Yahia Hassanein

900150800

**Heart Monitor Final Report**

The project chosen will be a real time heart monitor using the AD8232, which is an integrated signal conditioning block for ECG and other biopotential measurement applications. It is designed to extract, amplify, and filter small biopotential signals in the presence of noisy conditions. This design allows it to be used as an embedded microcontroller to acquire the output signal. The STM32F103C8 will be used along with The AD8232 to measure the electrical activity of the heart. This electrical activity will be transmitted and be charted as an ECG on a python code.

The AD8232 Heart Rate Monitor has nine pins Which are: SDN, LO+, LO-, OUTPUT, 3.3V, GND, which provide the basic needed components to be able to communicate with different microcontrollers, including the STM32F03C8. Also provided on this board are RA (Right Arm), LA (Left Arm), and RL (Right Leg) pins to attach and use custom sensors if needed. Additionally, there is an LED indicator light that will pulsate to the rhythm of a heartbeat. Biomedical Sensor Pads and Sensor Cable are provided and will be used for the heart monitor.

As for the general architecture of the application, The Biomedical Sensor Pads will be attached on the correct locations of the human body through the sensor cables, while the other end will be connected to the AD8323. The Pulse will be transmitted through the OUTPUT pin as an Analog signal to the ADC of the STM32F. After the necessary analysis on the signal and converting it to digital one, the values will be sent to a Python code for plotting through a Serial Communication (UART). The Python code will receive the values through the Pyserial, which is a module encapsulates the access for the serial port. It provides backends for Python running on Windows, OSX, Linux, BSD (possibly any POSIX compliant system). After successfully receiving the values, they will be plotted in real time using matplotlib and animation libraries.

The user will be able to specify the value of the sampling frequency required for the ADC as well as both the COM Port and the Baudrate to start the conversion. The data will be collected for one minute while plotted on a graph. The collected data will be used to report the heart rate in bpm using the HeartPy library. The user will only deal with the graphical user interface that is implemented using the Tkinter library.

**Code Implementation:**

This is the C file and it begins with waiting to acquire the Baudrate before acquiring the start signal. It calculates the delay that the timer is going to use to achieve the desired sampling rate. After that, it will calculate the required number of samples for 1-minute worth of data.A screenshot of a social media post

Description automatically generated

The code will start acquiring the signal using the ADC and transmit it via UART. The ADC Starts and Stops in order to not to be converting all the time, which wastes power as we require it to work in less number of conversions. The timer starts right after the conversion is completed. It stops conversion fully after the 1-minute worth of data is collected and transmitted successfully.

A screenshot of a social media post

Description automatically generated

This is the python code with the GUI that is implemented using Tkinter library. Window is displayed along with its title and starting size. Label is displayed with combobx for the user to choose the COM Port and the call back function to initialize the serial, which is implemented using the Pyserial library.

A screenshot of a cell phone

Description automatically generated

The Baudrate is then set using another combobox and it call back function. Then, the sampling rate is taken from the user. The user cannot set the sampling rate unless both the COM Port and the Baudrate are both acquired, or an error message will be shown to notify the user to do so. After the desired sampling rate is chosen, it will be transmitted to the STM through UART after the user presses the button “Set”. Once a value is chosen for each of the required fields, they cannot be changed until the application finishes a whole iteration.

A screenshot of a cell phone

Description automatically generated

The user then press the “start” button in order to signal the STM to begin the conversion.

A screenshot of a cell phone

Description automatically generated

The GUI Will look like the following:

A screenshot of a cell phone

Description automatically generated

In case that the user input the required fields:

A screenshot of a cell phone

Description automatically generated

The digital signal will then be plotted using animation library to be shown in real time.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

The graph will look like the following:

A screenshot of a cell phone

Description automatically generated

After acquiring and plotting all the data, the user will be in window where the user chooses whether he/she wants to view the reported heart rate in bpm or not using buttons “Yes” or “No”, respictively. If the user decides to view the heart rate, it will be showed in the Text widget.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

The corresponding GUI will look like the following:

A screenshot of a cell phone

Description automatically generated

**Hardware Connection:**

A picture containing indoor, sitting, table, shoes

Description automatically generated

**Deliverables:**

* Zip file that contains Cubemx and uvision files
* Power Point Presentation
* Text file that contains link to GH