SFWRENG/MECHTRON 3K04

Deliverable 1

Deliverable 1 is 20% of your final grade

Due: October 27, 2025 - 8 a.m.

Introduction

There are two major components to this deliverable. The first part involves creating real-time software on the hardware platform. The second part involves properly documenting the development that occurs during the time of the deliverable. The main objective of this first deliverable is for you to be able to parse documentation provided to distill the specific information necessary to build a functional pacemaker and Device Controller–Monitor (DCM). We expect you to properly identify the requirements for this project (for the pacemaker and DCM) and build a design that satisfies their requirements. You will need to implement their design faithfully (Simulink for the pacemaker, the language of your choice for the DCM) and write appropriate documentation for your efforts. If you are confused or unclear about information or tasks to complete, you are expected to reach out to a TA during labs.

Prerequisites

1. Familiarity with the (natural language) PACEMAKER Requirements Document. Specific sections that you need to understand at this stage are:

1	Introduction
2.1 - 2.2	Overview/Components
3.1 - 3.2	DCM *(English only, 3.2.2, 3.2.3, 3.2.4 only 1 and 2, for 3.2.5 display egrams without markers)*
3.4	Pacing Pulses
3.5 - 3.6	Modes and States
4.7	*Real-Time Electrograms (egrams)*
5	Modes and Their Programmable Parameters
Appendix A	Programmable Parameters

   Some of the details/specifics in the sections from the list above such as egrams, will be more relevant for the next deliverable. The document *srsVVI* rev 2 provides formal requirements for VVI mode. You should consult this document if you are unsure of the exact pacing requirements you will need in the future. Also, use this document for a *serial protocol for communications, and details regarding egrams (electrograms).*

2. Familiarity with Pacemaker Microcontrolled Shield document to understand how the pacemaker shield operates

3. Familiarity with software development on the hardware platform

Part 1 - Pacemaker Design

Use Simulink to implement stateflows for the following pacemaker modes for a pacemaker in permanent state: AOO, VOO, AAI, and VVI (Hint: Think of the labs and the charts on Simulink). The stateflow should use the programmable parameters listed in the requirements document. You can find them using the references in the 'Prerequisites' section above. Specifically: the pulse characteristics (width, and regulated amplitude), rate characteristics (limits, and delays) and what chamber(s) are being paced. We very much **DO NOT** want other pulse characteristics. Refer to Part 3 for instructions on mapping pins that are referenced by the model. The stateflow should not change if the pinmap is altered, but rather the correct pin should map to its corresponding component.

Hardware Hiding

Each team will be given a pacemaker board, shield, and testing set that allows you to see the pacing signal as it is output from the pacemaker and into the 'heart'. These signals can be observed using the HeartView desktop application. You are required to apply hardware hiding to map the correct pins to the Simulink model (Hint: use a Simulink Subsystem to do this). This will also help you in future labs where the models will become more complex. Remember that the idea is to abstract away the hardware from the design. Use the large table in the Pacemaker Microcontrolled Shield document to help with the mapping between hardware and design.

Part 2 - DCM Design

The DCM is detailed in the natural language PACEMAKER Requirements. There is also information in the *srsVVI* rev 2 document about what specific aspects of the DCM you should concentrate on. You are required to design and operate the DCM on your computer. Your team has the option of programming the DCM in a programming language of your choice. Hint: Python is a common choice to create a simple GUI (with the added bonus of a stable serial communication library, introduced in Deliverable 2).

For this deliverable, you are required to:

1. Develop an interface that includes a welcome screen, including the ability to register a new user (name and password), and to login as an existing user. A maximum of 10 users should be allowed to be stored locally.

2. Develop essential aspects of the user interface – with respect to 3.2.2 in PACEMAKER, you should include: 1, 2 (input buttons), 3, 4, and 7.

3. Develop interfaces to present all of the pacing modes mentioned in Part 1 to the user.

4. Make provision for storing programmable parameter data for checking inputs – for the purposes of this assignment the parameters we want to see specifically on the DCM are: Lower Rate Limit, Upper Rate Limit, Atrial Amplitude, Atrial Pules Width, Ventricular Amplitude, and Ventricular Pulse Width, VRP, ARP. The complete set is in PACEMAKER document on page 28.

5. Develop and document appropriate date structures for egram data required in future assignments.

6. Implement any other requirements you elicit from the documentation that is not explicitly stated in this assignment document.

For this deliverable, you are not required to implement the communications between the DCM and Pacemaker (see Information regarding the next deliverable below). The scope of the DCM portion of this deliverable is to implement the presentation layer (the "front-end") of the DCM user application.

Part 3 - Documentation

It is critical for safety intensive applications to have full and proper documentation of the systems that you are developing. Documentation should be clear and concise. Documentation should cover full product lifecycle development. This includes:
Part 1:

1. List all requirements. This should be done formally (as in not just a bunch of bullet points), but as a group you can decide what formalism you want to use (mathematical, informal, somewhere in between).

2. Requirements must be concise and disjoint. They should also be traceable to the design and vice versa

3. List and justify all design decisions.

4. The Simulink diagram must include necessary annotation to understand the model. There should be screenshots of the simulink model in the documentation. Provide an additional section in the document that describes testing you performed, and the results.

5. DCM code should be commented and easy to read/understand. Screenshots should be in documentation.

6. Validation and verification of the DCM and Pacemaker.

Part 2:

1. List all requirements that are likely to change.

2. List all design decisions that are likely to change.

3. For each module:

   (a) Describe the purpose of the module.

   (b) List public functions provided by the module – with their parameters.

   (c) Describe the black-box behaviour of each function (part of the Module Interface Specification).

   (d) Describe any global variables in the module that are within scope for all functions in the module – we call them state variables. You must describe the date structure where appropriate (part of the Module Internal Design).

   (e) List private functions in the module, if any (part of the Module Internal Design).

   (f) Describe the internal behaviour of each public and private function in enough detail that you can code from it with ease. Pay particular attention to how these functions maintain the state variables, or provide the values of state variables (part of the Module Internal Design).

   Describe testing and results of those tests. This includes whether the test passes or fails. Remember, you should document every part of development, even when things don't work the way you expect it to. There is a lot to learn from the failures that can occur during development!
   Note: Your documentation is a living document that evolves as you progress through the project. By the end of the project we should be able to see every step that was taken towards completion within the respective documentation. We will share a documentation outline that you are welcome to use as a template if you wish.

Deliverables

You are required to submit documentation as a PDF format. You may either submit separate documents for the DCM and Simulink, or combine the documentation of both software components into one PDF. The file(s) must be included in a .zip folder that is submitted to Avenue by any team member. Your code for the DCM and Simulink programs should be located in a code folder which must also be included in the .zip submission folder. The .zip submission folder should be of the format deliverable1_group#.zip (you will be able to see your group number on Avenue).

Documentation Folder	docs_group#
Simulink Folder	simulink_group#
DCM Folder	DCM_group#

Information regarding Deliverable 2

1. Implement AOOR, VOOR, AAIR, VVIR modes.

2. Generate assurance of the system.

Design Principles

This is a course on software development, and design principles are a huge part of being able to achieve dependable, safe and secure applications. As such, you should pay particular attention to the following design principles as you develop this project:

1. Separation of concerns – in particular, effective modularity.

2. Your designs must be robust with respect to anticipated changes – apply information hiding.

3. High cohesion and low coupling of modules.

4. Proper documentation so that we can onboard new group members if needed.

What do you need to do?

1. All design, documentation, and code pushed to your repository if you have one and in a zip file submitted on Avenue.
2. Demo your project in the lab to the TAs

Grading

1. Demo: **60% Marks**

Pacemaker/Simulink	25
DCM/GUI	25
Discussion	10

2. Documentation: **40% Marks**