ZnO Nanostructure-Based Biosensors with Mobile Device Interface

Student Team
Ryan Dallago
Pavel Litorovich
Ahmed Shehata
Tanzina Farzana
Lakeram Kissoon

Advising Faculty
Dr. Yicheng Lu and
Dr. Pavel Ivanoff Reyes

Spring 2014
1. Abstract

**Background:** Biosensors are an important topic of major scientific interest. They possess a wide variety of potential applications that are currently being used to identify and measure herbicides, bacterial contamination in foods, and blood glucose levels in diabetics to name a few. Along with its improvements, it is now possible to screen more than 75,000 different DNA sequences on a single DNA sensor. Biosensors are also used in other fields of research including identifying disease related mutations in human genes. Improvements in sensitivity and simplicity are important aims in this field of research. Zinc Oxide (ZnO) based biosensors developed by Dr. Reyes and Dr. Lu at Rutgers University have extended these capabilities with their ability to be grown on different types of substrates and ability to operate in semiconducting, piezoelectric, transparent and conducting, or ferromagnetic modes.

**Purpose:** The goal of our project is to develop a biosensor interface that can be easily accessed with a mobile device. An improvement of this sort extends the usefulness of ZnO sensors by enabling researchers to have quicker more convenient access to test data even when on the go.

**Method:** In order to get accurate test data, our mobile interface collects data from a network analyzer via the General Purpose Interface Bus (GPIB) and transmits the result to a mobile device over Bluetooth connection which runs our custom built user-friendly android application. We used a ZnO sensor to measure the mass of a bio-protein Biotin based on the shift in frequency on the sensor. This mobile interface first measures the resonant frequency of the sensor with no protein present to acquire a baseline. A protein is then added to the sensor which results in a shift in frequency. With a simple push of a button on the mobile app the interface measures the frequency again, processes that data, and displays the weight of the added protein.

**Result:** Our interface was successfully built and we were able to demonstrate the process with a mobile application. The correct mass of a bio-protein was identified and displayed using the procedure described above. While the interface works and displays the correct mass of the biomolecule, the response time is fairly slow due to the bottleneck effect of a slow network analyzer used to observe data.

**Conclusion:** The ZnO Nanostructure-based biosensor with a mobile phone interface is extremely helpful for the research community. It demonstrates the essence of engineering by simplifying an otherwise tedious and time consuming process with it's easy to use mobile app used to control the biosensor made from reasonably low cost parts. This project also contributes to better health facilities as it easily and accurately increases the speed of disease diagnosis for more scrupulous treatment.
2. Introduction and Definition of the Problem

A. Biosensors Technology

Biosensors research is emerging as a major focus in both academic and commercial sectors. According to global market analysts, the biosensor market is projected to reach $12 billion by 2015. The emergence of new technologies in materials design, device fabrication, microfluidics, and nanotechnology, and the maturation of biochemical analytics helped propel this market viability. The key areas of growth in biosensors research are in non-invasive sensor platforms, device design with increasingly high accuracy, sensitivity, selectivity, and compactness, real-time point-of-care testing, and development of high throughput, multi-species biochemical protocols. The applications for biosensors have widened in recent years to include drug discovery and development, clinical diagnostic tools, cancer and genetic research, agricultural and veterinary analytical tools, pollution and contamination monitoring, and food processing.

B. ZnO and its Nanostructures as a Multifunctional Sensor Material

ZnO is emerging as a wide bandgap semiconductor. A unique advantage of ZnO is its multifunctional properties. Through proper doping, ZnO can be made semiconducting, piezoelectric, transparent and conducting, or ferromagnetic, leading to versatile applications.

ZnO is attractive as a sensor material. High sensitivity to CH₄, CO, H₂O, H₂, NH₃, trimethylamine, ethanol, and NO₂ has been demonstrated for ZnO. ZnO and its ternary alloy, MgₓZn₁₋ₓO, are biocompatible oxides, in which Zn and Mg are important elements for neurotransmitter production and enzyme function. Furthermore, ZnO nanostructures can be grown on various substrates at relatively low temperature, leading to the promising application: ZnO-based sensors on glass and on flexible substrates. It is shown that the ZnO nanorods are compatible with intracellular material and highly sensitive to pH changes inside cellular environments. The nanorods are also able to detect the enzymatic reactions with target biochemicals. ZnO is promising for sensing proteins, antibodies, and DNA through specifically designed surface functionalization of ZnO-based devices.
C. Specific Goals of the Proposed Work

Personal health care tools rely heavily on the performance of its biosensing devices aimed at medical diagnostics and environmental monitoring. There has been an increasing requirement for high throughput biosensors arrays with high selectivity and sensitivity, and operating in the wireless mode. The development of such devices lies in the design of the active component such as the transistor-based sensors.

The aim of this is to extend mobile device user interface and functionality in the ZnO-based nanostructured biosensor technology through the development of the GPIB-Bluetooth data interface and mobile app software development for the manipulation of multi-modal sensor signals (e.g., acoustic, electrical, and optical). The sensors were developed through: (i) accurate and high quality growth of multifunctional ZnO thin and single crystal ZnO nanostructure arrays, (ii) organic and biomolecular functionalization, and (iii) development of sensor platforms and devices for applications in biochemical detection and dynamic and noninvasive techniques of monitoring biophysical activity.

D. Previous work in wireless biosensor interfaces

The following are some applications for mobile biosensor interfaces that have previously been developed.

**Wireless Hybrid Bio-Sensing with Mobile based Monitoring System**
A Low energy Bluetooth link is used to transmit information from wireless temperature sensors and EEG sensors to a cell phone.
*Advantages:* Small size, integrates analog amps and BT transceiver
*Limitations:* prototype power is higher than desired. Limited by MCU & LEDs.

**Query Processor in cell phone**
An app is written on a cell phone which communicates with wearable sensors. Data is collected from the sensors and displayed on the phone. The data is also sent to a central health server where doctors can view at their convenience.
*Advantages:* the patient can monitor their own data with their cell phone. Transmitting of data to a central server, virtually unlimited storage.
*Limitations:* Old cell phone design which needs external dongle for wireless communication.

**Wireless Lab-in-a-Pill Biosensor**
A biosensor which, when swallowed detects gastrointestinal bleeding and then transmits the results to a zigbee enabled device.
3. Approach

We divided our project into several stages that are outlined below.

a. Hardware Development

GPIB interface is relatively old interface that is still widely used. The challenge we faced was to find a solution that would allow MCU enabled board to control the instrument over GPIB connections. We did a thorough research on how GPIB protocol works and decided that the best choice for us will be to use Arduino board since it allows great flexibility in implementing the task.

Another advantage of using Arduino board is that it allowed us to cut several hardware tasks by implementing them directly in software. In particular GPIB interface uses “active low” method (zero output means the signal is asserted), hence numerous pull-up resistors needed to be implemented, but with Arduino board it allowed us to avoid using the bulk of resistors and implement them directly in software.

The next page shows how we connected Arduino’s board input/outputs with the pins of GPIB connector.
**GPIB to Arduino connection:**

<table>
<thead>
<tr>
<th>GPIB Connector</th>
<th>Arduino Connector--XIO</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPIB Pin</strong></td>
<td><strong>Signal</strong></td>
</tr>
<tr>
<td>1</td>
<td>D1</td>
</tr>
<tr>
<td>2</td>
<td>D2</td>
</tr>
<tr>
<td>3</td>
<td>D3</td>
</tr>
<tr>
<td>4</td>
<td>D4</td>
</tr>
<tr>
<td>5</td>
<td>EOI</td>
</tr>
<tr>
<td>6</td>
<td>DAV</td>
</tr>
<tr>
<td>7</td>
<td>NRFD</td>
</tr>
<tr>
<td>8</td>
<td>NDAC</td>
</tr>
<tr>
<td>9</td>
<td>IFC</td>
</tr>
<tr>
<td>10</td>
<td>SRQ</td>
</tr>
<tr>
<td>11</td>
<td>ATN</td>
</tr>
<tr>
<td>12</td>
<td>gnd</td>
</tr>
<tr>
<td>13</td>
<td>D5</td>
</tr>
<tr>
<td>14</td>
<td>D6</td>
</tr>
<tr>
<td>15</td>
<td>D7</td>
</tr>
<tr>
<td>16</td>
<td>D8</td>
</tr>
<tr>
<td>17</td>
<td>REN</td>
</tr>
<tr>
<td>18</td>
<td>gnd</td>
</tr>
</tbody>
</table>

**Bluetooth connection:**

<table>
<thead>
<tr>
<th>Bluetooth Module</th>
<th>Arduino Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPIB Pin</strong></td>
<td><strong>Signal</strong></td>
</tr>
<tr>
<td>RXD</td>
<td>receive data</td>
</tr>
<tr>
<td>TXD</td>
<td>transmit data</td>
</tr>
<tr>
<td>VCC</td>
<td>3.3V</td>
</tr>
<tr>
<td>GND</td>
<td>Ground</td>
</tr>
</tbody>
</table>
b. **Software Development**

The Software portion of the project is naturally divided into two parts: The MCU software development which is written in C, and the mobile application which is written in Java.

**MCU software operation**

The MCU software has two roles. First is to interface to the mobile application in order to receive commands and to send test data back to the application. The second function is to interface with the network analyzer in order to acquire measurement data.

The interface to the mobile application is implemented as a command processor. After some initial setup, the MCU will wait until it receives data over the Bluetooth interface, when data is received, it is interpreted and then action is taken depending on which command was received. Several commands were developed, both for the main application and for development purposes. Three types of commands are recognized by the interface: local commands, bio-sensor commands, and GPIB pass-through commands. A complete list of implemented commands and their responses is shown below.

**Local Commands**

The following commands were developed to enable/disable certain function on the MCU as well as aid in mobile application development and debug.

**GPIB Enable**

Puts the instrument into remote mode & initialize the GPIB IO signals.
The following command needs to be sent before the device will pass GPIB commands to an instrument.
Command: `<#gpibon>`
Response: `<#gpibon-done>`

**GPIB Disable**

Releases the instrument from remote mode. After this command is sent, the device will not pass GPIB commands to an instrument.
Command: `<#gpiboff>`
Response: `<#gpiboff-done>`

**Blinking LED:**

Start or stop the onboard LED from blinking.
Command: `<#tb>`
Response: `<#tb-blinkon>` or `<#tb-blinkoff>`
**Return Data Simulation**

Used for mobile application development when the interface is not connected to a network analyzer.

Return an integer: `<#retint>`, Response: `<1234567890>`
Return a floating point number: `<#retfloat>`, Response `<12345.09876`
Return a number in scientific notation: `<#retsci>`, Response `<12345.e-010>`

**Biosensor Commands**

These commands are used as macro functions to send and receive data from the biosensor platform.

**Setup the Interface**

Enables GPIB and then sends a series of GPIB commands to configure the network analyzer. Returns the current peak frequency and magnitude.

Command: `<#setup>`
Response: `<real magnitude, imaginary magnitude, frequency>`

**Get current peak value**

Sends a series of commands and returns the peak frequency and magnitude.

Command: `<#getpeak>`
Response: `<real magnitude, imaginary magnitude, frequency>`

**GPIB Commands**

After the `#gpibon` command is sent, any gpib command can be sent to the interface without the leading `'#'` character. The interface will return any data that the instrument returns.

A flow chart of the MCU software is shown below.
Upon power up, the software enters the Setup block which defines all of the IO pins as input or output and sets them to their initial values. It also initializes the serial port, which is the primary interface to the Bluetooth module. Next the string “Interface Ready” is sent over the Bluetooth link to indicate that the interface is available to accept commands. In reality, the application will only see this string if the interface is reset while connected to a mobile device, i.e. the #mcurst command is sent to the interface. The main program loop begins next: The program will wait until data is received from the mobile application. When data is received it will put the command(s) into a first in first out (FIFO) buffer. The commands are executed and responses sent according to their definitions given above. When the FIFO is empty the program will wait for additional commands to be received from the mobile application and the process begins again.
The Interface also implements the GPIB protocol to efficiently communicate with the network analyzer. As shown above in the hardware description, the GPIB interface contains eight data signals and three handshaking signals. In order to send and receive data to the network analyzer the handshaking protocol shown below must be followed.

When the GPIB interface is enabled using the #gpibon command, the interface sets itself as a “talker” and the instrument as a “listener” on the GPIB. If a command is to be sent to the instrument, the MCU puts the data on the data bus and waits for the NRFD (not ready for data) signal to go high. Once this happens, the MCU sets the DAV (Data valid) signal low to tell the “listener” that the data on the bus is valid. When the listener successfully reads the data, it sets the NDAC (Data not accepted) signal high. The process begins again with the next byte of data. If the command that was sent prompts the instrument to respond, the MCU will then set the instrument as the talker and itself as a listener and the above process is executed.
Mobile Application operation

Flow chat of the Mobile application development

The main means of communication for the mobile app to the MCU hardware is through serial Bluetooth connection. To initialize the Bluetooth we first need to get Bluetooth permission from the phone itself. Once permission has been granted the app checks if Bluetooth is turned on.

In the event that Bluetooth is not turned on, the app displays a “toast” to the user to enable it. If Bluetooth is turned on, the app then list all devices that the phone is currently paired with. The user would now select the name of the Bluetooth module that is connected to the MCU. Once a connection is established, app switches to the main activity (Bio-sensor Interface). This activity displays 3 means of communication to the user; the Calibrate button, the Get Mass button, and a text box.

To begin operation of the application, the user first taps on the calibrate button. This sends the #setup command to the MCU which processes the command to: set the scale to display the trace on the network analyzer, move the cursor to the peak of the trace, then read and return the resonance frequency of the sensor. As the reference frequency is shown on the display, the user can now apply the sample to the biosensor, which will cause the resonate frequency of the sensor to shift. The user can now tap the Get Mass button. This button sends the #getpeak command to the MCU via the Bluetooth terminal. The MCU then processes this command to get the peak of the shifted frequency and returns this value to the app. The app then subtracts the shifted frequency from the reference frequency to obtain the delta F. Using delta F, the mass of the added protein can be determined.
c. Result

In our project we used a ZnO Bio-sensor to measure the mass of a bio-protein (Biotin) based on the shift in frequency on the sensor. The mobile interface initially measures the resonant (Peak of the frequency) frequency of the sensor with no protein present in order to acquire a baseline. Then a protein solution is added to biosensor, the resonant frequency of the sensor shifts proportionally to the amount of protein that is added. Subtracting the new frequency from the baseline frequency yields delta f.

\[ \Delta m = \frac{\Delta f V_q \rho_q A}{2(f_0^2)} \]

- \( f_0 = 10 \text{ MHz} \)
- \( V_q = 3.336 \times 10^3 \text{ m/sec} \)
- \( \rho_q = 2.648 \times 10^3 \text{ kg/m}^3 \)
- \( A = 0.2047 \text{ cm}^2 \)

We used above formula to calculate mass of the protein. The mobile application will interpret this frequency shift and display the mass of the protein that was added.
The mobile application runs on android based device. With a press of button on this customize mobile application received data from network analyzer thorough GPIB command, measured the frequency, processes that data and display the weight of the added protein.

The picture on the left shows calibrated data (9.954 MHz) from resonant frequency of the sensor, when no protein added. The picture on the right shows the mass of the protein from shifted frequency (84.76 mg), when protein sample were added to the sensor.
4. Team member tasks:

Pavel Litorovich
- Researched GPIB protocol functionality
- Implemented MCU to GPIB and Bluetooth interface connections
- Researched and purchased necessary parts and accessories
- Assembled and tested the circuit
- Created finished product by putting the board into electronics enclosure box
- Assisted team members with other tasks

Ryan Dallago
- Researched network analyzer remote commands.
- Selected and purchased MCU development board.
- Built initial hardware prototype.
- MCU software
  - Implemented GPIB handshaking protocol in MCU software.
  - Designed and implemented command processor on MCU platform
- Coordinated/summarized team meetings and point of contact to advisors

Ahmed Shehata
- Created necessary schematic and diagrams.
- Researched Biosensor fabrication process “Solid state electronics“ as an interest and for the documentation
- Researched  Arduino Mega 2560 and GPIB specifications
- Tested hardware effectiveness in simulation software
- Worked closely with other team members
Tanzina Farzana

- Researched and gathered information about previous work in wireless biosensor interfaces
- Documented the reports and presentation slides with other team members
- Researched android programming and mobile application
- Assisted in mobile app development

Lakeram Kissoon

- Developed mobile application
- Troubleshot connection between Mobile app, MCU and network analyzer
- Researched and reverse engineered android programming.
Cost Analysis

From the beginning the goal for us was to keep the expenses at minimum. We budgeted our expenses under $200 that was the average cost of previous capstone projects.

The following table shows the equipment and materials used in the project the actual expenses we incurred:

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Vendor</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Module</td>
<td>Device under test</td>
<td>Available in EE lab</td>
<td>$0.00</td>
</tr>
<tr>
<td>Probe fixture</td>
<td>Holds the sensor module</td>
<td>Available in EE lab</td>
<td>$0.00</td>
</tr>
<tr>
<td>Network analyzer</td>
<td>Interface to sensor module</td>
<td>Available in EE lab</td>
<td>$0.00</td>
</tr>
<tr>
<td>GPIB Cable</td>
<td>cable from device to network analyzer</td>
<td>Available in EE lab</td>
<td>$0.00</td>
</tr>
<tr>
<td>MCU board</td>
<td>Arduino Mega2560 Revision 3</td>
<td>Jameco Electronics</td>
<td>$54.95</td>
</tr>
<tr>
<td>Bluetooth Module</td>
<td>Wireless interface to mobile device</td>
<td>Amazon</td>
<td>$10.00</td>
</tr>
<tr>
<td>Enclosure</td>
<td>Electronics Box</td>
<td>Ebay</td>
<td>$10.40</td>
</tr>
<tr>
<td>GPIB connector</td>
<td>24 pin Centronics connector</td>
<td>Digikey</td>
<td>$9.34</td>
</tr>
<tr>
<td>Misc.Hardware</td>
<td>proto boards, wire, resistors etc.</td>
<td>Radio Shack</td>
<td>$27.06</td>
</tr>
<tr>
<td><strong>Total Cost</strong></td>
<td></td>
<td></td>
<td><strong>$111.75</strong></td>
</tr>
</tbody>
</table>

As can be seen we were able to stay below the budget that makes our project very feasible cost wise. The most expensive part of the project was Arduino board, but that expense can also be dramatically minimized if we decide to build our own MCU board. The average cost of a microcontroller for a task like ours is about $10. But it all depends on particular application. Our design is very versatile and can be used in many applications. For example, since the command processor can pass GPIB commands from a mobile device to an instrument, any GPIB enabled instrument can be controlled with this interface assuming the mobile application is modified. Upon our research we found out that there are not many similar implementations exists on the market and they also cost a lot of money. For example GPIB to USB converters sell for over $300 and they do not have wireless capability.
5. Conclusion

The project appeared to be quite interesting and challenging at the same time. It required a lot of research to be done since none of the members in our group had prior experience in programming microcontrollers and controlling GPIB enabled instruments. There were many challenges we had to face. Among several of them are:

- Develop a working connection between microcontroller and GPIB connector.
- Implement a working GPIB hand-shaking protocol.
- Make the system stable without the need for hard reset.
- Develop a working mobile application.

The main goal for us was to not just obtain readings from a ZnO sensor, but to create a fully working universal interface that would allow two-way communications. Our design allows the user to not only see the data on the mobile phone but also request the data that needs to be measured.

The advantages of our project among others:

- Simple user interface.
- Outputs meaningful sensor readings.
- Frees PC resources for other data processing rather than instrument control.
- Researchers who may not be familiar with test equipment will now be able to collect sensor data.

We also understand that our project can benefit from further development. In particular a good application would be to build a device that will eliminate the need for a network analyzer. In that case the sensor can be directly implemented with our interface that will allow great flexibility and compactness. It can even be brought to a nano-scale which can find its usage in bio-medical industry as well as in many others industries.

Another improvement would be to expand the capabilities of the mobile application to allow additional processing of data and presenting various results on the fly. The mobile phone could be turned into a versatile instrument with great scientific capabilities.
6. References


Wireless Hybrid Bio-Sensing with Mobile based Monitoring System
Linlin Xu, School of Information and Communication Technology, 2013.

Query Processor in cell phone
“Development of Efficient Query Processor in Cell Phone for Sensor Data Processing Over a WSN” Sing-Hui Toh, Hoon-Jae Lee, Kyeong-Hoon Do Graduate School of Design & IT, Dongseo University, Busan 617-716, KoreaQuery “Processor in cell Phone.pdf”

Wireless Lab-in-a-Pill Biosensor