Indonesian Journal of Electrical Engineering and Computer Science
Vol. 8, No. 2, November 2017, pp. 428 ~ 440
DOI: 10.11591/ijeecs.v8.i2.pp428-440

Design of a Reconfigurable, Modular and Multi-Channel Bioimpedance Spectroscopy System

Ahmed Al-Hashimi, Anis Nurashikin Nordin*, Amelia Wong Azman
Department of Electrical and Computer Engineering, Kulliyah of Engineering,International Islamic University Malaysia,
Jalan Gombak, 53100 Kuala Lumpur, Malaysia, (+603) 6196 4478
*Corresponding author, e-mail; anisnn@iium.edu.my

Abstract

This paper presents the design and implementation of a multichannel bio-impedance spectroscopy system on field programmable gate arrays (FPGA). The proposed system is capable of acquiring multiple signals from multiple bio-impedance sensors, process the data on the FPGA and store the final data in the on-board Memory. The system employs the Digital Automatic Balance Bridge (DABB) method to acquire data from biosensors. The DABB measures initial data of a known impedance to extrapolate the value of the impedance for the device under test. This method offers a simpler design because the balancing of the circuit is done digitally in the FPGA rather than using an external circuit. Calculations of the impedance values for the device under test were done in the processor. The final data is sent to an onboard Flash Memory to be stored for later access. The control unit handles the interfacing and the scheduling between these different modules (Processor, Flash Memory) as well as interfacing to multiple Balance Bridge and multiple biosensors. The system has been simulated successfully and has comparable performance to other FPGA based solutions. The system has a robust design that is capable of handling and interfacing input from multiple biosensors. Data processing and storage is also performed with minimal resources on the FPGA.

Keywords*: Bioimpedance Spectroscopy (BIS)**; Field Programmable Gate Array (FPGA)**; Digital Auto Balance Bridge (DABB)**; Multichannel data acquisition.*

1. Introduction

Bioimpedance spectroscopy (BIS) is popular, non-invasive technique of measuring biological fluids in the body to estimate body composition as well as to assess the clinical conditions of a person. By definition, bioimpedance can be described as the ability of biological tissues to restrict electrical current [1–3]. At cellular levels, studying impedance properties of cells allow detection of various substances such as chemical toxins, bacteria, viruses, due to the presence of the many types of receptors on cell membranes [2–4].

Using this technique, currents of varying frequencies are sent to the biological samples. At lower frequencies, due to the high resistance of cells, current passes primarily through the fluids and not through the cell membranes [5]. At higher frequencies, the electrical signals can penetrate through the cell membranes. The impedance measurements of these cells are graphed in the shape of a Bode plot over time and reflect the changes of impedance as the cells progresses through its growth and death cycles. The impedance spectrum of the living cells can offer details about physiological and pathological information of the living cells and what affects them.

Conventionally, impedance characterization of cells and other elements under test is done with the aid of an Impedance Analyzer. This has several problems that this research tries to address: i) The impedance analyzer is a large device which limits the portability of the sensors, ii) Multichannel measurements are not supported [6], iii) a computer is required to store data and perform further analysis. To address these difficulties, this work presents a bio-impedance spectroscopy system on field programmable arrays (FPGAs) that is capable of generating alternating voltage signals at various frequencies as well as measuring bioimpedance. This system is mounted on an FPGA and has the advantage of being portable, can acquire and process multiple signals simultaneously on-board.

Received July 27, 2017; Revised October 5, 2017; Accepted October 21, 2017

This system employs the Digital Auto Balance Bridge (DABB) method [7][6] to calculate the value of the unknown bioimpedance. The DABB block is reconfigurable and can be duplicated in the FPGA as multiple instances to acquire bioimpedance data from multiple sensors simultaneously. In this case, each DABB block will be connected to a different biosensor. The system also includes a processor that performs mathematical calculations to convert the analog to digital data, as well as handle complex number processing. Both the raw data from the DABB and the processed data from the processor are stored on a flash memory. The system has a control unit which, governs the communications of all the different instances in the design. This paper is organized as follows: Section 2 starts with the system overview of the bioimpedance spectroscopy system and Section 3 describes implementation of the system on FPGA. Section 4 discusses the functional results of the system and also shows the allocation of resources on FPGA. Finally, Section 5 concludes the design.

2. System Overview

The designed bioimpedance spectroscopy system is equipped to handle multi-channel, *n* number of inputs from the biosensors. The user predefines the range of frequencies of the AC voltage to be applied (40Hz to 100 kHz). The overall block diagram of the bioimpedance spectroscopy system is shown in Figure 1. The system is composed of several modules namely memory controller, system control unit, processing unit, multiplexers and n number of bioimpedance spectroscopy modules which are placed on the FPGA Cyclone II board. The Memory Controller is used to interface with a flash memory, which is placed off-chip. The bioimpedance spectroscopy (BIS) modules are used to interface with Digital Auto Balance Bridge (DABB) circuit and perform all calculations of the unknown impedance. There are multiple instances of this module to allow multichannel data acquisition. The Processor module processes the raw data from the BIS module. Finally, the system control unit handles the scheduling and communications between all the modules.

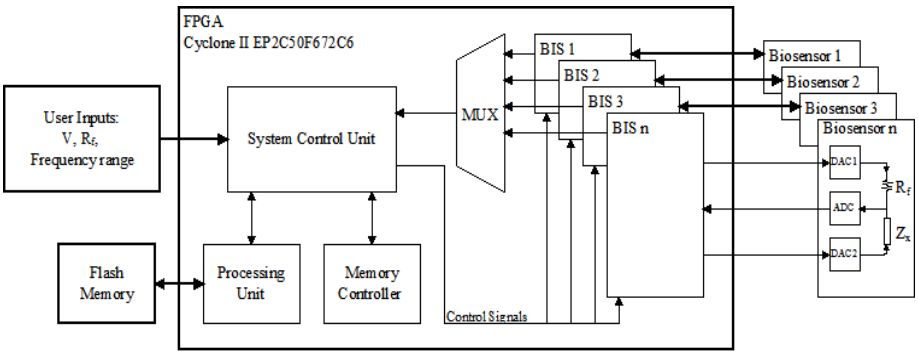

Figure 1. Bioimpedance System block diagram

In this work, the BIS module core has been designed to utilize a minimal number of resources as possible. The BIS module can be instantiated (with minimal additional logic elements) repeatedly depending on the number of resources available on the FPGA. As a proof of concept for multichannel data acquisition, the smallest number of BIS module that has been instantiated in this work is two.

To start, a user will pre-define three parameter inputs; V, R_f and the required frequency range. The BIS module then acquires raw (impedance) data from the device under test or the biosensor. After the data is gathered, the System Control Unit writes the data to the flash memory. It is important to highlight it here that the proposed system operates in real-time such that any unprocessed data that becomes available in the memory will be sent directly to the processor for computations into legible impedance data.

The bioimpedance under test, Z_x is obtained using a biosensor. The system utilizes the digital auto balance bridge (DABB) technique to calculate the unknown impedance value, Z_x. Using this technique; the balance bridge has a reference resistance (R_f) of known value that will

be used to deduce Z_x. First the voltage difference, V_e between the reference and unknown impedance is calculated using equation (1) as follows:

$$V_e = \frac{V_o R_f - V_f Z_x}{Z_x - R_f} \quad (1)$$

where V_o is the output voltage while V_f is the reference and R_f is the known reference resistance in the balance bridge circuit. Next the unknown impedance, Z_x is calculated using equation (2).

$$Z_x = \frac{V_o}{V_f} R_f \quad (2)$$

Z_x is a complex number and can be separated into its real, Z_{xR} and imaginary components, Z_{xI} as follows:

$$Z_{xR} = \frac{(V_{out} \cdot R_f \cdot V_{fR})}{(V_{fR}^2 + V_{fI}^2)} \quad (3)$$

$$Z_{xI} = \frac{(V_{out} \cdot R_f \cdot V_{fI})}{(V_{fR}^2 + V_{fI}^2)} \quad (4)$$

The processing unit handles all computations to obtain the values of Z_{xR} and Z_{xI}. To handle complex number computations on FPGA, a lookup table method was selected because of its simplicity in design and faster processing times. The system uses an on board Flash Memory to store the raw data of the BIS Module and the processed data of the Processor Module.

Figure 2 shows a detailed flowchart of the system operation. The system starts with obtaining input from the user regarding the voltage, reference resistor and frequency sweeps. Next, the systems checks for available raw data from the DABB circuitry. If there is raw data, the BIS module will first write it into the flash memory. The data will be recorded into memory for each frequency within the range specified by the user. The processor then reads the stored raw data in the memory and calculates the values of Z_{xR} and Z_{xI} using equations (1)- (4). Once the computation is complete, the processing unit then writes the processed data to memory.

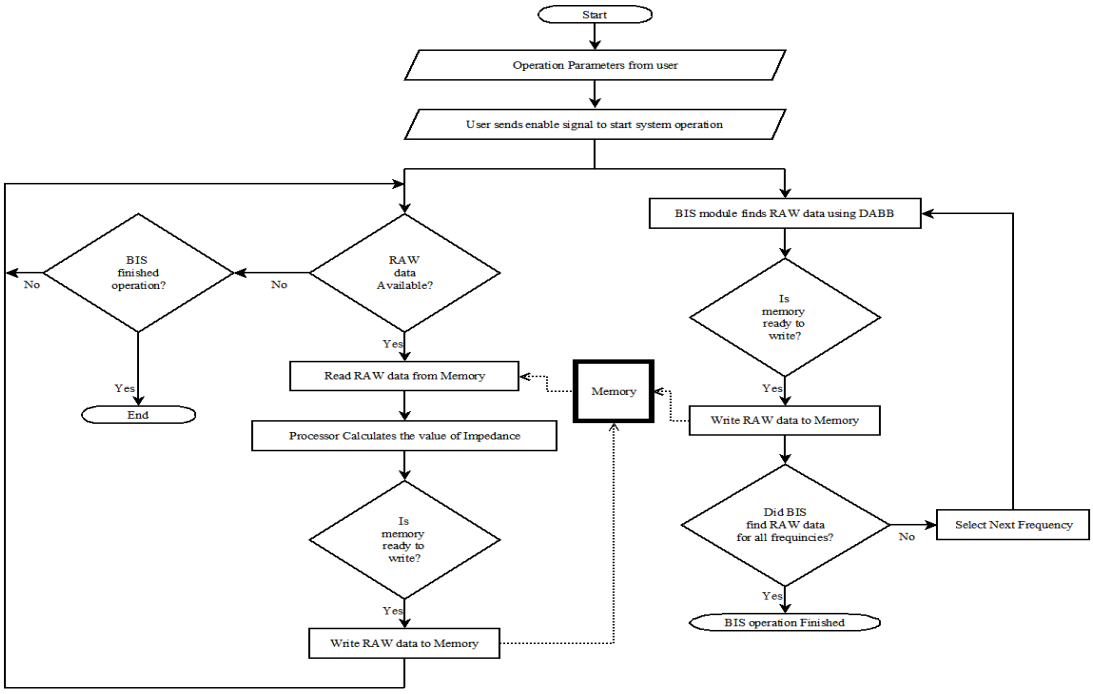

Figure 2. Flowchart for the overall system operation

3. Implementation on Low-Power Cyclone II FPGA

The system was implemented to be optimized on the low-cost Altera Cyclone II FPGA DE2 board. This FPGA utilizes Intel's high-performance, low-power 90nm technology which has an average of 60% higher performance and 50% lower power than other 90nm low-cost FPGAs [8]. Cyclone II devices consume only 50% static and dynamic power compare to other 90nm FPGAs. In addition to this, Cyclone II devices have zero inrush current, hot-socketing support and require fewer power supplies [9]. In terms of worst-case static power consumption, it has been reported that Cyclone II devices only consume a maximum of 0.6W compared to 1.8W for Spartan-3 at 85^{o}C when 70k equivalent logic elements are used [9]. This section describes the design details of each module for the bioimpedance spectroscopy system.

3.1 Bioimpedance Spectroscopy (BIS) Module

The BIS module is a key component of the bioimpedance spectroscopy system. It handles the interfacing between the biosensor and the balance bridge via two digital to analog converters (DACs) and one analog to digital converter. The BIS module also performs the calculations to obtain the amplitude and phase values of V_f that will allow the system to reach a balanced state (V_e = 0). To achieve this, a phase detector and amplitude detector are used to detect the phase and amplitude values of V_e respectively. The data from each BIS module is sent to a multiplexer and then to the processor to compute the values of Z_{xR} and Z_{xI}. The BIS module also acts as a control unit to manage the DAC drivers. The DAC1 driver is used to generate V_o based on the user's input parameters, and this value is constant throughout the entire operation of the system. The DAC2 driver generates V_f whose values changes based on feedback from the phase detector and amplitude detector units. The submodules and operational flow of the BIS modules is shown in Figure 3.

3.1.1 Clock divider

The Altera DE2 FPGA board provides a default 50 MHz for the clock [10]. For our system, clock division is necessary to cater for user specifications, which can be any frequency within the range of 40 Hz to 100 kHz. The input and outputs of the clock divider module are as follows: Inputs: Enable, ClockIn, Frequency [6..0]; Outputs:ClockOut. The *Enable* signal is controlled by the system control unit which allows the module to be active or inactive for power optimization. The user specifies the required frequency of operation through the BIS control unit as the *FrequencyIn* signal. The Clock divider module is basically a counter which divides the 50 MHz clock from the FPGA with a *Scale* value = *ClockIn/FrequencyIn* to produce the desired ClockOut frequency. The clock divider module provides the running clock (ClockOut) for the DAC 1 driver, DAC 2 driver and the amplitude detector modules.

3.1.2. DAC Drivers

The DAC 1 driver connects to the Digital to Analog Converters that generates V_o (amplitude and frequency) supplied to the biosensors The DAC1 driver code is simple since V_o does not need to change phase and amplitude throughout the operation of the system. This module has an 8 bit output connected to the external DAC. The external DAC 1 will generate the desired signal of V_o. The external DAC has a 9-bit resolution and produces full sinusoidal cycle of V_o with 510 segments.

The DAC 2 driver connects to the Digital to Analog Converter that generates V_f. Based on the DABB method, V_f needs to change its phase and amplitude until V_e returns a zero, thus, the DAC 2 driver needs some feedback signals both from the phase detector and amplitude detector modules. The *clock_in* input from the clock divider module dictates the frequency of V_f. Similar to DAC 1 driver, the *enable_in* signal is fed from the BIS control unit module. *Amplitude_in* input from the BIS control unit module is used to select the right amplitude to get the system to a balance state. The *add_in* and *sub_in* inputs which are from the phase detector module are used to inform this driver to shift its phase either by adding or subtracting 90, 45, 23, 11, 6, 3, 2 and 1 degrees. Phase shifting is performed by converting these numbers into binary and shifting the value of 45 = 101101_2 to the right by 1 bit. The DAC 2 driver also has am 8-bit *data_out* output that connects to the DAC 2. Similar to DAC 1, DAC 2 will generate the desired analog signal of *Vf*. Once V_f = *Ve*, a *no_fluctuate_out* is sent to the phase detector module allowing it to perform operations. Conversely, when the module receives either *add_in* or *sub_in* the module changes the phase of the output accordingly, and *no_fluctuation_out = 0.*

3.1.3. Phase Detector

The phase detector module detects the phase of V_f and V_e and sends appropriate signals to DAC 2 river module to be able to change the phase of V_f until the system is in phase. When this module finds that V_e and V_f are in phase it reports that it is in phase and value of the phase is sent to the BIS control unit module. The phase of V_f can only be equal to the phase of V_e when the phase of the signal across Z_x is equal the phase of the signal across R_f, and since R_f is purely resistive then it does not change the phase of V_f.

When the system detects that V_e is equal to V_f this modules operation is finished and the phase of V_f is ready to be sent to the memory to be stored as raw data, this raw data will then be sent to the processor to find the final impedance values of Z_x. The outputs *add_out* and *sub_out* signals are used to tell the DAC 2 Driver to change its output *data_out*. The output signal *in_phase_out* is used to tell the BIS control unit module and the DAC 2 driver module that the phase detection of the operation is done. The *phase_out* output bus sends the final result of the phase shift when this module finds it to the BIS control unit module.

3.1.4. Amplitude Detector

After the Phase Detector finishes its operation the Amplitude detector, takes over reading the data from the ADC to detect if V_e is equal to zero. The Amplitude Detector also gives feedback to DAC 2 Driver to allow the change in the amplitude of V_f that will help the system reach a balanced state. When this module detects that V_e is equal to Zero then its operation is finished. The DAC 2 driver sends the final amplitude to the memory and is stored as raw data.

3.1.5 BIS Module Operation and Results

The Bio-impedance Spectroscopy Module is activated when it receives a '1' on the *Enable Input* port and a '1' on the *Operate Input* port. The DAC drivers and Phase Detector unit are enabled next. Both DAC drivers would send signals to their respective DAC to generate V_o and V_f for balance bridge circuit. The Phase Detector unit reads the data from the ADC which converts the V_e signal of the balance bridge circuit into digital representation, as shown in Figure 3.

After a series of steps, that passes through the DAC drivers, DACs, phase detector. Amplitude detector and finally the BIS Control Unit. The BIS control unit schdules all interactions between the DAC drivers, DACs, ADC, phase detector and amplitude detector. Once the data is processed, the BIS control unit will send the Ready Out signal to the System control Unit, which indicates that the raw data is ready. In the case of multiple BIS modules, the BIS Control Unit also sends out the flags_out signal to identify which BIS module is sending the data. This powerful module also controls the address of the output data via BIS_next address_out.

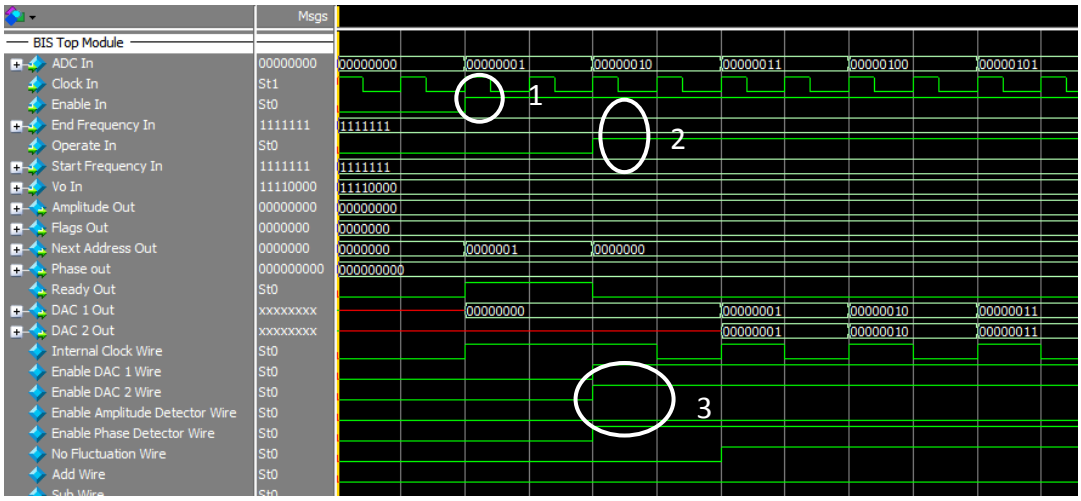

Figure 3. Initializing the BIS module. Step 1: Enable goes to 1. 2: Operate In goes to 1. 3: Enable DAC 1 and DAC 2 goes to 1

3.2 Processor Module

The processor module calculates the real and imaginary values of Z_x as shown in Equation (3) and Equation (4). The processor also has to convert the real and imaginary values of Z_x into its polar form to obtain the phase shifts between the input and output signals. FPGAs cannot perform sine and cosine operations automatically. A popular method to calculate sine and cosine is CORDIC by Volder [11], [12]. This method is based on rotating the phase of a complex number, by multiplying it by a succession of constant values. Since all multiplications can be powers of 2, a binary arithmetic can be done by shifting and adding without the need of an actual multiplier. The main advantage of using CORDIC is that the software implementation will take fewer resources on the FPGA when compared to other methods. However, this approach requires multiple clock cycles to find the answer since it needs to iterate the operation multiple times.

Another faster and easier method is to use a lookup table to determine the sine and cosine values [13]. The resources needed for the lookup table implementation are not significantly different from the resources taken by CORDIC algorithm. A lookup table has the sine and cosine values between 0 and 90 degrees. The trigonometric identities are then used to find the sine and cosine values of any angle. This method is more resource efficient as it requires less computation.

The block diagram of the processor module shown in Figure 4 where it comprises of the control unit, divider, multiplier and sine/cosine lookup table. The operation flow of the processor is shown in Figure 5. The processor takes the initial data provided by the user (V_{out} and R_f) and the raw data found by the Bioimpedance spectroscopy (BIS) unit ($|V_f|$ and $V_f(\theta)$). It then uses the sine/cosine lookup table to turn $|V_f|$ and $V_f(\theta)$ into V_{fR} and V_{fI}. These two values are then used to find the denominator of Equation (3) and Equation (4). Meanwhile, the initial data from the user allow the processor to find the nominators for Equation (3) and Equation (4). The denominator and the nominator are represented by L and M in Figure 5 respectively. Multiplier operations have been used to obtain these values. Finally, Divider operations are used to solve Equation (3) and Equation (4) by dividing the variable L and M with the value D. This module's final result will be the value of real and imaginary values of $Z_{x,}$ which is loaded onto its output port along the original raw values. The flag 'data ready' values change once computation is complete, to show that the data has been processed.

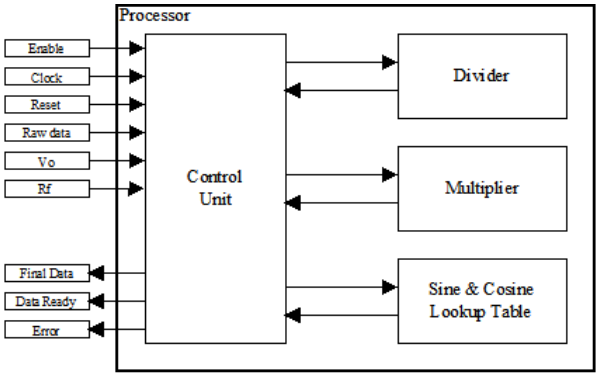

Figure 4. Processor module block diagram

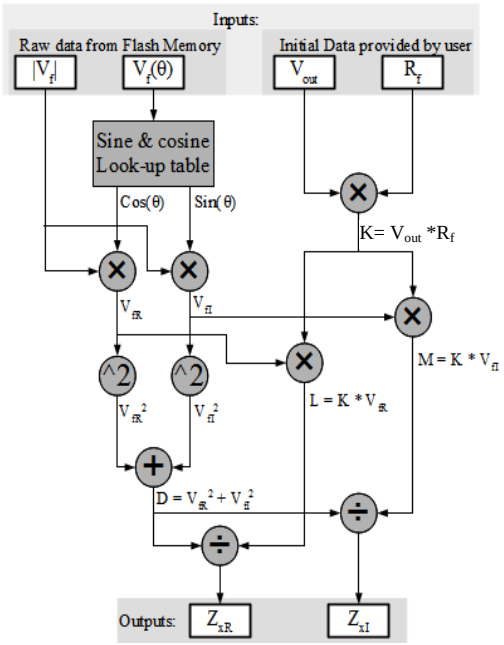

Figure 5. Operation flow for the processor to find the real and imaginary values of the unknown impedance Zx

The processor is initialized when it it receives the necessary data and the enable signal from the System Control Module. The Processor Module will also set the enable signal for the Multiplier Unit to '1' and sends the values of *phase shift* from the BIS module to the *dividend* port for it to start calculations. Once the Multiplier unit has completed calculations, the Processor Control Unit sets the *enable* signal for the divider to '1'.

The Sine and Cosine unit performs its operations via a lookup table. Once this step is completed, the processor next enables the multiplier unit to calculate the values of Z_{xR} and Z_{xI}. These Z_{xR} and Z_{xI} values were calculated for inputs: V_o = 5 V, V_f = 0.294 V, R_f = 10 Ω, Theta = 31.765° and can be compared with the calculated results of Z_{xR} = *143.7 Ohms* and $Z_{x\ I}$ =*89 Ohms.*

The design of the system allows each of the modules (Processor, and multiple BIS modules) to operate independently. This will decrease the overall time needed to find the impedance values. From the simulation results of a BIS module, it is noted that the system needs a varying amount of time to find the RAW data depending on the user predefined operation frequency ranging from 40 Hz and 100 kHz. Therefore, the time needed to perform 4 cycles would be between 0.1s and 0.04 ms for each frequency value. Meanwhile, the processor requires 1930 clock cycles at a constant clock speed to process the raw data. This means that proposed system running at a 50 MHz clock speed needs approximately 0.0386 µs to process an incoming raw data and to compute impedance values.

3.3 Flash Memory

The memory used in this system is the onboard flash memory available on Altera DE2 development board. The core used to access the memory and methods of usage is obtained from The Altera University Program, which provides all the intellectual property (IP) cores needed to operate all the integrated chips and devices on the Altera DE2 development board. The flash memory IP Core documentation by Altera Corporation [14] describes the IP core and its operation with the onboard Flash Memory. This work uses the standalone version of the flash Memory IP Core.

The system uses a flash memory to store the raw data $|V_f|$ and $V_f(\theta)$ as well as the output from the BIS module Z_{xR} and Z_{xI}. The raw data is 24-bit as shown in Figure 6(a) and comprises of flags, $|V_f|$ and $V_f(\theta)$. The flags, serves two purposes: i) to indicate completion of data processing and ii) as a BIS module identifier. Bit 17 represents the 'processed' flag and it is set to '0' for a raw data and a '1' when the data has been processed. For multi-channel usage, bits 23-18 indicate which BIS module has generated the data. Bits 16-8 stores $V_f(\theta)$ and bits 7-0 stores $|V_f|$. Since the operation of the bio-impedance spectroscopy module is sequential, it finds the raw data for each frequency step in the selected range in an ascending sequential order. The location of the data in the flash memory indicates the frequency at which the data was obtained.

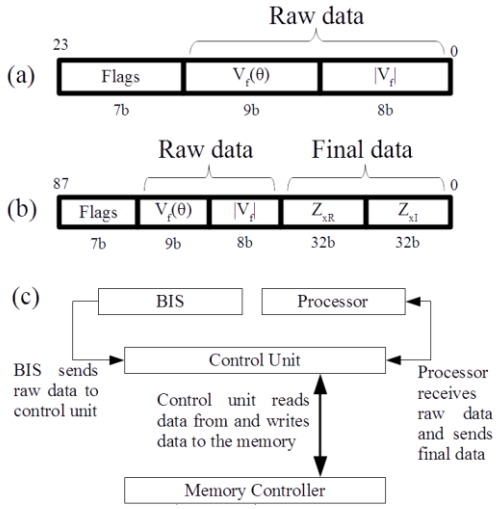

Figure 6. (a) Bit arrangement of raw data in memory (b) Bit arrangement of raw and processed data in memory (c) Data read and write flow diagram

When the processor receives the raw data from the memory, it will read all 24 bits and uses the phase and amplitude data to find the impedance values. Once the process is completed, the processor will set bit 17 in the Flags to '1' to indicate that the data has been computed. The processor outputs two values, Z_{xR} and Z_{xI}, which are the real and imaginary values of the unknown impedance Z_x respectively. Each of these values is 32-bits wide.

Figure 6(b) shows the bit arrangement for the processed data. The flash memory allocates an 8-bit (1 byte) sized locations to store data. Therefore, the values of Z_{xR} and Z_{xI} are broken into 4 bytes each to store all 32 bits. The flags field in this bit arrangement is identical to the one in Figure 6(a). For the phase value, the 8 least significant bits are stored in a byte and the flags along with the one remaining bit of the phase are stored in one byte. When writing the raw data to the memory the address chosen for each word is set to have 11 bytes of space and the 3 most significant bytes would hold the raw data, then when the processor finds the impedance values they will be written in the remaining 8 bytes and the most significant byte that holds the flags would be adjusted to set the 'processed' flag to '1'.Figure 6(c) shows the flowchart of reading and writing the data into the memory. Data management is scheduled by the control unit, which interfaces between the BIS module and the processing unit. Once data processing is complete, the control unit sends the data to be written to the flash memory via the memory controller unit. The processed data, Z_{xR} and Z_{xI} is stored as 64-bit LSB in the memory.

3.4 System Control Unit

The system control unit handles all processing, communications and scheduling between the processing units, BIS modules, multiplexer and flash memory. For this work, we have simulated the system control unit with 3 BIS Modules. To initialize the system, the user will input V_o peak, R_f, and the frequency range. When the system starts to operate, the control unit sets the enable signal for the BIS modules to high (Refer to Figure 7:1), sends the values of V_o and the frequency range to all of the BIS modules. The Control Unit implements a polling system, which cycles through each BIS module to check if the BIS module is ready (Refer to Figure 7: 2 and 3). Once the data is ready, the bis_operate_out is set to '110' to pause operations in the BIS module so that the system control unit can access the data in that particular BIS module and write it in the Flash Memory. Writing to the flash memory requires an erase operation to be performed first (refer to Figure 7:8) where mem_erase_out is set to '1'. Completion of the erase operation is indicated when the mem_done_in is '1' (refer to Figure 7:4). Next the first byte of the data from the BIS module will be written into the Flash Memory, mem_write_out = 1. (Refer to Figure 7:9). Once the write operation is finished, mem_done_in indicates a 1 (refer to Figure 7:5). This process is repeated for the next 2 bytes of data as the BIS Module occupies 3 bytes in the memory. Figure 7: 8 and 9 show all three read and write signals being sent to the memory.

The control unit also checks the state of the processor of its availability as well as the flash memory for any unprocessed data. When both the processor is free and there are unprocessed data, the control unit reads the data in the memory (refer to Figure 7:10) and sends it to the processor. When the processor has completed its process, it sets proc_data_ready_in to '1', and this data need to be written to the Flash Memory. The final data from the processor is an 88 bits wide (in the simulation it is broken to *proc_data_in_a, proc_data_in_b,* and *proc_data_in_c* because the simulation software (ModelSim Altera 10.1d) cannot simulate input wider than 32 bits). The final data includes the flags, and raw data, as well as processed data, and all of it needs to be written to 11 bytes in the memory. The writing operation is the same as described earlier but the System Control Unit will have to erase and write 11 times which is shown by the change in *mem_erase_out* and *mem_write_out* (refer to Figure 7: 12 and 13). The memory will also set its *mem_done_in* signal to '1' after each read and write operation.

The operation is similar for every BIS module, and since the control unit will cycle through all blocks to check if any flag is high, therefore, each BIS would have its turn to write to the memory. Once the write operation is complete, the control unit would then instruct the BIS module to move to the next frequency and find the needed raw data. It is important to highlight it here that whenever there is unprocessed data becomes available and the processor is free, the Control Unit would send data from the memory to the processor for it to process to find the values of real and imaginary Z_x.

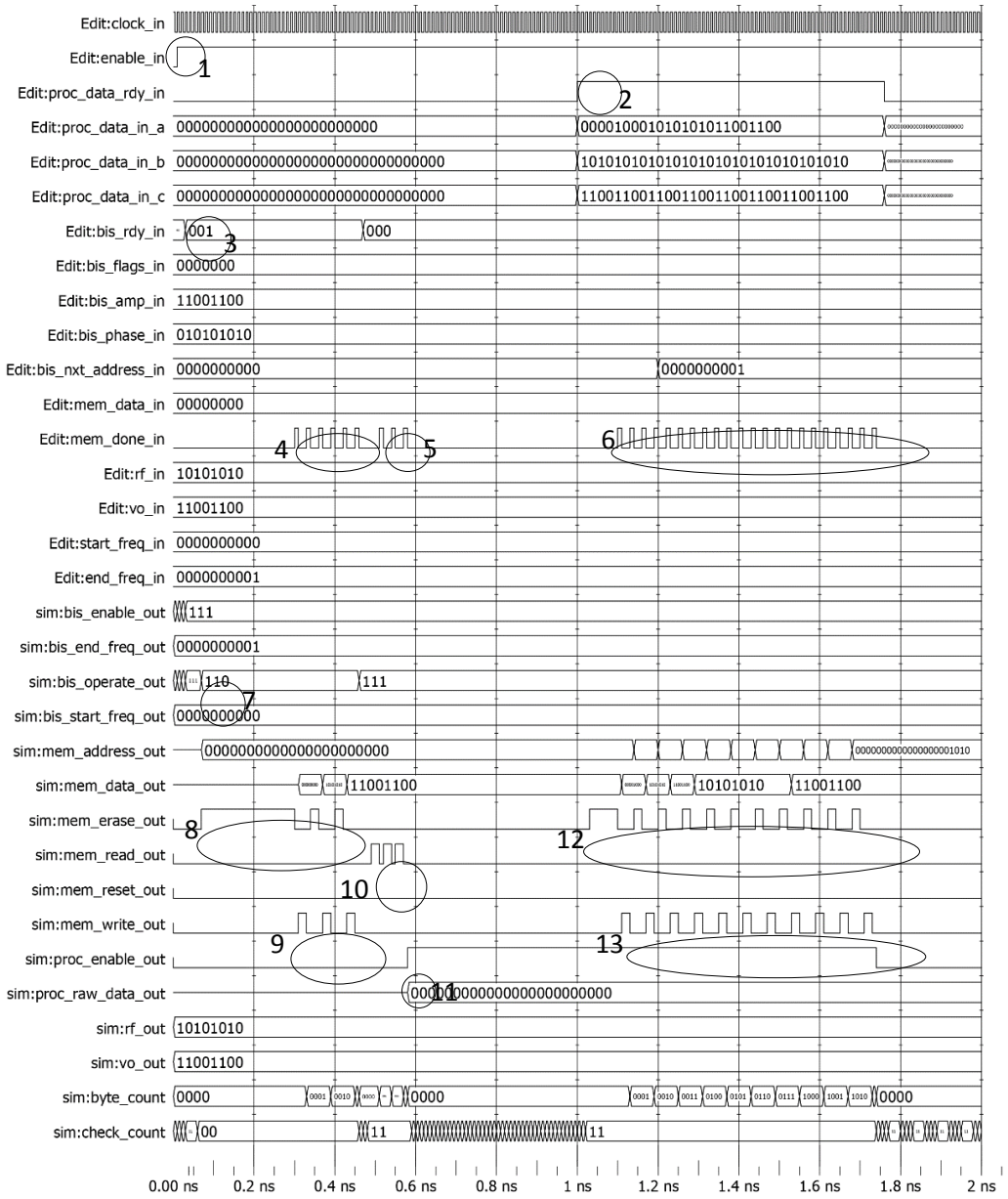

Figure 7. shows the full operation of the Control Module

3.5 System expandability

The system is designed be reconfigurable and can cater to multiple DABB circuits specified by the user. This modular topography is a distinct advantage as it allows the system to have multichannel connections to DABB circuits. To handle input from *n* number of biosensors, *n* number of BIS modules can be instantiated into the design. Each BIS module can then work in parallel to autonomously find $|V_f|$ and $V_f(\theta)$ for each sensor. The rest of the system only requires minimal changes to accommodate the additional modules since the control unit will evaluate every BIS module to identify if data becomes available at any of the modules. The multiplexer module can easily be expanded to accommodate more inputs. The size of control unit also is also expandable to handle the increased number of the ready signals coming from the BIS modules increases and the increased number of loops going through the BIS modules. In the current system, the 6-bit flags can support up to 2^6 or 64 BIS modules in total. Further expansion is also possible with code modifications of the flags to accommodate more than 64

BIS modules. This change to the Flags must be done in the Control unit, BIS, and Processor, because all of these blocks change or read the flags.

This modular design allows the system to be more versatile although it might sacrifice some speed as the number of BIS modules becomes larger since a bottleneck at the processor might occur when handling too many BIS modules. Having said that, the computational operation would not be affected since any raw data found by the BIS modules would be immediately stored in the memory for further computation.

4. Performance Evaluations

4.1 Resource allocations

To evaluate the performance of the modular bioimpedance system, we first study the effect of increasing the number of BIS modules on the amount of resources needed on the FPGA. The increase in number of BIS modules occurs when the user specifies multi-channel or usage of multiple biosensors in parallel. Table 1 summarizes the increase in total logic elements and combinational functions when the number of BIS units is increased to 5. It is notable that there is a nearly linear increase in the number of Logic Elements needed with the addition of each new BIS unit. It is also interesting to note that the system with three BIS units uses only 11% of the total available Logic Elements of the Cyclone II FPGA, indicating a great potential for system expansion. This shows that this system is very versatile and can be easily customized according to user specifications.

Table 1. Resource usage for system with 1 to 5 BIS modules

No. of BIS:	1	2	3	4	5
Number of Logic Elements / Total Number of Logic Elements	4128/ 50528	4743/ 50528	5347/ 50528	5895/ 50528	6447/ 50528
Total combinational functions	3024	3604	4174	4688	5206
By number of LUT inputs					
4 input functions	1063	1257	1463	1666	1850
3 input functions	1365	1555	1723	1830	2001
2 input functions	596	792	988	1192	1355
By mode					
Normal mode	2215	2628	3031	3405	3783
Arithmetic mode	809	976	1143	1283	1423
Dedicated logic registers	2563	2787	2983	3172	3359

The number of lookup table (LUT) input functions can also be customized. Here it can be seen that while doubling the number of LUT functions from 2 to 4 doubles the resource usage, the linear relation is less dramatic when the number of BIS modules is added. In terms of operations, the bioimpedance system can also be categorized into three different modes namely: normal, arithmetic and using dedicated logic registers. The less resource intensive mode is the arithmetic mode which uses half the number of resources compared to the normal mode.

For every extra BIS module added, the Control Unit resources will also increase accordingly. Based on the analysis, it is shown that when the system is scaled to include more BIS modules, the increase in system resources is not directly proportional to the BIS module. The system actually would use slightly less Logic Elements (LE) and Dedicated Logic Registers (DLR) with every subsequent addition of a BIS module. This is because that Quartus II is capable of efficiently optimizing onboard resources to ensure that each module is operating with minimal LEs and DLRs.

Table 2 summarizes the utilization resources of each of the main modules in the system. It is noted that the processor module requires the most resources due to the complex computation needed to convert the polar notations into Cartesian as well as the sine and cosine lookup tables. In contrast, both the control unit and the BIS modules require only minimal amount of resources. The small size of the BIS modules allows the user to be able to instantiate multiple instances of it depending on the available resources and the user requirements.

Table 2. Resource usage for BIS, Processor, and control unit modules

	BIS	Processor	Control Unit
Total Logic Elements	519	3125	593
Total combinational functions	485	2101	545
By number of LUT inputs			
4 input functions	179	730	217
3 input functions	152	1099	158
2 input functions	154	272	170
By mode			
Normal mode	336	1599	333
Arithmetic mode	149	502	212
Dedicated logic registers	183	2252	154

The system is also designed to have separate module for each BIS unit and for the processor. This allows each of these modules to operate independently in parallel, decreasing the overall time needed to find the impedance values. From the simulation results of a BIS module, it is noted that the system needs a varying amount of time to find the raw data depending on the operating frequencies. The time needed to perform 4 cycles would be between 0.1s and 0.04 ms for each frequency value. Meanwhile, the processor requires a total of 1930 clock cycles at a constant clock speed to process the raw data. This means that proposed system running at a 50 MHz clock speed needs approximately 0.0386 µs to process an incoming raw data and to compute impedance values.

4.2 Speed and Limitations

Looking at the speed of the system, one BIS running between 40Hz and 100 KHz at 4 cycles for each frequency for a total of 1001 steps would need 0.399s to find the raw data for all the frequencies. And since the processor needs 0.0386 µs to transform one piece of raw data into processed data, it is theoretically possible to have one processor for 10,347 BIS modules. The other limiting factors would be the resources available on the FPGA and the size of the Flash memory. There are other factors that will slow down the operation slightly that needs to be considered such as the time the control units need to cycle through all the different module, and the time to write the data to a Flash Memory. For this work, the design has been tailored to work at a frequency range of 40Hz to 100KHz, this is however reconfigurable and the user will be able to change the frequency range to what they need, but it will affect the time needed to find all raw data.

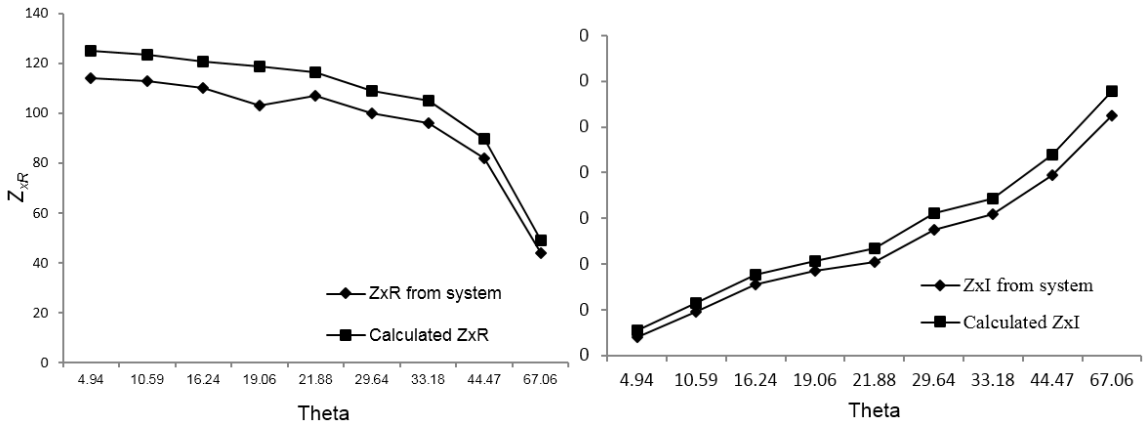

Figure 8. Left: Calculated and simulated values of Z_{xR} at multiple thetas, Right: Calculated and simulated values of Z_{xI} at multiple thetas

For functional verifications, the simulated results from FPGA are compared to theoretical calculations. Figure 8 compares FPGA simulations with the theoretical values of the real part of the impedance Z_{xR} at different theta angles and also illustrates the difference

between the theoretical and simulated values for the imaginary part of the impedance Z_{xI}. The average error between the simulated and calculated values were found to be e_{ZxR} = 9.29% for the real part of impedance Z_{xR} and e_{ZxI} = 13.21% for the imaginary part of the impedance Z_{xI}. This error is acceptable based on the target research by [15] as the change in bioimpedances of cells for e.g. is an order of magnitude greater these values.

5. Conclusion

This work presents a highly modular bioimpedance spectroscopy system on FPGA. The proposed system is capable of acquiring multiple signals from multiple bio-impedance sensors, process the data on the FPGA and store the final data in the on-board Memory. The system uses DABB circuits to obtain the value of the unknown impedance of the biosensor. This method offers a simpler design because the balancing of the circuit is done digitally in the FPGA rather than using an external circuit. Computations of the impedance are done in the processor and stored in the onboard flash memory. The main advantage of the system is that it is modular and can interface with more than one DABB circuits with minimal increase in system resources. In this work the system has been demonstrated for three multichannel measurements compared to single sensor measurements [6]. The system has been simulated successfully and has comparable performance to other FPGA based solutions [7,16]. The system also uses a small amount of resources similar to the research by [17].

Acknowledgments

This research was supported by Escience Fund: SF16-004-0073 under the Ministry of Science and Technology Malaysia.

References

[1] Yang, L.; Liu, W.; Chen, R.; Zhang, G.; Li, W.; Fu, F.; Dong, X. In Vivo Bioimpedance Spectroscopy Characterization of Healthy, Hemorrhagic and Ischemic Rabbit Brain within 10 Hz–1 MHz. *Sensors* 2017; 17, 791, doi:10.3390/s17040791.

[2] Radke, S. M.. Alocilj. E. C.. Design and fabrication of a microimpedance biosensor for bacterial detection. *IEEE Sens. J*. 2004; 4: 434–440, doi:10.1109/JSEN.2004.830300.

[3] Asphahani, F.. Zhang, M. Cellular impedance biosensors for drug screening and toxin detection. *The Analyst*. 2007; 132: 835, doi:10.1039/b704513a.

[4] Beach, R. D. Conlan, R. W. Godwin, M. C. Moussy, F. Towards a miniature implantable in vivo telemetry monitoring system dynamically configurable as a potentiostat or galvanostat for two- and three-electrode biosensors. *IEEE Trans. Instrum. Meas*. 2005; 54: 61–72, doi:10.1109/TIM.2004.839757.

[5] Palanisami, A. Mercier, G. T. Fang, J. Miller Jr., J. H. Nonlinear Impedance of Whole Cells Near an Electrode as a Probe of Mitochondrial Activity. *Biosensors.* 2011; 1: 46–57, doi:10.3390/bios1020046.

[6] Li, N. Guo, J. Nie, H. S. Yi, W. Liu, H. J. Xu, H. Design of Embedded Bio-Impedance Analyzer Based on Digital Auto Balancing Bridge Method. *Appl. Mech. Mater.* 2011; 135–136, 396–401, doi:10.4028/www.scientific.net/AMM.135-136.396.

[7] Li, N. Xu, H. Zhou, Z. Sun, Z. Xu, X. Wang, W. Wide bandwidth cell impedance spectroscopy based on digital auto balancing bridge method. In; *IEEE*, 2011; 53–56.

[8] Cyclone II FPGA Starter Development Kit Available online: https://www.altera.com/products/boards_and_kits/dev-kits/altera/cyc2-2c20n-kit.html (accessed on Aug 4, 2017).

[9] Cyclone II Power Comparison—1/2 the Power of Competing 90-nm Low-Cost FPGAs Available online: https://www.altera.com/products/fpga/cyclone-series/cyclone-ii/features/cy2-power-compare.html (accessed on Aug 4, 2017).

[10] Joost, R.; Salomon, R. CDL, A Precise, Low-Cost Coincidence Detector Latch. *Electronics* 2015; 4: 1018–1032, doi:10.3390/electronics4041018.

[11] Volder, J. E. The CORDIC Trigonometric Computing Technique. *IRE Trans. Electron. Comput*. 1959; EC-8: 330–334, doi:10.1109/TEC.1959.5222693.

[12] Macias-Bobadilla, G. Rodríguez-Reséndiz, J. Mota-Valtierra, G. Soto-Zarazúa, G. Méndez-Loyola, M. Garduño-Aparicio, M. Dual-Phase Lock-In Amplifier Based on FPGA for Low-Frequencies Experiments. *Sensors*. 2016; 16(379), doi:10.3390/s16030379.

[13] Aguiar Santos, S. Robens, A. Boehm, A. Leonhardt, S. Teichmann, D. System Description and First Application of an FPGA-Based Simultaneous Multi-Frequency Electrical Impedance Tomography. *Sensors*. 2016; 16: 1158, doi:10.3390/s16081158.
[14] University-IPCores Available online: https://www.altera.com/support/training/university/materials-ip-cores.html (accessed on Jul 17, 2017).
[15] Ibrahim, I.; Nordin, A. N.; Hashim, Y. H.-Y. The Study of Cell Attachment and Spreading on Polyaniline and Gelatin using Electric Cell-substrate Impedance Sensing. *J. Pure Appl. Microbiol.* 2014, 8, 827–831.
[16] Rossi, M.; Bennati, M.; Thei, F.; Tartagni, M. *A Low-cost, Portable System for Real-time Impedimetric Measurements and Impedance Spectroscopy of Sensors*. In; Rome, 2012.
[17] Hamed, A.; Tisserand, E.; Berviller, Y.; Schweitzer, P. *Embedded System Design for Impedance Measurement of Multi-piezo Senso*r. In; IEEE: Venice, 2010; 6–11.