Real-time Heart Monitoring and ECG Signal Processing

Fatima Bamarouf, Claire Crandell, and Shannon Tsuyuki

Advisors: Drs. Yufeng Lu and Jose Sanchez

Department of Electrical and Computer Engineering
Bradley University

April 26, 2016

Contents

Introduction and Overview

- Problem Description
- Objectives
- Constraints
- System Block Diagram
- Scope
- Division of Labor

Problem Description

- Arrhythmias
  - Are irregular heartbeats caused by defective electrical signals in the heart [1]
  - Include premature ventricular contractions (PVCs)
  - PVCs may lead to ventricular tachycardia (VT)

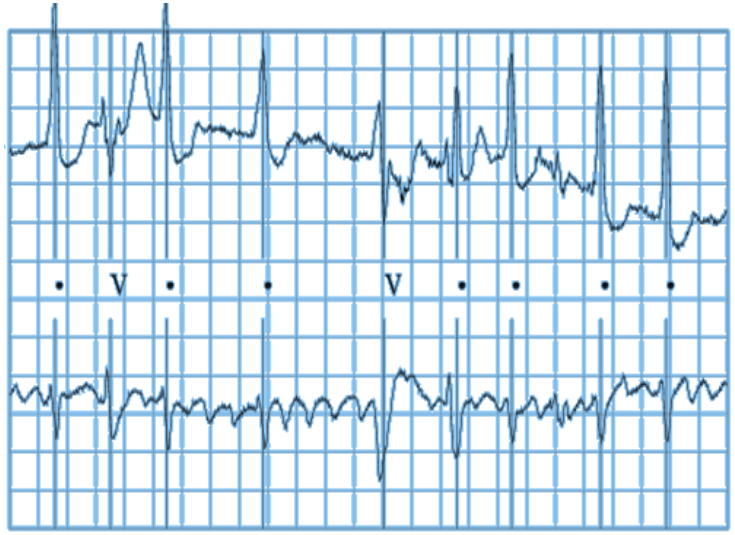
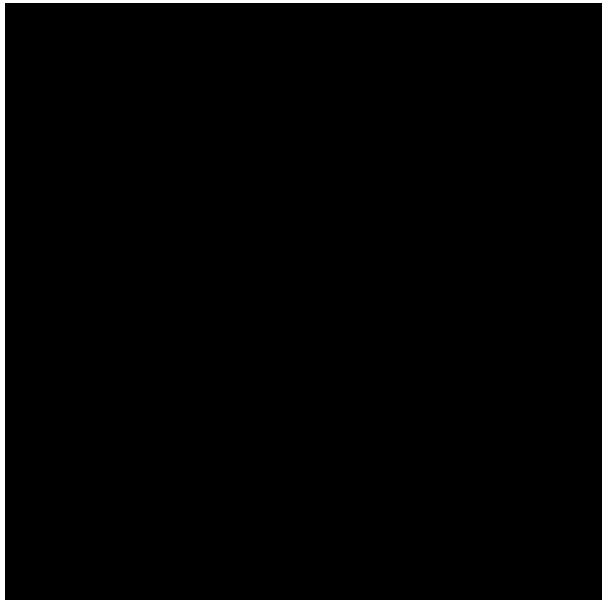
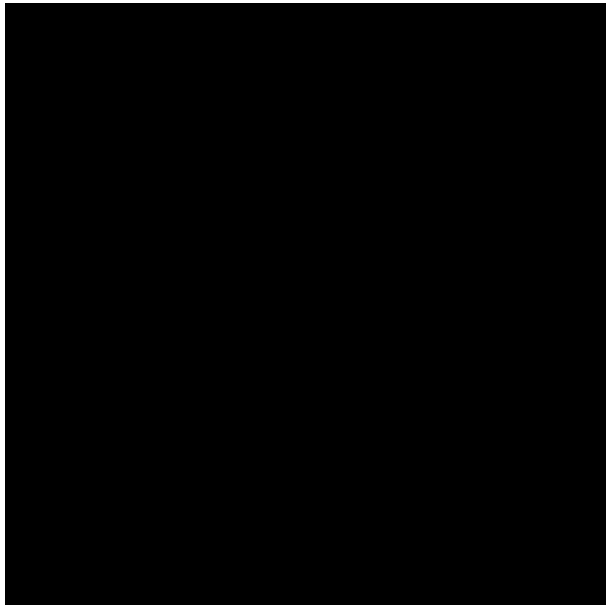


Figure 1. Electrocardiogram with “V” labels for PVCs [2]

Problem Description

- Normal vs. Arrhythmic Heart Rhythms





Source:
http://watchlearnlive.heart.org/ [1]

Problem Description

- An electrocardiogram (ECG) describes the heart's electrical activity
- An ECG can be recorded using a Holter monitor or event monitor

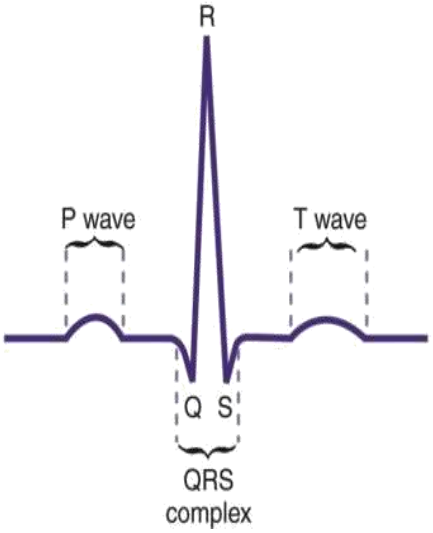


Figure 2. Features of a normal ECG [3]

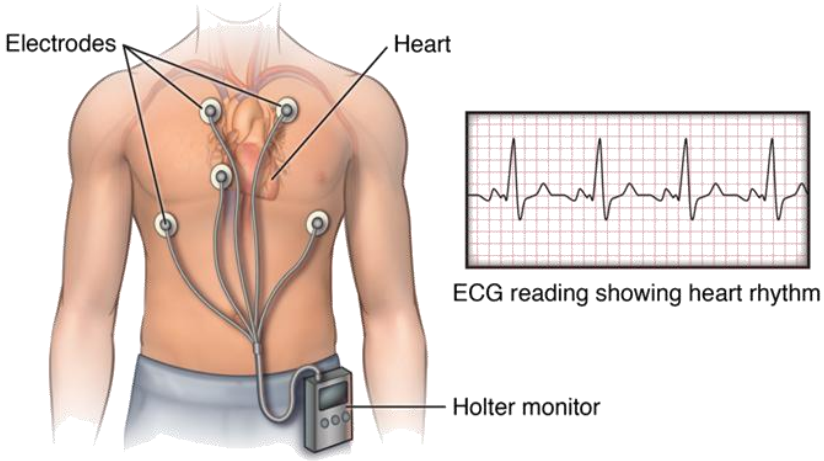


Figure 3. Holter monitor with ECG reading [4]

Problem Description

- Holter and event monitors are limited in functionality
  - Utilize some in-platform signal processing for diagnostic assistance
  - Must perform some signal processing offline
  - Are unable to address medical issues in real time

Objectives

- Develop a low-power, stand-alone embedded system for continuous heart monitoring that will
  - Process ECG data in real time
  - Detect PVCs accurately and consistently
  - Alert the patient's doctor wirelessly of ventricular tachycardia

Constraints

- Real-time ECG signal processing
- On-board signal processing computations
- Battery-powered functionality

System Block Diagram

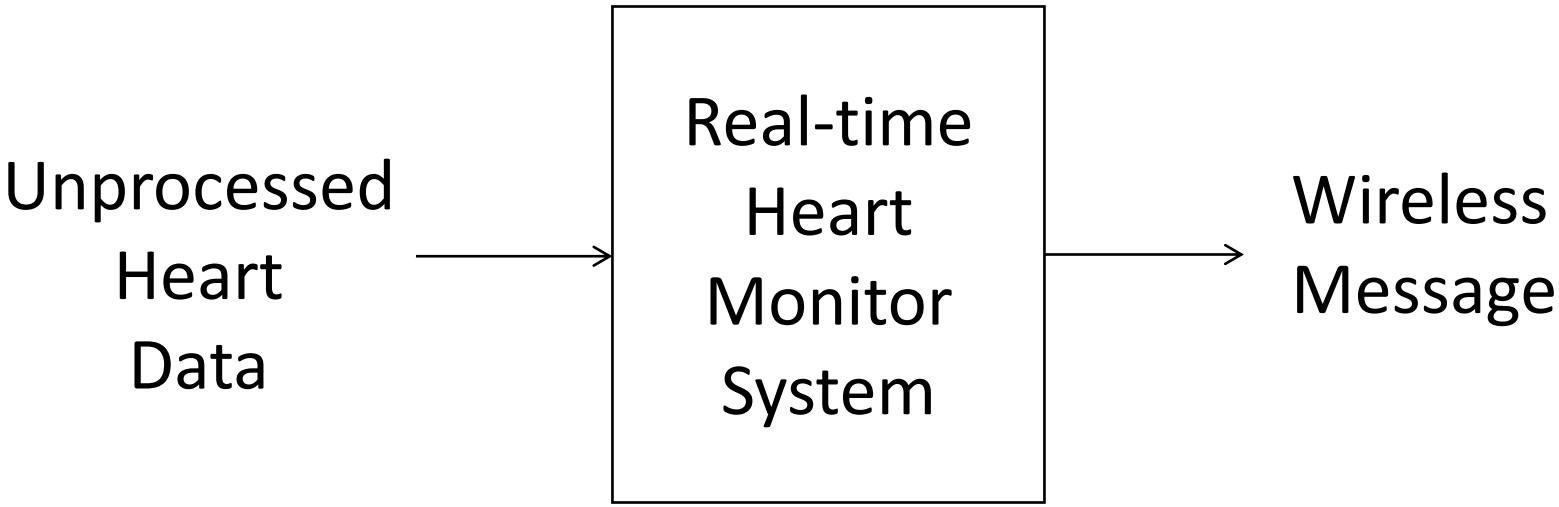


Figure 4. Overall heart monitoring system diagram

Scope

TABLE I. SCOPE OF HEART MONITORING SYSTEM

In Scope	Out of Scope
ECG signal processing	Electrode interfacing, battery circuit
PVC and VT detection	Detection of other types of cardiac arrhythmias
High-level wireless communication	Security issues (encryption, data integrity, etc.)

Division of Labor

- MATLAB Simulation (PVC detection)
  - Shannon/Fatima
- C Programming (PVC detection)
  - Claire/Shannon
- Wi-Fi Communication
  - Fatima/Claire/Shannon

Contents

Functional Requirements

TABLE II. FUNCTIONAL REQUIREMENTS (ALGORITHMS)

Functional Requirement	Specification	Specification Met?
Filtering and/or normalizing the heart data	**The filtering/normalization must ensure that the heart data is compatible with the QRS, PVC, and VT detection functionality.**	**Yes**
Identifying each QRS complex	**The QRS detection algorithm must return the R-peak indices for use in the PVC detection subsystem.**	**Yes**
Classifying each QRS complex as PVC or non-PVC	The PVC detection algorithm must return the PVC indices for use in the VT detection subsystem.	Yes

Problem Approach

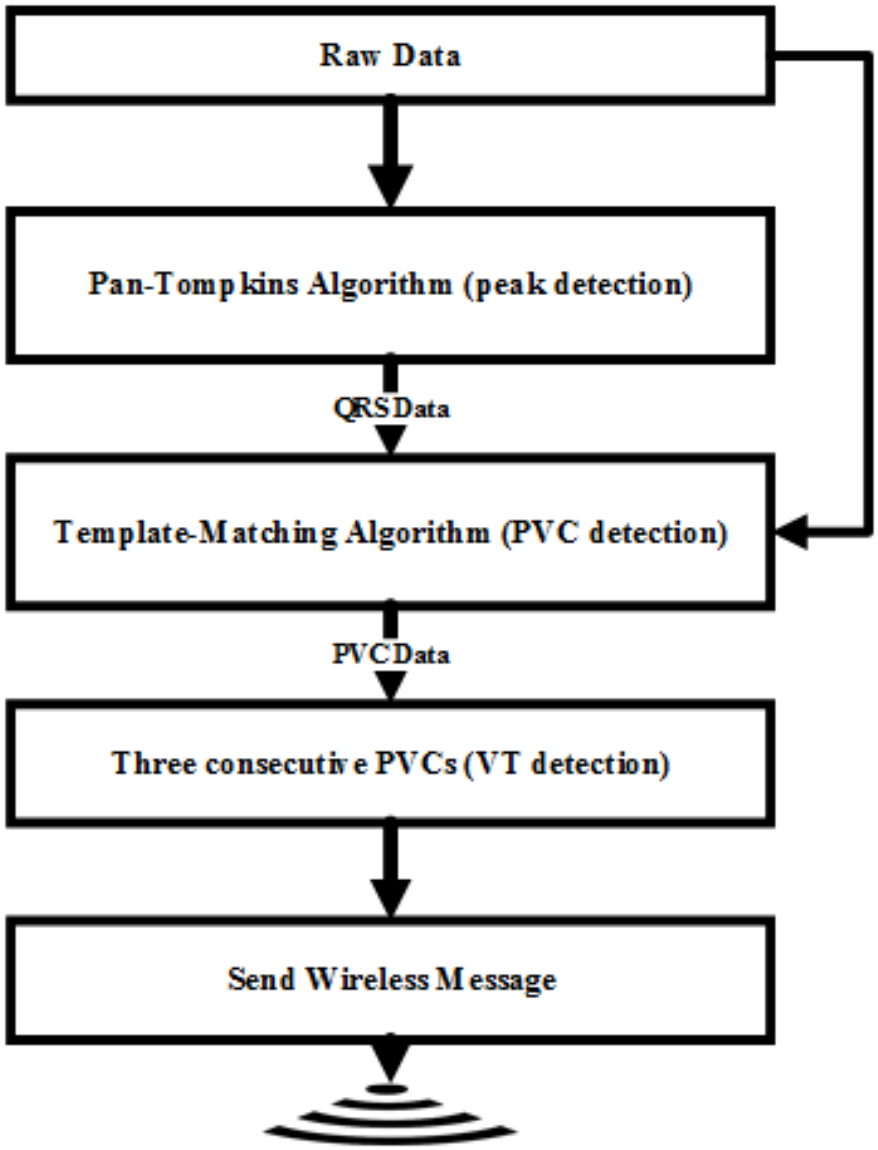


Figure 5. High-level flowchart for algorithms

Pan-Tompkins Algorithm [5]

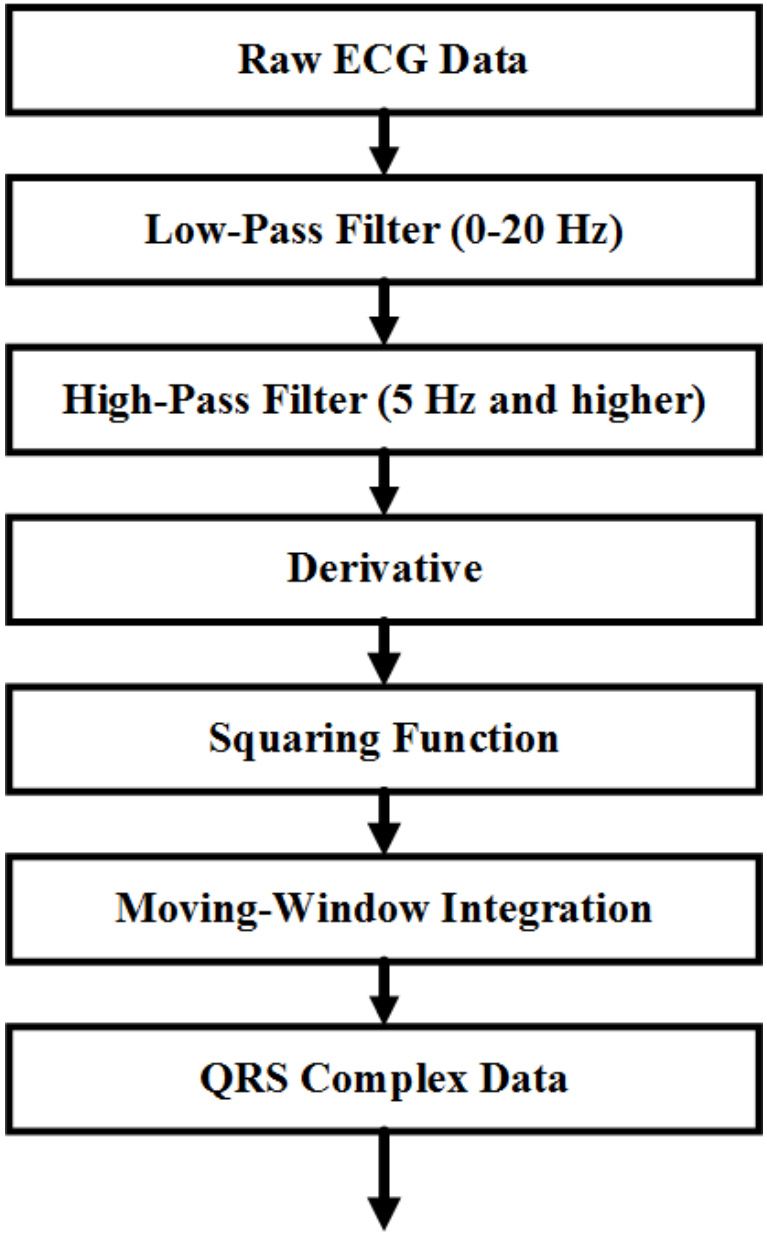


Figure 6. Flowchart for Pan-Tompkins algorithm

Functional Requirements

TABLE III. FUNCTIONAL REQUIREMENTS (ALGORITHMS)

Functional Requirement	Specification	Specification Met?
Filtering and/or normalizing the heart data	The filtering/normalization must ensure that the heart data is compatible with the QRS, PVC, and VT detection functionality.	Yes
Identifying each QRS complex	The QRS detection algorithm must return the R-peak indices for use in the PVC detection subsystem.	Yes
Classifying each QRS complex as PVC or non-PVC	**The PVC detection algorithm must return the PVC indices for use in the VT detection subsystem.**	**Yes**

Template Generation Algorithm [6]

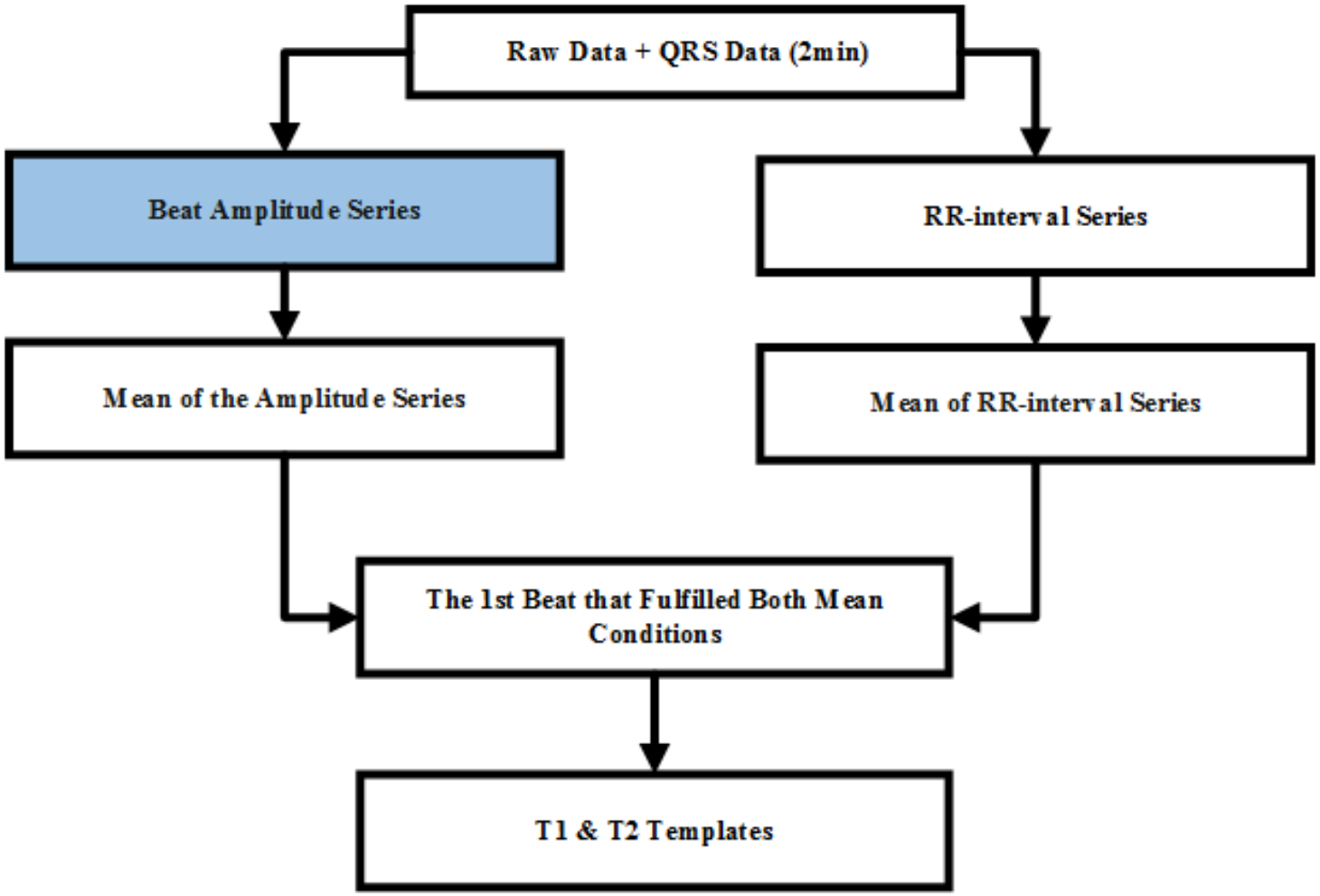


Figure 7. Flowchart for T1 and T2 generation

Beat Amplitude Series

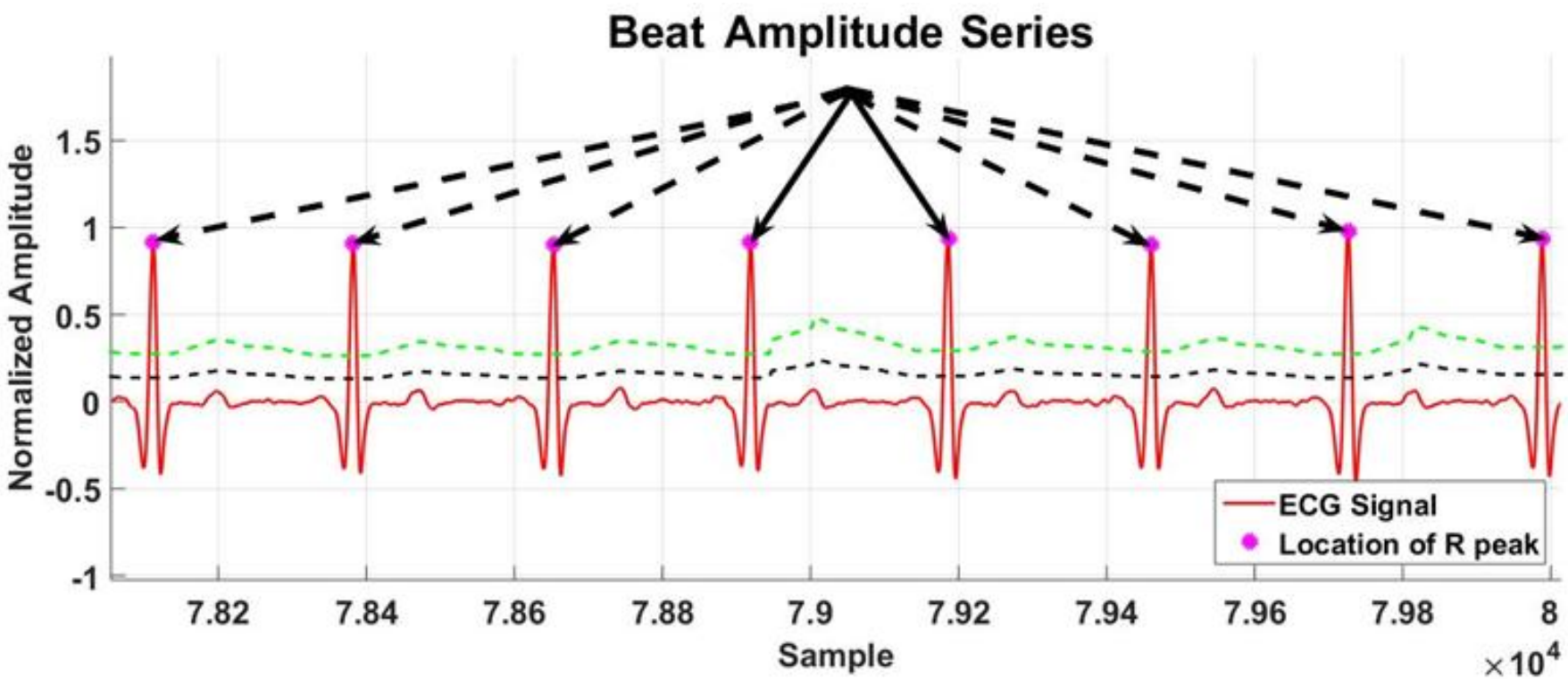


Figure 8. MATLAB plot of QRS detection results.

Template Generation Algorithm

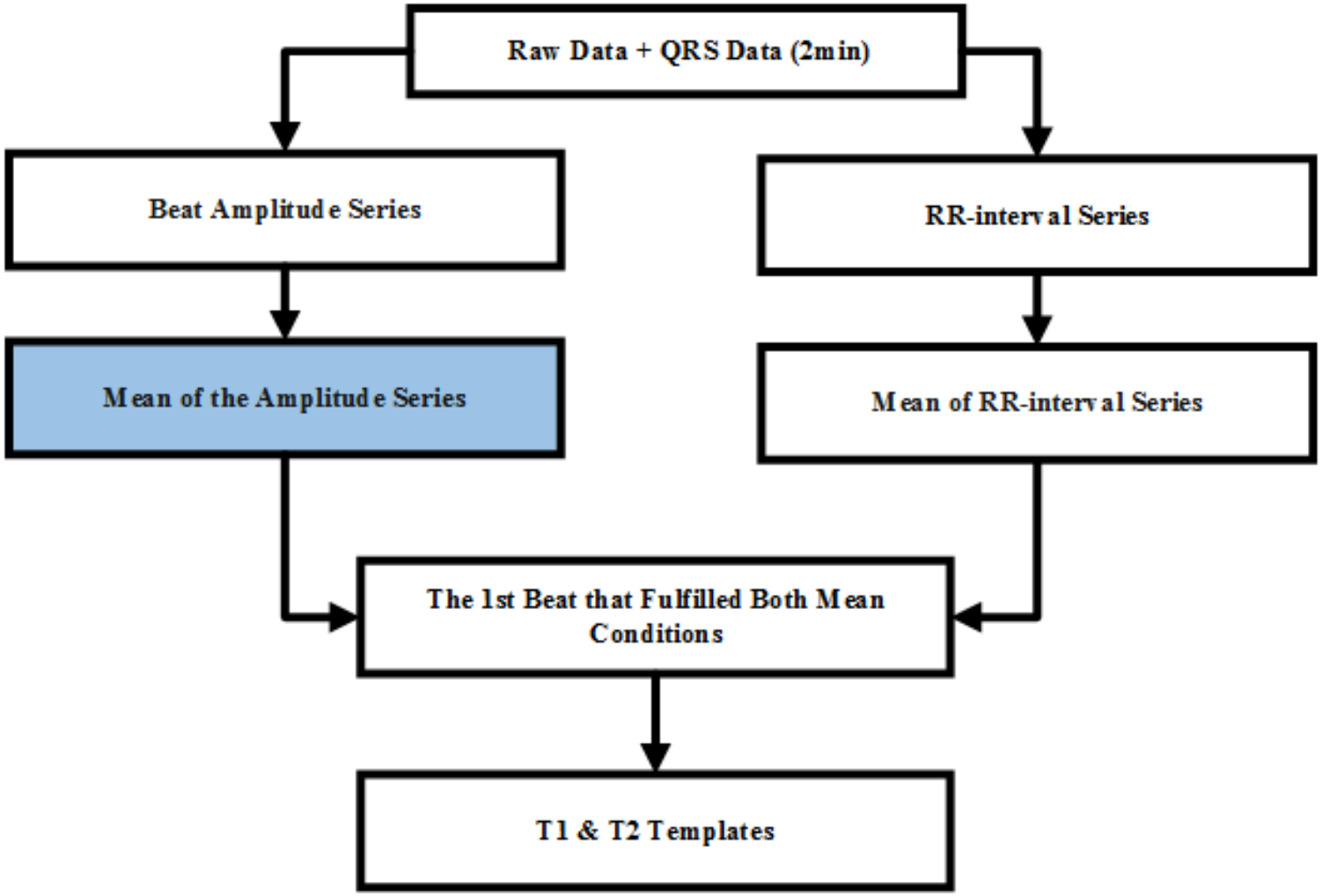


Figure 9. Flowchart for T1 and T2 generation

Template Generation Algorithm

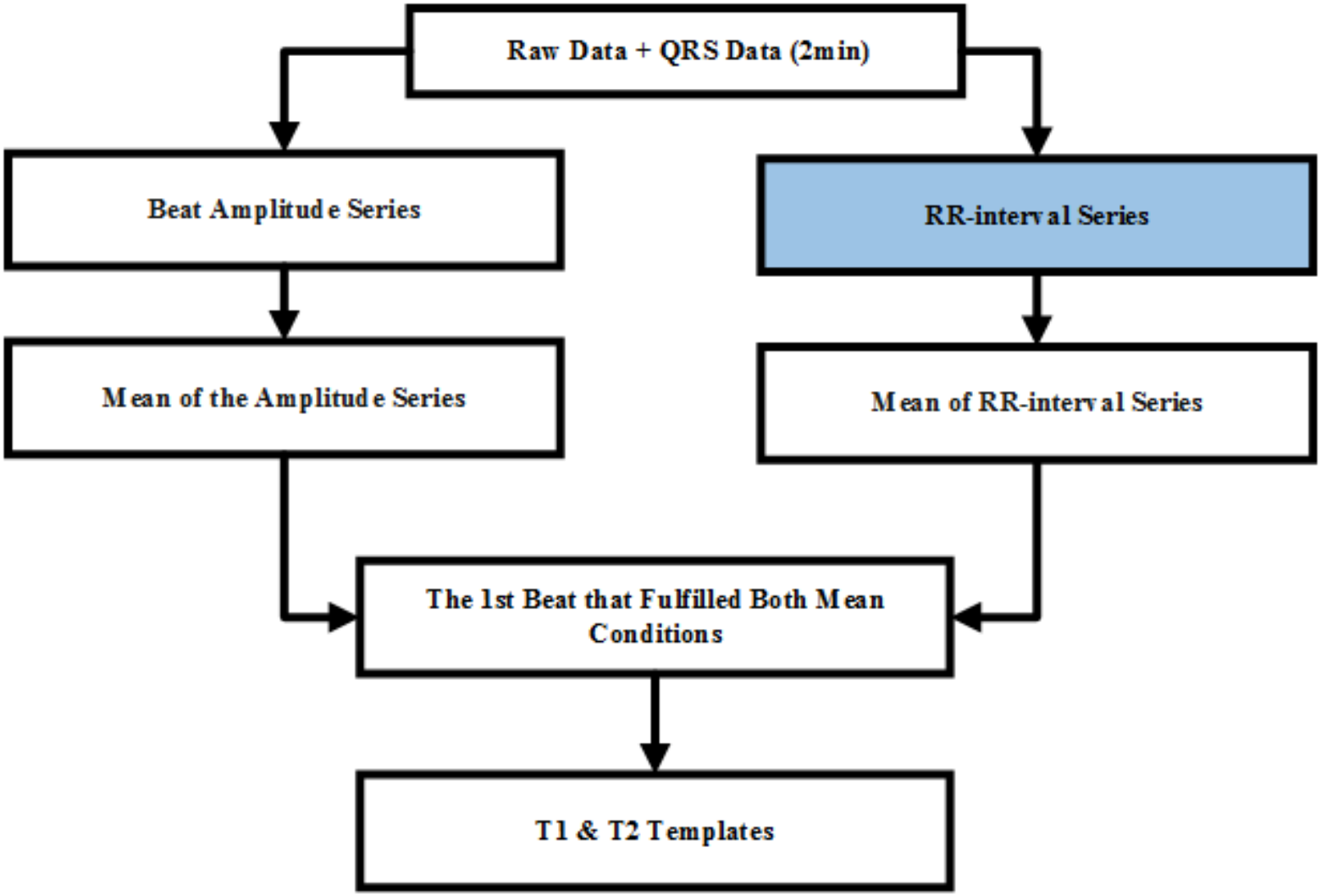


Figure 10. Flowchart for T1 and T2 generation

Data between Two R-Peaks

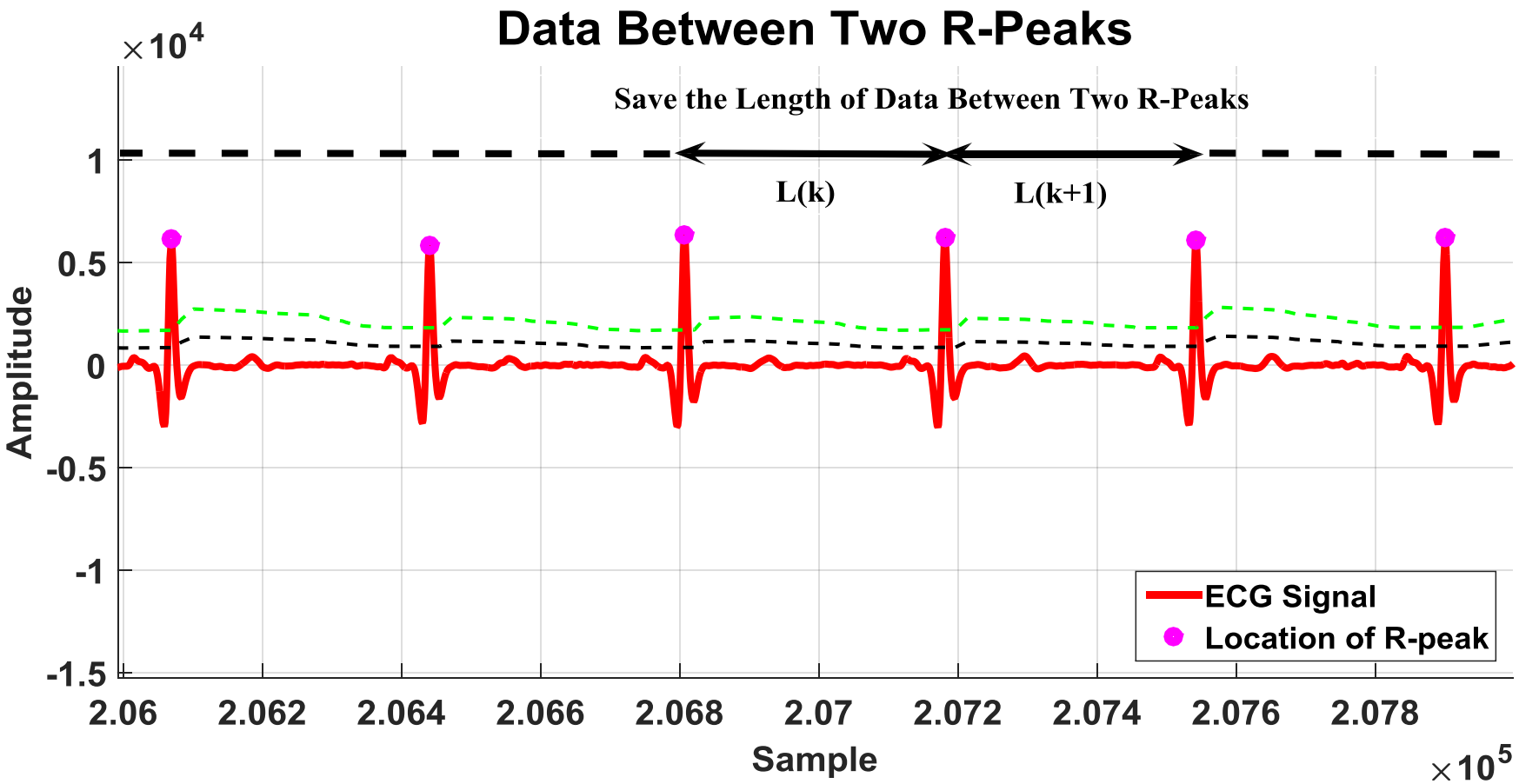


Figure 11. MATLAB plot of QRS detection results.

Template Generation Algorithm

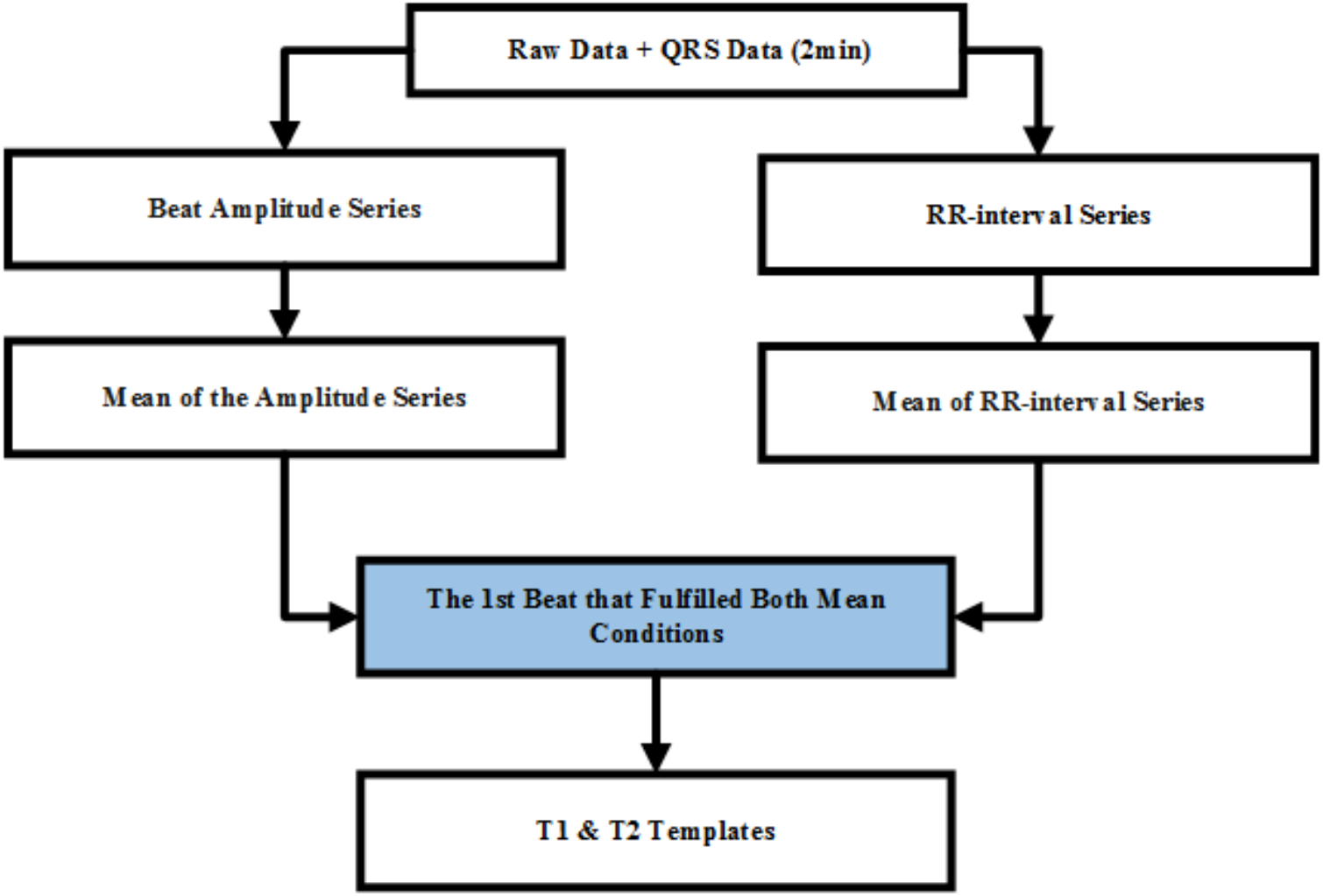


Figure 12. Flowchart for T1 and T2 generation

Template Generation Algorithm

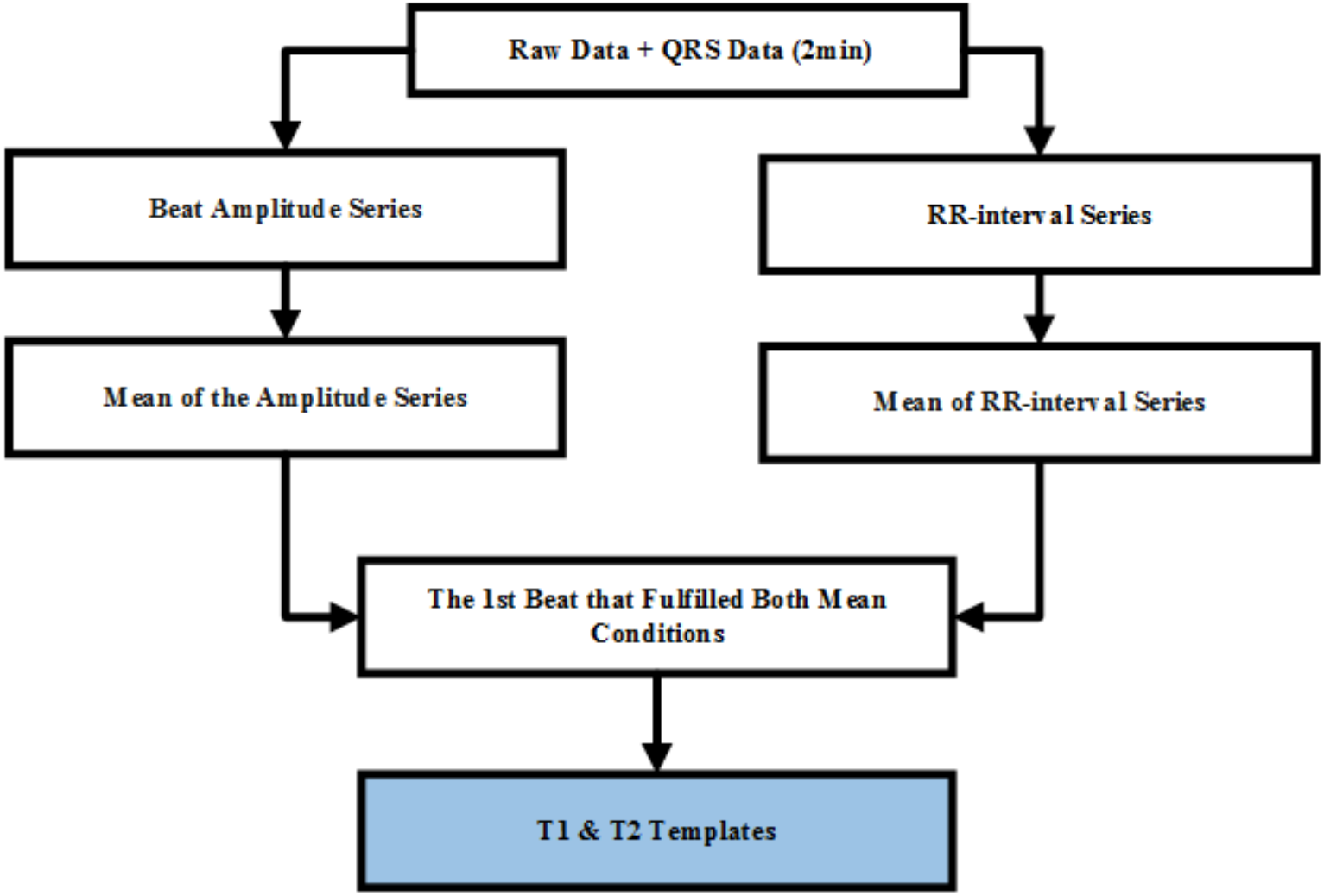


Figure 13. Flowchart for T1 and T2 generation

Wavelet Transform Algorithm [7]

- QRS (T1 template) detection method
- Zero crossings of quadratic spline wavelet transform used to find QRS onset and offset

Wavelet Transform Algorithm [8]

- First-order wavelet transform:

$$y(\mathrm{n}) = \left(\frac{1}{1.5}\right) * (-2x(\mathrm{n}) + 2x(\mathrm{n}-1))$$

Wavelet Transform Algorithm

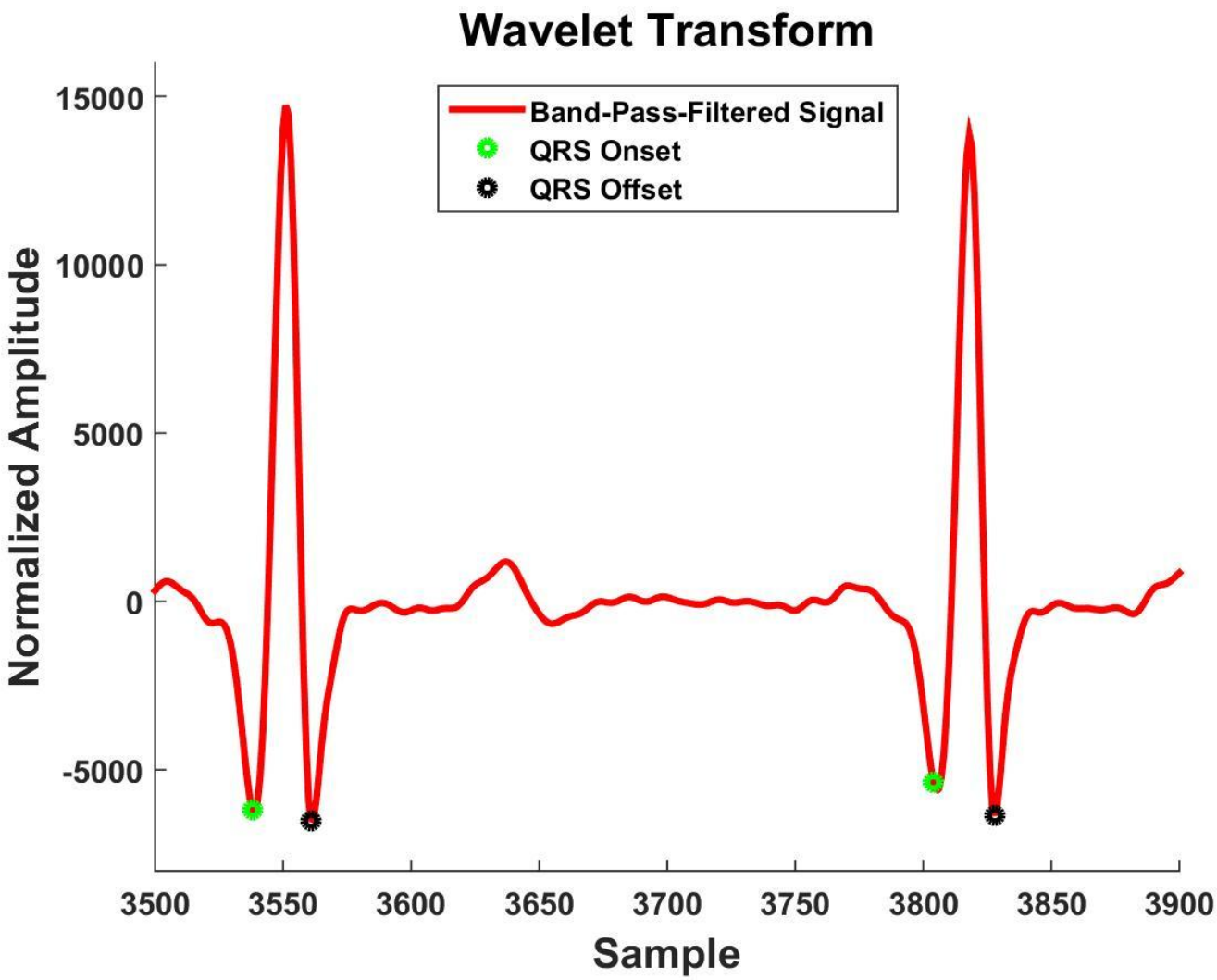


Figure 14. QRS onsets and offsets detected using the wavelet transform

Templates T1 and T2

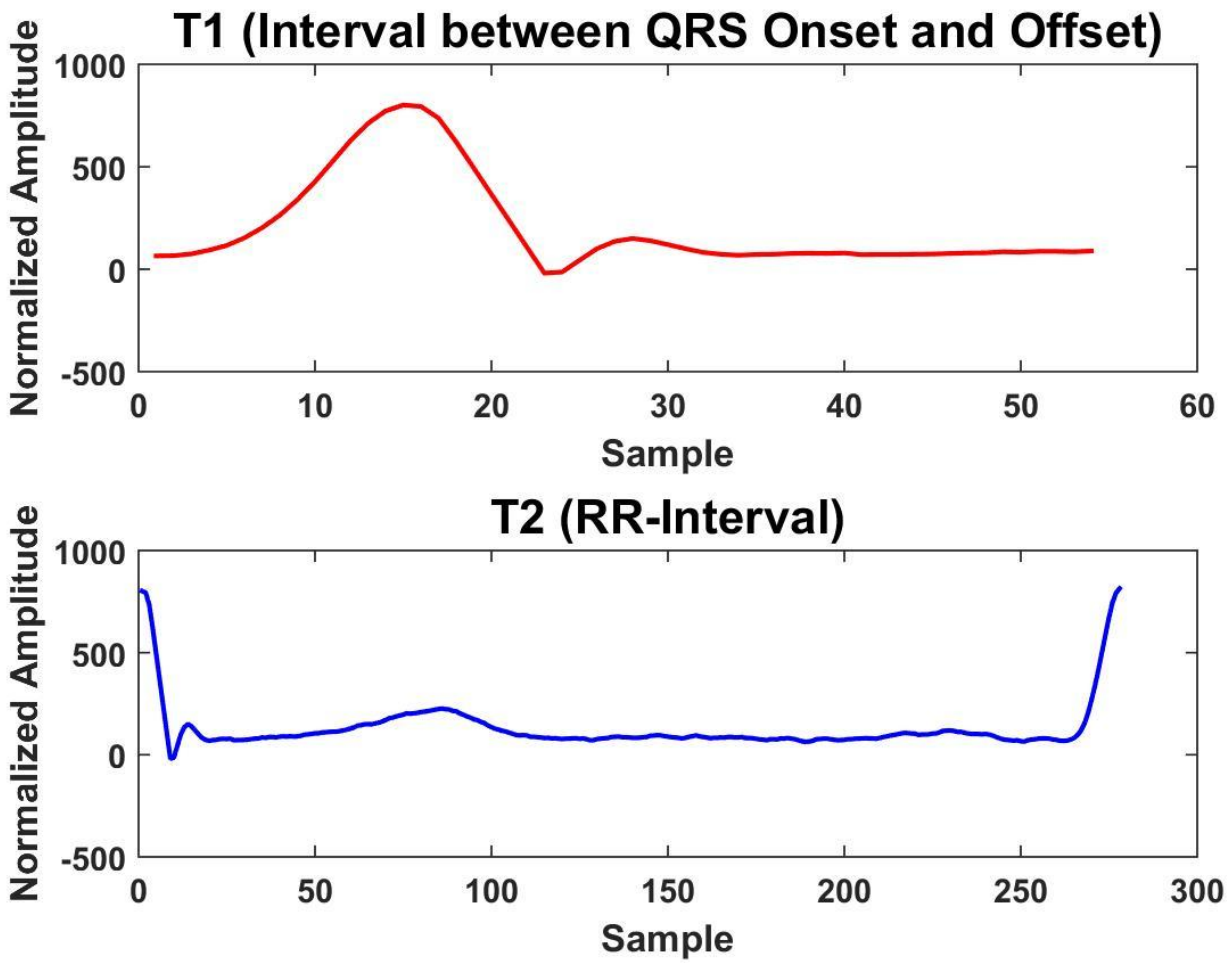


Figure 15. T1 and T2 generation

Template-Matching Algorithm

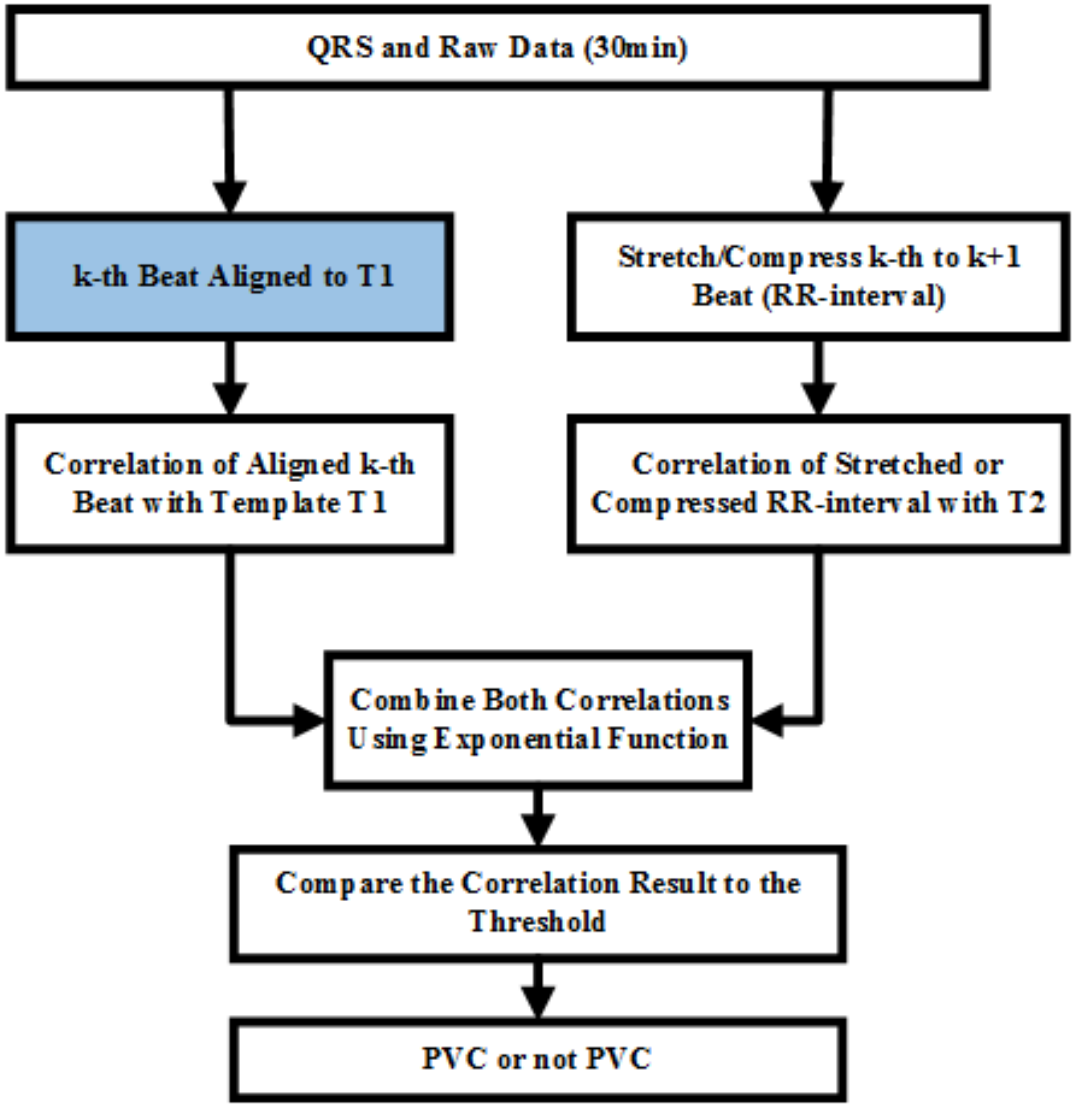


Figure 16. Flowchart for template-matching algorithm

Template-Matching Algorithm

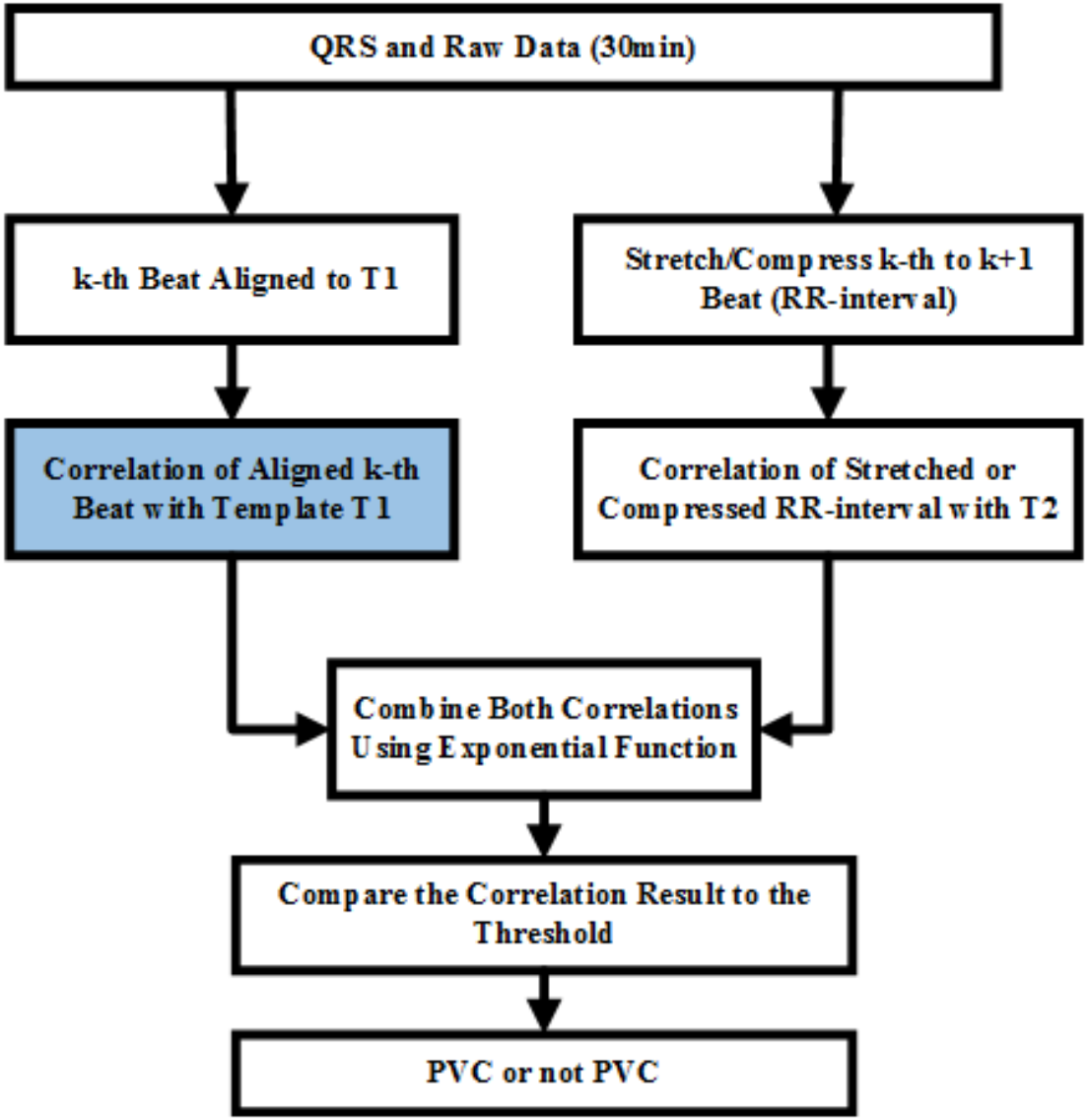


Figure 17. Flowchart for template-matching algorithm

Template-Matching Algorithm

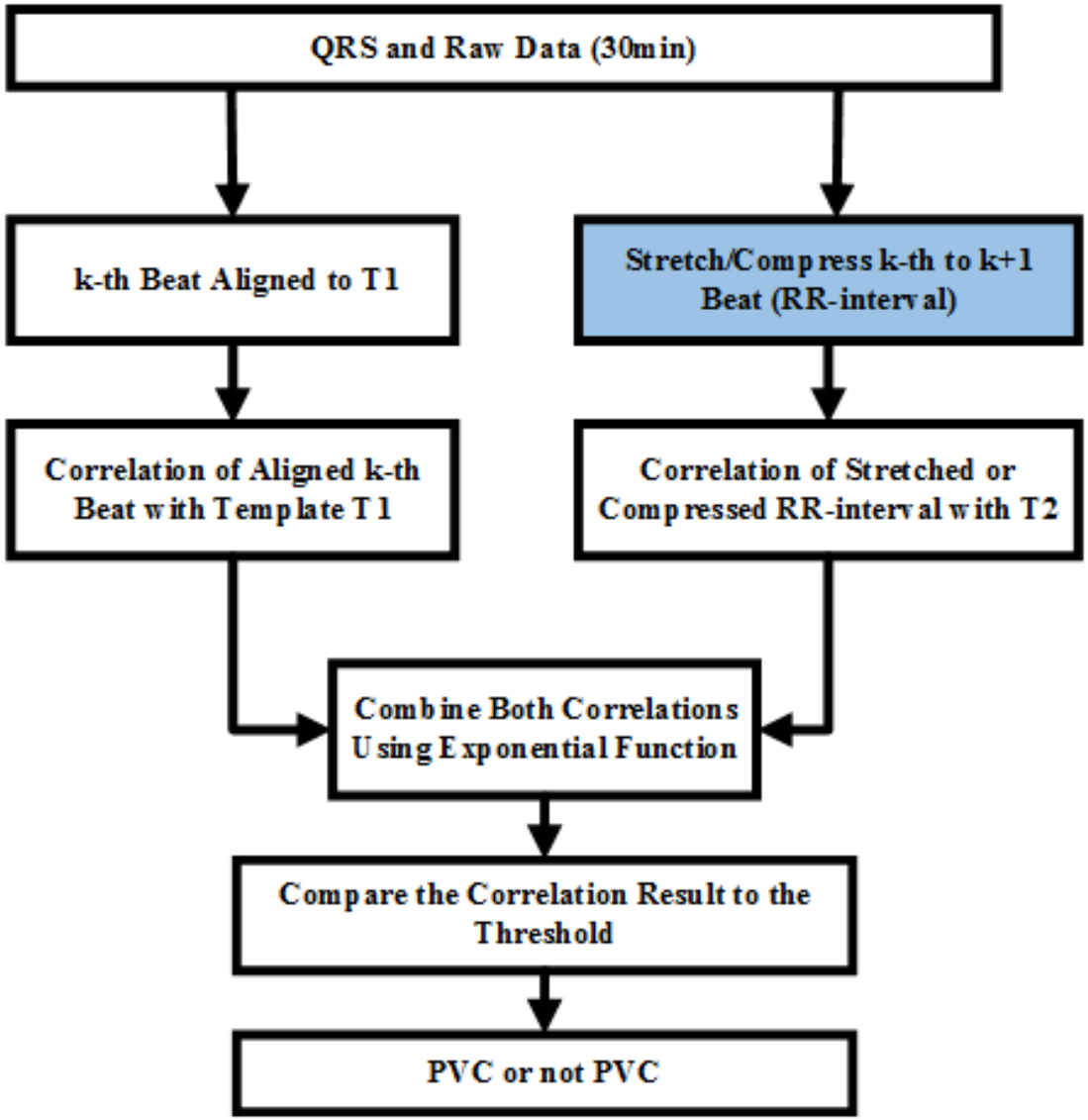


Figure 18. Flowchart for template-matching algorithm

Data between Two R-peaks

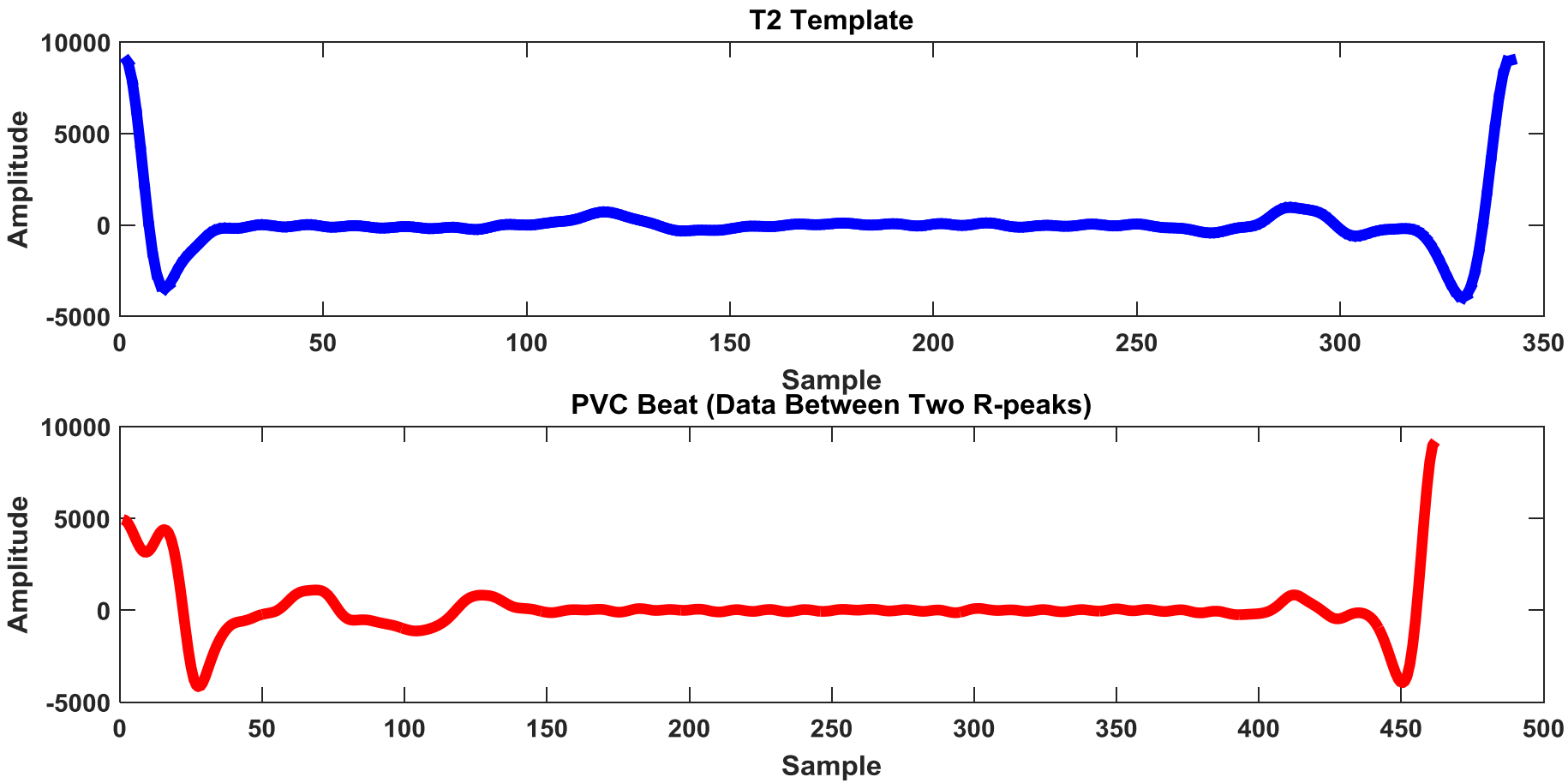


Figure 19. T2 (interval between R peaks) and RR-interval signal for PVC beat

Interpolation and Extrapolation

- Let b_k be the data between the kth beat and the $k+1^{\text{th}}$ beat.
- If length(b_k)$\leq$ length($T2$), we do extrapolation based on

$$new\_b_k = b_k[1 + \alpha(n-1)]\ (n = 1,2, \dots L)$$

$$\alpha = \frac{length(b_k)}{length(T2)}$$

Normalized Correlation Coefficient

$$x_k, y_k = \frac{\sum_{n=0}^{L-1}\left[b_k(n) - \overline{b_k}\right]\left[N(n) - \bar{N}\right]}{\sqrt{\sum_{n=0}^{L-1}\left[b_k(n) - \overline{b_k}\right]^2\left[N(n) - \bar{N}\right]^2}}$$

- x_k is the QRS complex correlation coefficient
- y_k is the RR-interval correlation coefficient
- $b_k(n)$ is the QRS complex/RR-interval in the kth beat
- L is the template length
- $N(n)$ is the template (T1 or T2)
- $\overline{b_k}$ and $\bar{N}$ are the signal mean and the template mean

Template-Matching Algorithm

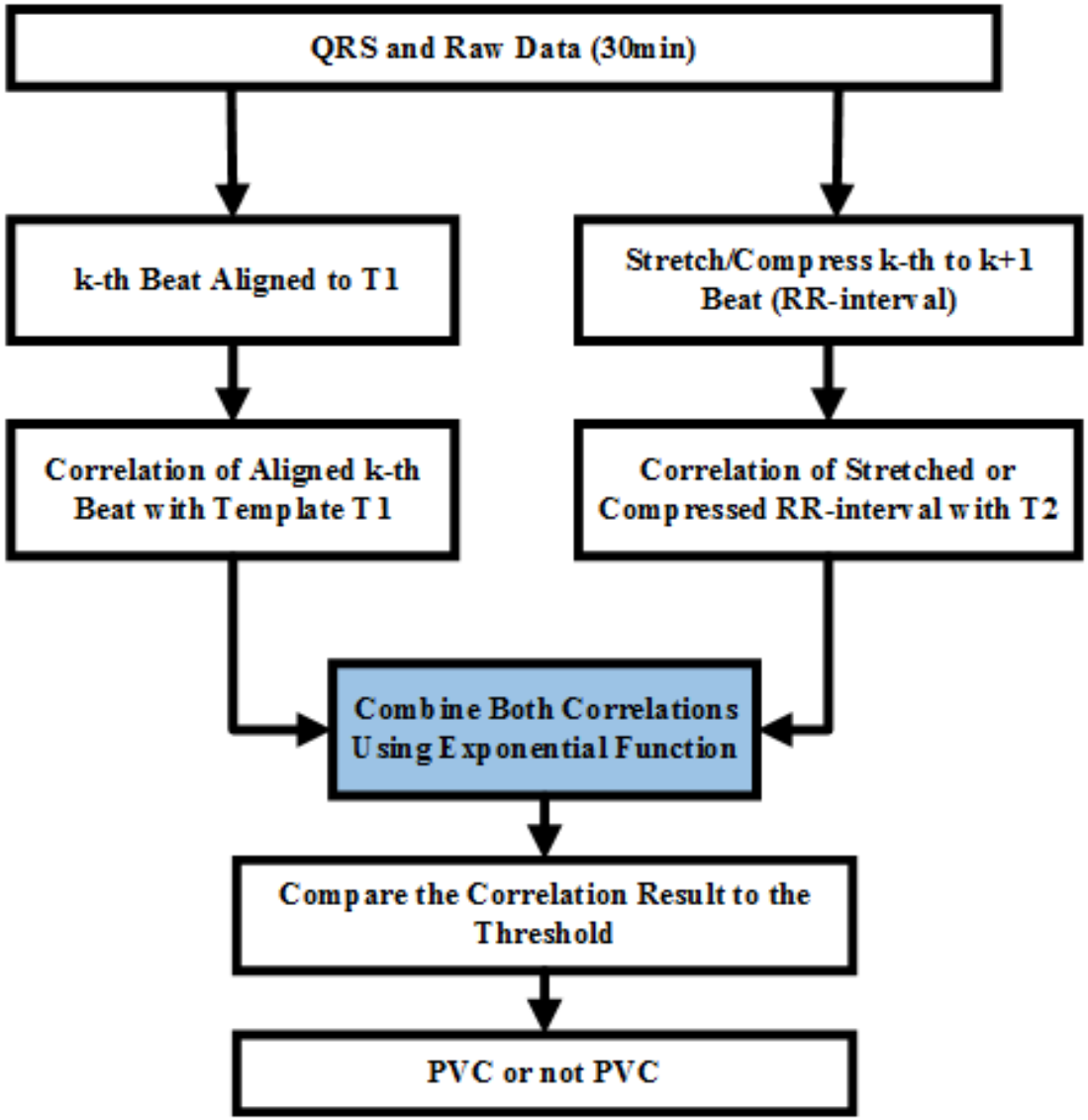


Figure 20. Flowchart for template-matching algorithm

Exponential Function Correlation

$$z_k = f\left(x_{k,}y_k\right) = \frac{\left(e^{{x_k}^r} + e^{{y_k}^r}\right)}{2e}$$

- r determines the increasing rate of the slope and $z_k \geq z_{thre}$ in a healthy beat
- We used $r = 5$, $z_k = 0.55$

Contents

Functional Requirements

TABLE IV. FUNCTIONAL REQUIREMENTS (IMPLEMENTATION)

Functions	Specifications	Specification Met?
Storing heart data input into memory	The internal memory of the device must be at least 25 kB.	Yes

Hardware

- SimpleLink Wi-Fi CC3200 Launchpad
  - Inexpensive: $30.00
  - Simplifies data transmission
  - 256 kB RAM

Figure 21. CC3200 Launchpad [9]

Communication

- µDMA (Micro Direct Memory Access) system with PC (USB connection)
  - CC3200 receives buffers of heart data
- WiFi connection for web services

µDMA Diagram

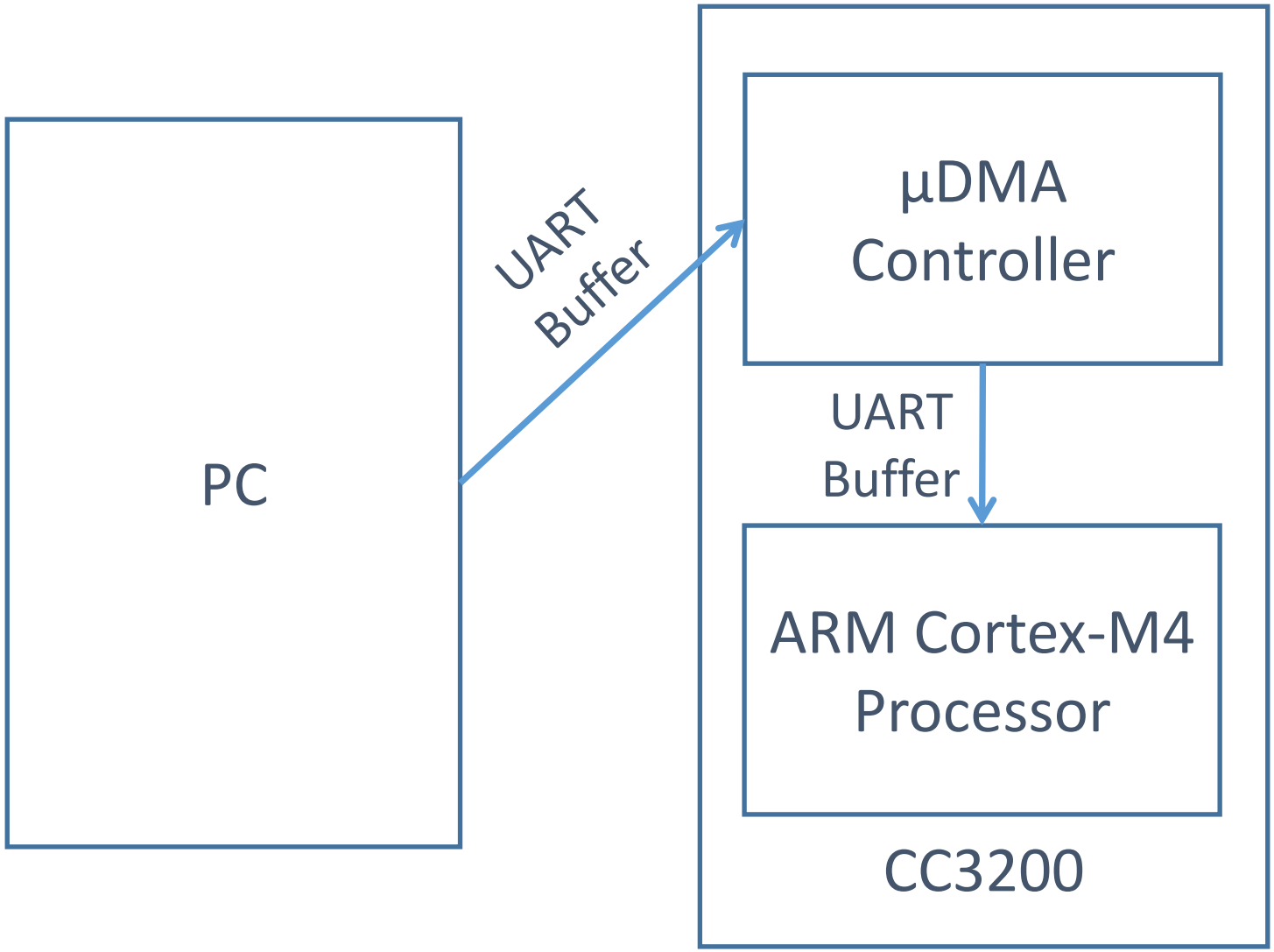


Figure 22. UART data flow between the PC and the CC3200

PC-side Program

- Uses WFDB (Waveform Database) library functions to access the MIT-BIH arrhythmia database
- Converts the integer values into characters for µDMA transfer

Functional Requirements

TABLE V. FUNCTIONAL REQUIREMENTS (IMPLEMENTATION)

Functions	Specifications	Specification Met?
Determining whether ventricular tachycardia is present using the calculated PVC information	The VT detection algorithm must return a 30-second interval of heart data surrounding the VT event for use in the wireless communication subsystem.	1000 data samples: about 3 seconds
Transmitting a wireless message to the patient's doctor in the event of ventricular tachycardia	The time delay between the device transmitting the message and the doctor receiving the message must be less than one second.	approximately 2 minute delay

CC3200 WiFi Setup

- The CC3200 was set up in station mode to access the internet

Figure 23. CC3200 set up in station mode [10]

Temboo

- Middleware that allows different devices (such as the LaunchPad) to access web-based services

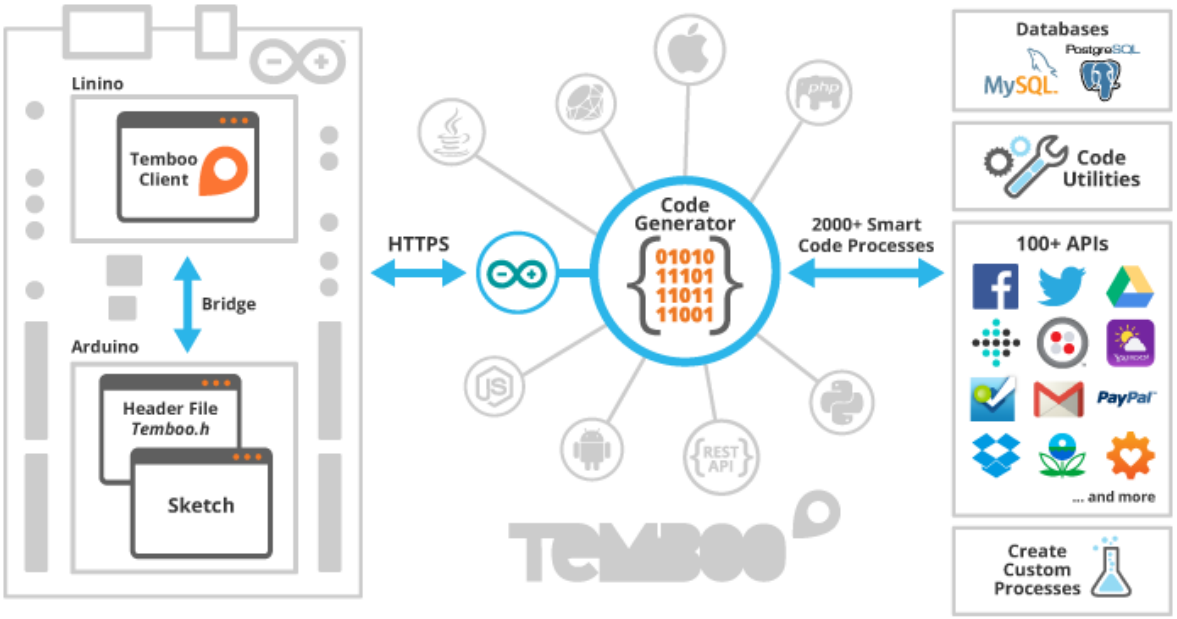


Figure 24. Temboo integration with an embedded device [11]

Temboo/Twilio

- Sending an SMS message using the CC3200 LaunchPad
- Message includes text and image file

Temboo/Twilio

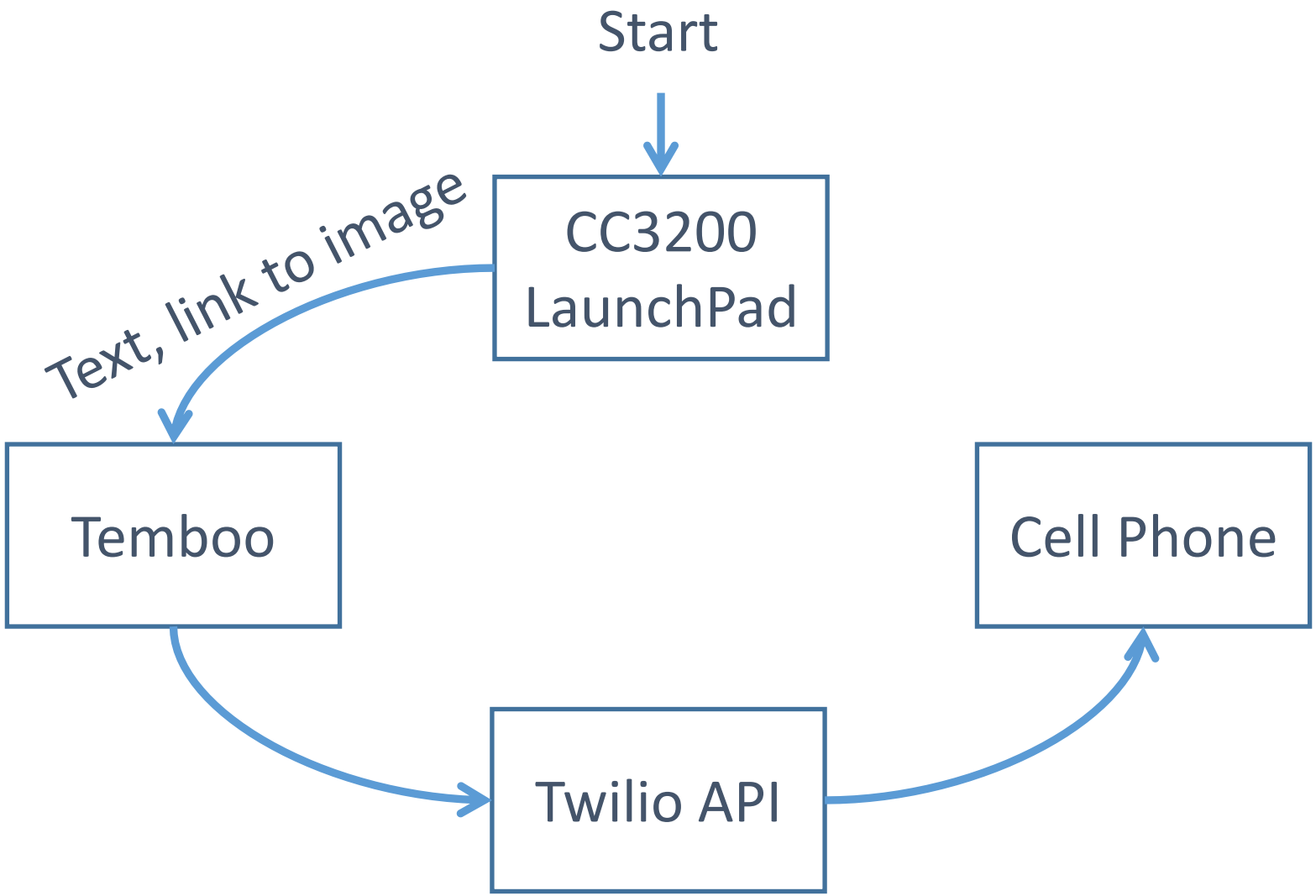


Figure 25. Transmitting an SMS message using the LaunchPad

Wireless - Plotly

Figure 26. Sample Plotly graph of 1000 samples of heart data

System Integration

- Combined project uses Energia sketch in Code Composer Studio
- GNU compiler replaced ARM compiler
- Mixed C/C++ code

Contents

Functional Requirements

TABLE VI. FUNCTIONAL REQUIREMENTS (ALGORITHM METRICS)

Function	Specification	Specification Met?
Identifying each QRS complex	The device must have at least 90% positive predictivity and at least 90% sensitivity.	Yes
Classifying each QRS complex as PVC or non-PVC	The device must have at least 90% accuracy.	Yes

QRS Detection Metrics [12]

$$PP = \frac{TP}{TP + FP} \qquad SE = \frac{TP}{TP + FN}$$

- PP: positive predictivity
- SE: sensitivity
- TP (True Positive): detected QRS complex that is present in the signal
- TN (True Negative): data point between QRS complexes that does not contain a QRS peak
- FP (False Positive): incorrect identification of QRS peak
- FN (False Negative): QRS peak that was not detected by the algorithm

PVC Detection Metric [12]

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

- ACC: accuracy
- TP (True Positive): detected PVC that is present in the signal
- TN (True Negative): healthy QRS complex
- FP (False Positive): incorrectly detected PVC
- FN (False Negative): PVC beat that was not detected by the algorithm

Solution Testing

- MATLAB simulation of QRS, PVC, and VT detection
  - Use MIT-BIH arrhythmia database for testing data
  - Ensure that accuracy, sensitivity, and specificity are at least 90% using the WFDB toolbox

Solution Testing

- C implementation of QRS, PVC, and VT detection
  - Store the heart data in the board's memory and export the detection results to a file

Solution Testing

TABLE VII. QRS SENSITIVITY COMPARISON

Record	QRS Sensitivity (MATLAB)	QRS Sensitivity (C implementation)
106	0.9994	0.9762
116	0.9889	0.9885
119	1.0000	1.0000
201	0.9815	0.9731
203	0.9903	0.9910
208	0.9654	0.9381
Total	0.9912	0.9909

Solution Testing

TABLE VIII. QRS POSITIVE PREDICTIVITY COMPARISON

Record	QRS Positive Predictivity (MATLAB)	QRS Positive Predictivity (C Implementation)
106	0.9946	0.9995
116	0.9978	0.9987
119	1.0000	1.0000
201	0.9943	0.9791
203	0.9871	0.9821
208	0.9985	0.9942
Total	0.9921	0.9918

Solution Testing

TABLE IX. OVERALL ACCURACY COMPARISON

Record	Accuracy (MATLAB)	Accuracy (C Implementation)
106	0.9390	0.9404
116	0.9957	0.9966
119	1.0000	1.0000
201	0.9682	0.9441
203	0.9284	0.8987
208	0.9702	0.9481
Total	0.9096	0.9316

Results

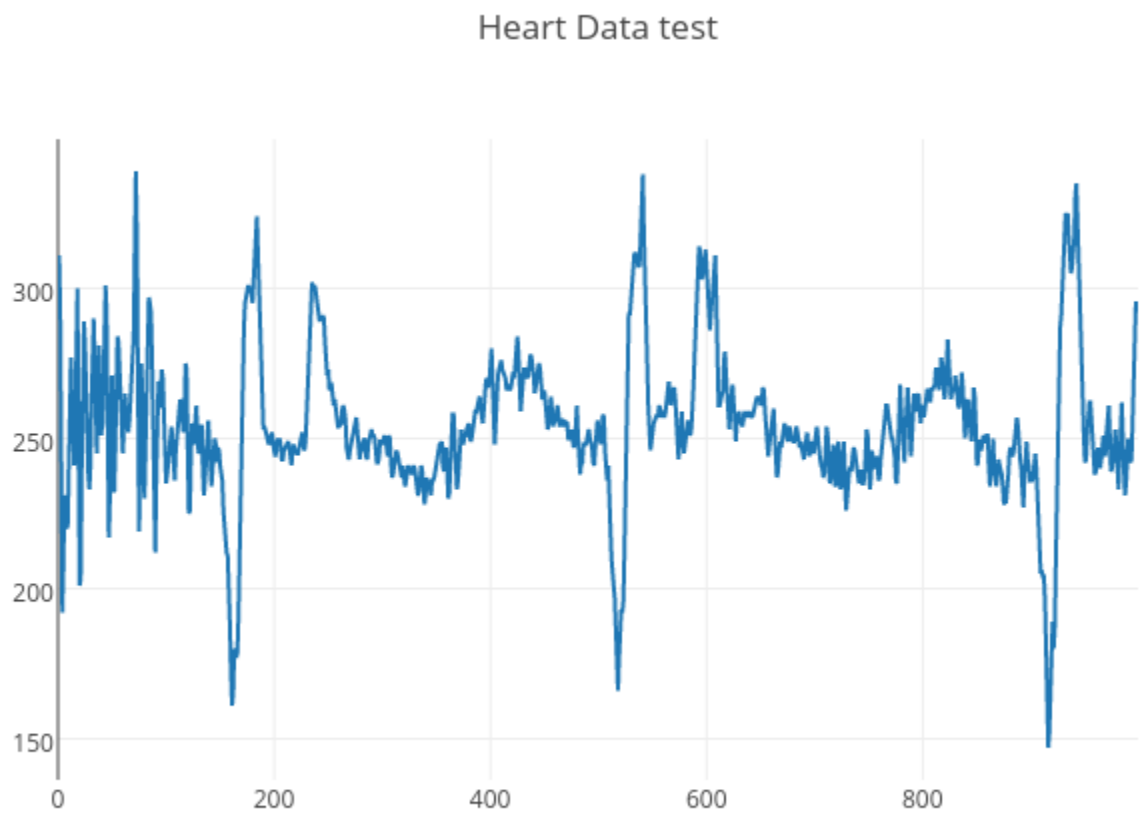


Figure 27. Plotly graph of VT event

Contents

Summary and Conclusions

- PVCs are irregular heartbeats that may lead to VT
- An embedded system has been developed that detects PVCs in real time and wirelessly alerts the patient’s doctor of VT

Summary and Conclusions

- Future Work
  - Interfacing the LaunchPad with electrodes
  - Additional improvements to the signal processing algorithms

Real-time Heart Monitoring and ECG Signal Processing

Fatima Bamarouf, Claire Crandell, and Shannon Tsuyuki

Advisors: Drs. Yufeng Lu and Jose Sanchez

Department of Electrical and Computer Engineering
Bradley University

April 26, 2016

References

- [1] *Arrhythmias.* [Online] Available: http://watchlearnlive.heart.org/CVML_Player.php?moduleSelect=arrhyt
- [2] *MIT-BIH Arrhythmia Database.* [Online] Available: http://www.physionet.org/physiobank/database/mitdb/
- [3] *Cardiovascular System Assessments.* [Online] Available: http://media.lanecc.edu/users/driscolln/RT116/softchalk/Cardia_Assessment/Cardia_Assessment_print.html
- [4] *Holter Monitor.* [Online] Available: http://www.hopkinsmedicine.org/healthlibrary/test_procedures/cardiovascular/holter_monitor_92,P07976/
- [5] J. Pan and W. Tompkins, "A Real-Time QRS Detection Algorithm," *IEEE Transactions on Biomedical Engineering*, vol. -32, no. 3, pp. 230-236, 1985. [Online] Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4122029
- [6] P. Li, et al., "A low-complexity data-adaptive approach for premature ventricular contraction recognition," *Signal, Image and Video Processing*, vol. 8, no. 1, pp. 111-120, 2013. [Online] Available: http://link.springer.com/article/10.1007%2Fs11760-013-0478-6
- [7] C. Li, et al., "Detection of ECG Characteristic Points Using Wavelet Transforms." [Online] Available: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=362922&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F10%2F8315%2F00362922.pdf%3Farnumber%3D362922
- [8] S. Mallat and S. Zhong, "Characterization of Signals from Multiscale Edges," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 14, no. 7, July 1992. [Online] Available: http://www.cmap.polytechnique.fr/~mallat/papiers/MallatEdgeCharact92.pdf
- [9] *CC3200-LAUNCHXL.* [Online] Available: http://www.ti.com/ww/en/launchpad/launchpads-connected-cc3200-launchxl.html?DCMP=cc3100cc3200&HQS=cc3200launchpad-oob
- [10] CC3200 –Getting Started with WLAN Station. [Online] Available: http://processors.wiki.ti.com/index.php/CC32xx_Getting_Started_with_WLAN_Station
- [11] Arduino: Dynamic Living-Room Lights. [Online] Available: https://www.element14.com/community/groups/arduino/blog/2014/12/22/dynamic-living-room-lights-the-yun-review--when-the-penguin-met-the-arduino
- [12] B. Ribeiro, et al., "Choosing Real-Time Predictors for Ventricular Arrhythmia Detection," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 21, no. 08, pp. 1249-1263, 2007. [Online] Available: https://www.researchgate.net/publication/220359330_Choosing_Real-Time_Predictors_for_Ventricular_Arrhythmia_Detection
- [13] *MSP430 Wireless Development Tool*. [Online] Available: http://www.ti.com/tool/ez430-rf2500
- [14] *REST API | Plotly Developers.* [Online] Available: https://plot.ly/rest/

Code Optimization

- CC3200 has only 256kb of RAM
- The template matching algorithm requires storing 20 seconds of heart data on-board
- Instead, we obtained used a simple average to find a suitable template:

$$0.9 * \text{average} \leq \text{template} \leq 1.1 * \text{average}$$

T1 Alignment

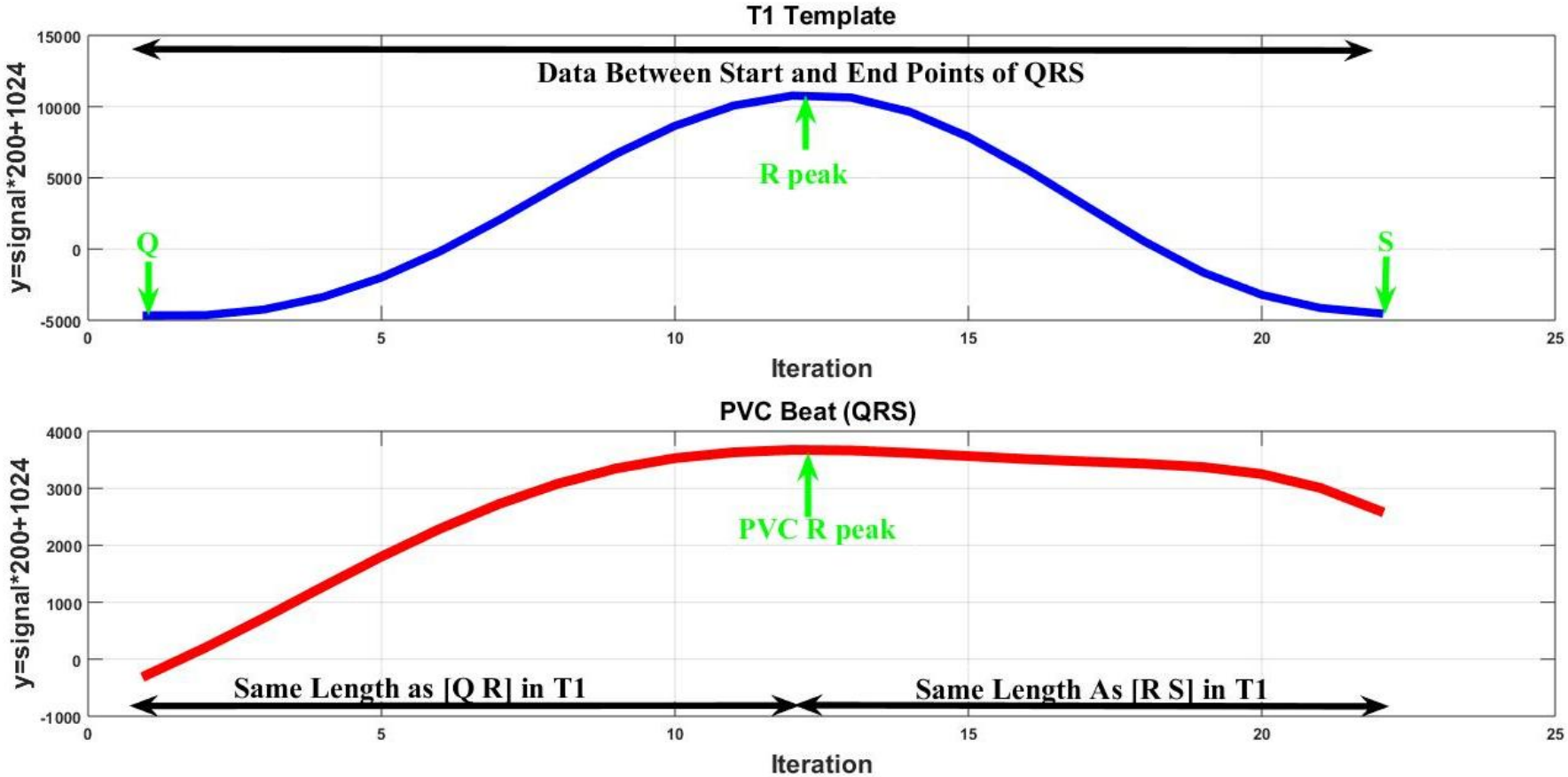


Figure 28. T1 template and PVC QRS complex aligned to T1's R peak index

Memory Requirements

- Sampling rate for ECG signal (MIT-BIH arrhythmia database): 360 Hz
- Number of samples required for 30 seconds of ECG data: 10,800
- Amount of memory required: 21 kB

Nonfunctional Requirements: Metrics

Objective: The device should be compatible with all patient data in the MIT-BIH database. [2]

Metric:

- Highly compatible: 10 points
- Very compatible: 7.5 points
- Compatible: 5.0 points
- Somewhat compatible: 2.5 points
- Not compatible: 0 points

Nonfunctional Requirements: Metrics

Objective: The device should be portable.

Metric:

- Very easy to carry around: 10 points
- Easy to carry around: 7.5 points
- Portable: 5.0 points
- Uncomfortable to carry around: 2.5 points
- Difficult to carry around: 0 points

Nonfunctional Requirements: Metrics

TABLE XI. QUANTITATIVE PERFORMANCE LEVELS FOR REAL-TIME HEART MONITORING

Power Consumption in 24 Hours of Continuous Use (W)	Price ($)	Value Scaled
1.50	500	10
2.50	600	7.5
3.25	700	5
4.00	800	2.5
4.75	900	0

Alternative Solution: Hardware

- eZ430-RF2500 (Texas Instruments)
  - MSP430F2274 MCU
  - CC2500 wireless transceiver
  - 32 kB flash memory

Figure 29. eZ430-RF2500 Development Kit [13]

Alternative Solution: Software

- PVC detection
  - Wavelet transform algorithm [13]
  - RR-interval algorithm [14]

Pan-Tompkins Algorithm [5]

1. Low-pass Filter
   - 11 Hz cut-off frequency
   - 5-sample delay
   - Gain of 36

$$y(n) = 2y(n-1) - y(n-2) + x(n) - 2x(n-6) + x(n-12)$$

2. High-pass Filter
   - 5 Hz cut-off frequency
   - 29-sample delay
   - Gain of 1

$$y(n) = y(n-1) - \frac{1}{32}x(n) + x(n-16) - x(n-17) + \frac{1}{32}x(n-32)$$

Pan-Tompkins Algorithm

3. Derivative
   - Provides information about QRS slope
   - Approximates derivative from 0-30 Hz
   - Has a 4-sample delay

$$y(n) = \frac{1}{8}[2x(n) + x(n-1) - x(n-3) - 2x(n-4)]$$

4. Squaring Function
   - Emphasizes higher frequencies of the ECG (caused by QRS complexes)

$$y(n) = x^2(n)$$

Pan-Tompkins Algorithm

5. Moving-Window Integration

- Detects long-duration and large-amplitude QRS complexes

$$y(nT) = \frac{1}{N}[x(nT - (N-1)T) + x(nT - (N-2)T + \cdots + x(nT)]$$

Algorithm Efficacy, 100s

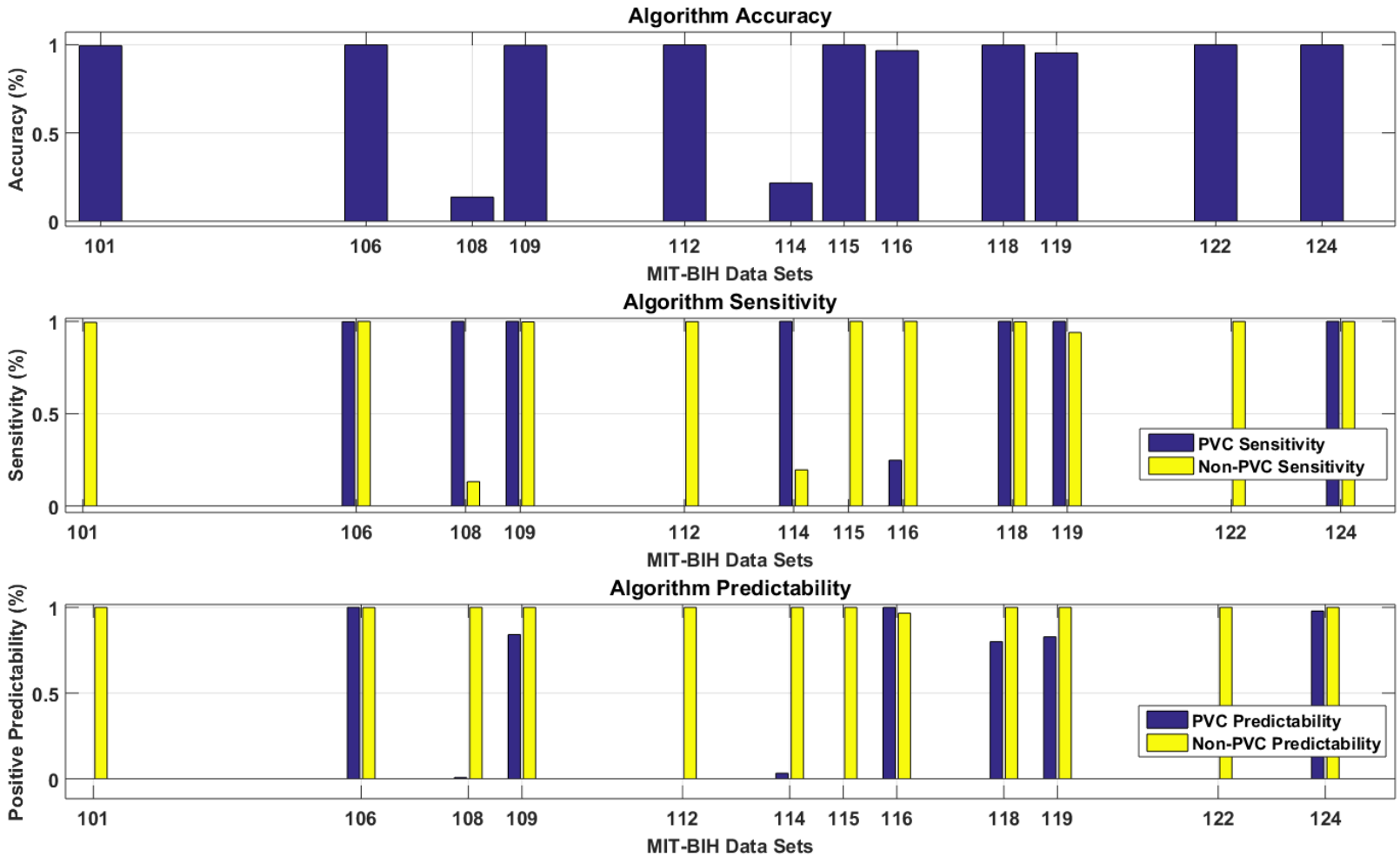


Figure 30. Performance of template-matching algorithm MATLAB simulation

Algorithm Efficacy, 200s

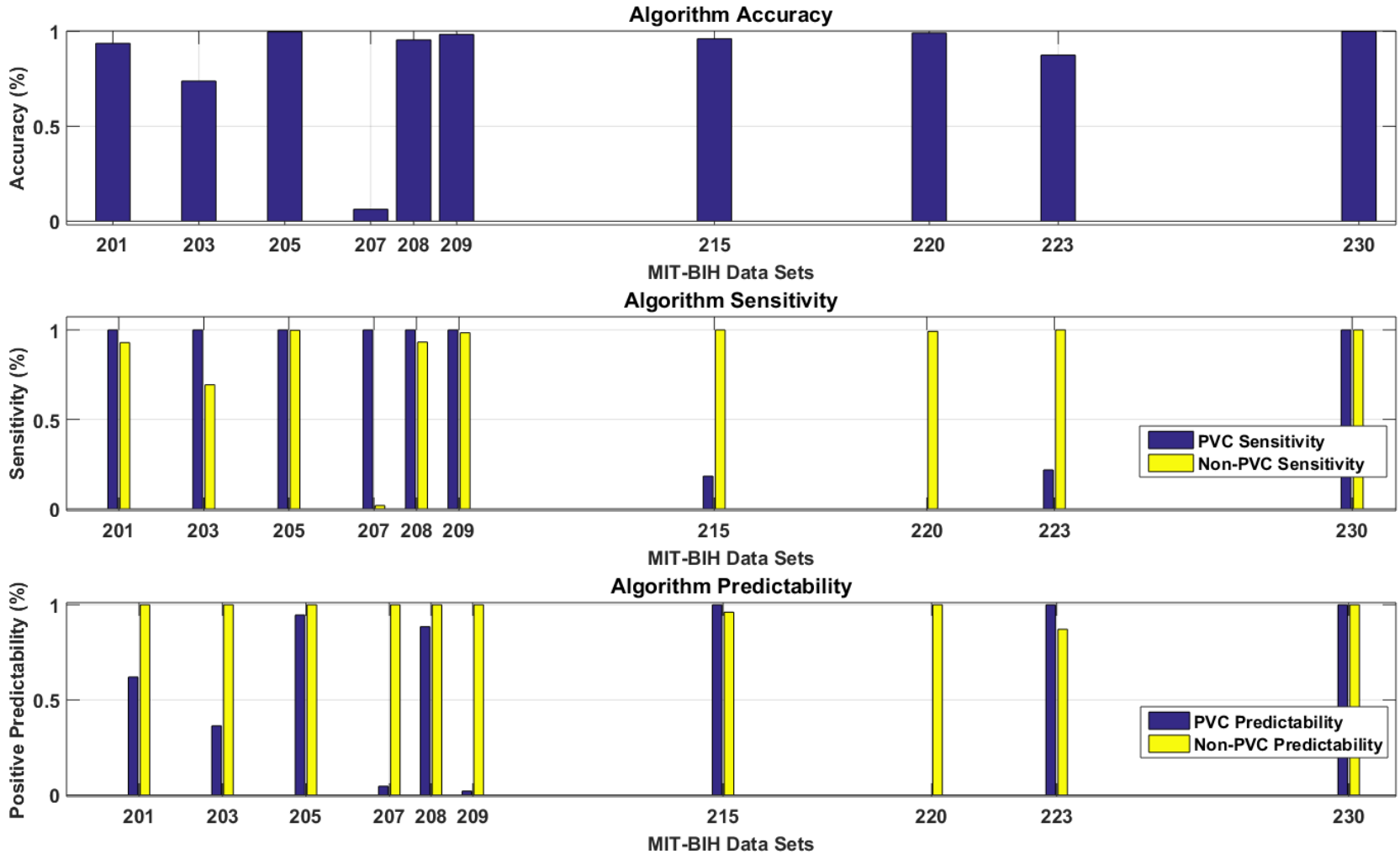


Figure 31. Performance of template-matching algorithm MATLAB simulation

WFDB Library (PC side)

- `isigopen()`: open a specific WFDB record
- `getvec()`: get the next sample in the record

WFDB Toolbox (MATLAB)

- `rdsamp()`: place samples from a WFDB record into a vector
- `rdann()`: place annotations (characters) from a WFDB record into a vector

WFDB Toolbox (MATLAB)

- `wrsann()`: write experimental annotations into a vector
- `bxb()`: generate a report (with accuracy and positive predictivity data) using experimental annotations

Sample BXB Report

```
Beat-by-beat comparison results for record mitdb/100
Reference annotator: atr
     Test annotator: test

                    Algorithm
          n     s     v     f     q     o     x
     ------------------------------------------
 N  |   356     0     0     0     0     8     0
 S  |     4     0     0     0     0     0     0
 V  |     0     0     0     0     0     0     0
 F  |     0     0     0     0     0     0     0
 Q  |     0     0     0     0     0     0     0
 O  |     0     0     0     0     0
 X  |     0     0     0     0     0

            QRS sensitivity:  97.83% (360/368)
 QRS positive predictivity: 100.00% (360/360)
            VEB sensitivity:      -  (0/0)
 VEB positive predictivity:       -  (0/0)
           SVEB sensitivity:   0.00% (0/4)
SVEB positive predictivity:       -  (0/0)
     RMS RR interval error: 168.87 ms
```

Figure 30. Text file generated using the `bxb()` function in the WFDB toolbox

Energia and Code Composer Studio

- To be able to load the Energia sketch code from CCS, the libraries were changed to be compatible with Energia's GNU compiler
  - Hardware libraries
  - DSP library

DSP library for CC3200

- `Corr_Coeff.c`
  - `arm_sub_f32()`
  - `arm_mult_f32()`
- `Normal_amplt.c, normIntRange.c`
  - `arm_mean_f32()`
  - `arm_std_f32()`

QRS Detection

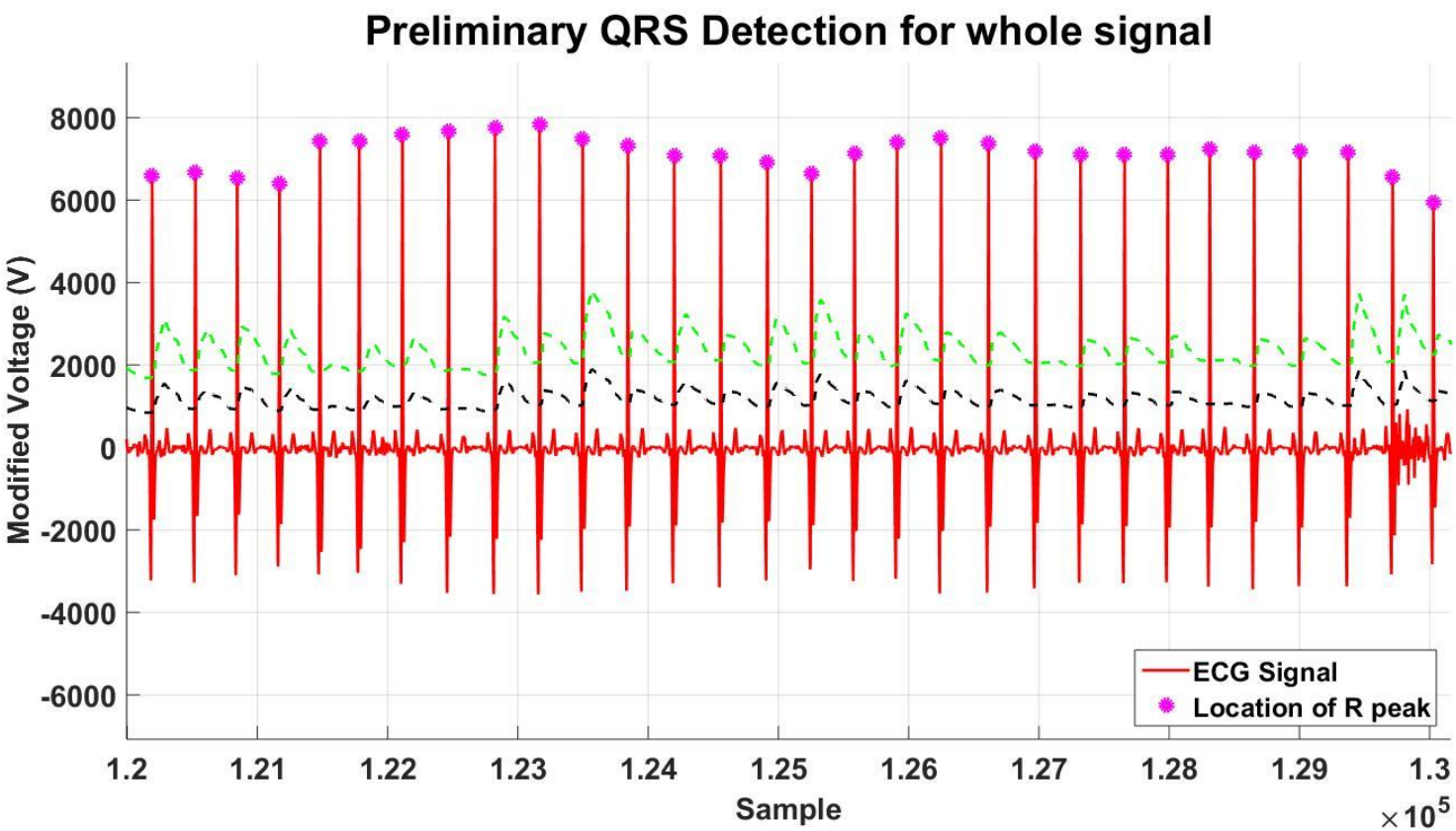


Figure 32. MATLAB plot of QRS detection results

UART Data Transfer

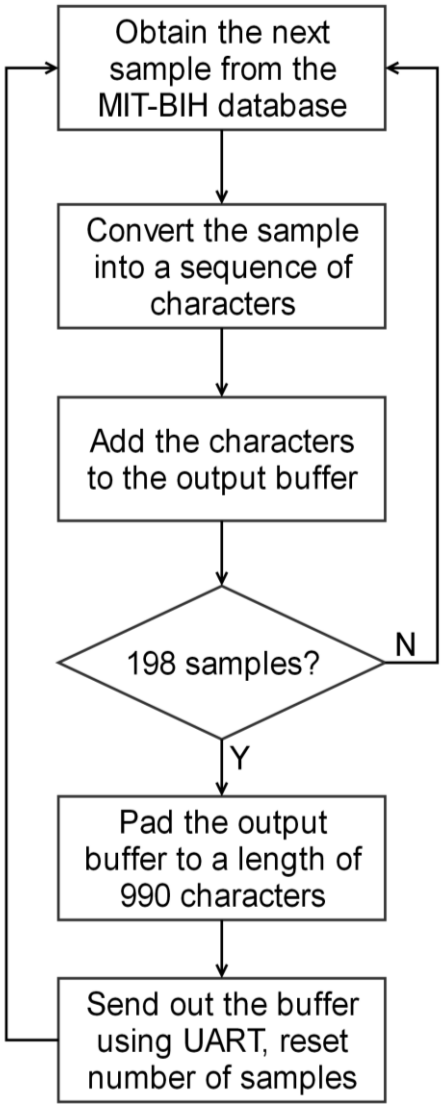


Figure 33. UART data transfer flowchart (PC)

UART Data Transfer

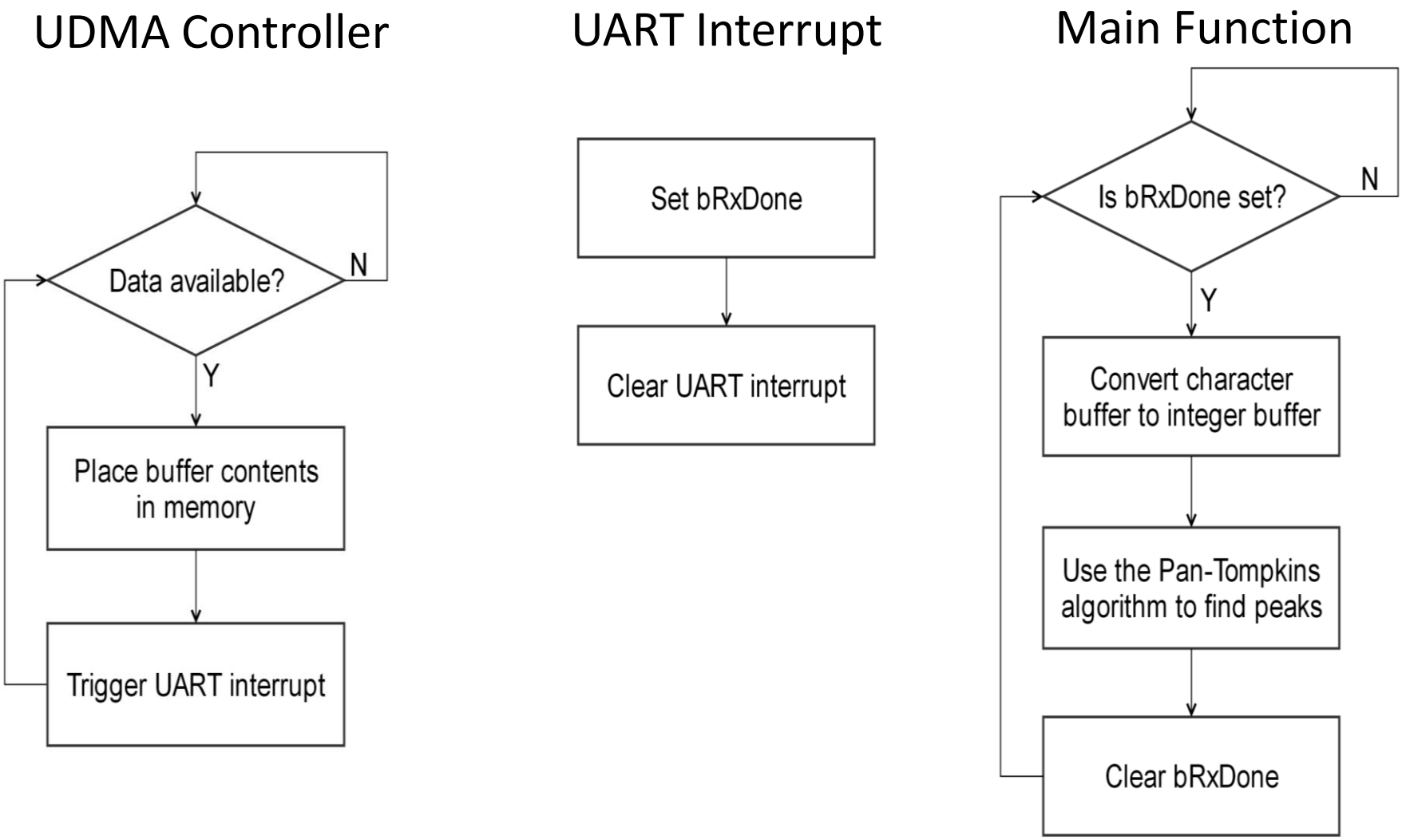


Figure 34. UART data transfer flowcharts (CC3200)

Alternative QRS Detection

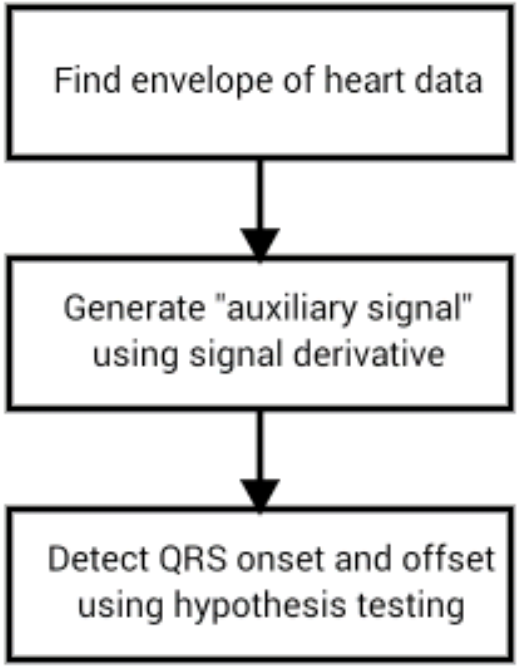


Figure 35. Flowchart for alternative QRS detection method

Alternative QRS Detection

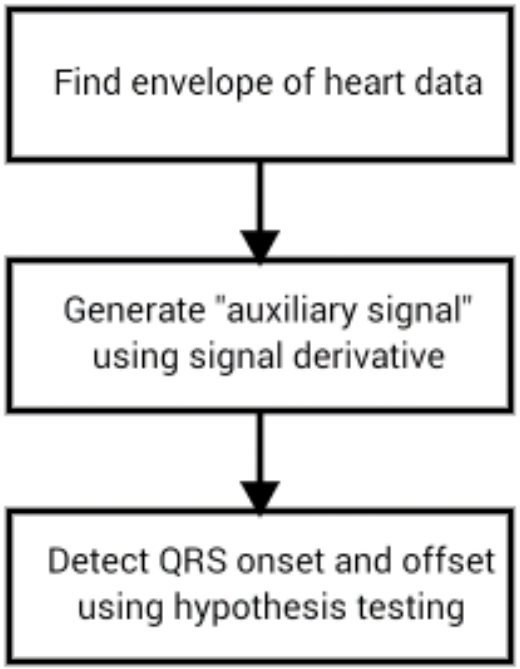


Figure 36. Flowchart for alternative QRS detection method

- Uses statistics to accurately locate QRS onset and offset
- Can be used to determine abnormal QRS complexes

Alternative QRS Detection

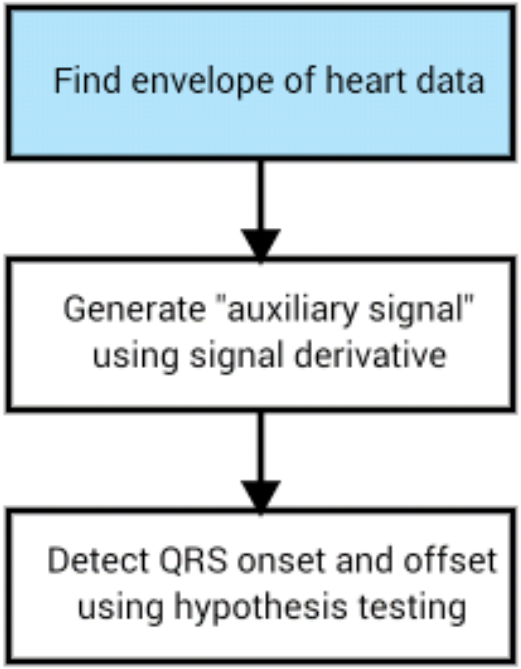


Figure 37. Flowchart for alternative QRS detection method

- Use the Hilbert transform to obtain the envelope

Alternative QRS Detection

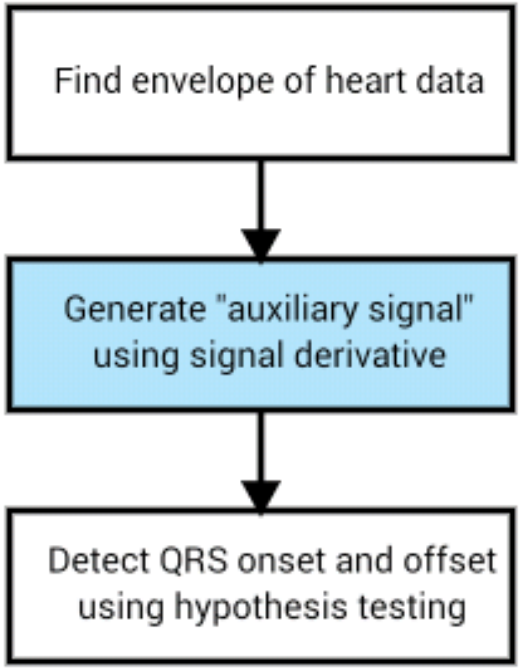


Figure 38. Flowchart for alternative QRS detection method

- Estimate the derivative using a parabola:

$$h'(k) = \frac{1}{10}\left(2\big(h(k+2r) - h(k-2r)\big) + h(k+r) - h(k-r)\right)$$

Alternative QRS Detection

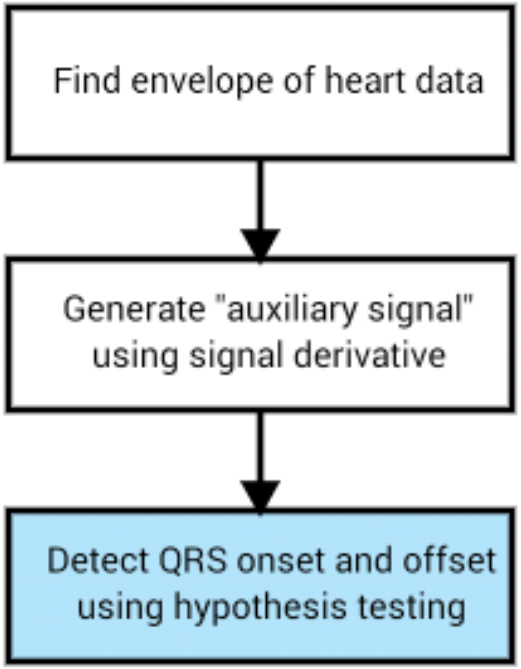


Figure 39. Flowchart for alternative QRS detection method

- Calculate a cumulative mean for the QRS onset and offset windows
- Determine the probability density functions

Envelope Signal

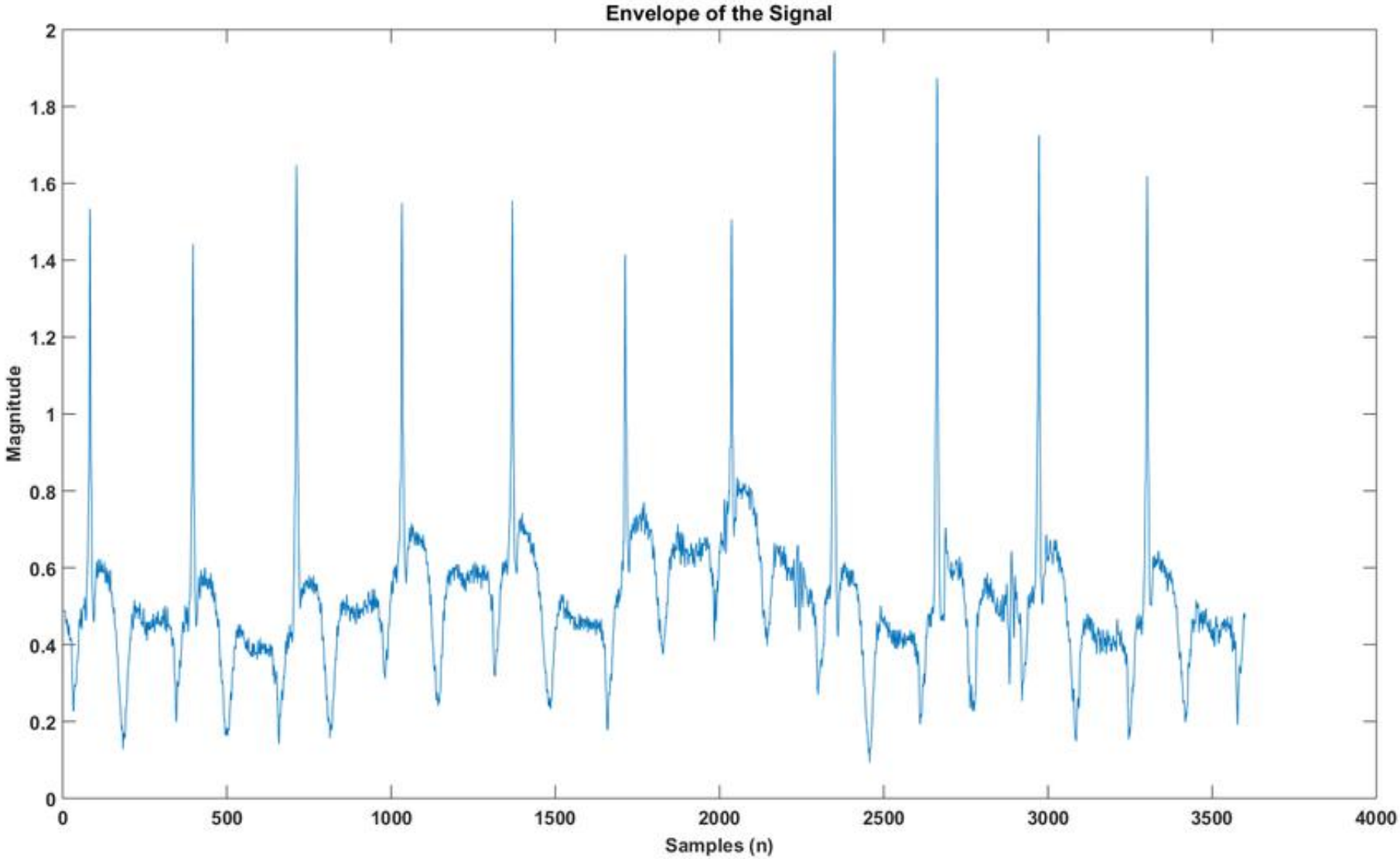


Figure 40. Envelope of the initial ECG signal

Auxiliary Signal

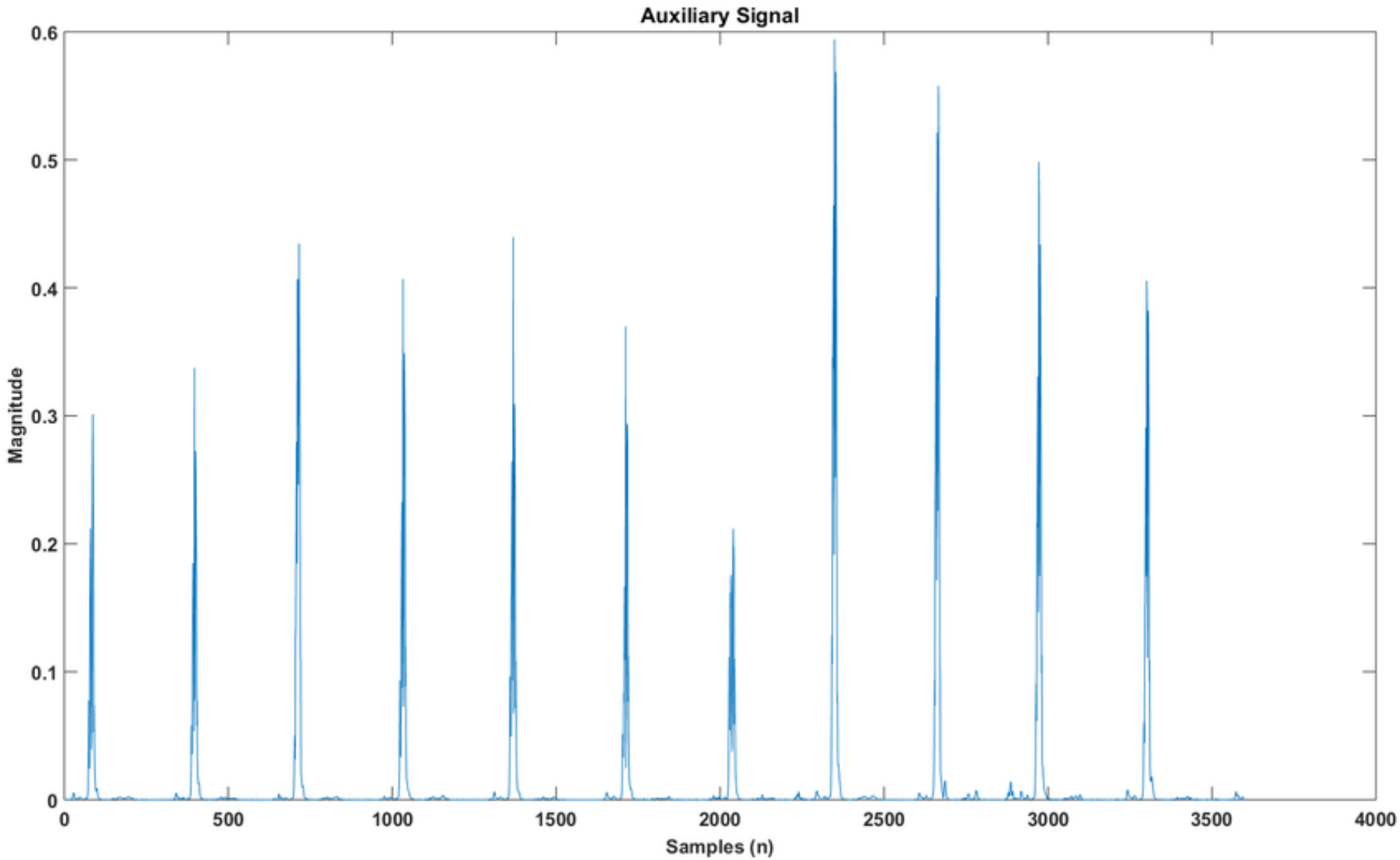


Figure 41. Auxiliary signal of the envelope

C Implementation Results

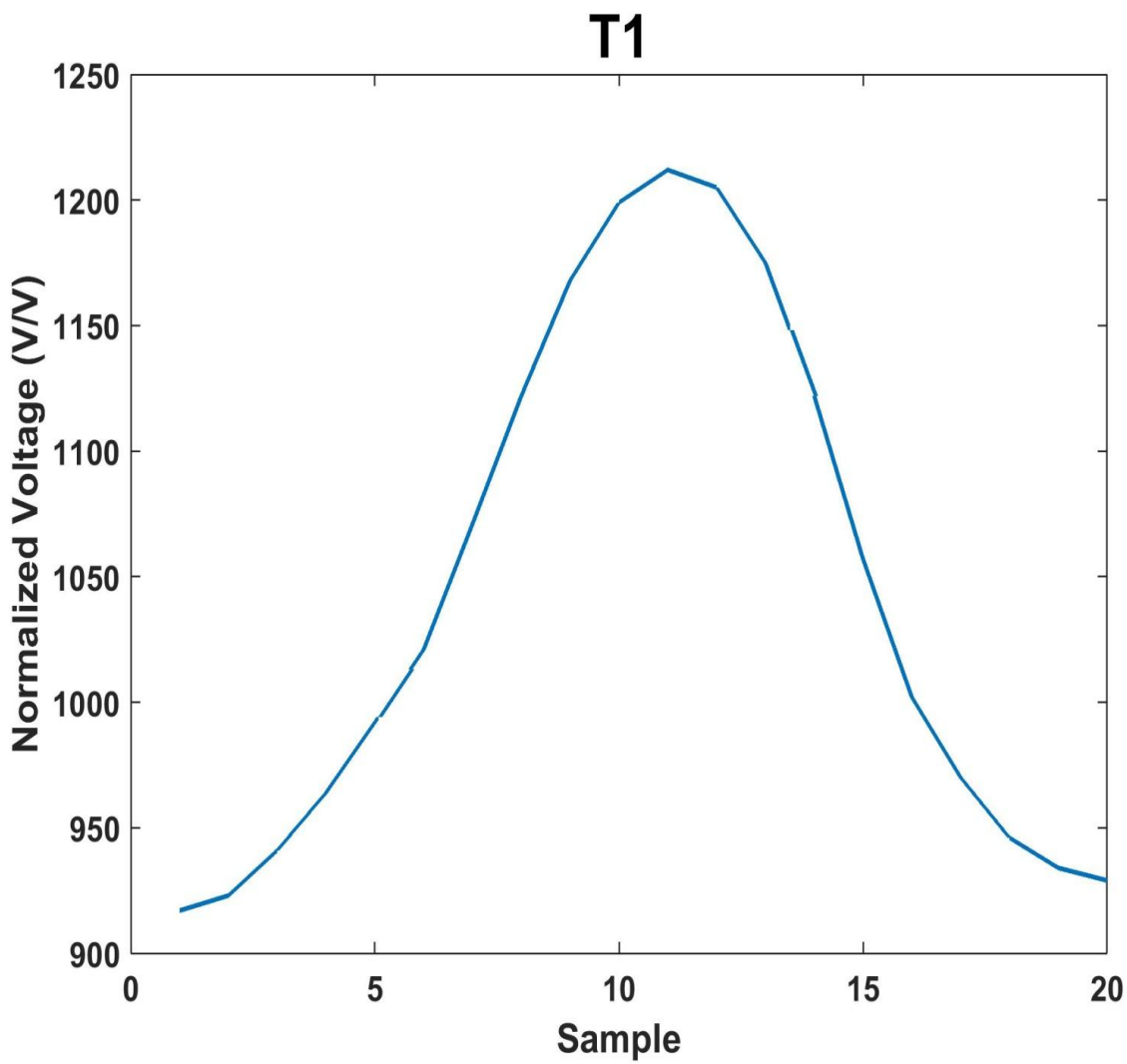


Figure 42. T1 (QRS complex) template generated on the CC3200

C Implementation Results

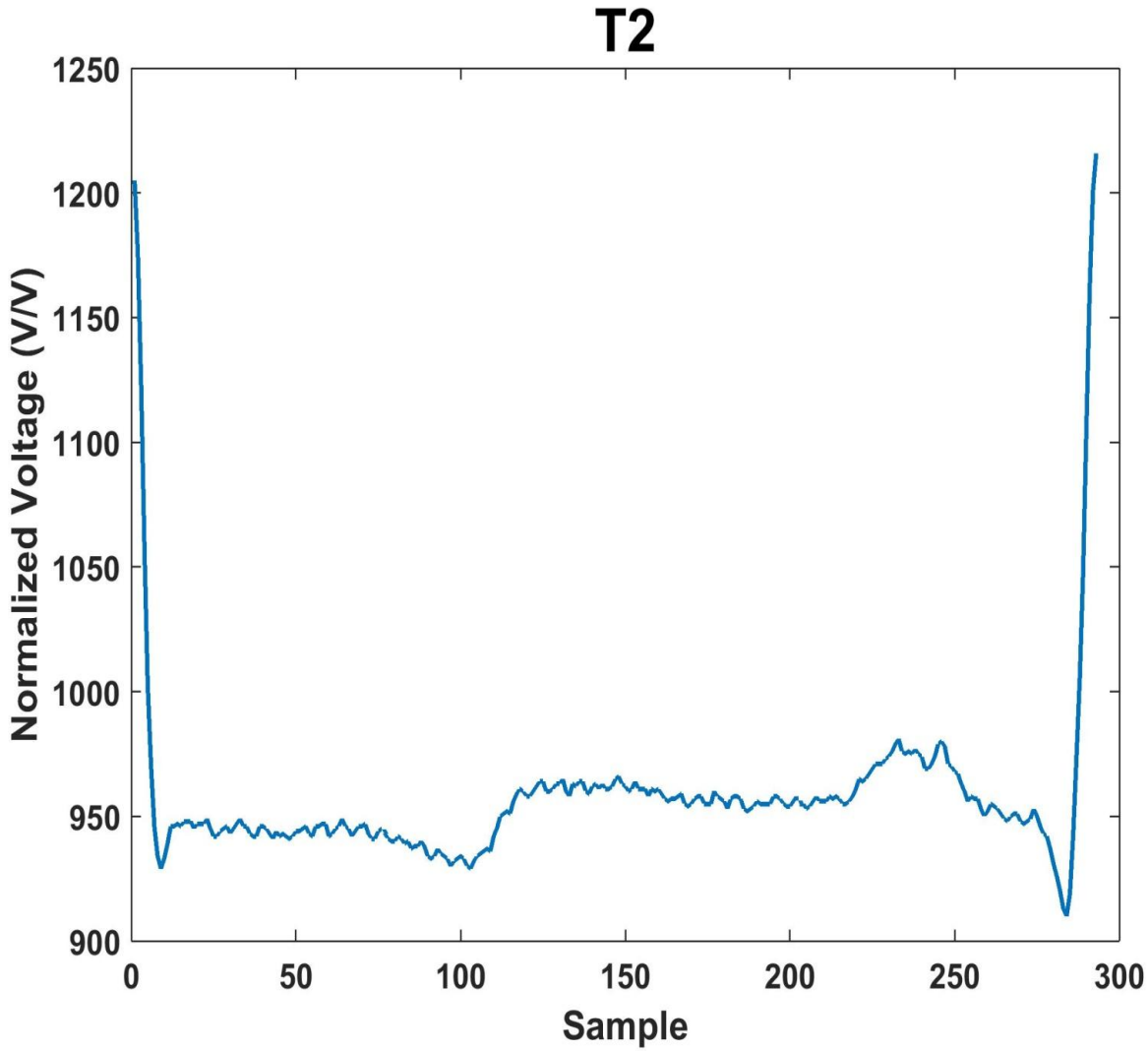


Figure 43. T2 (RR interval) template generated on the CC3200

System Integration

- Issues addressed
  - Memory configuration
  - Synchronization with PC

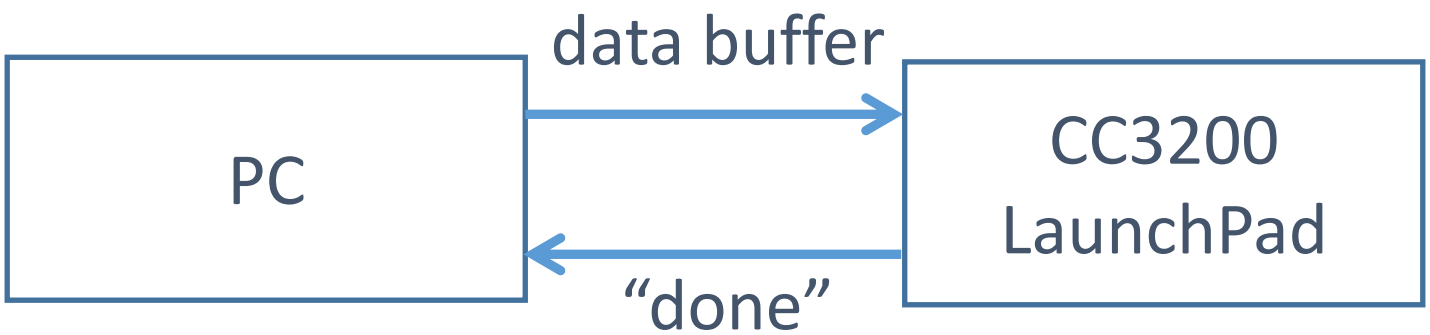


Figure 44. Improved UART testing system

Example (Extrapolation)

$$L_k = 10, \quad L = 21, \quad \alpha_k = \frac{10}{21} \approx 0.476$$

TABLE XII. INDICES AFTER EXTRAPOLATION

n value	Calculated Index
1	1
2	1.48
3	1.95
4	2.43
5	2.90

Wireless - Plotly

- Code language: JavaScript Object Notation (JSON)
  - Language independent
  - Based on JavaScript, C/C++/C#, Python, Perl, etc.

JSON Example [14]

```
un=chris&
key=kdfa3d&
origin=plot&
platform=lisp&
args=[[0, 1, 2], [3, 4, 5], [1, 2, 3],
 [6, 6, 5]]&
kwargs={"filename": "plot from api",
 "fileopt": "overwrite", "style": {
 "type": "bar" }, "traces": [1],
 "layout": { "title": "experimental
 data" }, "world_readable": true }
```

Quadratic Spline Wavelet Transform [8]

$j = 0$
while $(j < J)$
$\quad W^d_{2^{j+1}} f = \frac{1}{\lambda_j} \cdot S^d_{2^j} f * G_j$
$\quad S^d_{2^{j+1}} f = S^d_{2^j} f * H_j$
$\quad j = j + 1$
end of while.

TABLE XIII. WAVELET COEFFICIENTS

n (sample)	H	G
-1	0.125	
0	0.375	**-2.0**
1	0.375	**2.0**
2	0.125	

TABLE XIV. WAVELET TRANSFORM NORMALIZATION COEFFICIENTS

Order	Normalization Coefficient
1	**1.5**
2	1.12
3	1.03
4	1.01
5	1.00