Yueyang Cheng, Yingge He
CSE/EE 474 group #24
Final Project Report
March 14, 2018

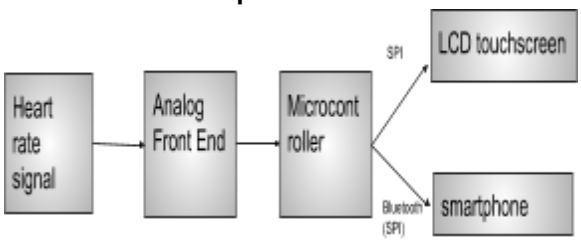
Project Abstract
We have developed a real-time heart monitor to detect Arrhythmia. Arrhythmia is the abnormal rate of heart contraction which is dangerous as it may also cause death. Thus, this project mainly deals with the implementation of an efficient arrhythmia detection algorithm in Teensy board. The idea of this project is from new emerging concepts such as “wireless hospital” or “mobile healthcare”, which require the development of bio-signal acquisition devices to be easily integrated for clinical purposes. In order to ensure the accuracy of this device, a subject we tested includes normal patients, patients with Bradycardia, Tachycardia, and patients with ventricular premature contraction(PVC) or Premature Atrial Contraction (PAC). The completed project was constructed by a simple amplifier circuit with a microcontroller. We add a Bluetooth as our extra future that could be helpful to achieve the wireless clinical diagnosis.

Introduction
Heat diseases are one of the most important reasons which cause death nowadays. Heart disease killed 631636 people in 2006 in US which are 26 percent of whole death population in US in this year [1]. According to the World Health Organization, Heart disease is the top one cause of death worldwide. However, 82% take place in low- or middle-income country due to the lack of adequate medical and healthcare services. Therefore, an cheap and effective device for heart health monitoring is on demand around the world.
In this final project, a system for Electrocardiogram acquisition and arrhythmia analysis is built. We used an amplifier circuit as analog front end for acquiring ECG wave, and a programmable microcontroller Teensy as processor to process arrhythmia detection algorithm. Whole process of measurement is efficient, patient only need to put fingers on dog tags for 30 second after system finished calibration. The 30s ECG wave data will be recorded for later review, and tentative diagnostic report will be provided. Bluetooth allows synchronous data transmission of heart beats per minute(BPM) to the prebuilt application in smartphone.

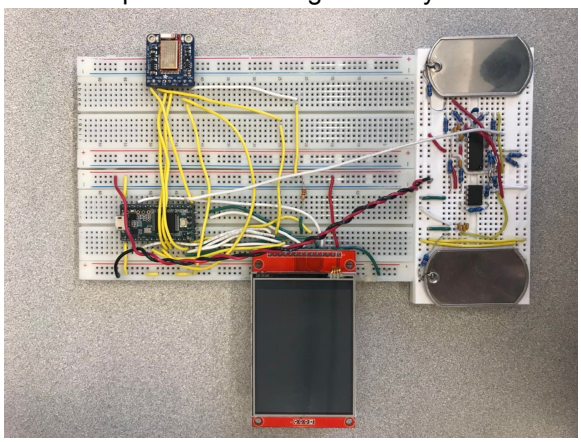
From software perspective, the main challenge of this project is to implementa arrhythmia detection algorithm in our system. Cardiac arrhythmias are disturbance in the behavior of the heart’s electrical activities. In ECG wave, such arrhythmias manifest themselves as irregularities in the observed waveform. Thus, we applied different filter to eliminate noisy signal and amplify useful signal. And then we used the idea of signal processing to find out QRS complex and RR interval to do further analysis. After we collected all the useful data, we used equation from Pan-Tompkins algorithm to generate results.

In sections below, theory of operation are explained, future work are discussed and overall conclusion are made.

Theory-of-operation
1. Hardware description



Graph#1. Block Diagram of System


Graph#2. Circuit layout

The hardware consists the Teensy 3.1 microcontroller, heart signal amplifier, ILI9341_t3 LCD and Adafruit nRF51 bluetooth module. At first, we put fingers on both metal plates. The amplifier will magnify the tiny heart signal and send it to Teensy port A0(14). Teensy then reads the signal every 4ms (250 Hz) and save in the “sample array”. After the program processes the data, Teensy drives the “BlueFruit” module and LCD display using the SPI interface. On the LCD display, we print out

our designed embedded system and we can see the result of measurement in our cellphone via bluetooth communication.

Inside the microcontroller, there are ADC, PDB, and DMA control the system to process input data in required frequency.

2.ECG hardware

Our analog front end for ECG consists seven parts. First is two electrodes, which detect voltage signal directly from subject's fingers. Second is the instrumentation amplifier which enlarges the voltage signal. The third is the bypass capacitor that shorts AC signals to ground, so that any AC noise that may be present on a DC signal is removed, producing a much cleaner and pure DC signal. The fourth component is the bandpass filter, which filters out unuseful data. The specific bandpass we have chosen is based on the purpose of detecting a heartbeat. The fifth component is a notch filter or a band-stop filter with a narrow stopband. It will pass most frequencies unaltered, but attenuates signal that in a specific range to very low levels. Another very important component is output buffer, which data that is ready to be sent is held until the receiver is ready. The last and most common component is the ground that protects subject from surges in electricity

3. ECG software

Basically our program is divided into four main stages -- "setup", "calibrating", "measuring" and "review".

- Setup

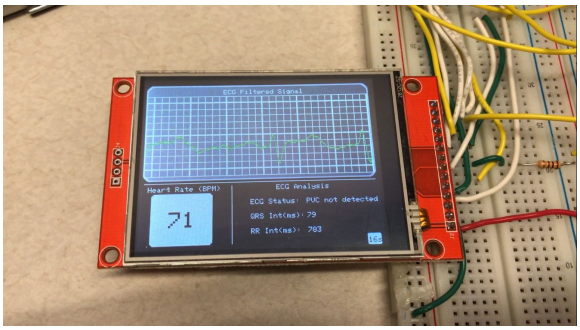
In this section, we initialized ADC, PDB and DMA. They are the cores of our program because they are responsible of reading data from the peripheral. Also in "setup", we set up user interface of our system.

- Calibrating

In this section, we calculate the variance last 100 sample data and keep updating the variance. Once the variance drops to less than 5000, which means the signal is stable, it goes to "measuring" stage. If someone has a large variance, the program goes automatically to "measuring stage" after 6 seconds.

- Measuring

In this section, we monitor user's ECG for 30 seconds and display all the real-time data in the following interface shown below. At the upper half of the LCD display, we drew grids where each grid are 5*5 pixels. The lower left part is user's average heart rate. The lower right part is user's ECG analysis which includes ECG Status, QRS and RR interval. We also put a small 30-second countdown timer at the lower right corner of the screen.



Graph#3. Measuring Page

Since we did all the calculations in the loop, which consumes some time, we used "micros()" to make sure each iteration takes exactly 4 ms so that it syncs with PDB interrupts. The measurement algorithm will be discussed blow.

- "Review"

In this section, users are allowed to review their diagnosis report and ECG data. We designed a menu page which has two buttons. The left one will go to the page of recording of 30-second measured ECG wave. We also added feature to scroll through the recorded data via touch screen. The right button generate the final report of this measurement. The final report shows user's average BPM, QRS, whether bradycardia, tachycardia, PVC, PAC are detected.

Graph#4. Menu Page

- Digital filters and Heart Rate Measurement

The core of ECG software is the implementation of digital filters and peak detection algorithm.

The sampling rate is the rate of an analog-to-digital converter(ADC) samples the ECG, in our system, the sampling rate is 250Hz. Before systems start heart rate measurement, it applied three linear digital filters in software. First filter is low pass filter, which can eliminate noisy data. Next is the differentiator. Its function is to provide the QRS complex slope information. And then, we used squaring function to make all points positive and does nonlinear amplification of the output of the derivative, which is helpful to emphasize the higher frequencies. Last, signal will pass a moving-window integrator that smooth the signal(refer to graph#6).

Heart rate measurement use the processed signal to detect peak with real-time adjustable thresholds. Whole measurement logic follows Pan-Tompkins algorithm to find the locations of R peaks.

SPKI = 0.125 PEAKI + 0.875 SPKI

(if PEAKI is the signal peak)

NPKI = 0.125 PEAKI + 0.875 NPKI

(if PEAKI is the noise peak)

THRESHOLD I1 = NPKI + 0.25 (SPKI - NPKI)

THRESHOLD I2 = 0.5 THRESHOLD I1

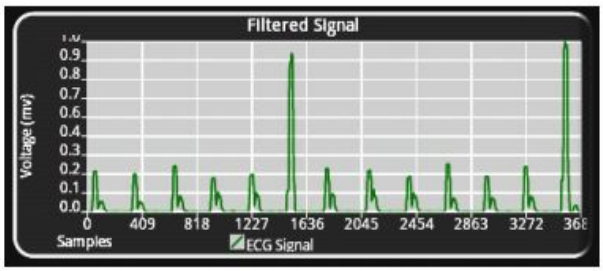
where all the variables refer to the integration waveform:

PEAKI is the overall peak,
SPKI is the running estimate of the signal peak,
NPKI is the running estimate of the noise peak,
THRESHOLD I1 is the first threshold applied, and
THRESHOLD I2 is the second threshold applied.

Graph#5. Equations from Pan-Tompkins[1]

I initialize PEAKI, SPKI, NPKI, THRESHOLD1, THRESHOLD2 equal to 0 at the beginning. Since normal BPM is between 60 to 100, I set up my window size to be 180, which equal to 720ms. And window size will be adjusted base on stabilized BPM.
Window_size = 7500/(BPM/2)
With this function, I can run peak detection accurately without worrying miss peak. In every about 180 samples, system will look for local maximum and record its location. PEAKI is the local max and recalculate all variables again by those equation(refer to graph#5). If this local maximum is greater than THRESHOLD 1, its a peak, otherwise it is a noise data. All these variable will be updated after R peak is detected. And BPM is calculated by time duration between previous R peak and current R peak.
BPM = 7500 * 2 / (RR interval)
As long as system keep running, RR interval will be averaged, so the BPM is relatively stable.

3. QRS detector & Arrhythmia detectors



Graph#6. Output after Moving-window Integrator[1]

After we applied all filters, we can see each heartbeat signal is a pulse in graph with a peak. The QRS complex that we want to detect is the rising edge of the pulse(refer to graph#6). The time duration of rising edge is equal to the width of the QRS complex. Our QRS detector only runs when heartbeat is detected, and it only needs to find start point of rising edge, because we know the endpoint of rising edge is the peak that has been recorded during heart rate measurement. Technically, the width of the QRS complex is between 60 to 120 ms for health heart. As long as measurement is processing, the ECG data will be saved in data buffer. So whenever the heartbeat is detected, I go back 200ms and calculate the slope between two adjacent point (4ms). If the slope suddenly increases, QRS detector will automatically determines this point as start point of rising edge. Since we know the start and end point of rising edge, and based on system runs on 250Hz, we calculate the difference and times 4 to get width of QRS complex in unit of ms. The overall result is what we expected by using this method.

All Arrhythmia detectors have to do with Heart Rate Measurement and QRS detector. First,

Bradycardia and Tachycardia depend on magnitude of BPM. When BPM is greater than 100, global boolean variable Bradycardia in the code will become true, and final report page will display “Bradycardia is detected”. When BPM is less than 60, global boolean variable Tachycardia in the code will become true, and final report page will display “Tachycardia is detected”. And then RR interval is time duration from last R peak to current R peak, which is used to detect PAC. In heart rate measurement process, the time of current R peak and previous R peak are marked. System just need to take previous and current RR interval compare with average of previous five RR interval.

$$\begin{cases} RR_{n-1} < 0.9\left(\sum_{i=n-7}^{n=2} RR_i\right)/5 \\ RR_n > 1.1\left(\sum_{i=n-7}^{n=2} RR_i\right)/5 \; and \; RR_{n-1} \\ \qquad\qquad \leq \left(\sum_{i=n-7}^{n=2} RR_i\right)/5 \; or \\ \dfrac{RR_n}{RR_{n-1}} > 1.2 \\ \text{Where n is the index of tested beat} \end{cases}$$

Graph#7. Equations to Detect PAC[1]

There are three condition to detect PAV, previous RR interval is less than 0.9 times the average, or current RR interval is greater than 1.1 times the average and previous RR interval is less or equal than the average, or current RR interval divide by previous RR interval greater than 1.2 (refer to graph#7). One of three condition exist will be determined as possible PAC. As we know, PAC is a very common occurrence and usually is not serious. Thus, in our system, only when 8 times possible PAC is detected will report “PAC is detected” in final report page.

Future Work

Although we have accomplished the project in an outstanding result, there are still improvements we can make to perfect our ECG detector in both hardware and software. In hardware, the biggest problem is that wires are not stably connected to breadboard, which sometimes causes our ECG detector not working. So, in the future, we want to solder all the wires together and integrate the circuit in a smaller size. In software, when displaying ECG waves in green, it turns out that users can’t distinguish them from grids clearly. To solve this problem, we will change the color of waves and grids with greater contrast.

Conclusion

We tested our project by repeatedly measuring a person’s BPM and the results for each time did not show great variance. The measurement also showed difference from person to person. For example, my average BPM is around 68 while my partner is around 75. Also, when measured after we climbing stairs, the BPM increased significantly. The above also applies to other measurements in our detector. Therefore, we conclude that our project is functional and reliable.

In order to implement all the features we mentioned, we had to declare many variables and arrays to save data. This lead to a high consumption of memory space in our program. I think this is the biggest trade off in our design.

Reference

[1]GholamHosseini, Hamid. *Automated diagnosis of heart arrhythmias: towards a clinically useful computer-Aided diagnosis system for heart rhythm abnormalities*. LAP Lambert Academic Publishing, 2012.