****

**Temperature Prediction Using Kalman Filter & Grey Model**

# **Description:**

This project aims to create a device that can measure and predict the surrounding temperature using Kalman Filter and Grey Model. The project is based on the research in [1] and on LAB 3 in the curriculum. The model communication will be done through Bluetooth and the actual temperature will be generated in the C main function as a saw tooth signal. In this project, a prediction–based energy saving solution combining Kalman Filter and Grey Model will be implemented in CYPRESS Bluetooth Low Energy (BLE) platform. The one step predictor of Kalman Filter (KF) algorithm and Ending-point fixed Discrete Grey Model (EDGM) will be combined with weight parameters for measured value prediction. Both algorithms have low computational complexity but high prediction accuracy. Experiment results show that the sensor can reduce almost high time to sending and updating value to receiver. From this scenario, if the clusters or receivers in WSNs run the same algorithms with sensors, they do not need to update value from sensors but use the predicted value from their integrated algorithms. The energy consumption from sensors will be reduced by this prediction approach. The lifetime of WSNs could be prolonged.

# **Equipment& Objectives:**

## Equipment:

Hardware: BLE Pioneer Kit (CY8CKIT-042-BLE)

Software : PSoC Creator 3.3 SP2, CySmart 1.2, and Android App

## Objectives:

1. Simulate a temperature changed from 10℃ to 40 ℃

2. Implement Kalman Filter one step predictor algorithm

3. Implement EDGM algorithm for one step prediction

4. Combine the predicted value from KF and EDGM as final prediction value

5. Calculate the percentage of energy saving

6. Sending the percentage value Android phone to by Bluetooth protocol by using heart rate profile and app.

# **Preparing and Understanding**

This project will be implemented two one-step prediction algorithms named Kalman Filter and EDGM. Every one step prediction will be combine two algorithms with equal weights (0.5 for KF, 0.5 for EDGM). If the error () of next step is less than threshold, this means that the accuracy of predicted value can be accepted. Based on this prediction result and the prediction-based aggregation approach from [1], an prediction-based energy saving solution can be implemented.

The prediction-based energy saving scenario in WSNs will be composed of sensors and clusters or receiver. Taking a peer-to-peer communication for example. The sensor and receiver will be embedded as same prediction algorithms. So if the error () of next step is less than threshold, the sensor will not send the latest value to receiver. The receiver will use this predicted value as t+1 step of value to instead of actual value. On the contrary, if error is greater than the threshold, the latest value will be sent to receiver, the matrix of value will be synchronized. It is stated that the power consumption of transmission is far more than the power consumption of computation[2]. In this scenario, the more accurate value can be predicted, the less data will be transmitted from sensor to receiver, and the less power will be consumed.

Based the above solution and approach, the one prediction algorithms including Kalman Filter and EDGM will be programmed and implemented in Cypress BLE system. If the whole energy saving scenario would be implemented, the same prediction algorithms should be executed both in BLE hardware and receiver (iOS or Android phone App). However, due to the time limitation, there is no time to develop Apps, which runs Kalman Filter and EDGM algorithms. The alternative solution is that we will calculate a percentage value, which is the numbers of saving energy over numbers of temperature value sampling. To show this result significantly, this percentage value will be shown on iOS or Android phone, which is based on the Lab03 heart rate BLE profile. In summary, a percentage value, which means the energy saving, will be shown on Apps of iOS and Android phone based on BLE heart rate App.

# **Block Diagram:**

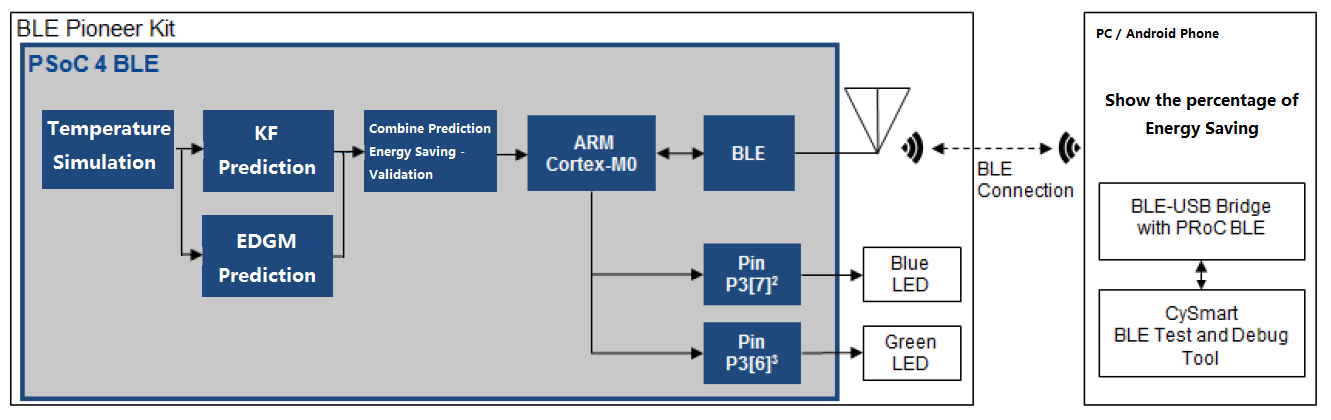


Figure 1: Block Diagram and Overview

# **Theory**

## Grey Model

The Grey Model is intervenient between the white system and the black system. This theory provides a powerful tool for modeling discrete series with a few data items and forecasting the near future value based on determination of an exponential equations[1]. The GM(1,1) is a basic model constructed on a single sequence in Grey Model theory. It uses only the behavioral sequence (output sequence or background values) of the system without considering any externally acting sequences (or referred to as input sequences, or driving quantities) [3].

The grey action quantity in the GM(1,1) model is a value derived from the background values. It reflects changes contained in the data and its exact intension is grey. This quantity realizes the extension of the relevant intension. To implement GM(1,1) into embedded system for on-line prediction, we need its discrete grey prediction models (DGM) . The details of DGM derivation will be introduced.

### Definition 1

The equation



is referred to as a discrete grey model (DGM) or a discretization of the GM(1,1)model.

### Theorem 2

Let  be a nonnegative sequence and its accumulation generation. , where  , . If  is the parametric sequence and



Then the least squares estimates of the parameters of the discrete model  satisfy , where









### Theorem 3

Let  be the same as defined in Theorem 2, and . Then it can be derived as following:

The iteration equations:  The restored values are



where . This form of the discrete grey model is referred with fixed starting points, so called SDGM.

### Theorem 4

Based on the similar derivation, the form of the discrete grey model with fixed end points, so called EDGM. The iteration equations is given by



where , and the restored valued given by



If we need a one-step prediction, so , the equation will be changed as



This equation will be implement in this project for one-step prediction.

## Kalman Filtering Predictor Equations

Form [4] the Kalman Filter Based on one-step prediction is shown as following

State equations:



Assumptions:

is gaussian with mean and covariance 

 are gaussian, zero-mean, white process independent with

Everything is Gaussian- linear operations preserve gaussian properties.



Kalman Predictor:











 satisfies a Riccati Difference Equation (RDE)

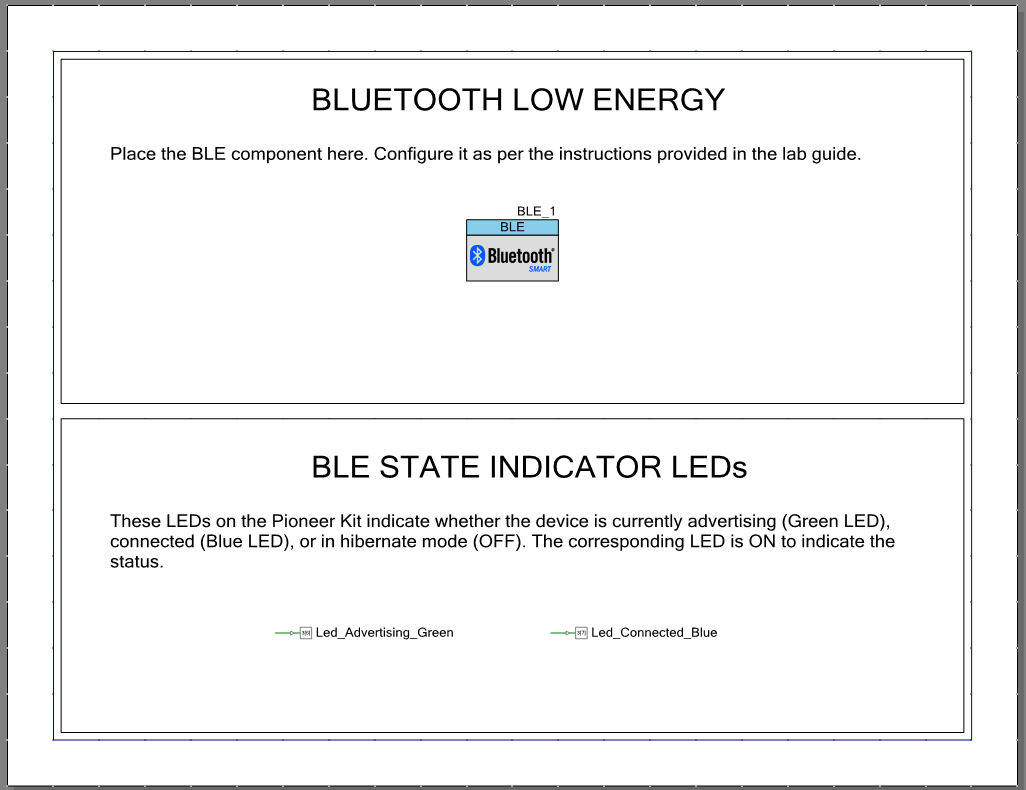
Based on the equation -, the one step prediction of Kalman Filter will be implement in this project. The correlation matrix of system noise Q could be set as 0.04, and the correlation matrix of measurement noise R could be set as 0.01. The initial value of covariance is set as 1.

# **Firmware:**

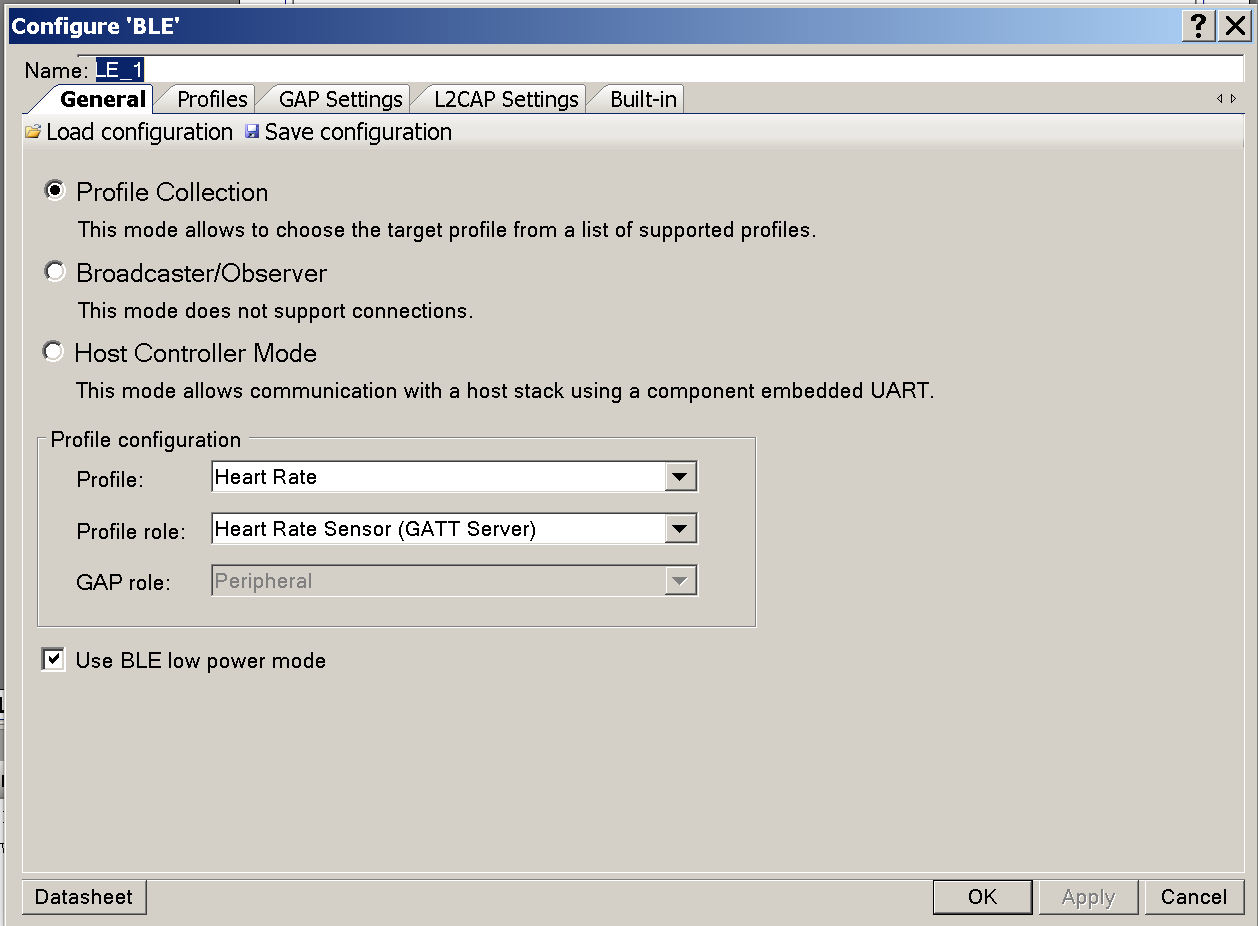
|  |  |  |
| --- | --- | --- |
| **File Name**  **(Sample time=1 sec)** | **Function** | **Description** |
| FinalProjectProgram | * Simulate Real Temperature * GM1stepprediction * KalmanFilterPredictor1 * EnergySavingValidation | * This is to generate a simulated temperature signal from 10 degrees to 40 degrees every sample time (1 second). It changes by 0.5 degrees every one second. We add a random noise from 0.02 to 0.2 degrees. * This uses the Grey Model prediction to predict one temperature for one step. * This uses the Kalman filter to predict temperature for one step. * This checks the energy saving percentage of the system. The sensor will not need to send an updated temperature as long as the difference between the predicted and the actual temperature is less than error threshold (0.1 degrees) |
| Heartrateprocessing.c | This file handles the energy saving validation percentage value. It repeats every one second to show the performance of the prediction. | HR\_WSNESPercentage=(unit8)(WSNESPercentageLF\*100.0); |

# **Steps:**

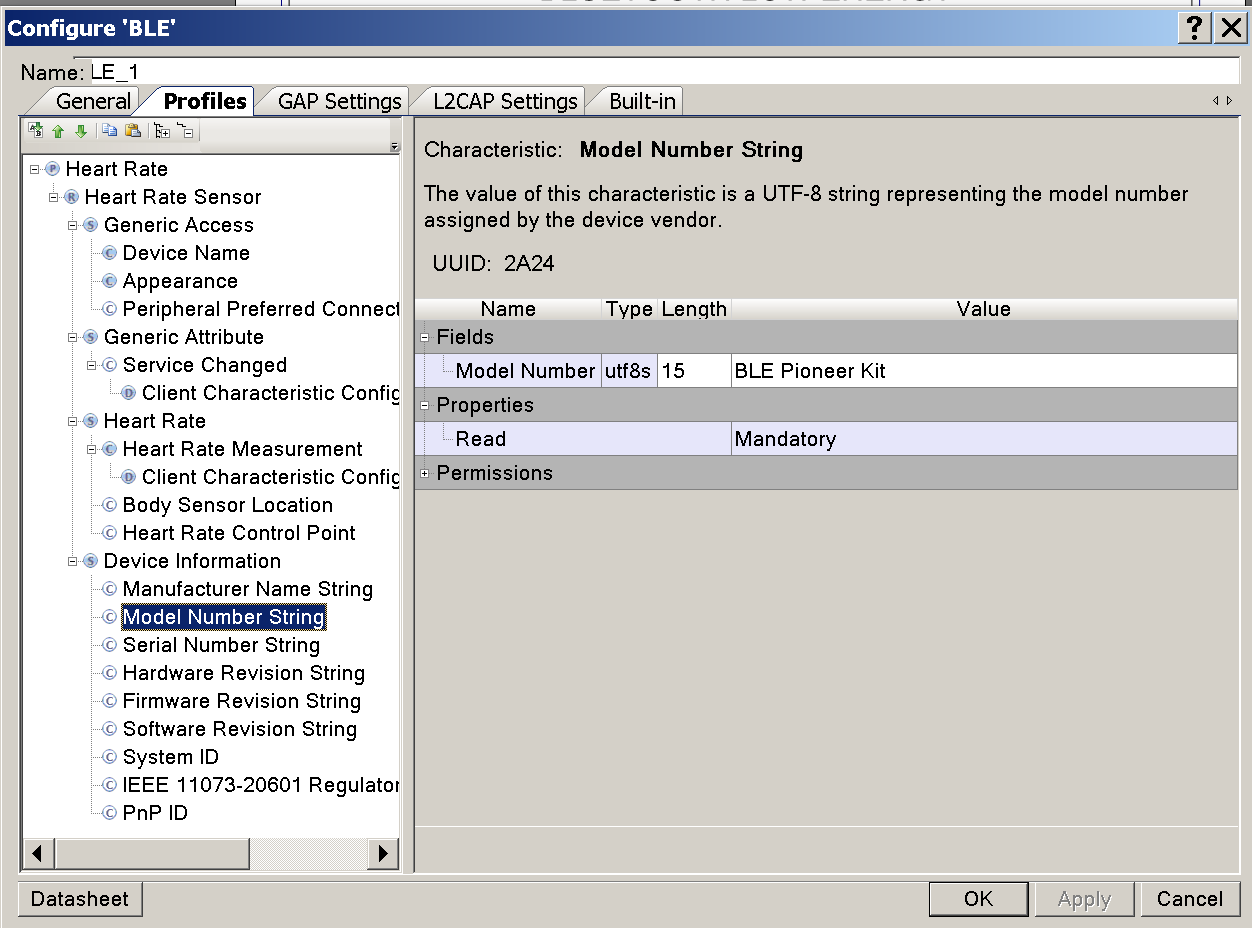
1. In the following, we used LAB 3 procedure and heart rate communication to send the energy saving percentage value. This has been done in addition to modifications in the C code.
2. Create a new project named Final Project: 



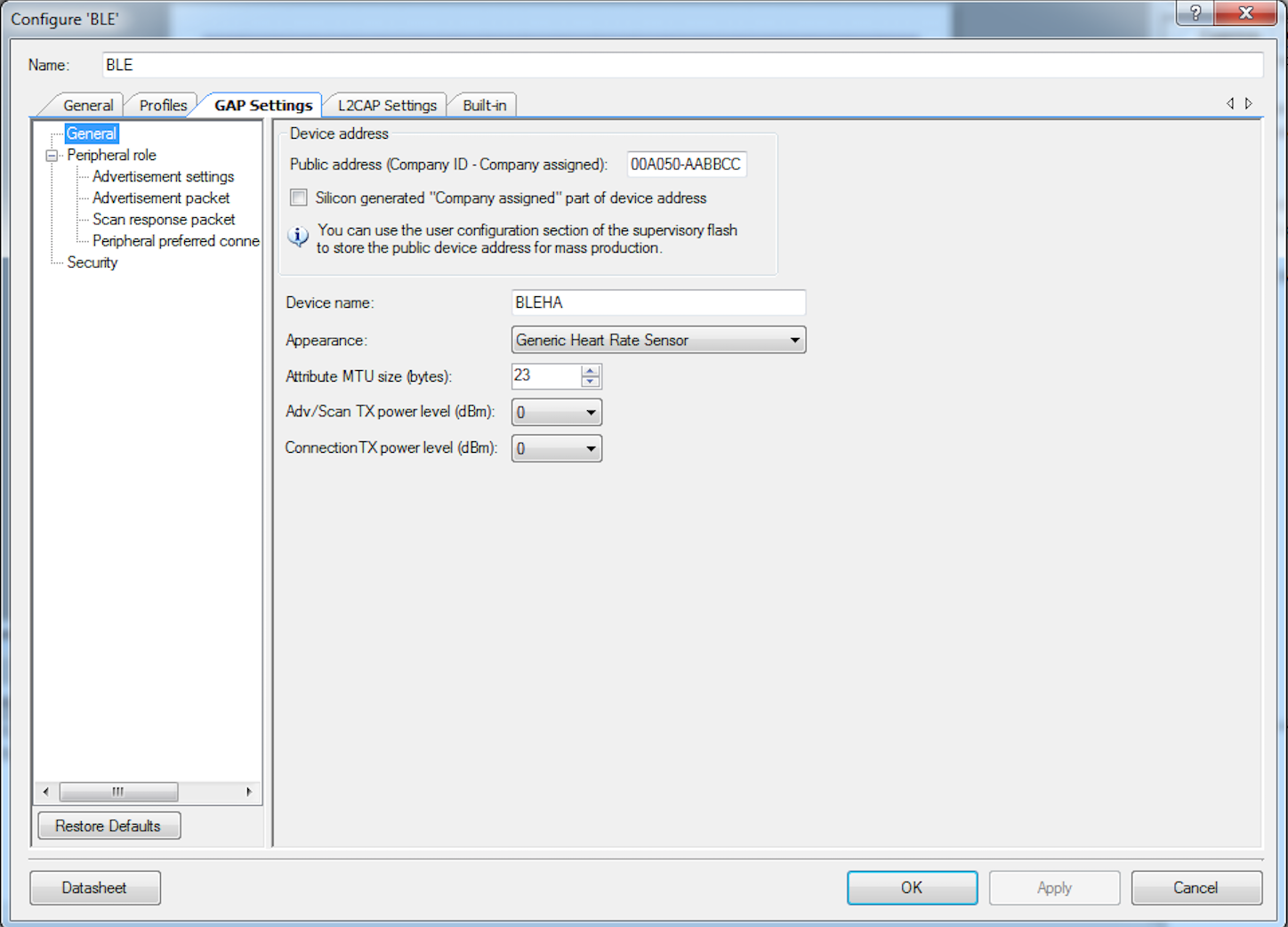
1. Click on the BLE component to open the configuration. Change the profile to Heart Rate, and make the role Heart Rate Sensor. This is shown in the figure below:



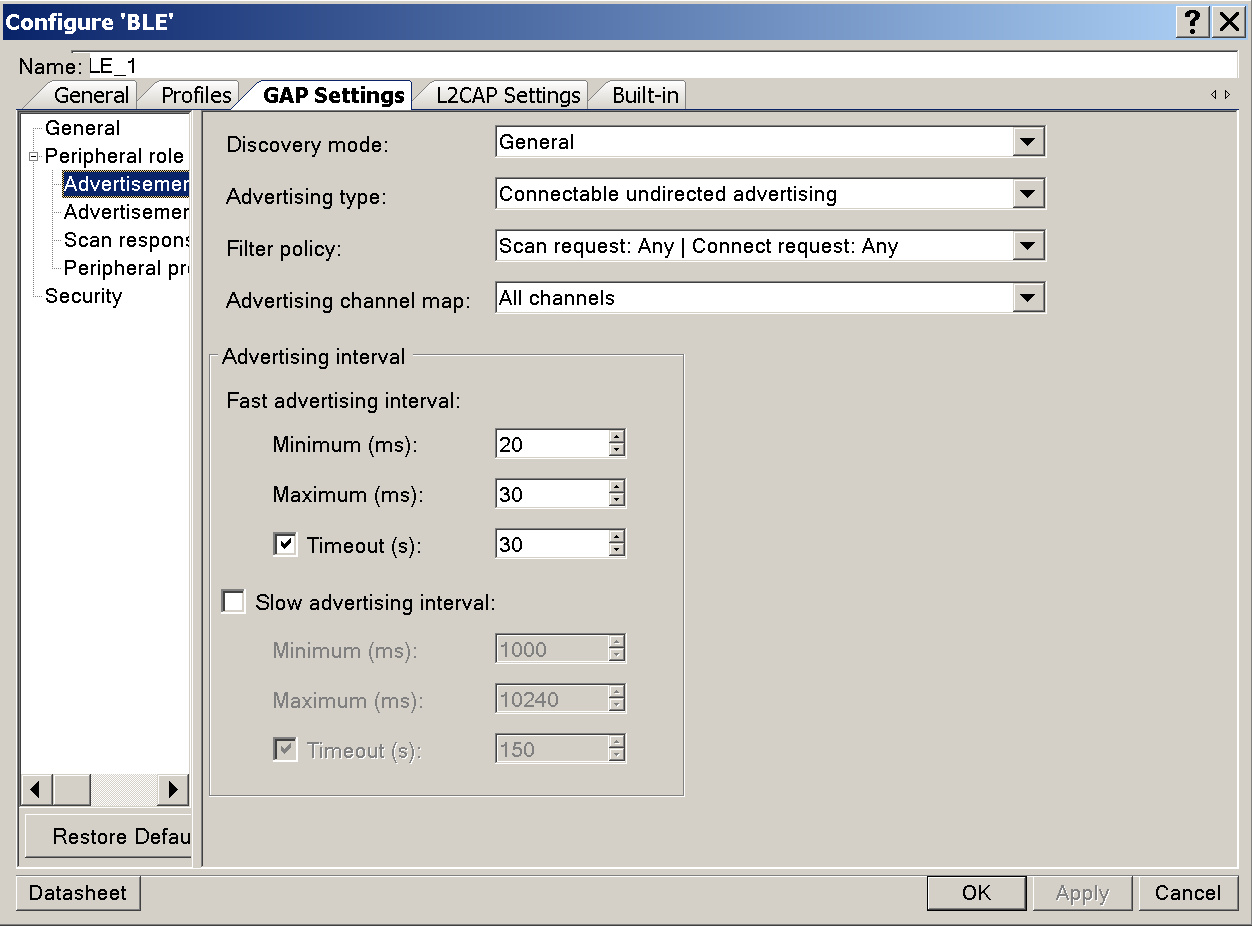
1. No changes are done in the profile tab as below:



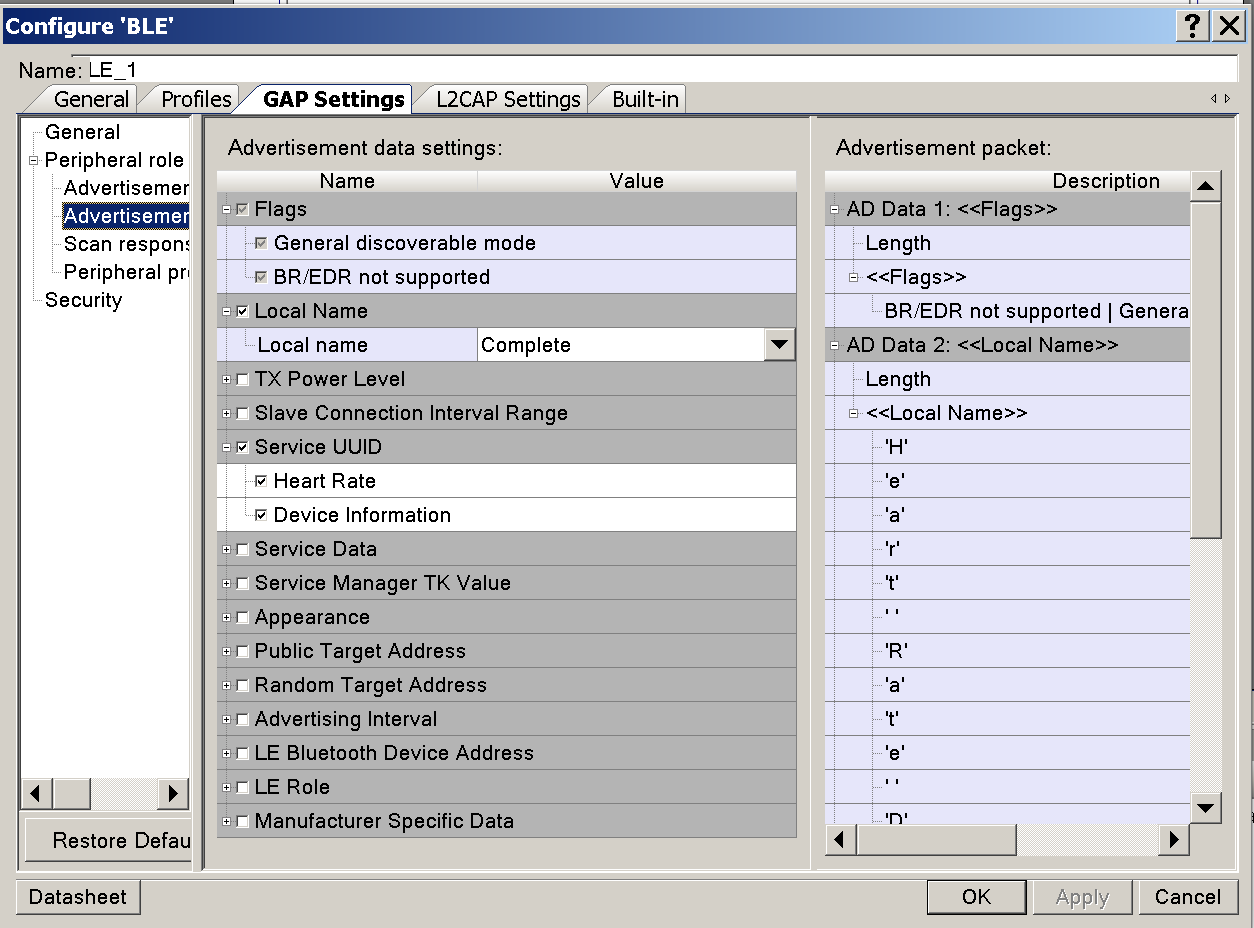
1. Click on the GAP settings, and change the device name and appearance as below:



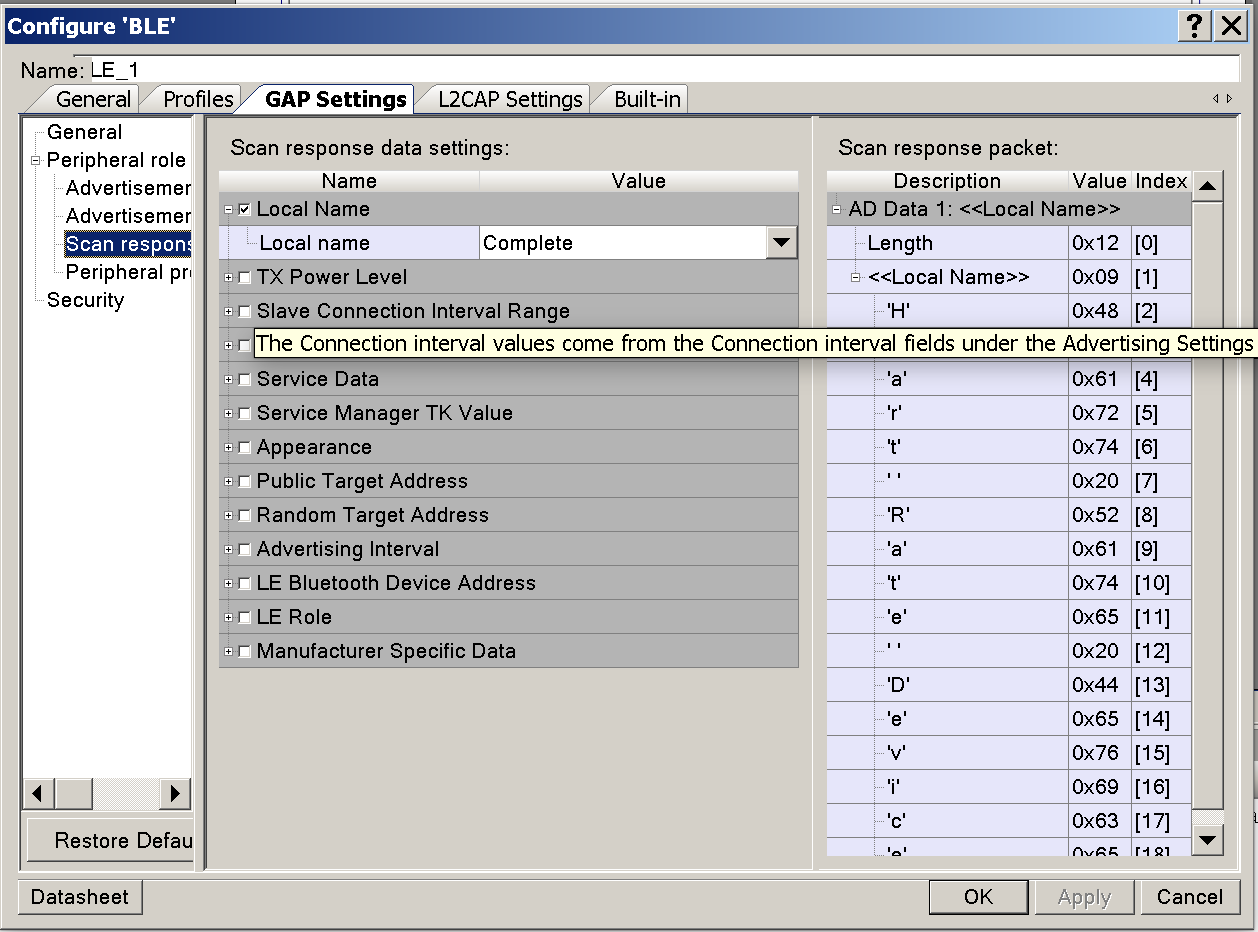
1. From the left click on advertisement settings then change the settings as below:



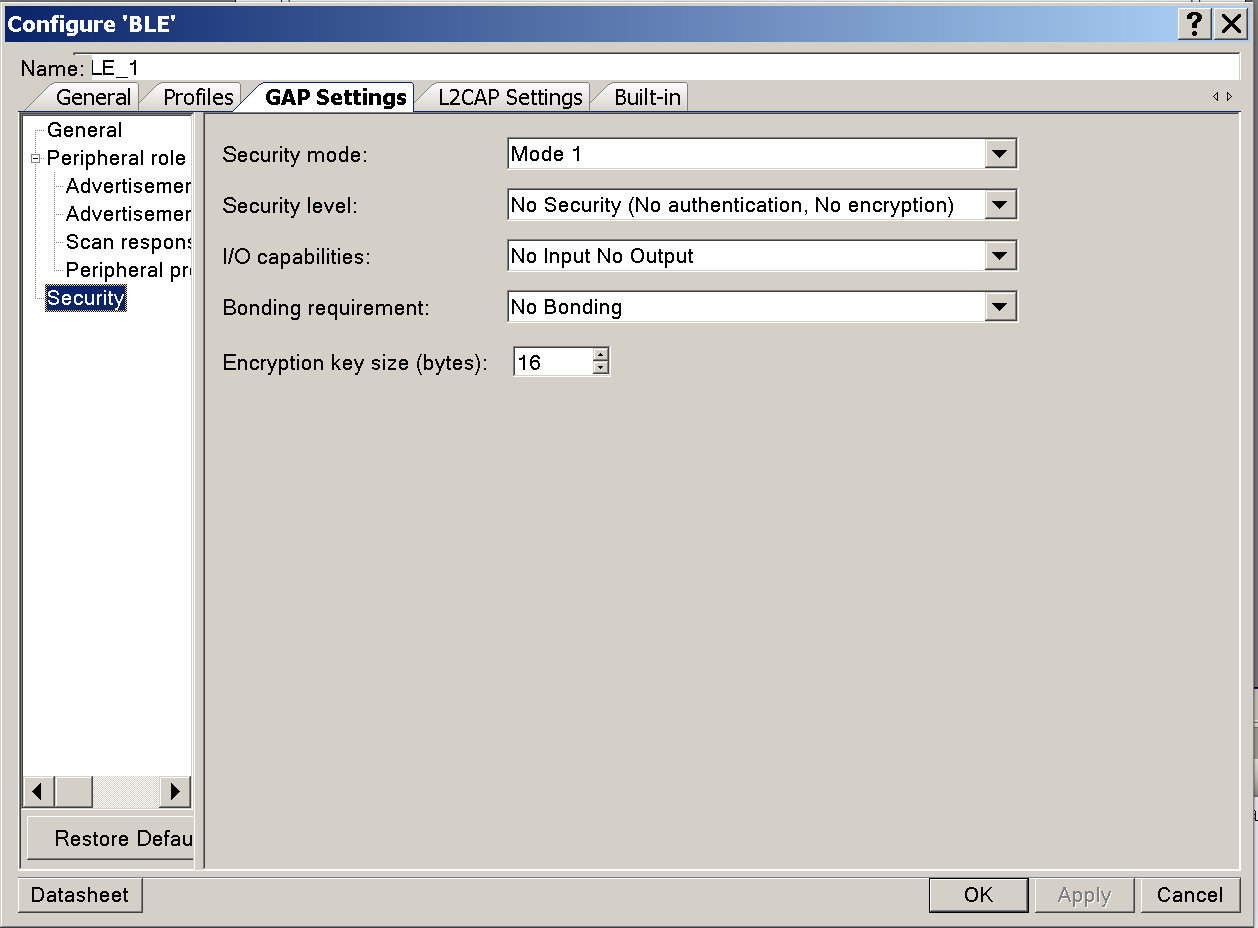
1. Click on Advertisement Packet from the left then apply the changes below:



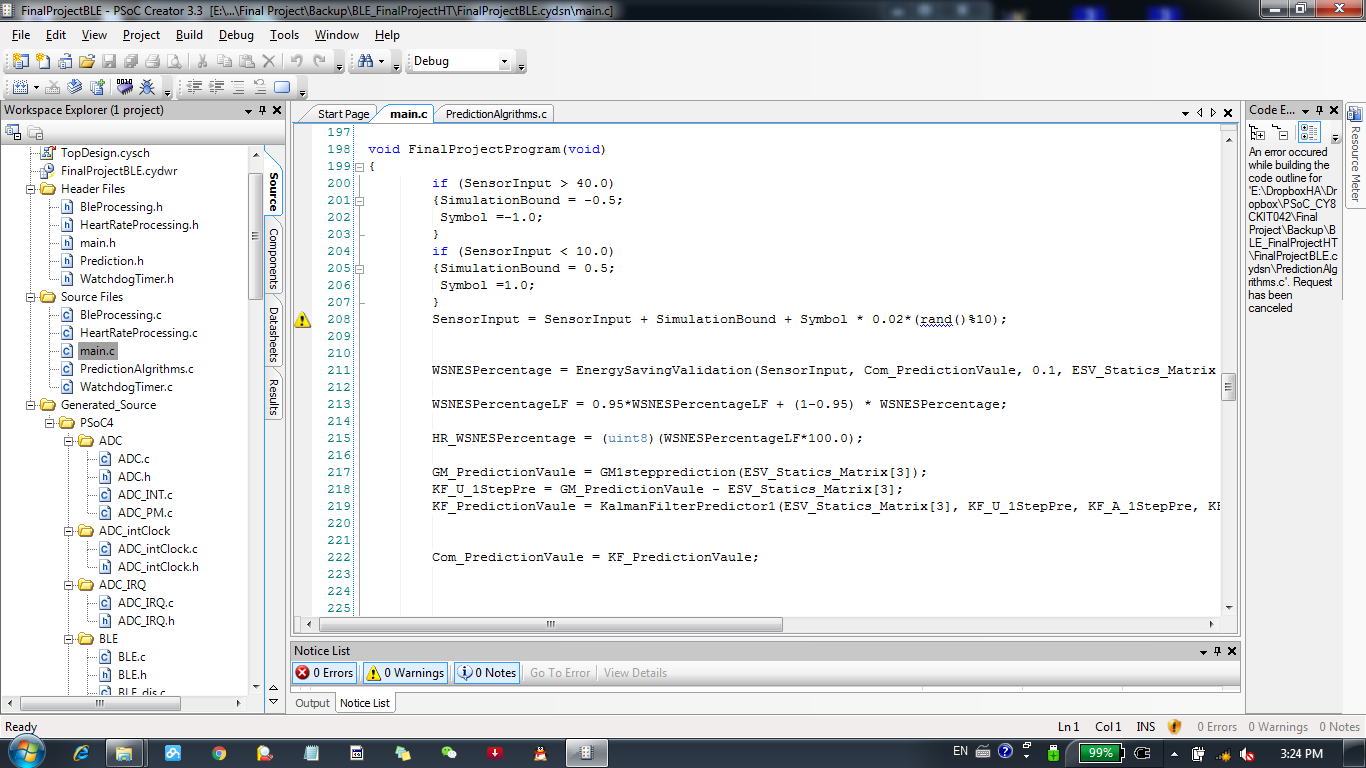
1. Change the properties of the scan response as below:

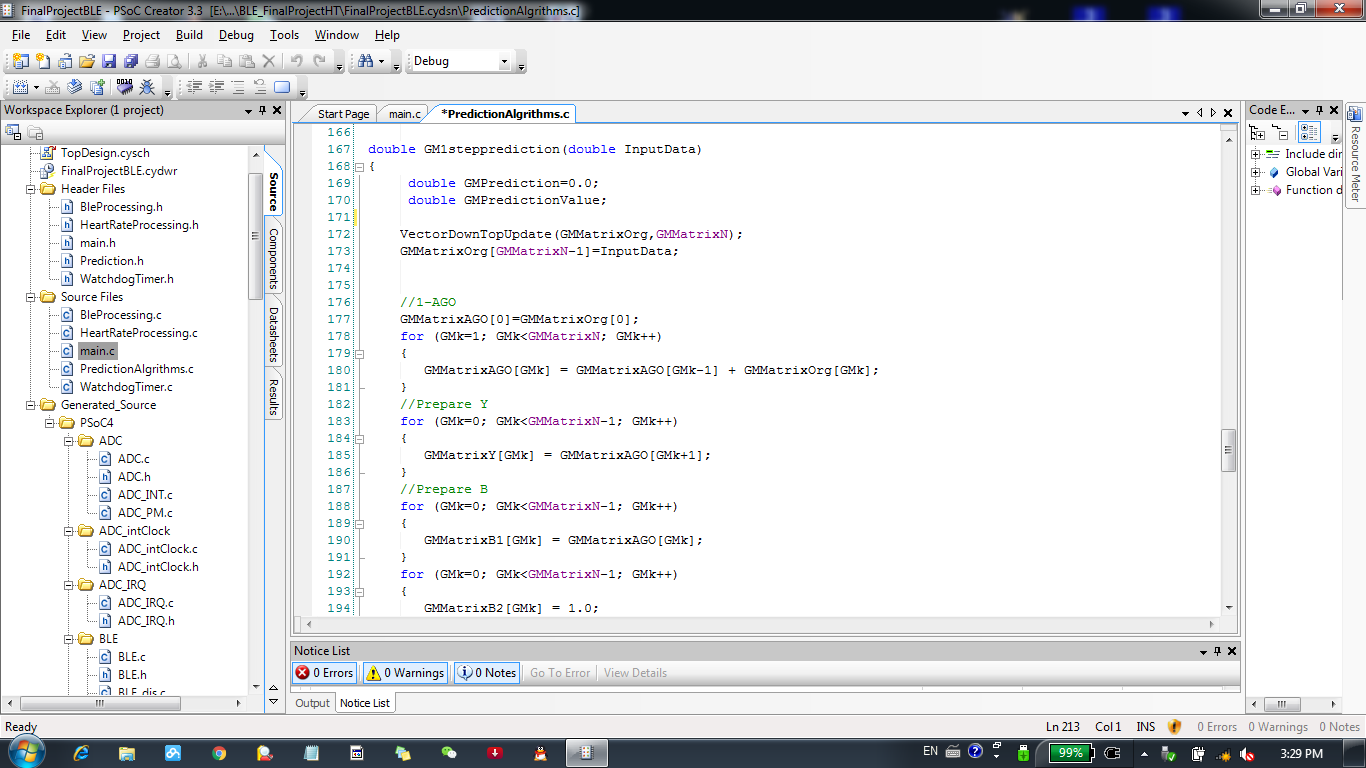


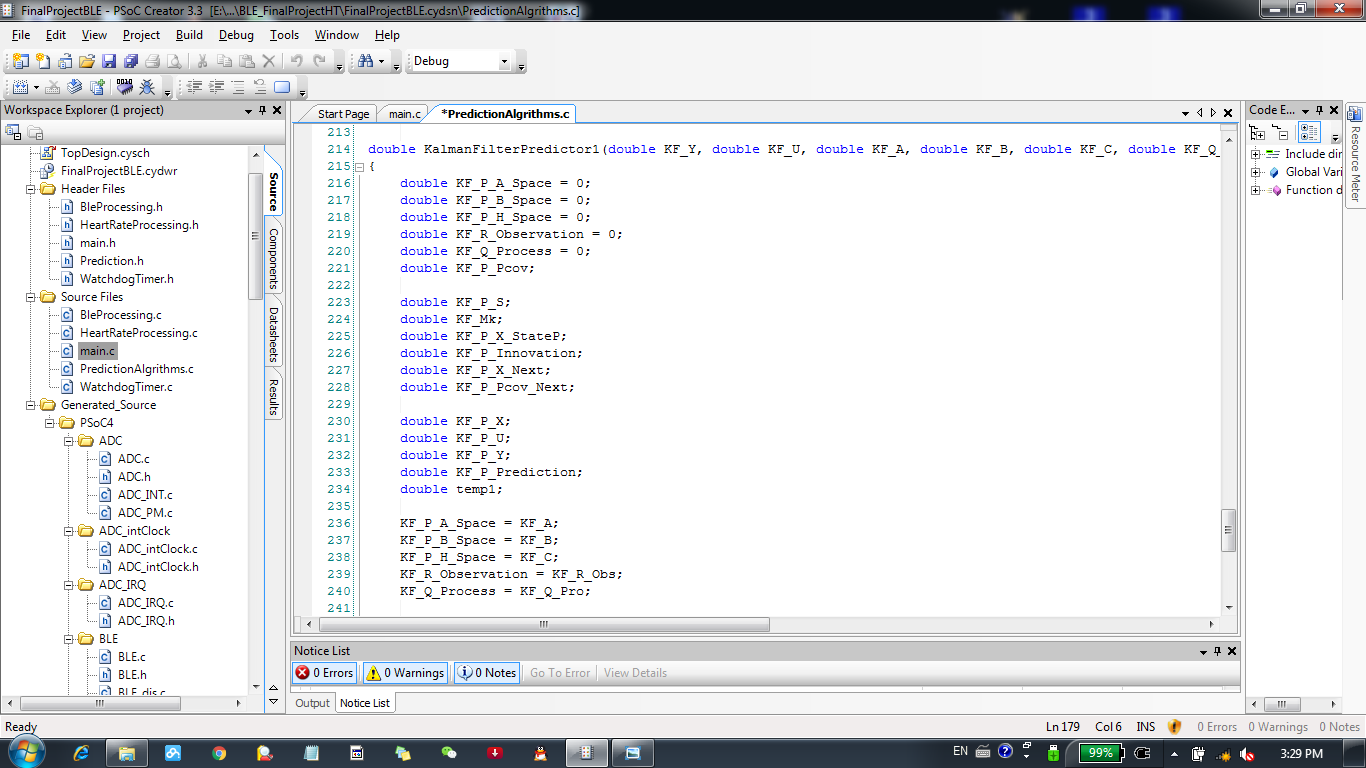
1. Change the security settings as below, then click ok to save and close the configuration settings.

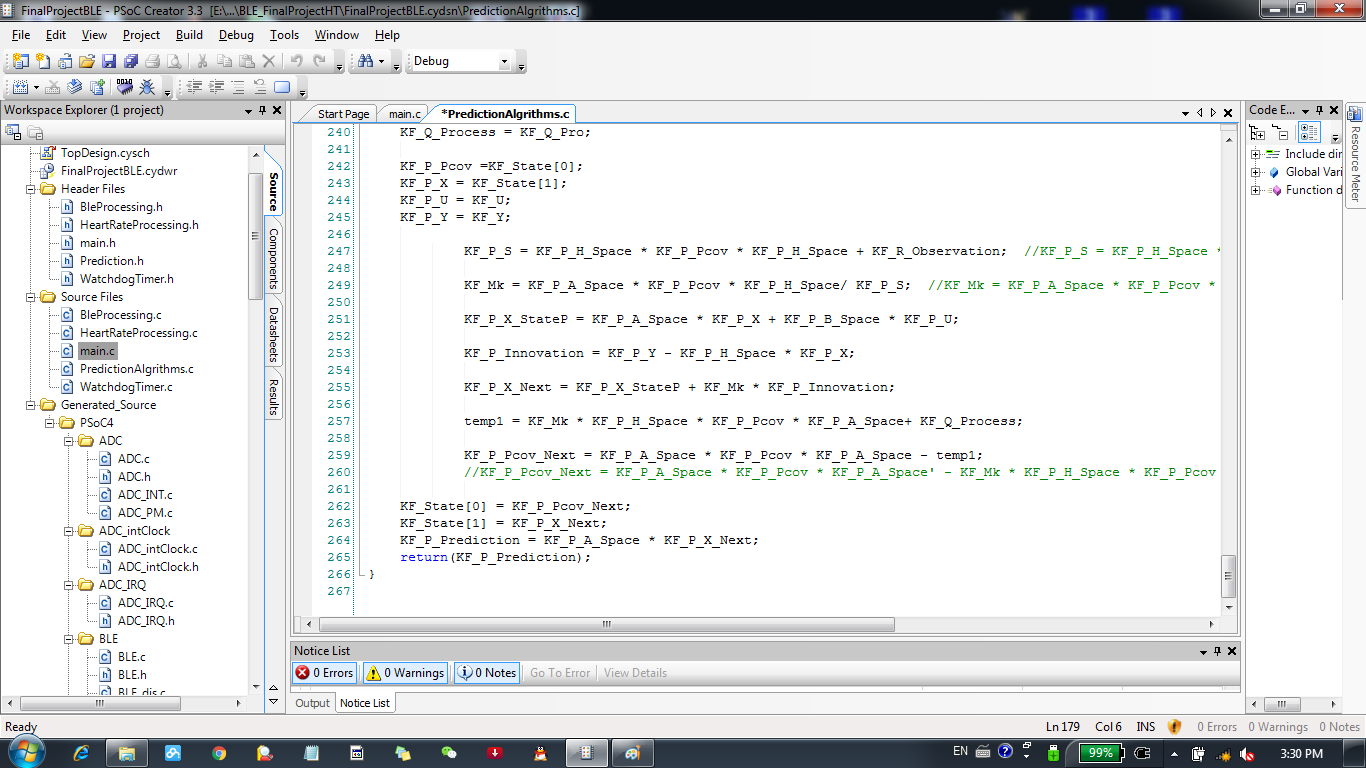


1. Click on Build the select Build BLE Final Project to compile the C code, make sure there are no errors after compiling the code.

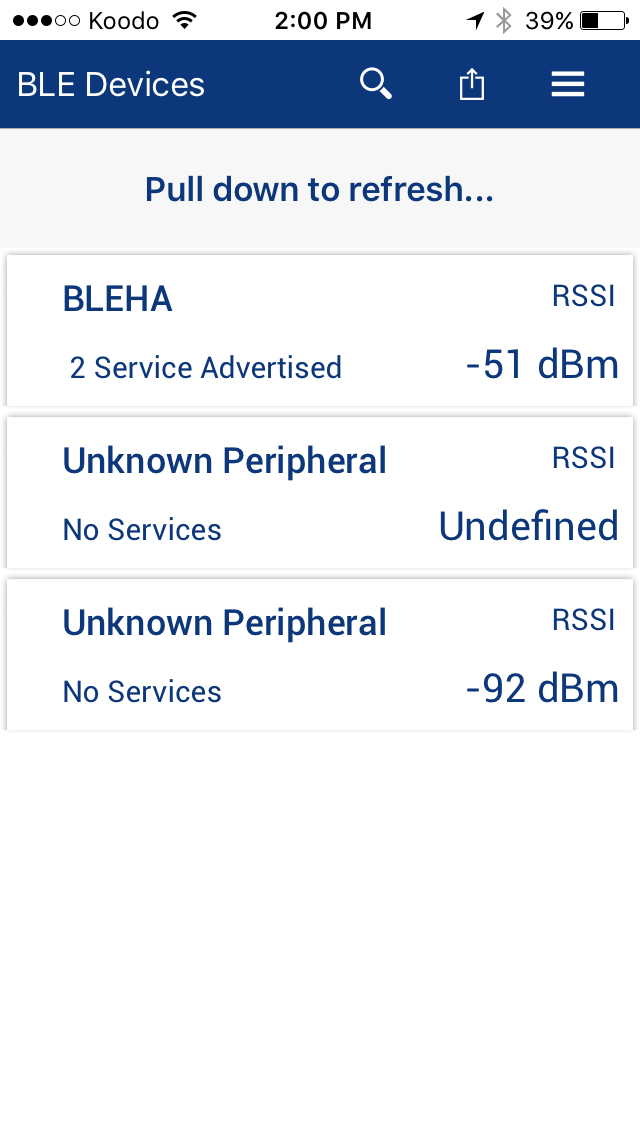




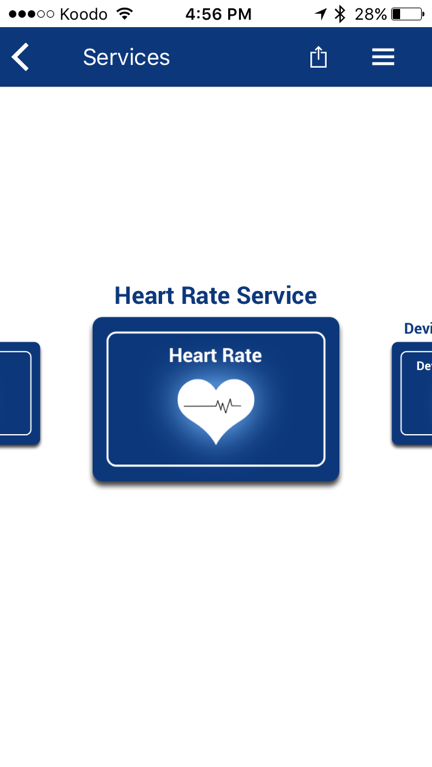




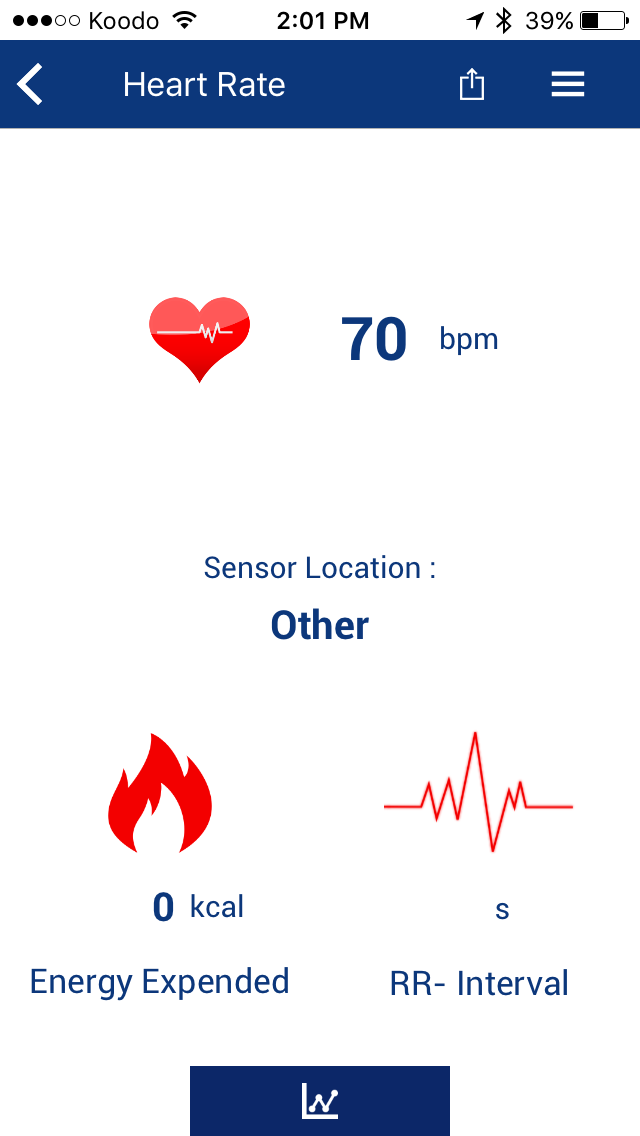
1. To test the configuration on IPhone, install CySmart App on the phone.



1. Choose the heart rate service as follows:



1. Read the value of energy saving:



# **Conclusion:**

The project aimed to simulate a practical temperature sensor and predictor. This sensor used the Grey Model and the Kalman Filter to predict the temperature. We used the Bluetooth model presented in lab 3 in the curriculum to send the energy saving percentage validation and used the heart rate measure as a tool to present this value. The percentage we got varied from 60 to 70 percent, which means that the sensor doesn’t need to send the updated measured temperature information around 70 percent of the time and save energy in the real case if the receiver and the sensor are running the same algorithms. For the future work, we can redesign the mobile application to make it a real temperature sensor and add another device as a receiver (or LCD ) to receive the measured temperature. It could be a very good approach to save energy in WSNs.

1. **References:**

[1] G. Wei, Y. Ling, B. Guo, B. Xiao, and A. V. Vasilakos, “Prediction-based data aggregation in wireless sensor networks: Combining grey model and Kalman Filter,” *Comput. Commun.*, vol. 34, no. 6, pp. 793–802, May 2011.

[2] M. Chen and M. L. Fowler, “Data compression trade-offs in sensor networks,” 2004, vol. 5561, pp. 96–107.

[3] S. Liu and Y. Lin, *Grey Systems: Theory and Applications*, vol. 68. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[4] S. S. Haykin, *Adaptive filter theory*, 4th ed. Upper Saddle River, N.J: Prentice Hall, 2002.