

CHAPTER 3 – RESEARCH METHODS

3.1 Complete System Design

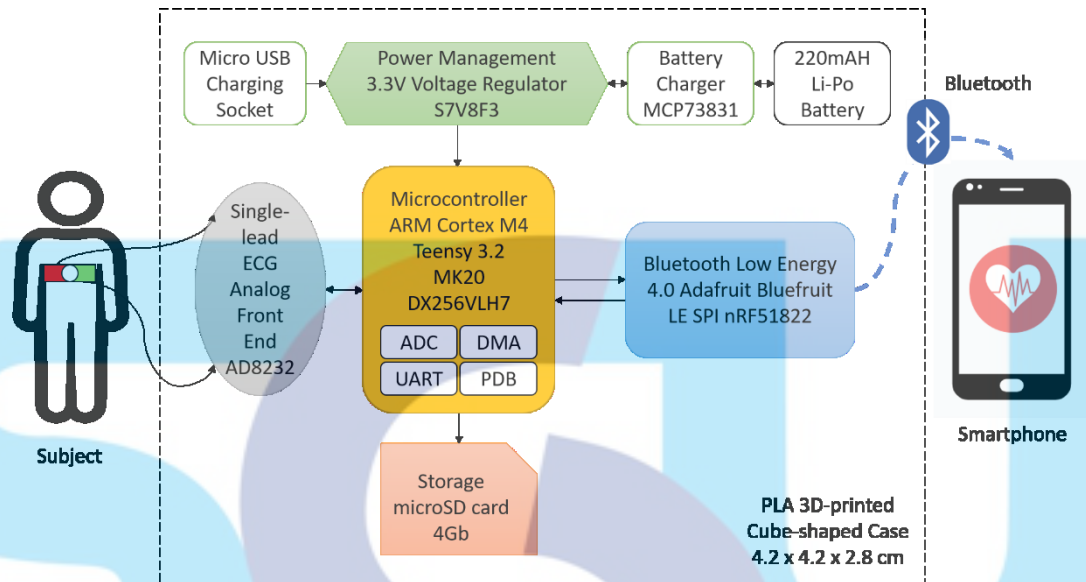


Figure 17. General Block Diagram of the Wearable Electrocardiograph.

This thesis aims to detect various cardiac arrhythmias based on ECG Lead I using a Wearable ECG device. The hardware used is a product of the previous study (Iskandar, 2017). The main system, shown in Figure 17, starts from AD8232 single lead ECG analog front end with passive electrodes as a sensor that acquires bio-potential signals of the heart from the subjects. The signals are then processed using a Teensy 3.2 development board with an ARM Cortex-M4 microprocessor. The processes include filtering, computing, detecting, and recording. These processes use the peripheral devices and computing power of the microprocessors simultaneously. Power management on this system includes a 3.3 V step-down voltage regulator and a Li-Ion battery recharging module. Finally, the outputs are transmitted out using Bluetooth 4.0 wireless connectivity to smartphone applications. The Adafruit Bluefruit Low Energy (BLE) SPI Friend was chosen for its low power consumption. This module comes with an application available for most smartphone operating system (OS) named Adafruit Bluefruit, which operations are based on serial peripheral connection. Another

smartphone application named nRF Toolbox is also capable of providing the user interface of heart rate measurement, complete with a chart, using a GATT connection.

3.2 ECG Signal Acquisition

As mentioned before, the acquisition process begins with two passive electrodes – attached on a chest strap – acquires the biopotential heart signals. Afterwards, the signals travel to an integrated circuit (IC) to be amplified and filtered from noises. The AD8232 has a built-in bio-instrumentation amplifier with analog filters. The microprocessor then conditions the signal to be readable before sending it to an analog-to-digital converter (ADC), with 10-bit conversion and 200 sample-per-second sampling rate.

3.3 Embedded Software Design

This section explains the real-time data processing design; consists of biopotential signal acquisition and signal processing, that runs simultaneously in multitasking software architecture. A scheduler schedules the task interval and determines the execution priority orders. The embedded system employs the Programmable Delay Block (PDB), Direct Memory Access (DMA), Analog-to-Digital Converter (ADC), Interrupt Service Routine (ISR), Serial Peripheral Interface (SPI), and Program Flash. The configurations of each resource are based on the microprocessor reference manual. The process, shown in Figure 18, begins with triggering the PDB with a delay of 5 ms so that the ADC sampling would be at 200 Hz. The acquired signal goes through the instrumentation amplifier and directly stored to the DMA on-board. The signal is stored in an array of buffer with the value of 10-bit digital conversion indexed using the sample number. Into the Pan-Tompkins algorithm, a low-pass filter is used to remove instrumentation amplifier noise. Then, the derivative and squaring functions convert the signal to a positive signal curve that response to the change in R-wave. Lastly, a moving window integration defines a window of a heartbeat.

Every 3 seconds, the buffer is passed to a subprocess without interfering the continuity of the acquisition process. The subprocess duration is less than 3 seconds. Firstly, the

ECG signal is passed to the R-wave peak detection algorithm to find the peak of QRS positive curve and define it as R-peak. There is a fine-tuning to adjust to filter delay.

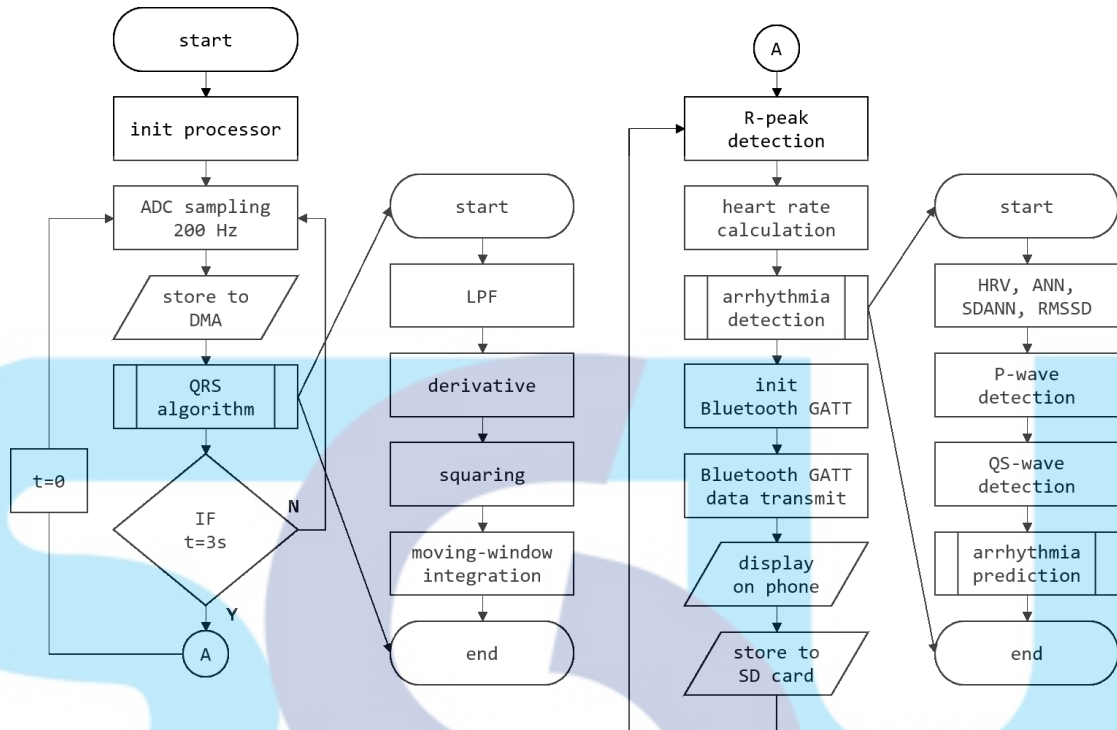


Figure 18. Software design flowchart.

Heart rate is then calculated from a distance between R-peaks and multiplied by sampling rate so that the R-R interval is in *ms* — one RR-interval is calculated as one heartbeat. Heart rate (*BPM* = beats per minute) is the average of R-R interval within 60 seconds.

Arrhythmia is an irregular heart rate. The standard deviation of the R-R interval is calculated: the higher the standard deviation, the more irregular, the higher possibility of arrhythmia — signs of many cardiac arrhythmias, such as irregular heartbeats, PVC, PAC, SVT and VT.

P-wave detection algorithm recognises P-wave by tracing backwards from R-peak with a range of half normal PQ-interval duration. Then, the Q-wave detection algorithm, improved from the Pan-Tompkins algorithm and the algorithm developed by Iskandar, combines the tracing methods and distance weight using additional cosine wave.

The arrhythmia recognition algorithm developed in this research is as follows. PVC is recognised when the QS-width is far more than 0.10 s , P-wave present, and medium *RMSSD* value. PAC is recognised when the QS-width is far less than 0.10 s , P-wave present, and medium *RMSSD* value. SVT is recognised when the QS-width is less than 0.10 s , P-wave absent, heart rate is above 100 bpm (tachycardia), and high *RMSSD* value. AFib is recognised when the QS-width is less than 0.10 s , P-wave present, and high *RMSSD* value. VT is recognised when the QS-width is far more than 0.10 s , P-wave hidden, heart rate is above 100 bpm (tachycardia), and low *RMSSD* value. Irregular heartbeat is recognised when *RMSSD* value is medium to high. Asystole is recognised when the heart rate is zero.

The heart rate value is shown in real-time using smartphone apps. NRF toolbox shows heart rate and chart so that any arrhythmia can be indicated on the inconstant chart. Bluefruit app shows heart rate and QRS width and the CA recognition, if possible, including the diagnosis/the type.

3.3.1 Programmable Delay Block (PDB)

The desired clock speed is 24 MHz to generate ADC sampling rate of 200 Hz . The period of the programmable delay block counter is defined by the microprocessor clock speed and a prescaler to achieve a continuous 5 ms timer for $200\text{ sample-per-second}$ sampling rate. The period calculation is shown in Equation (12).

$$\begin{aligned} PDB_PERIOD &= F_BUS \times \frac{1}{PDB_SC_PRESCALER} \times \frac{1}{PDB_SC_MULT} \times \frac{1}{Freq} \quad (12) \\ &= 24\,000\,000 \times \frac{1}{8} \times \frac{1}{10} \times \frac{1}{200} = \mathbf{1500} \end{aligned}$$

So, the result is used to define the PDB firmware `PDB_PERIOD` (1500).

3.3.2 Analog-to-Digital Converter (ADC)

The ADC converts the analog signal into a digital signal of 10-bit data. It has four registers that need to be configured based on the manual guide. The configurations control the mode of operation, clock source, clock division, and low power usage. The ADC must be calibrated before usage; it has a self-calibrating function that is connected to the DAC and make a stable condition to maintain accuracy.

3.3.3 Direct Memory Access (DMA)

Direct memory access or DMA holds a major role in processing the data in real-time. DMA transfers the data seamlessly between the main memory and the peripheral devices such as the ADC without interfering with the CPU system so that the CPU will not be overloaded and still able to handle other tasks. DMA can only hold the maximum of 16 samples at a time, or in this research, we needed 600 samples every 3 seconds. Therefore, the buffer in the DMA is overloaded, and the data is held in the memory. Despite the buffer size, DMA is useful for its work without disturbing the CPU workload.

3.3.4 Serial Communication Port

We made several operating modes for the device; one of them is the serial mode. For the Arduino serial monitor/plotter display, it is needed to code `Serial.begin(115200)` and `Serial.print()` or `Serial.println()` to transfer data to the communication channel. Meanwhile, for the Adafruit Bluefruit application plotter display, the code needed is `ble.print()` or `ble.println()` and of course a series of Bluetooth initiating process.

3.3.5 Bluetooth GATT

By using GATT protocol, communication is possible among Bluetooth Low Energy devices. The communication demands a characteristic consisting of a type (UUID), a value, and a set of properties. The protocols used in this research are heart rate service, heart rate measurement characteristic, and body sensor location characteristic. The heart rate measurement with the characteristic of body sensor location value set to 1 (chest). Other options for the value are 2 (finger) and 3 (wrist). The command is sent in the form of AT command using hexadecimal numbers. The heart rate value is shown in

real-time using smartphone apps. NRF toolbox shows heart rate and chart so that any arrhythmia can be indicated on the inconstant chart. Bluefruit app shows heart rate and QRS width and the CA recognition, if possible, including the diagnosis/the type.

3.4 ECG Signal Conditioning

3.4.1 Fast Fourier Transform

FFT analyses and converts a signal from the time domain to the frequency domain. FFT is useful to analyse the spectrum of a signal and to determine at which frequency a noise signal interferes with the desired ECG signal. After acquiring the ECG signal, through the filters and amplifier, some noises still can be found, which commonly is electrical noise of 50-60 Hz. Algorithm 5 is the algorithm to employ the FFT to analyse interfering noises.

Algorithm 5. Fast Fourier Transform (FFT) spectrum analysis.

```
Result: ECG signal plot in frequency domain.
Read the ECG signal file (.csv);
Specify the parameters of the signal:
    Sampling frequency,
    File length;
Convert the sampling number to its position in time domain (ms);
Apply the FFT;
Compute the two-sided spectrum;
Compute the single-sided spectrum;
Define the frequency domain;
Plot the single-sided power spectrum;
```

3.4.2 IIR Notch Filter

Cleaner ECG signals can be obtained by adding an IIR notch filter to filter electrical noise or power line interference of 50 *Hz* in the acquired ECG signal. Algorithm 6 removes the 50 *Hz* electrical noise to obtain cleaner ECG.

Algorithm 6. Notch filter for electrical noise removal.

```
Result: Filtered ECG signal after the removal of the electrical noise.  
Read the ECG signal file (.csv);  
Specify the parameters of the signal:  
    Sampling frequency;  
Implement IIR notch filter:  
    Set the notch location,  
    Set the bandwidth at -3 dB point,  
    Return the numerator coefficients and denominator coefficients of  
    the digital notching,  
    Filter the original signal using the coefficients;  
Save the filtered signal to .csv file;  
Plot the filtered signal;
```

SWISS GERMAN UNIVERSITY

3.5 ECG Signal Feature Extraction Algorithm

3.5.1 QRS-wave Recognition

Pan-Tompkins QRS detection algorithm is applied onto the signals acquired from the wearable ECG device, which already processed before to be in the form of raw ADC digital format. By detecting the occurrence of the QRS-complexes in every 3 seconds, the position of every heartbeat and the R-R interval can be obtained. Thus, the heart rate with the unit of *BPM* (beats per minute) can be calculated/recognised.

Cardiac arrhythmia, or irregular heartbeat, can be detected by calculating the standard deviation of these heart rate averages. The more the standard deviation, the more significant the variance, and the higher the possibility of cardiac arrhythmia. Next, the heart rate is sent to the smartphone via Bluetooth as the smartphone application is drawing a heart rate graph over time. All the acquired and processed data, including indications if any, are stored in a microSD card.

The original algorithm uses a band-pass filter consisting of a low-pass filter and a high-pass filter, but only its low-pass filter is operated in this research to remove artefacts and 50 *Hz* power interference noises due to the filter delays. The addition of a high-pass filter would add more delays to the signal. The second-order low pass filter used has a cutoff frequency of 11 *Hz* and gain of 36.

3.5.2 P-wave Recognition

P-wave detection algorithm, as mentioned in Subsection 2.4.3, is a part of the algorithm in this thesis, thus used as it is. The output/result from this algorithm will further be analysed together with the other parameters to create a better cardiac arrhythmia recognition.

3.5.3 Q-wave and S-wave Recognition

In this research, we tried to combine the algorithm developed from Kim-Lee formula and the Pan-Tompkins algorithm to locate the Q-wave and S-wave locations better as well as calculating the QRS-onset and offset. Since some of the methods overlap, the highlight from Kim-Lee formula is the distance weight method. Thus, obtaining QS-width from the calculation that can be used for cardiac arrhythmia recognitions. Algorithm 7 shows the steps of Kim-Lee formula implementation in an algorithm.

Algorithm 7. Q-wave and S-wave Detection Algorithm.

```

Result:  $Q_{wave}$  and  $S_{wave}$  locations and occurrence.
Read the RR buffer;
for every  $RR_{buffer}$  do
| Let  $R_{left} = RR_{buffer}$  AND  $R_{right} = next\ RR_{buffer}$ ;
| Calculate distance  $D = \frac{|(y_R - y_L)x_i - (x_R - x_L)y_i + x_R y_L - y_R x_L|}{\sqrt{(y_R - y_L)^2 + (x_R - x_L)^2}}$ ;
| Calculate weight  $W = \cos(2\pi \frac{x_i - x_L}{x_R - x_L})$ ;
| Calculate distance weight  $WD = W \times D$ ;
| if  $WD$  reaches the  $WD_{min}$  between  $2 R_{peak}$  then
| | Set the point as a  $i_{stop}$ ;
| | if  $WD$  reaches the  $WD_{max}$  to the right of  $i_{stop}$  then
| | |  $Q_{wave}$  found and stored;
| | end
| | if  $WD$  reaches the  $WD_{max}$  to the left of  $i_{stop}$  then
| | |  $S_{wave}$  found and stored;
| | end
| end
| Keep on tracing;
end

```

3.5.4 QS-width Calculation

After locating all locations of found Q-waves and S-waves, we can start calculating every QS-interval by subtracting the sample number indicating the location of S-wave by the sample number indicating the location of Q-wave within one ECG heartbeat. The algorithm implementation is shown in Algorithm 8.

Algorithm 8. QS-width calculation.

```

Result:  $QS_{width}$  of every heartbeat.
Read the  $Q_{buffer}$  and  $S_{buffer}$ ;
for every  $RR_{buffer}$  do
|  $QS_{width} = S_{buffer} - Q_{buffer}$ ;
| Store  $QS_{width}$ ;
end

```

3.6 Cardiac Arrhythmia Detection Algorithm

The identification of the cardiac arrhythmias will be based on the flowchart in Figure 19, as a continued detailed process from the firmware flowchart in Figure 18. The arrhythmia recognition algorithm developed in this research is as follows. PVC is recognised when the QS-width is far more than 0.10 s, P-wave present, and medium $RMSSD$ value. PAC is recognised when the QS-width is far less than 0.10 s, P-wave present, and medium $RMSSD$ value. SVT is recognised when the QS-width is less than 0.10 s, P-wave absent; heart rate is above 100 bpm (tachycardia), and high $RMSSD$ value. AFib is recognised when the QS-width is less than 0.10 s, P-wave present, and high $RMSSD$ value. VT is recognised when the QS-width is far more than 0.10 s, P-wave hidden, heart rate is above 100 bpm (tachycardia), and low $RMSSD$ value. Irregular heartbeat is recognised when $RMSSD$ value is medium to high. Asystole is recognised when the heart rate is zero. The structure of this algorithm is multiple if-else/decisions, as shown in Figure 19.

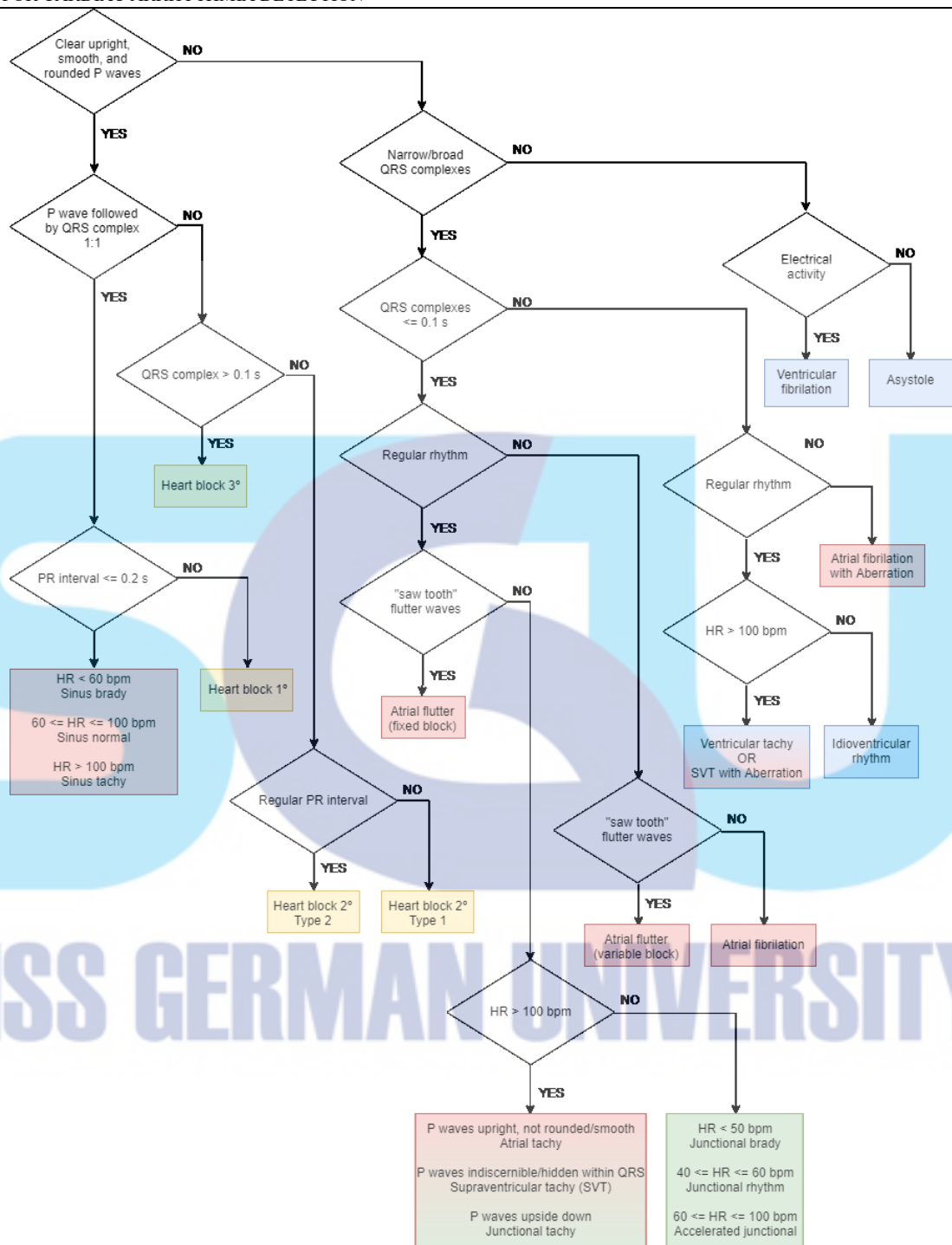


Figure 19. Cardiac rhythm analysis flowchart.

Conclusion

The highlight of the research is the QS-width algorithm developed under this study, to be tested in Chapter 4. The algorithm combines an adaptation of Kim-Lee distance weight formula for better Q-wave and S-wave recognition and Pan-Tompkins algorithm for R-peak recognition. The obtained locations of Q-wave and S-wave is further processed to obtain the QS-interval, that is an essential parameter for recognising several types of cardiac arrhythmias.

