BabyWatcher

January 2020

BY

Gilad Apelstein

Tal Berger

Ethgar Daniely

SUPERVISOR

Prof. Sivan Toledo

PROJECT CARRIED OUT AT

Tel Aviv University

Introduction

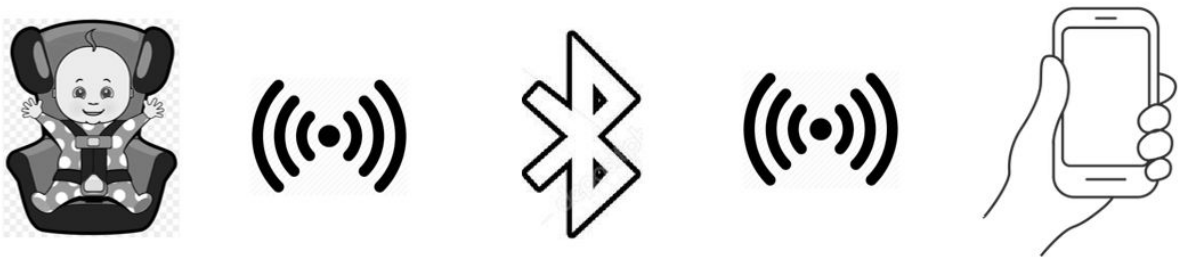
Description

BabyWatcher is an IoT solution designed to prevent adults from forgetting their babies in the rear seat of their cars. The parent uses a designated application that alerts in case a child is forgotten in the car.

Motivation

- From the beginning of 2008 until mid 2013, more than 200 parents in Israel forgot their children in the car. This number is 1% of the total children that died in the same years.
- In the USA, between the years 1998 and 2013 there were 650 incidents of forgotten kids in the car that led to their death.
- In 2019 a law was legislated in Italy, which obligated every car that drives kids until the age of 4 to have a sensor that prevents forgetting kids in the car.

Project Architecture



TI CC1350 Launchpad

We decided to implement a BLE-based application using TI CC1350 launchpad.

'CC1350 is a wireless MCU targeting low power, long range wireless applications with Bluetooth low energy implementations.'

(Wikipedia)

Board Application

- We used Code Composer Studio 7.3.0 for developing the application running on the board.
- The board application is based on the BLE stack implementation provided by TI as part of their examples (simple_peripheral_cc1350lp_app_FlashROM).
- We used the built-in ADC buffer of the board to acquire the measurements from the pressure sensor.
- We implemented a custom GATT profile containing 2 characteristics:
    - Mode characteristic

      The value of the characteristic controls the mode of the BabyWatcher according to the following specifications:
        - 0x00 – Off mode. The BabyWatcher is switched off by stopping the ADC conversion and the periodic clock that updates the profile. When BabyWatcher's state is switched from off to on, it is first

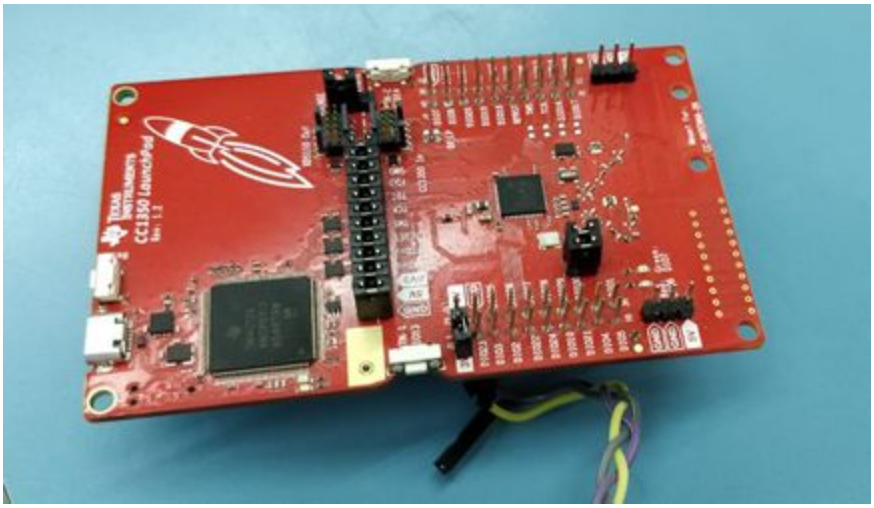
"turned on" by starting the ADC conversion and the periodic clock that updates the profile.

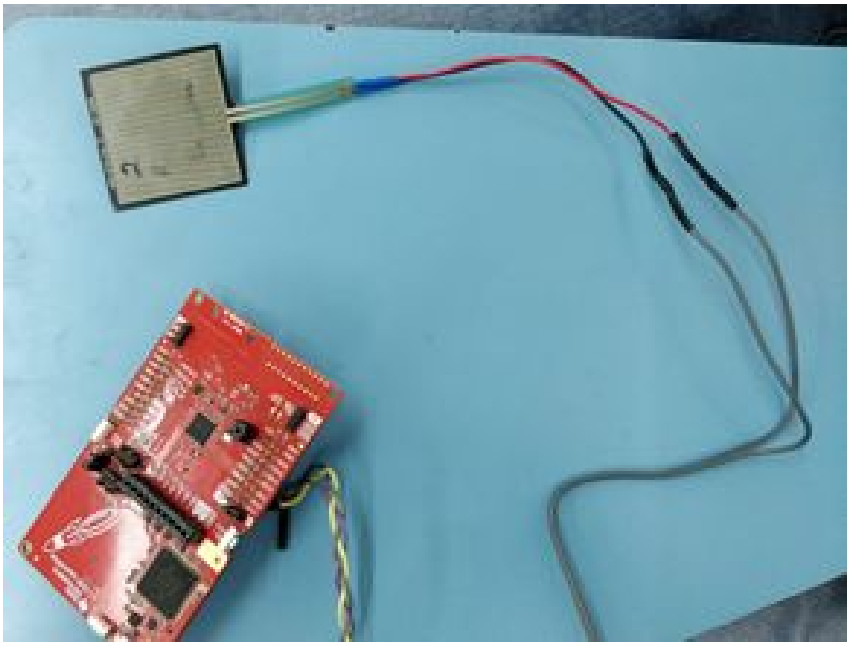
    - 0x01 – On mode. The MCU is switched on. It starts the ADC conversion and sends the data through BLE to the application.

- Measurement characteristic

  The value of the characteristic represents the value of the measurements according to the BabyWatcher's mode:

- Mode On – the value is the measured voltage in microvolts and is sent to the application by bluetooth (BLE).
- Mode Off – nothing is measured and nothing is sent to the application.
- The measurements are done using the ADCBuf driver. The sampling is done via pin DIO03.
- Pictures of the Board

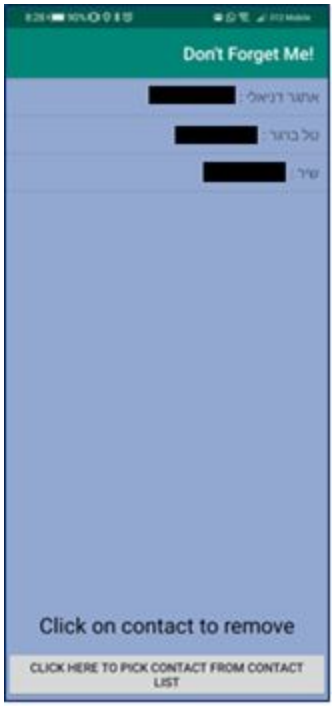


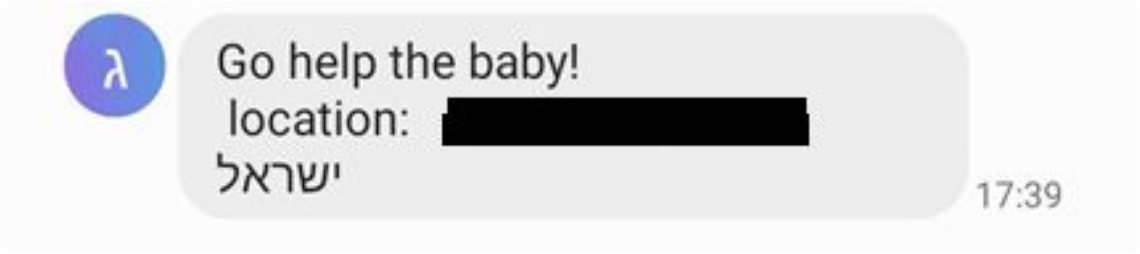


Android Application

- We developed an android app using Android studio and implemented a GUI that provides an interface to interact with the BLE BabyWatcher.
- We developed our application over Android's BLE example app.

- Different States of the Android Application





Issues During Development

Board Issues

a. Issue with installing framework

After downloading simple_peripheral example from the internet, we tried to compile it in Code Composer Studio 9.2.0. The project did not compile even after reinstalling the software and redownloading the project.
After a few tries, we understood that we didn't have the proper drivers installed on our Windows OS. Only after installing the drivers, did the project compile and we could start coding and testing.

b. Bugs with sending data from board to android app

i. We found out that many apps in the app store that are meant to communicate via BLE with the board are not supported by our smartphone versions. Finally, we found that LightBlue is indeed supporting our new versions and we were able to send and receive data from and to the board.

ii. It took us time to implement sending and receiving data from and to the board. Finally, we understood that are several characteristics, and each one holds a unique role.

c. Board was not connecting to application after one use

The scenario was connecting the application to the board via BLE, disconnecting, and then reconnecting. Upon the latter reconnect, the board did not send signals via BLE.

Finally, we found out that the problem was that disconnection has to be bidirectional - not only from the Android app to the board app, but also from the board app to the Android app. After fixing the disconnection from the board to the android app, the problem was solved.

Application issues

a. Could not connect to board

After finding the board BLE on the application, we could not connect to it. After debugging this issue, we discovered that we didn't enable BLE service in the XML application manifest.

Upon adding it, the problem was solved.

b. Make the application run in the background

It was hard for us to make the app run in background and wake up upon being close to the board - we were debugging it for a few days.

The solution was to make a service that runs in the background and scans for bluetooth devices on a regular basis.

Appendix

- Android application source code
- Board application source code