CIS 441 & 541: Embedded Software for Life-Critical Applications

Cloud-based Smart Pacemaker-Challenge Project

9 Feb 2022

Objective

This project is to familiarize the students with the basic aspects of the implementation, verification, and validation of life-critical CPS/IoT applications. In particular, the project is to learn how to develop life-critical embedded real-time systems and cloud-based IoT applications.

Problem Description

You are to design a cloud-based smart pacemaker. This IoT application contains three parts: a heart, a pacemaker, and a cloud application which collects and displays the heart-pace data from the pacemaker.

(a) Heart

A simulated heart will be implemented to generate an ECG signal belonging to either a healthy or unhealthy heart. When abnormal heartbeats occur, the signal will be “paced” by inputs from the pacemaker.

(b) Pacemaker

A pacemaker is a medical device, which analyzes ECG data and sends electrical impulses to the heart in order to maintain the proper heart rhythm of a patient.

Timing constraints are one important aspect of correctness criteria. In this project, you will mainly focus on the implementation, verification, and validation of a VVI mode pacemaker controller with timing constraints. The heart implemented above will be essential for comprehensively testing your pacemaker controller.

More information of the Pacemaker can be found at

- The specification document http://sqrl.mcmaster.ca/_SQRLDocuments/PACEMAKER.pdf.

(b) Thingsboard application

The cloud application needs to have three functionalities: 1) receiving and storing the heart and pace data from your mbed and 2) displaying this data on a real-time plot in Thingsboard

Your project is divided into the following milestones that are to be carried out in sequence. **Each milestone has its own deadline**. **Please start early.**

Milestone 1: Implementation of the Smart Pacemaker

You are to actually implement the smart pacemaker with MBED devices and thingsboard. First, you will implement a random heart simulator that will generate an ECG signal on one MBED. Second, you will send the ECG amplitude to a second MBED containing the Pacemaker. The Pacemaker will analyze the ECG data over time and send a pace back to the heart if needed. Third, the Pacemaker will send data to Thingsboard via MQTT to display the data in a real-time plot. The Pacemaker should also accept commands from thingsboard. These commands can be to start or stop the pacemaker, or update its parameters.

Milestone 2:Verification, and Security of the Smart Pacemaker

In the first part, you are asked to design a UPPAAL (http://www.uppaal.com) model of the implementation of the smart pacemaker. Using UPPAAL, you can create timed automata models and can verify properties specified in computational tree logics (CTL). Based on the model, you will verify properties of the pacemaker to make sure the design is (conceptually) correct.

The UPPAAL model must be consistent with your implementation of the pacemaker controller. This consistency will be used as an argument for the correctness of the implementation in your assurance case (Milestone 3). This phase will involve reading the pacemaker specification document, identifying correctness properties, specifying the properties in CTL, and employing the UPPAAL tool to verify the correctness of your design.

In the second part, you will secure the pacemaker to thingsboard communication layer. You can use existing security features of MQTT or your own cryptographic protocol.

Milestone 3: Validation, Demo, and Final Report

In this milestone, you are asked to develop an assurance case to argue the correctness of your system as a whole. This is done by profiling and/or testing of your implementation to gather enough evidence. Moreover, you are also required to validate the monitoring of the heart, communication between your mbed and the cloud, and the application deployed over the cloud using test cases. All the evidence is put together in the assurance case to justify your system-level claim. The assurance case also involves providing arguments that the properties you have verified in Milestone 2 are also not violated in your implementation.

Each group must submit a final report that includes the implementation architecture, design models and correctness properties, and assurance case. Also, each team is expected to demo their system either in-class or in a lab.

Collaboration Policy

The project can be done in groups of two or three(ideal). It is **not** advised that each member take one milestone and work alone for the following reasons: First, the workloads of the milestones are different. Second, it would be beneficial for all members to get to know the process of the model-based software design of an embedded and cyber-physical system.

Assessment

Final grade for the project is based on the items below. All of the following should appear in the project report.

(20%)	Milestone 1(a): Heart and Pacemaker MBEDs
(10%)	Milestone 1(b): Thingsboard and Remote Control
(20%)	Milestone 2(a): Verification
(20%)	Milestone 2(b): Security
(20%)	Milestone 3(a): Validation with Assurance Cases
(5%)	Milestone 3(b): Teamwork and Member Contribution
(5%)	Milestone 3(c): Demo

Notes

Please note that there isn't a single correct solution to the smart pacemaker project. You are encouraged to find creative solutions for any aspect of this project.

You are also encouraged to design and implement a cloud-based smart wearable device other than a smart pacemaker. If so, please talk to the instructor or TAs. You will be asked to write a short proposal to state the functionality of the device. We will evaluate the feasibility and difficulty of your proposed project and provide feedback/approval.