  ACCb2, W
        addwf   ACCc3, F
        btfsc   STATUS, C
        incf    ACCc4,F
        clrc

nobit_l:
        btfss   ACCa1, 7
        goto    nobit_h
        movf    ACCb1,W
        addwf   ACCc3,F
        movf    ACCb2,W
        btfsc   STATUS, C
        incfsz  ACCb2,W
        addwf   ACCc4,F

nobit_h:
        rrf     ACCc4,F
        rrf     ACCc3,F
        rrf     ACCc2,F
        rrf     ACCc1,F
        btfss   STATUS, C
        goto    nextbit

        return


;;; **********************************************************
;;; Subroutine Name:     DIV32U
;;; Description:         Divides two 32-bit unsigned numbers and
;;;                      Stores the result as a 32-bit unsigned
;;;                      Number.
;;;
;;;                      ACCa / ACCb --> ACCa; R --> ACCc
;;;
;;; Requires:            Unsigned 32-bit values in the following
;;;                      accumulator registers:
;;;                              Dividend:       ACCa[4:1]
;;;                              Divisor:        ACCb[4:1]
;;; Returns:             Unsigned 32-bit values in the following
;;;                      accumulator registers:
;;;                              Quotient:       ACCa[4:1]
;;;                              Remainder:      ACCc[4:1]
;;; Locations Affected: W is destroyed
;;; **********************************************************

DIV32U:
        global  DIV32U

        movlw   .32             ; 32-bit divide by 32-bit
        movwf   bitcount
        clrf    ACCc1           ; Clear remainder
        clrf    ACCc2
        clrf    ACCc3
        clrf    ACCc4

dvloop:
        clrc                    ; Set quotient bit to 0
        rlf     ACCa1,F         ; Shift left dividend and quotient
        rlf     ACCa2,F         ; LSB.
        rlf     ACCa3,F
        rlf     ACCa4,F         ; LSB into carry...
        rlf     ACCc1,F         ; ...and then into the partial
        rlf     ACCc2,F         ; remainder.
        rlf     ACCc3,F
        rlf     ACCc4,F

        skpnc                   ; Check for overflow.
        goto    subd
        movfw   ACCb4           ; Compare partial remainder and
        subwf   ACCc4,W         ; divisor.
        skpz
        goto    testgt          ; Not equal so test if remdrH is
                                ; greater.
        movfw   ACCb3           ; Compare partial remainder and
        subwf   ACCc3,W         ; divisor.
        skpz
        goto    testgt          ; Not equal so test if remdrH is
                                ; greater.
        movfw   ACCb2           ; Compare partial remainder and
        subwf   ACCc2,W         ; divisor.
        skpz
        goto    testgt          ; Not equal so test if remdrH is
                                ; greater.
        movfw   ACCb1           ; High bytes are equal, compare
        subwf   ACCc1,W         ; low bytes
testgt:
        skpc                    ; Carry set if remainder>=divisor
        goto    remrlt

subd:
        movfw   ACCb1           ; Subtract divisor from partial
        subwf   ACCc1,F         ; remainder.
        skpc                    ; Test for borrow

        decf    ACCc2,F         ; Subtract borrow
        movfw   ACCb2
        subwf   ACCc2,F
        skpc                    ; Test for borrow

        decf    ACCc3,F         ; Subtract borrow
        movfw   ACCb3
        subwf   ACCc3,F
        skpc                    ; Test for borrow

        decf    ACCc4,F         ; Subtract borrow
        movfw   ACCb4
        subwf   ACCc4,F
        bsf     ACCa1,0         ; Set quotient bit to 1.
                                ; Quotient replaces dividend which
                                ; is lost.
remrlt:
        decfsz  bitcount,F
        goto    dvloop
        return

        END
```

```
;;; ************************************************************
;;; Filename:   zanalyzer.asm
;;; Purpose:    Main program
;;; Revision:   0.0
;;; Date:       20 April, 2011
;;; Author:     D. Smetaniuk
;;; Copyright:  Public Domain
;;; ************************************************************
;;; Revision History:
;;;     0.0     20 April 2011 - DPS
;;;             Creation.
;;; ************************************************************


        list                    ; Turn on list output.

        ;; Include register, bit, and other info specific to
        ;; the specified device.
        ifdef   __16F873
        #include <p16f873.inc>
        messg "Assembling␣for␣PIC16F873."
        endif
        ifdef   __16F874
        #include <p16f874.inc>
        messg "Assembling␣for␣PIC16F874."
        endif
        ifdef   __16F876
        #include <p16f876.inc>
        messg "Assembling␣for␣PIC16F876."
        endif
        ifdef   __16F877
        #include <p16f877.inc>
        messg "Assembling␣for␣PIC16F877."
        endif
        ifdef   __16F873A
        #define A_Device
        #include <p16f873a.inc>
        messg "Assembling␣for␣PIC16F873A."
        endif
        ifdef   __16F874A
        #include <p16f874a.inc>
        #define A_Device
        messg "Assembling␣for␣PIC16F874A."
        endif
        ifdef   __16F876A
        #define A_Device
        #include <p16f876a.inc>
        messg "Assembling␣for␣PIC16F876A."
        endif
        ifdef   __16F877A
        #define A_Device
        #include <p16f877a.inc>
        messg "Assembling␣for␣PIC16F877A."
        endif
```

```
;;; ----------------------------------------------------------------------
;;; Included files (including library headers).
;;; ----------------------------------------------------------------------

        #include ”branches.h” ; Conditional branching macros
        #include ”del_lib.h”  ; Delay routine headers.
        #include ”dbus_lib.h” ; FIFO databus library headers
        #include ”i2c_lib.h”  ; I2C library headers
        #include ”ad5933.h”   ; AD5933 register definitions
        #include ”math_lib.h”

;;; ----------------------------------------------------------------------
;;; Assembler Equates Section. Define assembly−time constants here
;;;                            using the EQU assembler directive.
;;; ----------------------------------------------------------------------
RESETVECTOR      EQU    0x000   ; Address of RESET vector.
INTERRUPTVECTOR  EQU    0x004   ; Address of peripheral interrupt
                                ; vector.
CODESTART        EQU    0x008   ; Starting location of program
                                ; code.

;; Port pin definitions

;; PORT A
PIN_A0           EQU    0
PIN_A1           EQU    1
PIN_A2           EQU    2
PIN_RA0          EQU    3
PIN_RA1          EQU    4
PIN_RA2          EQU    5
;; PORT B
PIN_RD           EQU    1
PIN_WR           EQU    2
PIN_RXF          EQU    4
;; PORT C
PIN_E_3V         EQU    0
PIN_E_LTRAN      EQU    1
PIN_MCLK         EQU    2
PIN_SCL          EQU    3
PIN_SDA          EQU    4
PIN_EN_B         EQU    5
PIN_EN_A         EQU    6
PIN_EN_R         EQU    7
;; PORT E
PIN_LED          EQU    0
PIN_TXE          EQU    2


;; FLAGS register bits
F_TIMER          EQU    0
```

```
;; FREQ2 Register values for specific frequencies
FREQLOW         EQU     0x01
FREQ1K          EQU     0x0A
FREQ10K         EQU     0x14

;; For cycle time
;; 1500 counts = 2:30
CHTIME_HI       EQU     0x05
CHTIME_LO       EQU     0xDC
;; 3000 counts = 5:00
SETLTIME_HI     EQU     0x0B
SETLTIME_LO     EQU     0xB8



;;; ----------------------------------------------------------------------------
;;; Defines
;;; ----------------------------------------------------------------------------


        #define MAXTIME_MSB     0x10    ; ~5 Hz display cycle

;; PORT A
        #define bitA0           PORTA, PIN_A0
        #define bitA1           PORTA, PIN_A1
        #define bitA2           PORTA, PIN_A2
        #define bitRA0          PORTA, PIN_RA0
        #define bitRA1          PORTA, PIN_RA1
        #define bitRA2          PORTA, PIN_RA2
;; PORT B
        #define bitRD           PORTB, PIN_RD
        #define bitWR           PORTB, PIN_WR
        #define bitRXF          PORTB, PIN_RXF
;; PORT C
        #define bitE_3V         PORTC, PIN_E_3V
        #define bitE_LTRAN      PORTC, PIN_E_LTRAN
        #define bitMCLK         PORTC, PIN_MCLK
        #define bitSCL          PORTC, PIN_SCL
        #define bitSDA          PORTC, PIN_SDA
        #define bitEN_B         PORTC, PIN_EN_B
        #define bitEN_A         PORTC, PIN_EN_A
        #define bitEN_R         PORTC, PIN_EN_R
;; PORT D
        #define DBUS            PORTD
;; PORT E
        #define bitLED          PORTE, PIN_LED
        #define bitTXE          PORTE, PIN_TXE


        #define __TIMER1_OFF            bcf     T1CON,TMR1ON
        #define __TIMER1_ON             bsf     T1CON,TMR1ON
        #define __LED_OFF               bsf     PORTE, PIN_LED
        #define __LED_ON                bcf     PORTE, PIN_LED
        #define __INT_ENABLE_CCP1         bsf     PIE1,CCP1IE
        #define __INT_ENABLE_CCP2         bsf     PIE2,CCP2IE
        #define __INT_ENABLE_TMR2         bsf     PIE1,TMR2IE
        #define __INT_ENABLE_PERI         bsf INTCON,PEIE

        #define __INT_DISABLE_CCP1        bcf PIE1,CCP1IE
        #define __INT_DISABLE_CCP2        bcf     PIE2,CCP2IE
        #define __INT_DISABLE_TMR2        bcf     PIE1,TMR2IE
        #define __INT_DISABLE_PERI        bcf INTCON,PEIE

        ; enable global interrupts
        #define sei                       bsf     INTCON,GIE
        ; disable global interrupts
        #define cli                       bcf     INTCON,GIE

;;; ---------------------------------------------------------------------------
;;; Macros
;;; ---------------------------------------------------------------------------

; This Macro Saves register contents
PUSH_MACRO macro
        movwf W_TEMP              ; Copy W to a Temporary Register
                                  ; regardless of current bank
        swapf STATUS,W            ; Swap STATUS nibbles and place
                                  ; into W register
        movwf STATUS_TEMP         ; Save STATUS to a Temporary
                                  ; register in Bank0
        clrf  STATUS
        endm                      ; End this Macro

; This Macro Restores register contents
POP_MACRO macro
        swapf STATUS_TEMP,W       ; Swap original STATUS register
                                  ; value into W (restores original
                                  ; bank)
        movwf STATUS              ; Restore STATUS register from
                                  ; W register
        swapf W_TEMP,F            ; Swap W_Temp nibbles and return
                                  ; value to W_Temp
        swapf W_TEMP,W            ; Swap W_Temp to W to restore
                                  ; original W value without
                                  ; affecting STATUS
        ENDM                      ; End this Macro

mov32 macro src, dest
        movf    src,W
        movwf   dest
        movf    src+1,W
        movwf   dest+1
        movf    src+2,W
        movwf   dest+2
        movf    src+3,W
        movwf   dest+3
        endm

mov16 macro src, dest
        movf    src,W
        movwf   dest
        movf    src+1,W
        movwf   dest+1

        endm

;; Sets the PWM output to 819.2 Hz
;; Can't do 819.2 Hz with PWM. Need to use TMR2 and a counter
;; register
__PWM_800 macro
        banksel PR2
        movlw   0x7C
        movwf   PR2
        banksel CCP1CON                 ; Set up TMR2 with 1:4
        movlw   b'00100101'             ; prescale, 1:5 postscale,
        movwf   T2CON                   ; turn on.
        movlw   b'00000000'
        movwf   CCP1CON                 ; CCP1 disabled
        endm

;; Sets the PWM output to 8.192 kHz
__PWM_8K macro
        banksel PR2
        movlw   0x7C        ;0x7D
        movwf   PR2
        banksel CCP1CON
        movlw   0x3E
        movwf   CCPR1L
        movlw   b'00000101'             ; Set up TMR2 with 1:4
        movwf   T2CON                   ; prescale, turn on.
        movlw   b'00101100'
        movwf   CCP1CON                 ; Set up CCP1 for PWM
        endm

;; Sets the PWM output to 81.92 kHz
__PWM_80K macro
        banksel PR2
        movlw   0x31
        movwf   PR2
        banksel CCP1CON
        movlw   0x19
        movwf   CCPR1L
        movlw   b'00000100'             ; Set up TMR2 with 1:1
        movwf   T2CON                   ; prescale, turn on
        movlw   b'00001100'
        movwf   CCP1CON                 ; Set up CCP1 for PWM
        endm

;; Sets the PWM output to 819.2 kHz
__PWM_800K macro
```

```
        banksel PR2
        movlw   0x04
        movwf   PR2
        banksel CCP1CON
        movlw   0x02
        movwf   CCPR1L
        movlw   b'00000100'          ; Set up TMR2 with 1:1
        movwf   T2CON                ; prescale, turn on
        movlw   b'00101100'
        movwf   CCP1CON              ; Set up CCP1 for PWM
        endm

;; Sets the PWM output to 4.096 MHz
__PWM_4M macro
        banksel PR2
        clrf    PR2
        banksel CCP1CON
        clrf    CCPR1L
        movlw   b'00000100'          ; Set up TMR2 with 1:1
        movwf   T2CON                ; prescale, turn on
        movlw   b'00101100'
        movwf   CCP1CON              ; Set up CCP1 for PWM
        endm

;;; --------------------------------------------------------------------
;;; Variable Address Assignments.
;;; --------------------------------------------------------------------

        UDATA                    ; Start of uninitialized data
                                 ; section. Reserve memory for
                                 ; variables using the "RES"
                                 ; directive here. Note that the
                                 ; linker will assign addresses.

W_TEMP          RES     1        ; Temp storage for W register
STATUS_TEMP     RES     1        ; Temp storage for STATUS register
FLAGS           RES     1
TICKCNT         RES     1
CHANNELS        RES     1
COUNTER         RES     1
HIGH_CHAR       RES     1
LOW_CHAR        RES     1
TIME_HI         RES     1
TIME_LO         RES     1
TIMEB_HI        RES     1
TIMEB_LO        RES     1
RRANGE          RES     1
FREQVAL         RES     1
FREQINC         RES     1
FREQCNT         RES     1
TEMPA           RES     4
TEMPB           RES     4
ZDATA_A
REAL_A          RES     2
```

```
IMAG_A          RES     2
ZDATA_B
REAL_B          RES     2
IMAG_B          RES     2


;;; ---------------------------------------------------------------------
;;; Establish the OPTION register bit values.
;;; ---------------------------------------------------------------------
;;; The '__CONFIG' directive is used to embed PIC configuration
;;; data within an assembly file. The labels following the
;;; directive are located in the .inc file.  See the device data
;;; sheet for additional information on the configuration word.
ifndef A_Device
        __CONFIG _CP_OFF & _WDT_OFF & _BODEN_OFF & _PWRTE_ON &
            _HS_OSC & _WRT_ENABLE_ON & _LVP_ON & _CPD_OFF
endif
;;; _CP_OFF:      turn off code protection.  Don't change this
;;;               unless you want a device that can never be
;;;               programmed again. This is a "bug" in some PIC
;;;               devices.
;;; _WDT_OFF:     turn off the watchdog timer.
;;; _BODEN_ON:    turn on power brown-out reset.
;;; _PWRTE_ON:    turn on power-up timer.
;;; _XT_OSC:      specify that the device is using an XT oscillator.
;;; _WRT_ENABLE_ON:  enable writing to data EEPROM.
;;; _LVP_OFF:     disable low-voltage in-circuit programming.
;;; _CPD_OFF:     disable data EEPROM write protection.

ifdef A_Device
        messg "A␣revision␣device."
         __CONFIG _CP_OFF & _WDT_OFF & _BODEN_OFF & _PWRTE_ON &
             _HS_OSC & _WRT_OFF & _LVP_ON & _CPD_OFF
endif

;;; _CP_OFF:      turn off code protection.  Don't change this
;;;               unless you want a device that can never be
;;;               programmed again. This is a "bug" in some PIC
;;;               devices.
;;; _WDT_OFF:     turn off the watchdog timer.
;;; _BODEN_ON:    turn on power brown-out reset.
;;; _PWRTE_ON:    turn on power-up timer.
;;; _XT_OSC:      specify that the device is using an XT oscillator.
;;; _WRT_OFF:     disable write-protection of program FLASH.
;;; _LVP_OFF:     disable low-voltage in-circuit programming.
;;; _CPD_OFF:     disable data EEPROM write protection.


;;; ---------------------------------------------------------------------
;;; Main Program.
;;; ---------------------------------------------------------------------
        CODE                            ; Start of code section.
Main:
Init:
        bcf     STATUS, RP0             ; Select Bank 0.
        clrf    PORTA
        clrf    PORTB
        bsf     bitRD                   ; RD = HIGH
        movlw   b'00011110'     .
        movwf   PORTC
        clrf    PORTD
        clrf    PORTE
        clrf    FLAGS
Inputs:
        bsf     STATUS, RP0             ; Select Bank 1.
        movlw   b'00000110'             ; Configure PORTA as
        movwf   ADCON1                  ; digital I/O.
        clrf    TRISA
        clrf    TRISB
        bsf     TRISB, PIN_RXF          ; RXF = Input.
        clrf    TRISC
        movlw   0xFF
        movwf   TRISD                   ; DBUS Port = Input.
        clrf    TRISE                   ; RE PORT output.
        bsf     TRISE, PIN_TXE          ; TXE = Input.
Interrupts:
        __INT_ENABLE_CCP2
        bcf     STATUS, RP0             ; Select Bank 0.
        __INT_ENABLE_PERI               ; Enable peripheral ints.

Peripherals:
        call    I2C_Init
        __LED_OFF
        __TIMER1_OFF
        clrf    TIME_HI                 ; Setup CCP2 for an
        clrf    TIME_LO                 ; interrupt every 100 ms.
        clrf    TMR1H                   ; This is used to time the
        clrf    TMR1L                   ; settle and measurement
        movlw   0xC8                    ; cycle.
        movwf   CCPR2H
        movlw   0x00
        movwf   CCPR2L
        movlw   b'00001011'
        movwf   CCP2CON
        movlw   b'00110000'             ; Set 1:8 prescale value
        movwf   T1CON                   ; for Timer 1.
        call    DBUS_Init

        call    DBUS_RxByte             ; Wait until a key is
                                        ; pressed.

        banksel PTRLOW                  ; Establish a pointer to
        movlw   low Sline1              ; the NULL-terminated
        movwf   PTRLOW                  ; string.
        movlw   high Sline1
        movwf   PTRHIGH
        call    DBUS_TxStringK
        call    InitChannels

        clrf    RRANGE
        clrf    COUNTER

        banksel PTRLOW                  ; Prompt user to press
        movlw   low Sline4              ; a key.
        movwf   PTRLOW
        movlw   high Sline4
        movwf   PTRHIGH
        call    DBUS_TxStringK
        call    DBUS_RxByte             ; Wait until a key is
                                        ; pressed.
        __TIMER1_ON
        sei
        __LED_ON

MainLoop:
        movlw   SETLTIME_HI             ; Load settle time.
        movwf   TIME_HI
        movlw   SETLTIME_LO
        movwf   TIME_LO

Wait1:                                  ; Wait for RH settle time
                                        ; (5 mins).
        btfss   FLAGS,F_TIMER
         goto   Wait1
        bcf     FLAGS,F_TIMER

        mov16   TIMEB_HI, TIME_HI       ; Load known delay time to
                                        ; allow measurements for
                                        ; each channel.

MeasureNext:
        btfss   CHANNELS,0              ; Test if channel present.
         goto   IncChannel

        banksel PTRLOW                  ; Establish a pointer to
        movlw   low Sline3              ; the NULL-terminated
        movwf   PTRLOW                  ; string.
        movlw   high Sline3
        movwf   PTRHIGH
        call    DBUS_TxStringK
        movlw   0x31                    ; Convert channel # to
        addwf   COUNTER,W               ; ASCII (and add 1).
        call    DBUS_TxByte             ; Print ASCII char.
        movlw   ","
        call    DBUS_TxByte

        movlw   0xF8                    ; Mask for channel address
        andwf   PORTA,W                 ; bits on PORTA.
        iorwf   COUNTER,W
        movwf   PORTA
```

```
cli
__PWM_800                               ; Set clock to 819.2 Hz
                                        ; for 0.1-0.9 Hz sweep.
bsf      STATUS, RP0                    ; Select Bank 1
__INT_ENABLE_TMR2
bcf      STATUS, RP0                    ; Select Bank 0
sei

movlw    SFREQ2                         ; Start freq register 2
movwf    I2C_REG
movlw    FREQLOW
movwf    I2C_DAT
call     I2C_Write                      ; Change start frequency

call     Measure

cli
bsf      STATUS, RP0                    ; Select Bank 1
__INT_DISABLE_TMR2
bcf      STATUS, RP0                    ; Select Bank 0
sei

__PWM_8K                                ; Set clock to 8.192 kHz
movlw    FREQLOW                        ; for 1-9 Hz sweep.
movwf    FREQVAL
movwf    FREQINC
movlw    .9
movwf    FREQCNT
call     FSweep

__PWM_80K                               ; Set clock to 81.92 kHz
movlw    FREQLOW                        ; for 10-90 Hz sweep.
movwf    FREQVAL
movwf    FREQINC
movlw    .9
movwf    FREQCNT
call     FSweep

__PWM_800K                              ; Set clock to 819.2 kHz
movlw    FREQLOW                        ; for 100-900 Hz sweep.
movwf    FREQVAL
movwf    FREQINC
movlw    .9
movwf    FREQCNT
call     FSweep

movlw    FREQ1K                         ; And now 1-9 kHz sweep.
movwf    FREQVAL
movwf    FREQINC
movlw    .9
movwf    FREQCNT
call     FSweep

__PWM_4M                                ; Set clock to 4.096 MHz
```

```
        movlw   FREQ10K                 ; for 10-100 kHz sweep.
        movwf   FREQVAL
        movwf   FREQINC
        movlw   .10
        movwf   FREQCNT
        call    FSweep

        movlw   "\r"
        call    DBUS_TxByte
        movlw   "\n"
        call    DBUS_TxByte

IncChannel:
        clrc
        rrf     CHANNELS,F
        skpnc
         bsf    CHANNELS,7              ; Rotate carry into bit 7
        incf    COUNTER,F
        movlw   0x08
        subwf   COUNTER,W
        skpnz
         goto   MeasureDone
        goto    MeasureNext

MeasureDone:
        clrf    COUNTER

Wait2:
        btfss   FLAGS,F_TIMER
         goto   Wait2
        bcf     FLAGS,F_TIMER

        goto    MainLoop
;;; Note that execution of the main program never gets past here!


;;; ----------------------------------------------------------------------
;;; Application-specific Subroutines.
;;; ----------------------------------------------------------------------

PrintHex:
        movwf   LOW_CHAR                ; Mask off lower and upper
        movwf   HIGH_CHAR               ; nibbles and put in
        swapf   HIGH_CHAR,F             ; registers.
        movlw   0x0F
        andwf   LOW_CHAR,F
        andwf   HIGH_CHAR,F

        movf    LOW_CHAR,W              ; Check to see if the
            lower
        bcf     STATUS,C                ; character is a number or
        sublw   0x09                    ; alpha and convert.
        movlw   0x30
        btfss   STATUS,C
        movlw   0x37

        addwf   LOW_CHAR, F

        movf    HIGH_CHAR,W             ; Check to see if the high
        bcf     STATUS,C                ; character is a number or
        sublw   0x09                    ; alpha and convert.
        movlw   0x30
        btfss   STATUS,C
        movlw   0x37

        addwf   HIGH_CHAR, F

        movf    HIGH_CHAR,W             ; Print ASCII chars
        call    DBUS_TxByte
        movf    LOW_CHAR,W
        call    DBUS_TxByte

        return

;;; ************************************************************
;;; Subroutine Name:    InitChannels
;;; Description:        This subroutine scans through all 8
;;;                     channels, identifying which ones are
;;;                     present and initializes the register
;;;                     values on the AD5933's
;;; Requires:           None
;;; Returns:            NULL
;;; Locations Affected: W is destroyed
;;; ************************************************************
InitChannels:
        clrf    CHANNELS
        clrf    COUNTER
        clrf    TIMEB_HI
        clrf    TIMEB_LO

        movlw   (ADDR5933<<1)           ; AD5933 serial bus
                                        ; address. The LSB is the
        movwf   I2C_ADDR                ; W/R bit, hence the shift

ThisChannel:
        banksel I2C_PTRLOW
        movlw   low Init5933
        movwf   I2C_PTRLOW
        movlw   high Init5933
        movwf   I2C_PTRHIGH

        movlw   ADRPTR                  ; Address pointer code.
        movwf   I2C_REG
        movlw   CTRLH                   ; Control register HIGH.
        movwf   I2C_DAT
        call    I2C_Write               ; Write to address pointer
        call    I2C_BlockWrite          ; Initalize all R/W
                                        ; registers (via table).
        movlw   ADRPTR               ; Address pointer code
        movwf   I2C_REG
        movlw   CTRLL                ; Status register
        movwf   I2C_DAT
        call    I2C_Write            ; Write to address pointer
        call    I2C_Read             ; Read register value back
        movf    I2C_DAT,W            ; to see if there is a
        sublw   0x08                 ; device on the channel.
        skpz
         goto   Next
        bsf     CHANNELS,0           ; If found, set bit to
                                     ; indicate channel present

        banksel PTRLOW               ; Establish a pointer to
        movlw   low Sline2           ; the NULL-terminated
        movwf   PTRLOW               ; string.
        movlw   high Sline2
        movwf   PTRHIGH
        call    DBUS_TxStringK       ; Write line
        movlw   0x31                 ; Convert channel # to
        addwf   COUNTER,W            ; ASCII (and add 1).
        call    DBUS_TxByte          ; Print ASCII character.
        movlw   ”\r”                 ; Print Newline.
        call    DBUS_TxByte
        movlw   ”\n”
        call    DBUS_TxByte

        movlw   CHTIME_LO            ; Increase delay time
        addwf   TIMEB_LO,F           ; (TIMEB) by another
        skpnc                        ; channel (CHTIME).
         incf   TIMEB_HI,F           ; Note: does not check for
        movlw   CHTIME_HI            ; overflow.
        addwf   TIMEB_HI,F

Next:
        clrc
        rrf     CHANNELS,F
        skpnc
         bsf    CHANNELS,7           ; Rotate carry into bit 7
        incf    COUNTER,F
        movlw   0x08
        subwf   COUNTER,W
        skpnz
         goto   ChannelsDone
        movlw   0xF8                 ; Mask for channel address
        andwf   PORTA,W              ; bits on PORTA.
        iorwf   COUNTER,W            ; Set channel address bits
        movwf   PORTA                ; on PORTA.
        goto    ThisChannel
ChannelsDone:
        clrf    COUNTER
        clrf    PORTA

        return
```

```
;;; *******************************************************
;;; Subroutine Name:    SetRange
;;; Description:        Sets the proper range resistor based on a
;;;                     measured impedance value
;;; Requires:           Measured values in REAL_A, REAL_B, IMAG_A,
;;;                     IMAG_B
;;; Returns:            NULL
;;; Locations Affected: W is destroyed, uses I2C_REG, I2C_DAT
;;; *******************************************************
SetRange:
                                        ; Uu
        mov16   REAL_B,ACCa2            ; Square Real.
        call    ABS16
        mov16   ACCa2,ACCb2
        call    MUL16U
        mov32   ACCc4,TEMPB
        mov16   IMAG_B,ACCa2            ; Square Imag.
        call    ABS16
        mov16   ACCa2,ACCb2
        call    MUL16U
        mov32   TEMPB,ACCa4             ; Add both squares.
        mov32   ACCc4,ACCb4
        call    ADD32U
        mov32   ACCb4,TEMPB

                                        ; Ui
        mov16   REAL_A,ACCa2            ; Square Real.
        call    ABS16
        mov16   ACCa2,ACCb2
        call    MUL16U
        mov32   ACCc4,TEMPA
        mov16   IMAG_A,ACCa2            ; Square Imag.
        call    ABS16
        mov16   ACCa2,ACCb2
        call    MUL16U
        mov32   TEMPA,ACCa4             ; Add both squares.
        mov32   ACCc4,ACCb4
        call    ADD32U
        mov32   TEMPB,ACCa4
        call    DIV32U

ChkByte4:
        movf    ACCa4,W                 ; Test if zero
        skpnz
         goto   ChkByte3
        sublw   0x06
        skpc
         goto   FoundR4
        goto    FoundR3
ChkByte3:
        movf    ACCa3,W                 ; Test if zero
        skpnz
         goto   ChkByte2
```

```
            sublw    0x0F
            skpc
             goto    FoundR3
            skpz
             goto    FoundR2
            movf     ACCa2,W
            sublw    0x42
            skpc
             goto    FoundR3
            skpz
             goto    FoundR2
            movf     ACCa1,W
            sublw    0x39
            skpc
             goto    FoundR3
            goto     FoundR2
ChkByte2:
            movf     ACCa2,W                    ; Test if zero
            skpnz
             goto    ChkByte1
            sublw    0x27
            skpc
             goto    FoundR2
            skpz
             goto    FoundR1
            movf     ACCa1,W
            sublw    0x09
            skpc
             goto    FoundR2
            goto     FoundR1
ChkByte1:
            movf     ACCa1,W
            sublw    0x63
            skpc
             goto    FoundR1
FoundR0:
            movlw    0x00
            goto     Found
FoundR1:
            movlw    0x08
            goto     Found
FoundR2:
            movlw    0x10
            goto     Found
FoundR3:
            movlw    0x18
            goto     Found
FoundR4:
            movlw    0x20
Found:
            movwf    RRANGE

            movlw    0xC7                       ; Mask for range resistor
            andwf    PORTA,W                    ; bits on PORTA.
```

```
        iorwf   RRANGE,W
        movwf   PORTA                   ; Set range resistor bits
                                        ; on PORTA
        return


;;; *********************************************************
;;; Subroutine Name:    ReadImpedance
;;; Description:        Reads values stored in the AD5933 memory
;;;                     via I2C bus
;;; Requires:
;;; Returns:            NULL
;;; Locations Affected: W is destroyed, uses I2C_REG, I2C_DAT
;;; *********************************************************
ReadImpedance:
        bsf     bitEN_B                 ; disable I2C for module B
WaitA:
        movlw   ADRPTR                  ; Address pointer code
        movwf   I2C_REG
        movlw   STAT                    ; Status register
        movwf   I2C_DAT
        cli
        call    I2C_Write               ; Write to address pointer
        call    I2C_Read
        sei
        btfss   I2C_DAT,1
         goto   WaitA

        movlw   REALH
        movwf   I2C_DAT
        call    I2C_Write               ; Change address pointer
        movlw   low ZDATA_A
        movwf   FSR
        movlw   .4

        cli
        call    I2C_BlockRead
        sei

        bcf     bitEN_B                 ; enable I2C for module B
        bsf     bitEN_A                 ; disable I2C for module A

WaitB:
        movlw   ADRPTR                  ; Address pointer code
        movwf   I2C_REG
        movlw   STAT                    ; Status register
        movwf   I2C_DAT
        cli
        call    I2C_Write               ; Write to address pointer
        call    I2C_Read
        sei
        btfss   I2C_DAT,1
         goto   WaitB
        movlw    REALH
        movwf    I2C_DAT
        call     I2C_Write                ; Change address pointer
        movlw    low ZDATA_B
        movwf    FSR
        movlw    .4

        cli
        call     I2C_BlockRead
        sei

        bcf      bitEN_A                  ; enable I2C for module A

        return


;;; *************************************************************
;;; Subroutine Name:    Measure
;;; Description:        Performs an impedance measurement
;;; Requires:
;;; Returns:            NULL
;;; Locations Affected: W is destroyed, modifies ZDATA_A, ZDATA_B,
;;;                     uses I2C_REG, I2C_DAT
;;; *************************************************************
Measure:
                                          ; Reset to make sure both
                                          ; AD5933's are sync'd
        movlw    CTRLL                    ; Low control register.
        movwf    I2C_REG
        movlw    0x18
        movwf    I2C_DAT                  ; Reset.
        call     I2C_Write
        movlw    CTRLH                    ; High control register.
        movwf    I2C_REG
        movlw    0x17                     ; Initialize with start
        movwf    I2C_DAT                  ; frequency.
        call     I2C_Write

        movlw    .50                      ; Delay 100 ms to let
        call     DELAY_Wx10ms             ; things settle.

        movlw    0x27                     ; Start frequency sweep.
        movwf    I2C_DAT
        call     I2C_Write

        movlw    .1
        call     DELAY_Wx10ms             ; Delay 10 ms

        call     ReadImpedance
        call     SetRange                 ; Set range resistor.


                                          ; Reset to make sure both
                                          ; AD5933's are sync'd
```

```
movlw   CTRLL                   ; Low control register.
movwf   I2C_REG
movlw   0x18
movwf   I2C_DAT                 ; Reset.
call    I2C_Write
movlw   CTRLH                   ; High control register.
movwf   I2C_REG
movlw   0x17                    ; Initialize with start
movwf   I2C_DAT                 ; frequency.
call    I2C_Write

movlw   .100                    ; Delay 2s to let things
call    DELAY_Wx10ms            ; settle.

movlw   0x27                    ; Start frequency sweep.
movwf   I2C_DAT
call    I2C_Write

movlw   .1
call    DELAY_Wx10ms            ; Delay 10 ms.

call    ReadImpedance

movlw   CTRLL                   ; Low control register.
movwf   I2C_REG
movlw   0x18                    ; Reset.
movwf   I2C_DAT
call    I2C_Write

movlw   0xC7                    ; Mask for range resistor
andwf   PORTA,W                 ; bits on PORTA.
movwf   PORTA                   ; Set range resistor to 0.

movf    ZDATA_A,W
call    PrintHex
movf    ZDATA_A + 1,W
call    PrintHex
movlw   ”,”
call    DBUS_TxByte
movf    ZDATA_A + 2,W
call    PrintHex
movf    ZDATA_A + 3,W
call    PrintHex
movlw   ”,”
call    DBUS_TxByte
movf    ZDATA_B,W
call    PrintHex
movf    ZDATA_B + 1,W
call    PrintHex
movlw   ”,”
call    DBUS_TxByte
movf    ZDATA_B + 2,W
call    PrintHex
movf    ZDATA_B + 3,W
```

```
        call    PrintHex
        movlw   ”,”
        call    DBUS_TxByte

        rrf     RRANGE,F
        rrf     RRANGE,F
        rrf     RRANGE,F
        movlw   0x30                ; Convert range to ASCII
        addwf   RRANGE,W
        call    DBUS_TxByte         ; Print ASCII char
        movlw   ”,”
        call    DBUS_TxByte

        return

;;; *********************************************************
;;; Subroutine Name:    FSweep
;;; Description:        Sweeps through a series of frequencies on
;;;                     the AD5933, calling Measure to perform an
;;;                     impedance measurement at each.
;;; Requires:           – Starting frequency register value in
;;;                             FREQVAL.
;;;                     – Frequency value increment in FREQINC
;;;                     –Number of frequencies to sweep in FREQCNT
;;; Returns:            NULL
;;; Locations Affected: W is destroyed, uses I2C_REG, I2C_DAT
;;; *********************************************************
FSweep:
        movlw   SFREQ2              ; Start freq register 2
        movwf   I2C_REG
        movf    FREQVAL,W
        movwf   I2C_DAT
        cli                         ; Change start frequency
        call    I2C_Write           ; on AD5933.
        sei

        call    Measure

        movf    FREQINC,W           ; Increment FREQVAL
        addwf   FREQVAL,F           ; by FREQVAL.
        decfsz  FREQCNT,F
         goto   FSweep

        return


;;; ————————————————————————————————————————————————————————
;;; Interrupt Service Routines
;;; ————————————————————————————————————————————————————————


;;; *********************************************************
;;; Subroutine Name:    IntISR
;;; Description:        General interrupt ISR. The interrupt
;;;                     vector jumps to this subroutine. It polls
;;;                    the interrupt flag bits and calls the
;;;                    appropriate subroutine.
;;; Requires:                 Nothing
;;; Returns:                  RETFIE - restores program counter
;;; Locations Affected: W_TEMP, STATUS_TEMP
;;; ************************************************************
IntISR:
        PUSH_MACRO                          ; Saves W and STATUS
                                            ; registers.
        btfsc   PIR1, TMR2IF
                call    Timer2ISR           ; Service Timer2.
        btfsc   PIR2, CCP2IF
                call    CompareISR          ; Service CCP2 compare
                                            ; match on TMR1.
        POP_MACRO                           ; Restores W and STATUS
                                            ; registers.
        retfie

;;; ************************************************************
;;; Subroutine Name:   Timer2ISR
;;; Description:       Handles TMR2 match with PR2.
;;;                    PIN_MCLK is toggled on PORTC to generate
;;;                    a squarewave output.
;;; Requires:
;;; Returns:           NULL
;;; Locations Affected: PORTC, PIN_MCLK, clears TMR2IF
;;; ************************************************************
Timer2ISR:
        banksel PORTC
        movlw   (1<<PIN_MCLK)               ; Toggle PIN_MCLK
        xorwf   PORTC,F                     ; (flip bit).
        bcf     PIR1, TMR2IF                ; Clear TMR2IF interrupt
                                            ; flag.
        return


;;; ************************************************************
;;; Subroutine Name:   CompareISR
;;; Description:       Handles CCP compare match on TMR1.
;;;                    [TIMEHI:TIMELO] is decremented. F_TIMER
;;;                    bit in FLAGS register is asserted if zero.
;;; Requires:
;;; Returns:           NULL
;;; Locations Affected: TIME_HI, TIME_LO, FLAGS, clears CCP2IF
;;; ************************************************************
CompareISR:
        bcf     STATUS, RP0                 ; Select Bank 0.
        movf    TIME_LO,F                   ; Test if zero.
        skpnz
         decf   TIME_HI,F
        decfsz  TIME_LO,F
         goto   ClearInt
        movf    TIME_HI,F
        skpnz
```

```
         bsf     FLAGS,F_TIMER               ; Assert flag if zero
ClearInt:
        bcf      PIR2, CCP2IF                ; clear CCP2IF interrupt
                                             ; flag
        return

;;; --------------------------------------------------------------------
;;; Code-section Constant Data.
;;;             This data will appear in the PIC's code space.
;;; --------------------------------------------------------------------
Sline1: dt       "\fZ-Analyzer␣v0.1\r\n",0
Sline2: dt       "Found␣Channel␣",0
Sline3: dt       "Channel␣",0
Sline4: dt       "\r\nPress␣any␣key␣to␣start\r\n",0


Init5933:                                    ; Table of initialization
                                             ; values for AD5933
                                             ; registers.
        dt       .12
        dt       0xB3                        ; CTRLH: Standby mode,
                                             ; 1.0 Vp-p.
        dt       0x18                        ; CTRLL: use external
                                             ; clock, reset.
        dt       0x00,0x00,0x00              ; Start frequency.
        dt       0x00,0x00,0x00              ; Frequency increment.
        dt       0x00,0x00                   ; Number of increments.
        dt       0x00,0x00                   ; Number of settling
                                             ; cycles.

;;; --------------------------------------------------------------------
;;; Vectors Section.
;;; --------------------------------------------------------------------
Reset:  CODE     0x000                       ; Establish the reset
                                             ; vector.
        pagesel Main                         ; Selects proper page in
                                             ; memory for main program.
        goto     Main                        ; Goes to the actual
                                             ; program.

IntServeRoutine:         CODE       0x004    ; Establish the ISR
        goto     IntISR                      ; Jump to the general
                                             ; interrupt ISR.


        END                                  ; End of file.
```