Microsensors, Implantable Devices and Rodent Surgeries for Biomedical Applications

TA: Kaushik Lakshmiramanan, Course Instructor: Dr. Hardik J. Pandya
Department of Electronic Systems Engineering
Indian Institute of Science, Bangalore

Week - 10
Lecture – 44

Greetings, and welcome to the NPTEL course on Microsensors, Implantable Devices, and Rodent Surgeries for Biomedical Applications. I'm Kaushik, and I will be guiding you through the intricate process of designing and developing an electronic system specifically tailored for the stimulation and acquisition of brain signals, focusing on rodent-based models. This is the second installment in a series of three lectures. In this particular lecture, we will delve into the design of a comprehensive circuit capable of fulfilling our established objectives. For those who haven't had the chance to view the first lecture, I will provide a concise recap of its key points. In the first lecture, we explored the crucial design considerations involved in developing our system. We began by defining the system's primary goal, which is to both stimulate and acquire biopotentials from rodents. This fundamental objective necessitates that our system maintains an exceptionally small footprint, be wirelessly operable, and function on battery power. These considerations played a pivotal role in shaping our design choices.

To achieve these objectives, we meticulously examined various possibilities, ranging from utilizing development boards to employing integrated circuits (ICs) or even application-specific integrated circuits (ASICs) to construct our custom board. Ultimately, we settled on the specialized ASIC RHS2116, which possesses the remarkable capability to both stimulate and acquire biopotentials across any of its 16 channels. In this lecture, we will shift our focus to the circuit design and explore how to effectively interface with the RHS2116. It's crucial to recognize that this chip alone cannot accomplish the entire process. We require additional controllers to acquire the signals and translate them into meaningful information for the user. We will meticulously examine how to integrate these essential components.

I will commence by outlining the design process, including the creation of a system prototype and the seamless interfacing of various components. As I mentioned in the previous lecture, the RHS2116 incorporates its own built-in modules that empower it to acquire the subtle electrical potentials generated by the brain. These biopotentials typically fall within the range of a few hundred microvolts to millivolts. While numerous analog-to-digital converters (ADCs) are readily available in the market, capable of acquiring analog signals, we require preamplifier stages to amplify these faint signals, rendering them discernible to the machine during the digital conversion process.

These amplification stages can be implemented either internally or externally. However, external stages would inevitably increase the overall size of our system. The beauty of the RHS2116 ASIC lies in its internal integration of these stages, making it an exceptionally compact solution. To underscore its remarkably small size, you can observe its juxtaposition with the US currency in the provided image. Let's now proceed to the heart of this lecture and explore the intricacies of circuit design, component interfacing, and prototype development.

This aspect is crucial, especially when dealing with rodent-based models. Allow me to elaborate on the overall structure so that you can gain a clearer understanding of how these biopotentials are acquired. As you observe in the diagram, there are 16 electrodes, numbered 0 to 15, which establish connections with the sensors implanted in the brain. These sensors capture the brain's electrical signals, which are then channeled through AC-coupled high-gain amplifiers integrated into the system. These high-gain amplifiers are specifically designed to amplify the typically weak biopotentials. Notably, each electrode is equipped with both an AC-coupled high-gain amplifier and a DC-coupled low-gain amplifier, along with a current stimulator selector. This selector determines whether a particular electrode will function in stimulation mode or acquisition mode at any given time.

In acquisition mode, no current flows through the electrode, allowing the biopotentials to be captured by the AC high-gain amplifier. These acquired signals are referenced to a dedicated reference electrode to eliminate common-mode noise, a prevalent issue in the microvolt to millivolt range of biopotentials. Conversely, when an electrode is switched to constant current stimulation mode, capable of delivering biphasic stimulation, the DC low-gain amplifier, referenced to the common ground, measures the electrode potential resulting from the stimulation. Simultaneously, the AC high-gain amplifier continues to capture electrode potentials arising from local field potentials and action potentials. Maintaining charge neutrality is paramount in any stimulation setup. To achieve this, a charge recovery chip switch is employed to effectively ground any excess charge accumulating on the electrode. This entire process is replicated across all 16 electrodes concurrently through the utilization of an analog multiplexer, which rapidly switches between electrodes, creating the illusion of parallel operation. The constant current supplied for stimulation typically ranges from nanoamperes to microamperes, and the stimulation voltage is determined by the bias provided at Vstimplus and Vstimminus. These are the integral stages embedded within the RSS 2116. Our next step is to interface this chip with other components to realize a fully functional system.

As depicted in the schematic provided in the datasheet, the RSS 2116 can be readily connected to a CPU, microcontroller, or FPGA via SPI lines. CPUs, also referred to as microprocessors when operating independently, present a viable option for interfacing. Let's now embark on a step-by-step journey to understand how we can control the RSS 2116 chip based on the information we have gathered thus far. To begin, I propose that we opt for microcontrollers. The rationale behind this choice is that microcontrollers are generally cost-effective, readily available in the market, and offer a reasonably compact form factor when used as standalone chips rather than as part of development boards. With this foundation established, we will progressively build upon our understanding and explore the intricacies of interfacing and controlling the RSS 2116 to achieve our desired system functionality.

I am opting to connect the microcontroller to the RSS 2116 via SPI. Now, as you can see, the schematic provided is quite rudimentary and necessitates extensive back-end development, including programming in either embedded C or CircuitPython. However, my primary intention here is to provide viewers with a general overview of the design process. I encourage you all to delve deeper into the workings of a development board and how to program it, as the principles can be extrapolated to this integrated circuit as well. SPI, which stands for Serial Peripheral Interface, is a communication protocol commonly employed in embedded systems. It typically involves four lines: two data lines, one clock line, and one select line. The two data lines facilitate bidirectional communication between the microcontroller and the peripheral device, which, in our case, is the RSS 2116. The clock line synchronizes the data transfer between the two devices, ensuring they operate in lockstep. The select line, also known as the chip select, is used when multiple devices are connected to the same SPI bus. It allows the microcontroller to address a specific peripheral device. In our scenario, with only one peripheral connected, we'll simply use the default address for the select line.

The data lines are typically designated as master in slave out (MISO) and master out slave in (MOSI). These lines are connected in a crossover fashion, meaning the MISO line of the microcontroller is connected to the MOSI line of the peripheral, and vice versa. The clock lines are directly connected, as are the select lines. Upon connecting these, you'll notice that the chip has multiple lines for the same terms. This is because the RSS 2116 supports low-voltage differential signaling (LVDS) SPI. LVDS is a technique used for high-speed, low-power data transmission. It employs differential signaling, where each line has a positive and a negative counterpart that operate simultaneously. This approach allows for data transmission at high rates while consuming minimal power. Specialized comparators are used at the receiving end to interpret the differential signals.

Consequently, each line has its corresponding counterpart, resulting in a total of eight lines on the chip that need to be connected to the eight lines on the peripheral device. In this scenario, we'll utilize differential signaling for enhanced speed and performance. We'll establish the connection using LVDS SPI. However, it's important to be aware that not all microcontrollers support LVDS SPI. Furthermore, the datasheet indicates that the RSS 2116 is capable of transmitting 40,000 samples per second. When distributed across its 16 channels, this translates to a substantial sampling rate per channel. Considering a 16-bit ADC, the overall speed requirement amounts to a considerable 10.24 megabits per second. Microcontrollers and microprocessors typically have an upper limit of 2 megabits per second, even when overclocked, they might reach 6 to 10 megabits per second, but this is uncommon and often involves expensive microprocessors.

This is where FPGAs, or field-programmable gate arrays, come into play. FPGAs are reprogrammable even after manufacturing and comprise an array of logic blocks that can be configured to execute digital functions. Their unique strength lies in their flexibility and ability to handle high data sampling rates, making them indispensable for brain signal acquisition. Neuro potentials and other local field potentials and EEG signals span a wide frequency range, from 1 to 10 kilohertz. To capture all these signals effectively, a high sampling rate is imperative. In light of this, I will be replacing the microcontroller with an FPGA to accommodate the demanding data rate requirements of our brain signal acquisition system.

The reason I initially opted for a microcontroller or microprocessor is that most individuals are likely more familiar and comfortable working with them. Additionally, they tend to be readily available and relatively inexpensive. However, when developing a system intended for neurosurgical applications, where data rates are critical, we cannot afford to compromise on speed. Consequently, we must opt for an FPGA and employ LVDS SPI for communication and interconnection with the RHS 2116.

Now that we have settled on using an FPGA, there's an additional aspect to consider. FPGAs are typically programmed using Verilog code, and controlling the stimulation or acquisition electrodes, along with providing stimulation parameters and acquiring signals in real time, can be somewhat challenging with just the FPGA alone. To simplify this process, we can interface the FPGA with a microcontroller. This microcontroller will enable real-time programmability by facilitating the translation of signals and communication with the FPGA. This brings us to the circuit configuration we'll be working with. As you can see, there's a register file responsible for controlling amplification and stimulation. This register file will, in turn, be controlled by the SPI interface digital controller.

With the circuit structure established, we'll utilize a microcontroller to communicate with the FPGA using SPI or differential SPI. The FPGA will then interact with the RHS 2116. Since we're aiming for a battery-powered and wireless system, we need a means to communicate with the graphical user interface (GUI). To power the device, we'll incorporate a battery module that will also supply power to the FPGA and the brain stimulation process. It's worth noting that this is a simplified representation, and in reality, we would require various regulators, protection circuits, and other components. However, I'm focusing on providing a general overview.

To establish communication with the GUI, we can utilize either Wi-Fi or Bluetooth Low Energy (BLE) modules. Many BLE antennas are available, and we could select one with suitable specifications. Alternatively, we could opt for readily available BLE modules that boast a compact form factor, ensuring our overall device remains reasonably sized. Let's now consolidate our understanding and construct a preliminary schematic based on the components we've discussed so far. We have a microcontroller that will communicate with the FPGA via SPI. SPI is a preferred choice for communication between the microcontroller and FPGA due to its speed and prevalence in FPGA interfacing. Other communication protocols like UART or I2C could also be considered, but SPI is well-established, and since we're already utilizing LVDS SPI, it makes sense to maintain consistency.

Next, we have the stimulation and acquisition chip, the RHS 2116, with its 16 electrodes that will connect to the sensors implanted in the rodent's brain.

To complete the system, we need to transmit data from the microcontroller to the GUI. Let's incorporate a BLE module for this purpose. While Wi-Fi is also an option, we'll proceed with BLE, and I'll elaborate on the reasons behind this choice later. The BLE module will establish the wireless link with our GUI. This outlines the overarching schematic required to build a system capable of achieving our objectives. Now, let me present the complete system diagram I prepared beforehand. This diagram provides a more detailed illustration of the components and connections we just discussed.

In this comprehensive schematic, you'll find detailed descriptions corresponding to the elements we outlined in the previous slide. We have the microcontroller communicating with the FPGA via SPI, the FPGA interfacing with the RHS 2116, the 16 electrodes extending to the brain sensors, the BLE module enabling wireless communication with the GUI, and the battery module providing power to the entire system. This holistic view encapsulates the system architecture we'll be working with to realize our goal of stimulating and acquiring brain signals in rodent-based models.

This diagram illustrates the sensors that will be implanted within the rodent's brain. These sensors have the dual capability of sending signals, which the RHS-2116 will then acquire, and receiving stimulation signals from the RHS-2116. This stimulation can be targeted either at surface electrodes or deep brain electrodes. The intricacies of both these types of electrodes will be comprehensively discussed in the subsequent lecture by Professor Hardik. The signals acquired from the brain will be transmitted to the FPGA, which, due to its inherent capabilities, can adeptly handle the high sampling rate generated by the RHS-2116. Once processed by the FPGA, the data will be relayed to the microcontroller. The microcontroller's primary function is to transmit this data to the GUI or to modify stimulation patterns based on user input.

The entire system will be powered by a battery, and selecting an appropriate battery with a compact form factor is crucial. A lithium polymer battery, readily available and comparatively cost-effective, presents itself as a suitable option. These batteries typically offer a voltage range of 3.3 to 3.7 volts, with varying sizes and charge capacities. The battery will serve as the power source for all microcontrollers within the system. It will also supply the positive stimulation voltage and the necessary bias voltages. Biasing is required for the microcontroller, the FPGA, and the RHS-2116 to facilitate amplification and acquisition processes. To streamline the design and minimize the number of components, we will identify a common voltage suitable for all these chips and employ a single regulator capable of powering them all. This approach will contribute to a more compact form factor.

Additionally, we can utilize voltage boosting or bucking techniques to adjust voltage levels as needed. Some microcontrollers operate at 5 volts, while others function at 3.3 volts. To achieve the desired logic levels, we can employ charge pumps or boost regulators. However, I lean towards charge pumps as they eliminate the need for inductors in voltage boosting.

For stimulation purposes, we'll need a dedicated stimulation supply that aligns with our specific requirements. If the regulator's output voltage matches the stimulation needs, we can utilize it for stimulation as well. To generate a negative stimulation supply, we might need to incorporate inverters to invert the existing supply voltage.

Before proceeding with BLE communication, it's imperative to pre-program the microcontroller so that it can subsequently program the FPGA according to the user's specifications. Programming the microcontroller is most conveniently achieved through UART (Universal Asynchronous Receiver/Transmitter) communication. UART, as the name implies, is an asynchronous communication protocol that employs only two lines for transmission and reception, eliminating the need for a clock signal. In our setup, we'll utilize a USB-to-UART interface to convert the UART signals from the microcontroller into the standard USB format, allowing us to connect it to the GUI via a wired connection. This facilitates both pre-debugging and programming the microcontroller according to our specific requirements. This comprehensive overview encapsulates the fundamental components and interconnections of our system. We've addressed power management, signal acquisition and processing, communication protocols, and the overall system architecture.

Once connected to the GUI, we can program or debug the microcontroller via a wired connection. The USB-to-UART interface seamlessly converts the UART communication from the microcontroller into a standard USB format, allowing for bidirectional data transfer with the GUI. This capability proves invaluable for programming the microcontroller according to our specific requirements or for conducting preliminary debugging to ensure smooth communication before finalizing any parameters.

Although we are employing low-voltage differential signaling (LVDS), which effectively mitigates common-mode noise in the signal lines, it's important to acknowledge that the amplitudes of the brain signals we're dealing with are exceptionally small. Consequently, there's a potential for static discharges to accumulate on the lines. To safeguard against this, we'll incorporate dedicated electrostatic discharge (ESD) protection chips. These chips utilize reverse-biased diodes to safely channel any accumulated static charges to the ground. The ESD protection will be directly tapped into the SPI lines, ensuring their protection from electrostatic discharge.

I hope this overview has provided you with a comprehensive understanding of the schematic. While I haven't delved into the specific details of the microcontroller, BLE/Wi-Fi module, or FPGA chip we'll be using, I want to emphasize the vast array of options available. The internet is a treasure trove of information on these components, and I encourage you to explore them further. My primary focus was to highlight the biopotential acquisition and stimulation chip, which holds immense potential for biomedical applications. This circuit configuration is just one example of how to harness the chip's capabilities; numerous other configurations can be equally effective or even more so, depending on your specific needs.

Let's recap what we've covered in this module. We began by examining the RSS-2116 chip in detail, exploring its internal structure and the ingenious preamplifier stages that enable it to amplify biopotentials and measure electrode potentials arising from stimulation. We also discussed the critical role of the charge recovery switch in maintaining charge neutrality.

Furthermore, we explored how to interface this chip with microcontrollers or FPGAs to create a complete, functional system. We delved into the control mechanisms, highlighting how this configuration simplifies system control and facilitates the achievement of our desired goals. We also examined the overall schematic, illustrating how the various components interconnect. Although this representation is simplified, I urge you to investigate the specifics of how each component connects. Remember that any basic interfacing involves power lines and data lines, which can sometimes introduce complexities. However, the fundamental principles remain straightforward, and I encourage you to explore the intricacies of interfacing with any chip.

In the next module, we will transition from the schematic to PCB design. We'll learn how to translate our circuit into a PCB layout that can be fabricated using a Gerber file. I will guide you through the steps involved in creating your own schematic, generating the PCB layout, and producing the Gerber file, empowering you to manufacture your own PCBs using electronic design automation (EDA) or computer-aided design (CAD) software.

Popular EDA/CAD software options include EasyEDA, a free and web-based tool, and KiCAD, which is also free and open-source. Additionally, there are licensed software packages such as Altium Designer, OrCAD, and Eagle. I recommend starting with EasyEDA or KiCAD to familiarize yourself with the fundamentals of PCB design. This knowledge will prove valuable not only for biomedical applications but for any circuit you may wish to design in the future. I keep reiterating the importance of minimizing the form factor for rodent-based applications, and our ultimate goal is to encapsulate our system design within a compact PCB.

If you have any questions or uncertainties regarding this lecture, please don't hesitate to reach out to us on the forum. We are committed to providing you with the best possible support. I look forward to seeing you in the next lecture. Thank you.