

3K04 Deliverable 2: Documentation

Group 37

Last Updated: 2025-11-28

Contents

List of Figures	vi
List of Tables	vi
1 Group Members	1
2 Abbreviations	2
2.1 General Abbreviations	2
2.2 Bradycardia Operating Abbreviations	2
3 Introduction	3
4 Requirements	3
4.1 System Requirements	3
4.2 Programming Requirements	4
4.3 Hardware Requirements	4
4.4 DCM Requirements	5
4.5 Serial Requirements	5
4.6 Egram Requirements	6
4.7 Rate Adaptivity Requirements	6
5 Design	7
5.1 Simulink Design - Deliverable 1	7
5.1.1 Overall Design	7
5.1.2 Input Constant Variables	8
5.1.3 Monitored Input Variables	9
5.1.4 Stateflow Modules	10
5.1.5 AOO Stateflow Model	11
5.1.6 VOO Stateflow Model	12
5.1.7 VII Stateflow Model	13
5.1.8 AAI Stateflow Model	14
5.1.9 Hardware Hiding	15
5.2 Simulink Design - Deliverable 2	16
5.2.1 General Overview	16
5.2.2 Serial Communication Module	16
5.2.3 Rate Response	19
5.2.3.1 All Modes	19
5.2.3.2 Mode Stateflow	20
5.2.4 Outputs	26
5.2.4.1 Ventricular Inhibiting	26
5.3 DCM Design	27
5.3.1 Model	27
User Model	27
Pacing Model	27
Egram Model	27

5.3.2	View	28
Login View		28
Main View		28
5.3.3	Controller	28
Controller		28
Main		29
5.3.4	Views of DCM GUI	29
5.4	DCM & Serial Design - Deliverable 2	32
Serial_Comms		32
5.4.1	Serial Packet Structure	32
5.4.2	Main View Design & Decisions	36
5.4.3	Parameter Pages Design & Decisions	38
5.4.4	EKG View Design & Decisions	40
5.5	Assurance Cases	42
5.5.1	Clamping of Heart Rate For Rate Adapative Pacing	42
5.5.2	Echo Parameter Verification	43
5.5.3	Reset Between Mode Switching	45
5.5.4	Incorrect Device Handling	46
5.5.5	Push Button Shutoff	48
6	Requirements Potential Changes	49
7	Design Decision Potential Changes	49
8	Module Description	50
8.1	Simulink System (Model-Based Design)	50
8.2	DCM Application	50
9	Testing	53
9.1	SimuLink Mode Testing	53
9.1.1	Testing of AOO	53
9.1.2	Testing of VOO	54
9.1.3	Testing of AAI	56
No Natural Heart Rate		56
Natural Heart Rate of 45 BPM		57
Natural Heart Rate of 75 BPM		58
9.1.4	Testing of VVI	59
No Natural Heart Rate		59
Natural Heart Rate at 45 BPM		60
Natural Heart Rate at 75 BPM		61
9.1.5	Hysteresis Testing	62
Hysteresis Test 1 (60 BPM)		62
Hysteresis Test 2 (50 BPM)		63
Hysteresis Test 3 (40 BPM)		64
9.2	Testing of AOOR	65
9.2.1	Test 1: AOOR at Rest (No Movement)	65
9.2.2	Test 2: AOOR with Increased Movement	67

9.2.3	Test 3: AOOR Maximum Sensor Rate	68
9.3	Testing of VOOR	69
9.3.1	Test 1: VOOR at Rest (No Movement)	69
9.3.2	Test 2: VOOR with Increased Movement	71
9.3.3	Test 3: VOOR Maximum Sensor Rate	72
9.4	Testing of AAIR	73
9.4.1	Test 1: AAIR at Rest (0 BPM)	73
9.4.2	Test 2: AAIR at Rest (45 BPM)	74
9.4.3	Test 3: AAIR at Rest (75 BPM)	75
9.4.4	Test 4: AAIR with Increased Movement (45 BPM)	76
9.4.5	Test 5: AAIR Maximum Sensor Rate (45 BPM)	77
9.4.6	Test 6: AAIR Maximum Sensor Rate (75 BPM)	78
9.5	Testing of VVIR	79
9.5.1	Test 1: VVIR at Rest (0 BPM)	79
9.5.2	Test 2: VVIR at Rest (45 BPM)	80
9.5.3	Test 3: VVIR at Rest (75 BPM)	81
9.5.4	Test 4: VVIR with Increased Movement (45 BPM)	82
9.5.5	Test 5: VVIR Maximum Sensor Rate (45 BPM)	83
9.5.6	Test 6: VVIR Maximum Sensor Rate (75 BPM)	84
9.6	Rate Adaptive Tuning Testing	85
9.6.1	Test 1: Recovery Time Testing (30 seconds)	85
9.6.2	Test 2: Recovery Time Testing (60 seconds)	86
9.6.3	Test 3: Reaction Time Testing (10 seconds)	89
9.6.4	Test 4: Reaction Time Testing (30 seconds)	91
9.7	DCM Testing	93
9.7.1	Login and Registration	93
9.7.2	Parameter Input Validation	96
9.7.3	Mode Selection and Data Retrieval	99
9.7.4	Font Size Accessibility Controls	100
9.7.5	Manual Entry of “OFF” in Programmable Field	100
9.7.6	Error When Incrementing Values Not by 0.1	101
9.7.7	Amplitude Range 0.1–5.0 V	102

10 GenAI Usage	106
-----------------------	------------

List of Figures

1	Overall Simulink Mapping	7
2	Constant Input Variables	8
3	Monitored Input Variables	9
4	Stateflow Modules	10
5	AOO Stateflow Model	11
6	VOO Stateflow Model	12
7	VVI Stateflow Model	13
8	AAI Stateflow Model	14
9	Hardware Hiding of Model	15
10	General Overview of Simulink Model - Deliverable 2	16
11	Serial Communication Module Simulink - Deliverable 2	17
12	Communication Input Block - Deliverable 2	17
13	Send Parameters Block - Deliverable 2	18
14	All Modes and Default State Stateflow Diagram - Deliverable 2	19
15	AOOR Stateflow - Deliverable 2	20
16	VOOR Stateflow - Deliverable 2	20
17	AAIR Stateflow - Deliverable 2	21
18	VVIR Stateflow - Deliverable 2	21
19	Rate Adaptive Logic - Deliverable 2	22
20	Desired Pace Logic - Deliverable 2	22
21	Rate Adaptive Interval Block - Deliverable 2	23
22	Stateflow for Mode Switching of Rate Adaptive Logic - Deliverable 2	23
23	Rate Adaptive Mode Reset When Switching Logic - Deliverable 2	24
24	Accelerometer Reading Block Diagram - Deliverable 2	24
25	Accelerometer and Inputs Sampling Rates - Deliverable 2	25
26	Custom Moving Average - Deliverable 2	25
27	Outputs Block - Deliverable 2	26
28	Ventricular Inhibiting Logic - Deliverable 2	26
29	Login Screen of DCM	29
30	Main Menu of DCM	30
31	Data Entry View	30
32	Switch in Device Detection from DCM	31
33	Conversion of Data In Simulink	34
34	Mapping of Modes to Integers In Simulink	35
35	DCM Main Page - Deliverable 2	36
36	Warning of Potentially Wrong Device	37
37	Updated Parameter Page for AOO Deliverable 2	38
38	Serial Communication Simulink - Deliverable 2	39
39	DCM View of AAIR - Deliverable 2	40
40	View of EKG Page	41
41	EKG View with Data	41
42	Clamping of Heart Rate 1	42
43	Clamping of Heart Rate 2	42
44	Echo Stateflow in Simulink	43

45	Send Params Function in Simulink	44
46	Echo Appearing in DCM	45
47	Default State in Simulink	46
48	Potentially Wrong Device Warning in DCM	47
49	Simulink Inhibiting Ventricular Pulses	48
50	AOO Test	53
51	Close-up of AOO Pulse	54
52	VOO Test	55
53	Close-up of VOO Pulse	55
54	AAI Test No Heart Rate	56
55	AAI Test 45 BPM	57
56	AAI Test 75 BPM	58
57	VVI Test No Heart Rate	59
58	VVI Test 45 BPM	60
59	VVI Test 75 BPM	61
60	Hysteresis Test 1	62
61	Hysteresis Test 2	63
62	Hysteresis Test 3	64
63	AOOR at Rest Ignoring Pulses	65
64	AOOR at Resting Pace Rate	66
65	AOOR at Increased Pace Rate from Movement	67
66	AOOR at Maximum Pace Rate	68
67	VOOR at Rest and No Natural Heart Rate	69
68	VOOR at Rest with a Natural Heart Rate	70
69	VOOR with an Increased Heart Rate from Movement	71
70	VOOR at Maximum Sensor Rate	72
71	AAIR with no Natural Heart Rate at Rest	73
72	AAIR with a Natural Heart Rate at 45BPM at Rest	74
73	AAIR with a Natural Heart Rate at 75BPM at Rest	75
74	AAIR with a Natural Heart Rate at 45BPM at with Movement	76
75	AAIR with a Natural Heart Rate at 47BPM at Max Sensor Rate	77
76	AAIR with a Natural Heart Rate at 75BPM at Max Sensor Rate	78
77	VVIR with no Nratural Heart Rate at Rest	79
78	VVIR with a Nratural Heart Rate of 45BPM at Rest	80
79	VVIR with a Nratural Heart Rate of 75BPM at Rest	81
80	VVIR with a Nratural Heart Rate of 45BPM with Movement	82
81	VVIR with a Nratural Heart Rate of 45BPM with Max Sensor Rate	83
82	VVIR with a Nratural Heart Rate of 75BPM with Max Sensor Rate	84
83	AAIR at Time 0s After Movement	85
84	AAIR at Time 30s After Movement	86
85	AAIR at Time 0s After Movement	87
86	AAIR at Time 60s After Movement	88
87	AAIR at Time 0s (Before Movement)	89
88	AAIR After 10s of Shaking	90
89	AAIR at Time 0s (Before Movement)	91
90	AAIR After 30s of Shaking	92

91	Registration Test	93
92	Registration Result	94
93	Login Test	94
94	Login Result	95
95	DCM Parameter Page with Values Filled	96
96	DCM Parameter Error When Number Inputted is Out of Range	97
97	DCM Parameter Error When Non-Number is Inputted	97
98	DCM Parameter Error When URL is Larger Than LRL	98
99	Stored Parameter File	99
100	Stored Users File	99
101	Accessibility Interface: Font Size Adjustment	100
102	Allowing of OFF In Parameters	101
103	Incorrect Incrementing of Value Test	102
104	Input Validation Error Dialogue (Too high)	103
105	Input Validation Error Dialogue (Too low)	104
106	Input Validation Error Dialogue (Negative Values)	105

List of Tables

1	Table of Group Members	1
2	Bradycardia Operating Abbreviations	2
3	System Requirements	3
4	Programming Requirements	4
5	Hardware Requirements	4
6	DCM Requirements	5
7	Serial & Data Requirements	5
8	Serial & Data Requirements	6
9	Rate Adaptivity Requirements	6
10	Parameter Settings	33

1 Group Members

Table 1: Table of Group Members

Name	MacID	Student Number
Ryan Su	sur21	400507973
Cameron Lin	lin422	400535393
Braden McEachern	mceacb1	400527617
Damian Szydlowski	szydlowd	400512629
Menakan Thamilchelvan	thamilcm	400510755

2 Abbreviations

2.1 General Abbreviations

BPM - Beats Per Minute

CCS - Cardiac Conduction System

DCM - Device Controller-Monitor

GPIO - General Purpose Input Output

GUI - Graphical User Interface

PWM - Pulse Width Modulation

2.2 Bradycardia Operating Abbreviations

Table 2: Bradycardia Operating Abbreviations

Category	Chambers Paced	Chambers Sensed	Response to Sensing	Rate Modulation
Letters	O-None	O-None	O-None	R-Rate Modulation
	A-Atrium	A-Atrium	T-Triggered	
	V-Ventricle	V-Ventricle	I-Inhibited	
	D-Dual	D-Dual	D-Tracked	

3 Introduction

It is hard to underestimate the importance of the human heart. The heart is the core part of the cardiovascular system; supplying nutrients and oxygen to all the cells and removing carbon dioxide, especially to vital organs such as the brain, it is imperative for it to be working flawlessly and harmoniously at all times. Unfortunately, however, cardiovascular diseases are a leading cause of death globally, many of which are caused from complications with abnormal heart rhythms. A pacemaker is an implantable device capable of sending timed electrical impulses causing contractions at appropriate intervals. Understanding the operation and design of this life saving device will aid in developing more efficient and reliable cardiac assistive technology.

The purpose of this project is to design and implement a system that operates a cardiac pacemaker under specified modes. This project will be accomplished through an understanding of embedded systems and through engineering principles of software development.

The scope of this deliverable is to design and implement the embedded pacemaker software, driver software and user interface for the DCM while updating and maintaining documentation.

4 Requirements

4.1 System Requirements

Table 3: System Requirements

ID	Requirement	Rationale
SR1	System Modes	The system shall implement AOO, VOO, VVI, and AAI pacing modes as the basic bradycardia modes required by the spec.
SR2	Hardware Hiding	The system shall use a hardware abstraction layer that maps logical control signals to GPIO pins, which improves maintainability and supports future hardware changes.
SR3	Simulink and DCM Separation	The pacing logic shall be implemented in Simulink and the DCM as a separate GUI so that embedded logic and clinical interface can evolve independently.

4.2 Programming Requirements

Table 4: Programming Requirements

ID	Requirement	Description / Rationale
PR1	Programmable Pulse Amplitude	The pacemaker shall generate atrial and ventricular pacing signals with amplitudes configurable by the user. Adjustable amplitudes allow tuning of pacing strength.
PR2	Programmable Pulse Width	The pacemaker shall generate atrial and ventricular signals with pulse widths configurable by the user so timing of stimulation can be adjusted.
PR3	Programmable Rate Timing	The pacemaker shall have programmable lower rate limit (LRL) and upper rate limit (URL) that control the minimum and maximum pacing rates, preventing bradycardia and tachycardia.
PR4	Programmable Refractory Periods	Atrial and ventricular pulse modes shall implement refractory periods during which sensed events are ignored to avoid double sensing and ringing.
PR5	Programmable Sensitivity	A programmable sensitivity threshold for event detection shall be adjustable from the DCM so that sensing can be adapted to patient signals and noise.
PR6	Pacing Responses	Each pacing mode shall follow the response implied by its letters: O for asynchronous pacing that ignores sensing, I for inhibited pacing, and T for triggered pacing from sensed pulses.

4.3 Hardware Requirements

Table 5: Hardware Requirements

ID	Requirement	Description / Rationale
HR1	Hardware Hiding Layer	A hardware abstraction layer shall map digital logic signals to analog front end hardware pins so that the pacing model does not reference physical pins directly, improving readability and portability.
HR2	Front End Enable Control	A dedicated signal shall enable or disable the front end sensing circuitry so ADCs and amplifiers can be powered only when needed, reducing noise and power consumption.

4.4 DCM Requirements

Table 6: DCM Requirements

ID	Requirement	Description / Rationale
DCM1	User Authentication	The DCM GUI shall provide user registration and login functionality supporting up to ten stored users to control access to the programming interface.
DCM2	Parameter Display and Editing	The DCM shall display and allow editing of pacemaker parameters such as LRL, URL, atrial and ventricular amplitude, pulse width, and sensitivity so that clinicians can program the device.
DCM3	Status Indicators	The DCM shall provide visible indicators of device connection status and communication loss so the user remains aware of system state and faults.

4.5 Serial Requirements

Table 7: Serial & Data Requirements

ID	Requirement	Description / Rationale
COM1	The DCM and Pacemaker shall communicate via a defined Serial Protocol (UART).	Establishes communication between the pacemaker and DCM, replacing the mock communication.
COM2	The DCM shall verify the integrity of sent parameters by reading them back (Echo Verification) from the device.	Ensures safety by confirming data was stored correctly.
DAT1	The system shall support programmable Pulse Widths from 1ms to 30ms (1ms increment).	Updated requirements from the previous deliverable.
DAT2	The system shall support programmable Pulse Amplitudes from 0.1V to 5.0V (0.1V increment).	Updated requirements from the previous deliverable.

4.6 Egram Requirements

Table 8: Serial & Data Requirements

ID	Requirement	Description / Rationale
EKG1	The DCM shall receive real-time electrogram (egram) data points from the pacemaker via Serial.	Allows the doctor to visualize heart activity.
EKG2	The DCM shall plot two simultaneous graphs (Atrial and Ventricular) in real-time.	Required for diagnostic visibility.

4.7 Rate Adaptivity Requirements

Table 9: Rate Adaptivity Requirements

ID	Requirement	Description / Rationale
EKG1	The DCM shall receive real-time electrogram (egram) data points from the pacemaker via Serial.	Allows the doctor to visualize heart activity.
EKG2	The DCM shall plot two simultaneous graphs (Atrial and Ventricular) in real-time.	Required for diagnostic visibility.

5 Design

5.1 Simulink Design - Deliverable 1

5.1.1 Overall Design

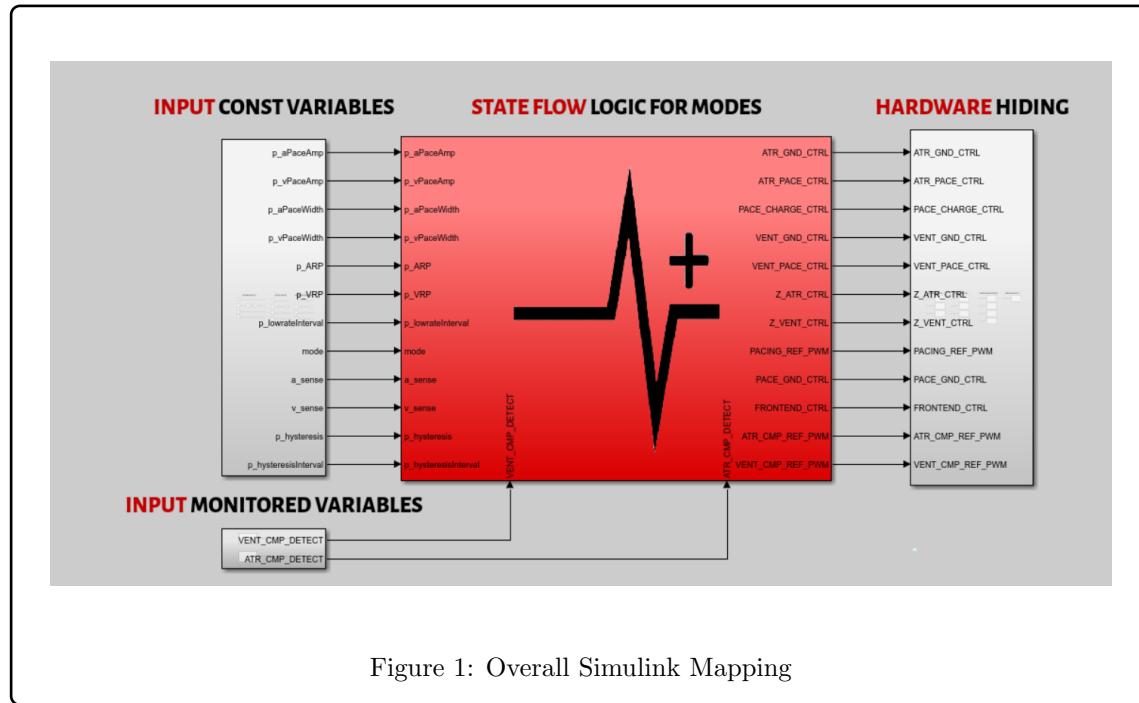


Figure 1: Overall Simulink Mapping

The Pacemaker architecture can be split up into 4 main modules, input constant variables, input monitor variables, stateflow logic for modes, and hardware hiding. Figure 1 below shows the overarching workflow of the system:

5.1.2 Input Constant Variables

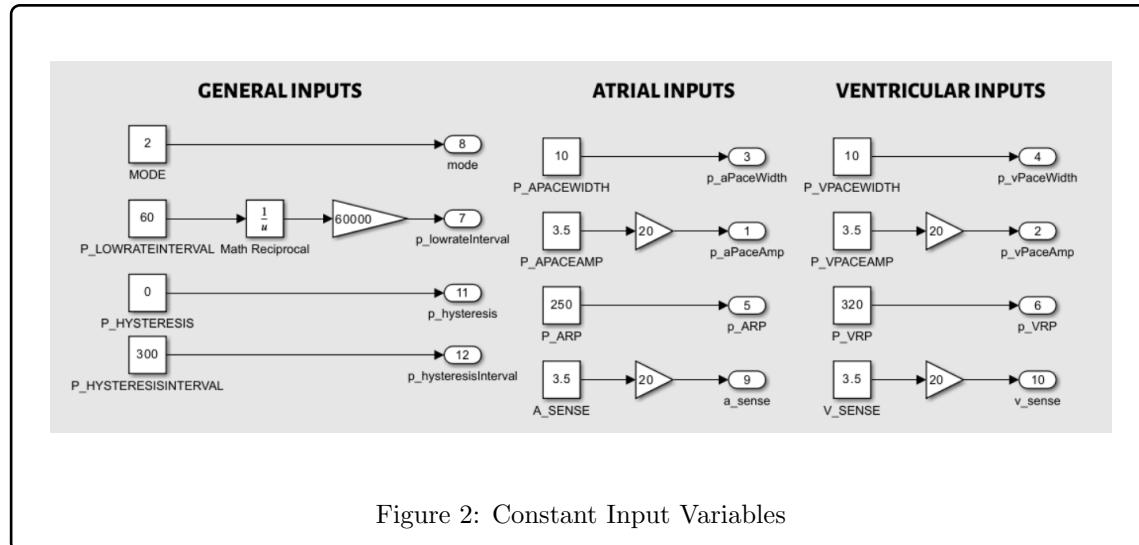


Figure 2: Constant Input Variables

In the above image, Figure 2, we find changeable variables relating to pacemaker operation. For general inputs, the changeable variables are:

- **Mode** - Refers to bradycardia operating modes, e.g AOO, VOO, AAI and VVI.
- **Low Rate Interval** - The number of generated pace pulses per minute, converted from a millisecond time period.
- **Hysteresis Pace** - When enabled, 1, a longer period is waited before pacing after sensing an event to prevent unwanted pacing pulses from ringing from an event.
- **Hysteresis Interval** - Specifies the time interval waited in the hysteresis mode in milliseconds.

The modifiable atrial variables are:

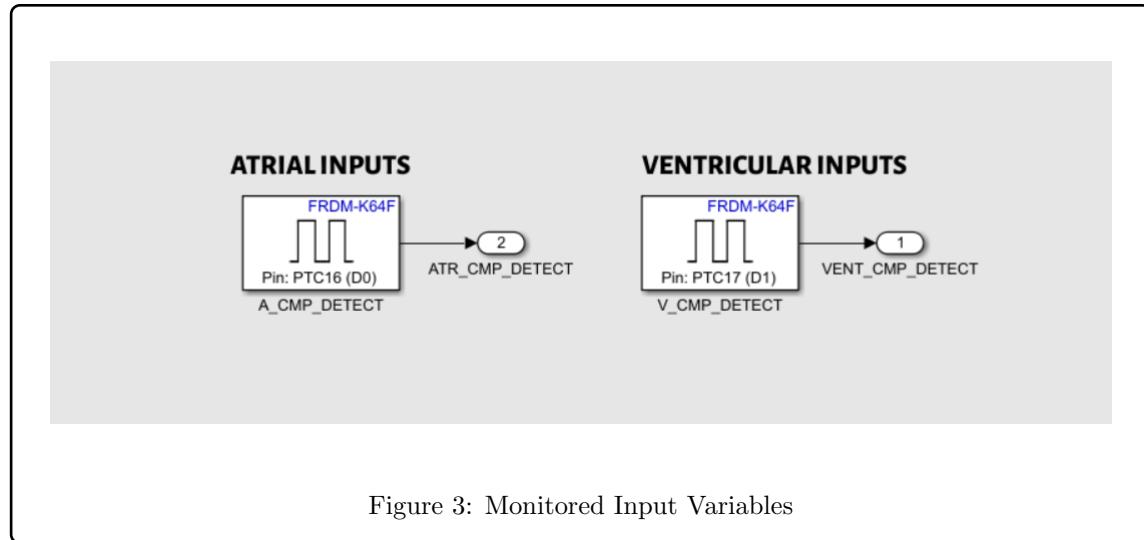
- **Pace Pulse Width** - Changes the width, length of time, of the pace pulse.
- **Pace Pulse Amplitude** - Changes the amplitude, voltage, of the pace pulse.
- **ARP (Atrial Refactory Period)** - The programmed time interval following an atrial event during which atrial events shall not inhibit nor trigger pacing
- **Sense (Sensitivity)** - Determines the minimum value an atrial signal must be to be considered by the pacemaker.

The modifiable ventricular variables are:

- **Pace Pulse Width** - Changes the width, length of time, of the pace pulse.
- **Pace Pulse Amplitude** - Changes the amplitude, voltage, of the pace pulse.

- **VRP (Ventricle Refactory Period)** - The programmed time interval following an ventricle event during which time atrial events shall not inhibit nor trigger pacing.
- **Sense (Sensitivity)** - Determines the minimum value a ventricle signal must be to be considered by the pacemaker.

5.1.3 Monitored Input Variables



The monitored input variables can be seen in the above Figure 3. These are the atrial and ventricle detection variables. The pulses are sensed with through GPIO pins connecting to a board simulating heart conditions.

5.1.4 Stateflow Modules

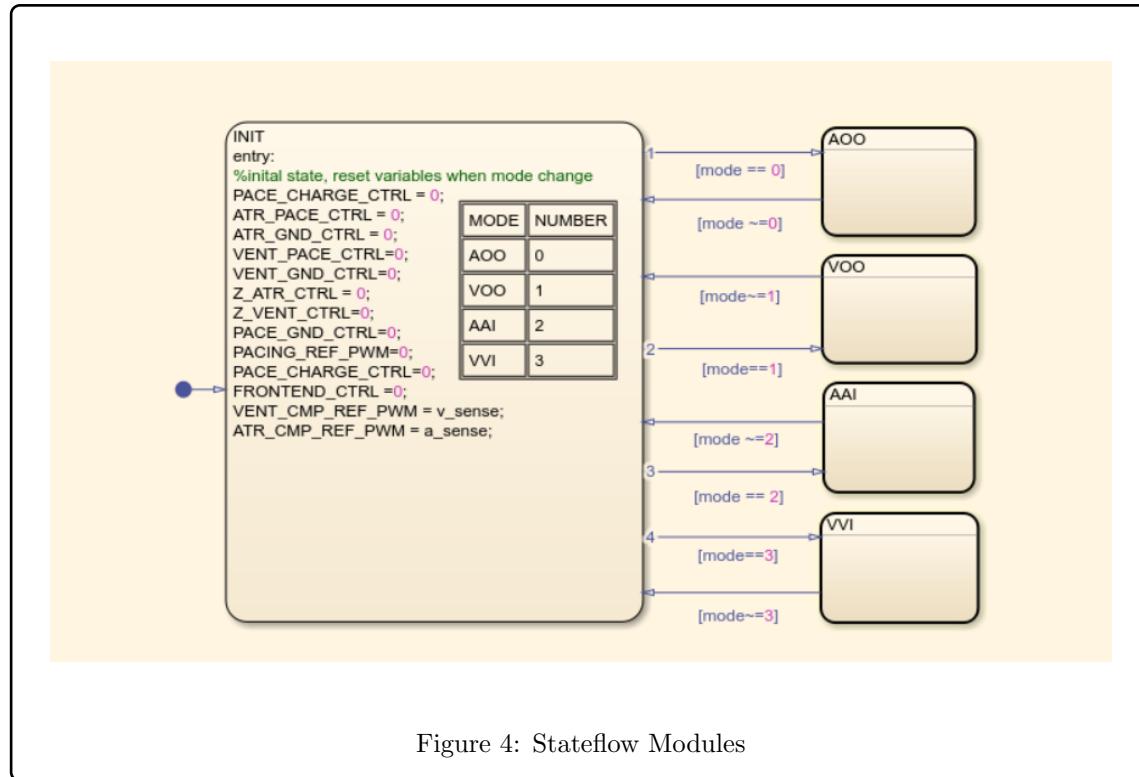
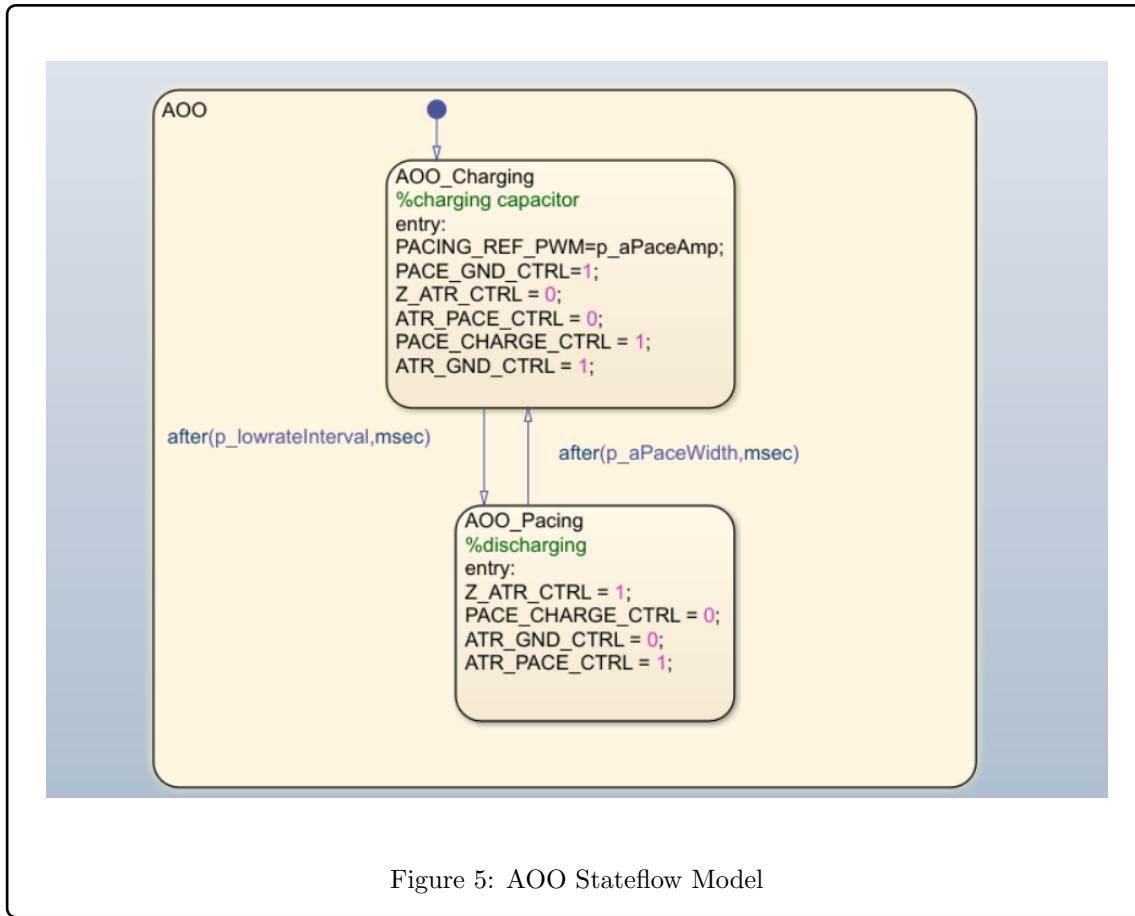


Figure 4: Stateflow Modules

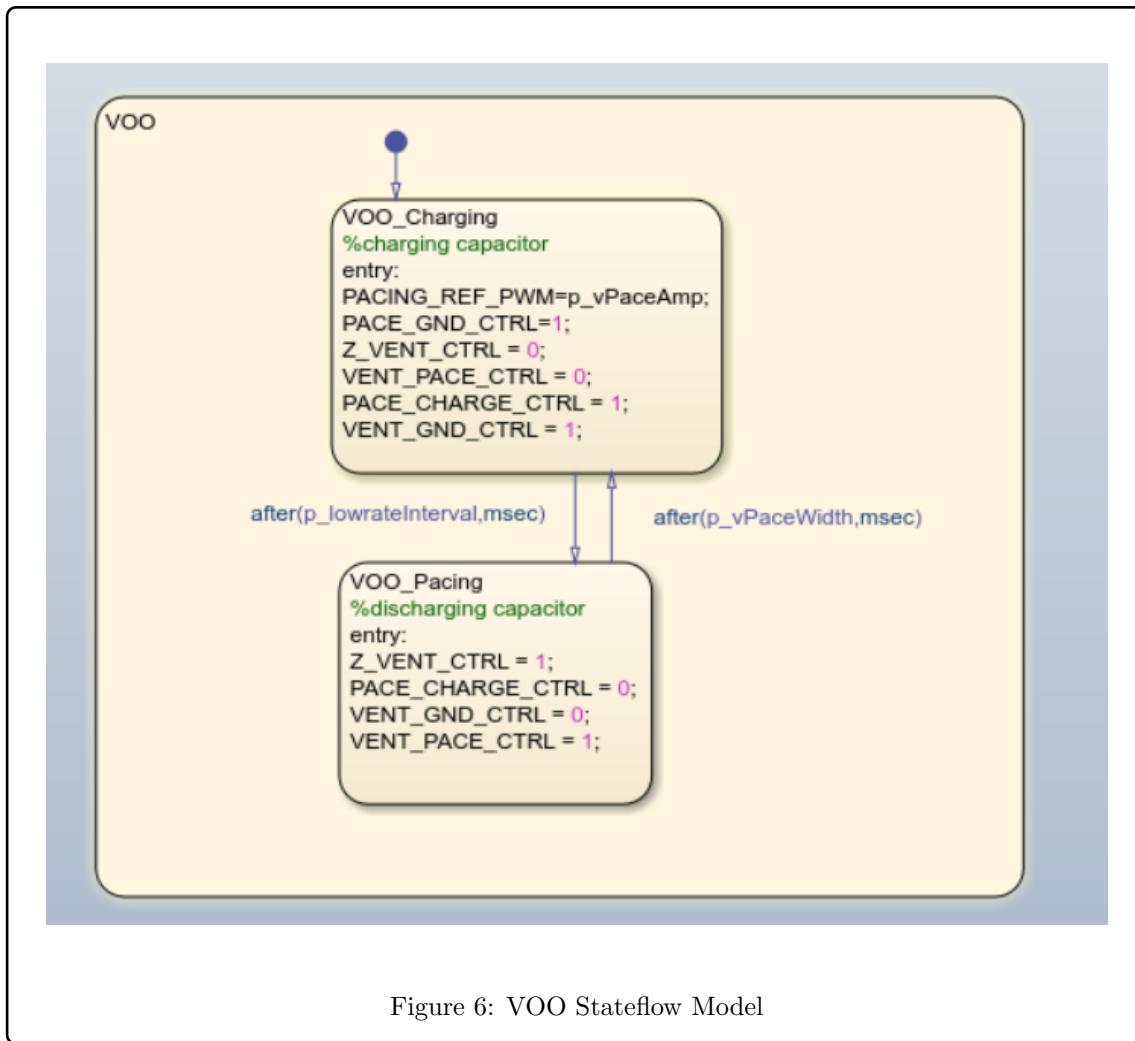
The stateflow diagram in Simulink is shown above. It shows the transition between each mode given the mode input shown in Figure 2. When switching between modes, a core state is returned to that resets variables to their nominal values.

5.1.5 AOO Stateflow Model



The above stateflow, Figure 5, shows the stateflow for the AOO mode. It sets values controlling discharge and charging of the capacitor to specified nominal values.

5.1.6 VOO Stateflow Model



Similar to the AOO stateflow, the above figure, Figure 6, shows the states of charging and discharging of the capacitor using specified nominal values.

5.1.7 VII Stateflow Model

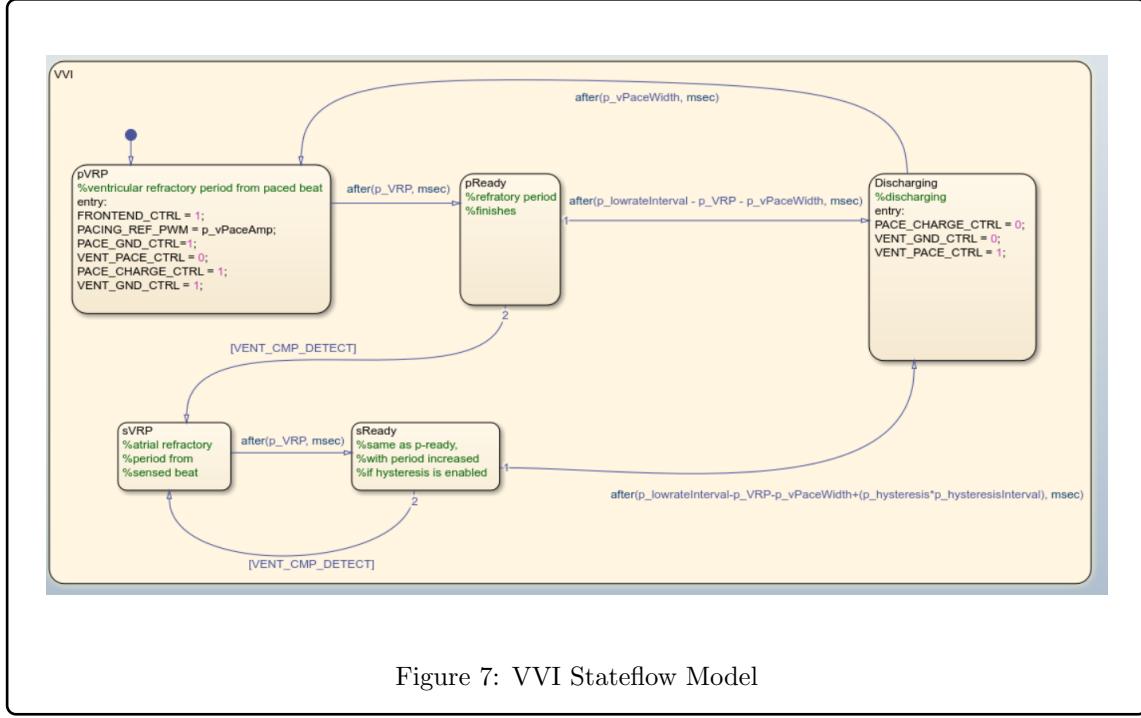


Figure 7: VVI Stateflow Model

The above stateflow, Figure 7 shows the FSM for the VVI model. The initial state, pVRP, is the state that occurs right after the pacemaker delivers a ventricle pacing pulse. A refractory period occurs during this time, where the pacemaker ignores sensor inputs to prevent sensing of its own produced pulse or electrical ringing. After a certain amount of time, the pReady state is transitioned to where sensor inputs are allowed. This allows for the pacemaker to detect natural heart rhythms and correct heart rhythms accordingly, or solely deliver pacing pulses. The last state before going to the initial state is the discharging state where the pacing pulse is produced.

5.1.8 AAI Stateflow Model

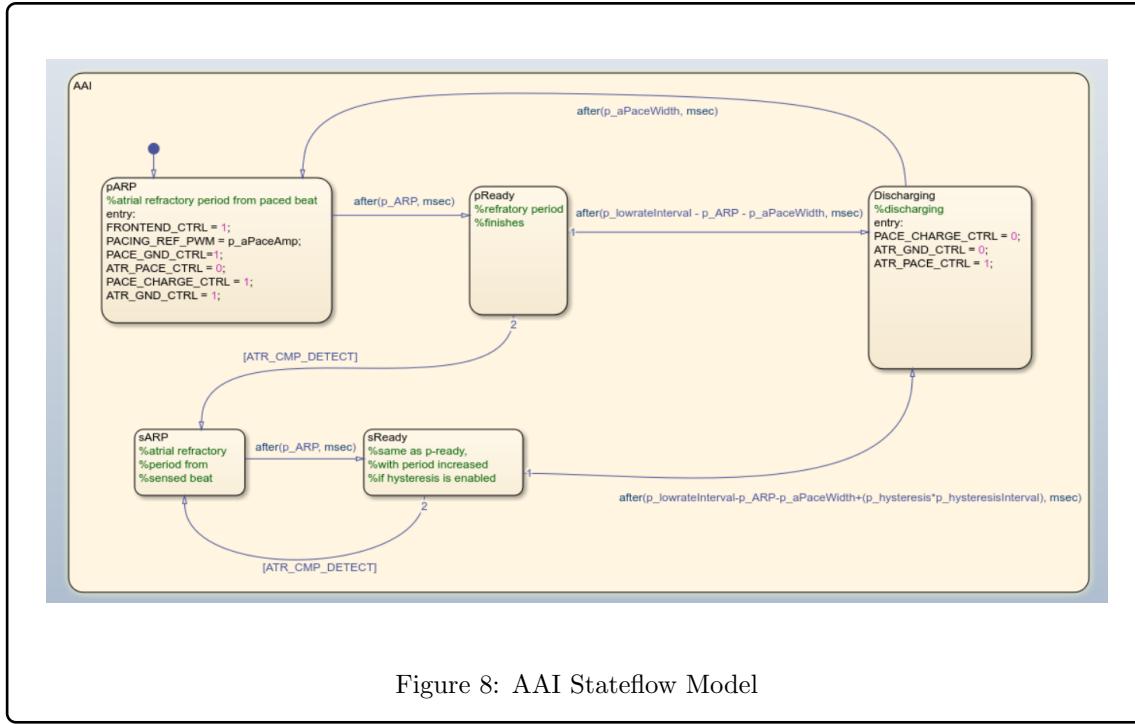


Figure 8: AAI Stateflow Model

The above stateflow, Figure 8 shows the process for the AAI mode. The initial state, pARP, is the state occurs right after the pacemaker delivers an atrial pacing pulse. During this time, the pacemaker ignores sensing events to prevent it from sensing its own delivered pulse. The next state is transitioned to after a set time period, the refractory period, if no natural heartbeat is detected after a certain time interval, the discharging state is then transitioned to. However, if a natural heartbeat is detected, another refractory period occurs to prevent the pacemaker from sensing ringing. This state then transitions to the discharging state once an appropriate time has passed.

5.1.9 Hardware Hiding

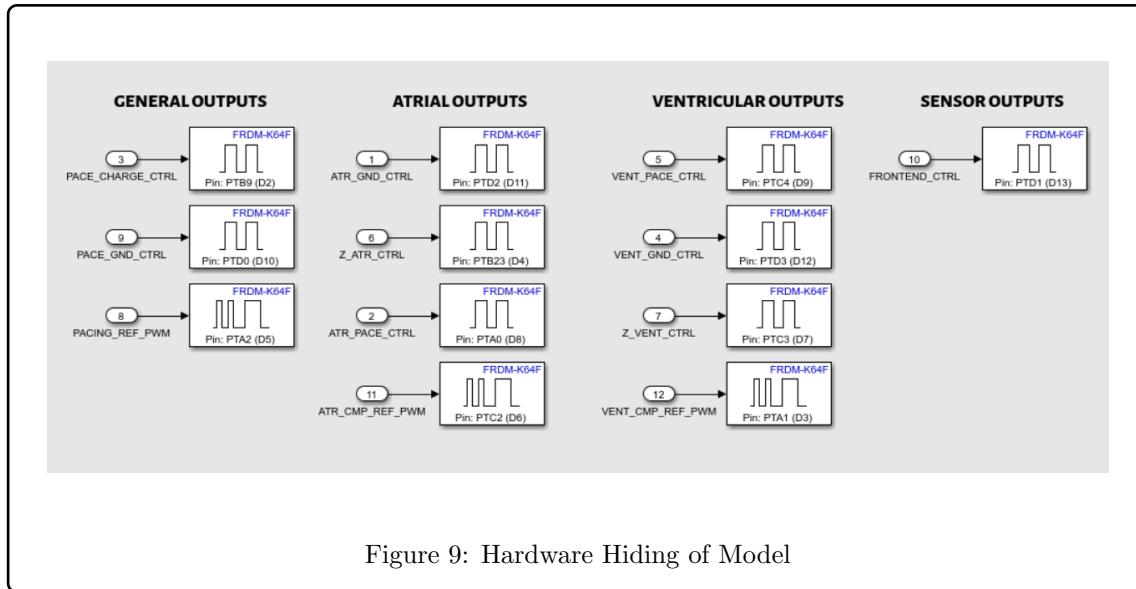


Figure 9: Hardware Hiding of Model

The above image, Figure 9 shows abstraction of the GPIO pin functions. It connects logical output signals to the pins on the FRDM-K64F. This allows for better readability, thus making the code easier to maintain, debug and safer.

5.2 Simulink Design - Deliverable 2

5.2.1 General Overview

The below image shows a high-level overview of the Simulink model with all subsystems. The newly added modules include the serial communication and rate adaptive logic modules for the newly added serial communication with DCM and pacing modes. The input monitored variables now also includes accelerometer values for adaptive rates based on accelerometer values.

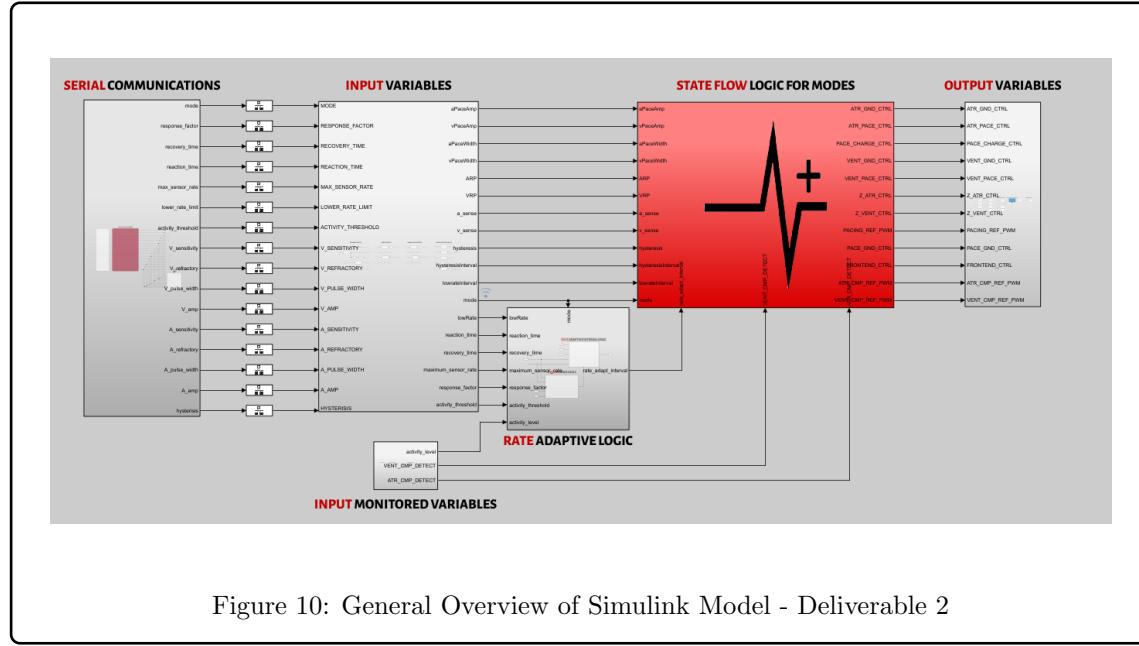


Figure 10: General Overview of Simulink Model - Deliverable 2

5.2.2 Serial Communication Module

The serial communication module can be seen in the figure below. The communication input block is stateflow logic that handles how the rxdata being transmitted from the serial receive block is being processed and unpacked and the echo function. The communication output is how data is outputted from the pacemaker back to the DCM via the send_params function packing all data back and sending via the serial transmit block. For simplicity, we chose to use a stateflow logic similar to that of tutorial 3.

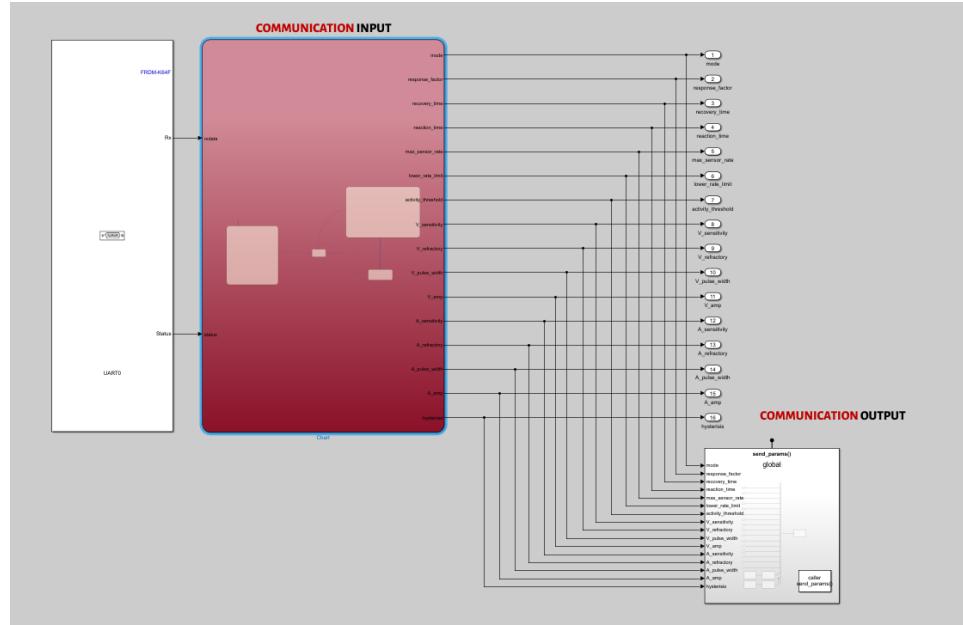


Figure 11: Serial Communication Module Simulink - Deliverable 2

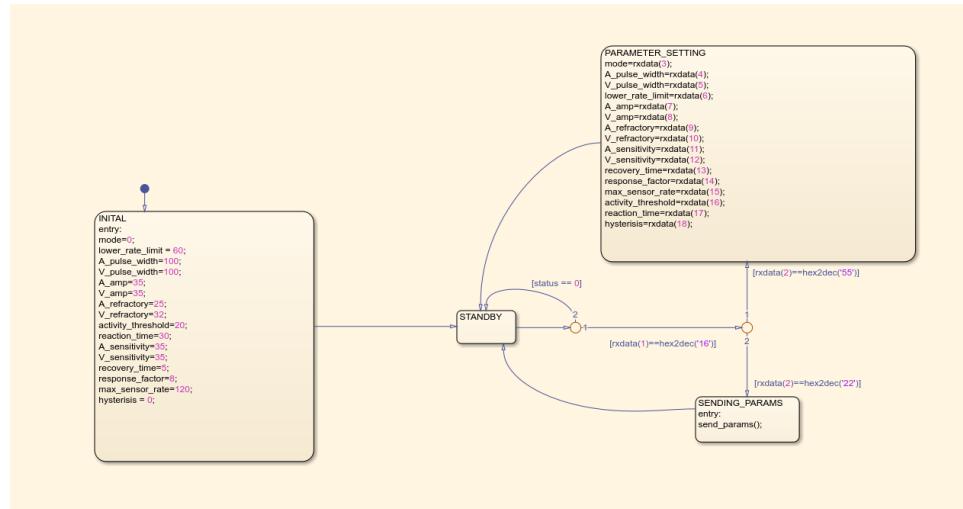


Figure 12: Communication Input Block - Deliverable 2

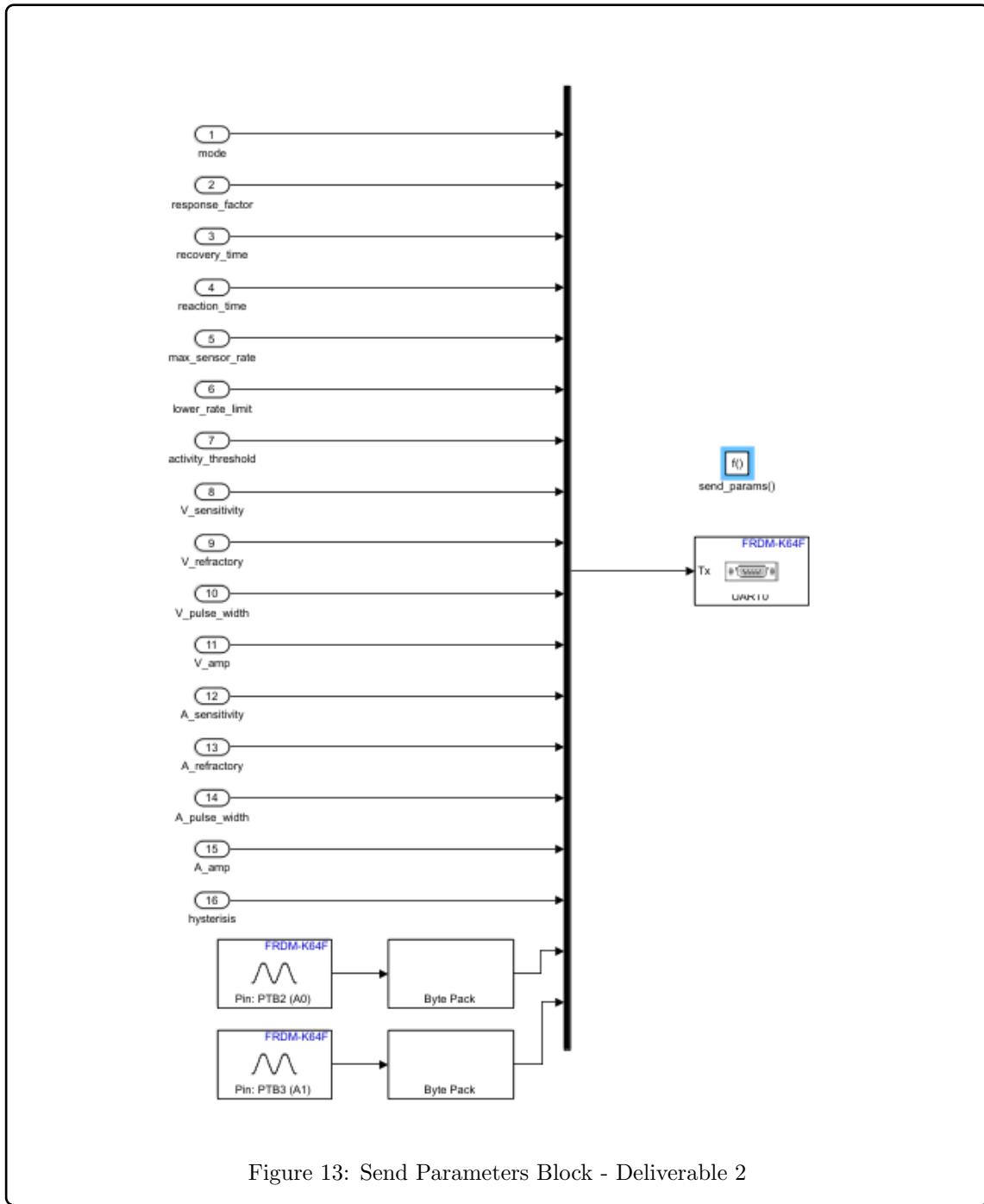


Figure 13: Send Parameters Block - Deliverable 2

5.2.3 Rate Response

5.2.3.1 All Modes

The requirements for the newly added modes were fulfilled and can be seen in the below images. Building off the stateflow logic in deliverable 1, the addition of modes was easily added to the default reset state block.

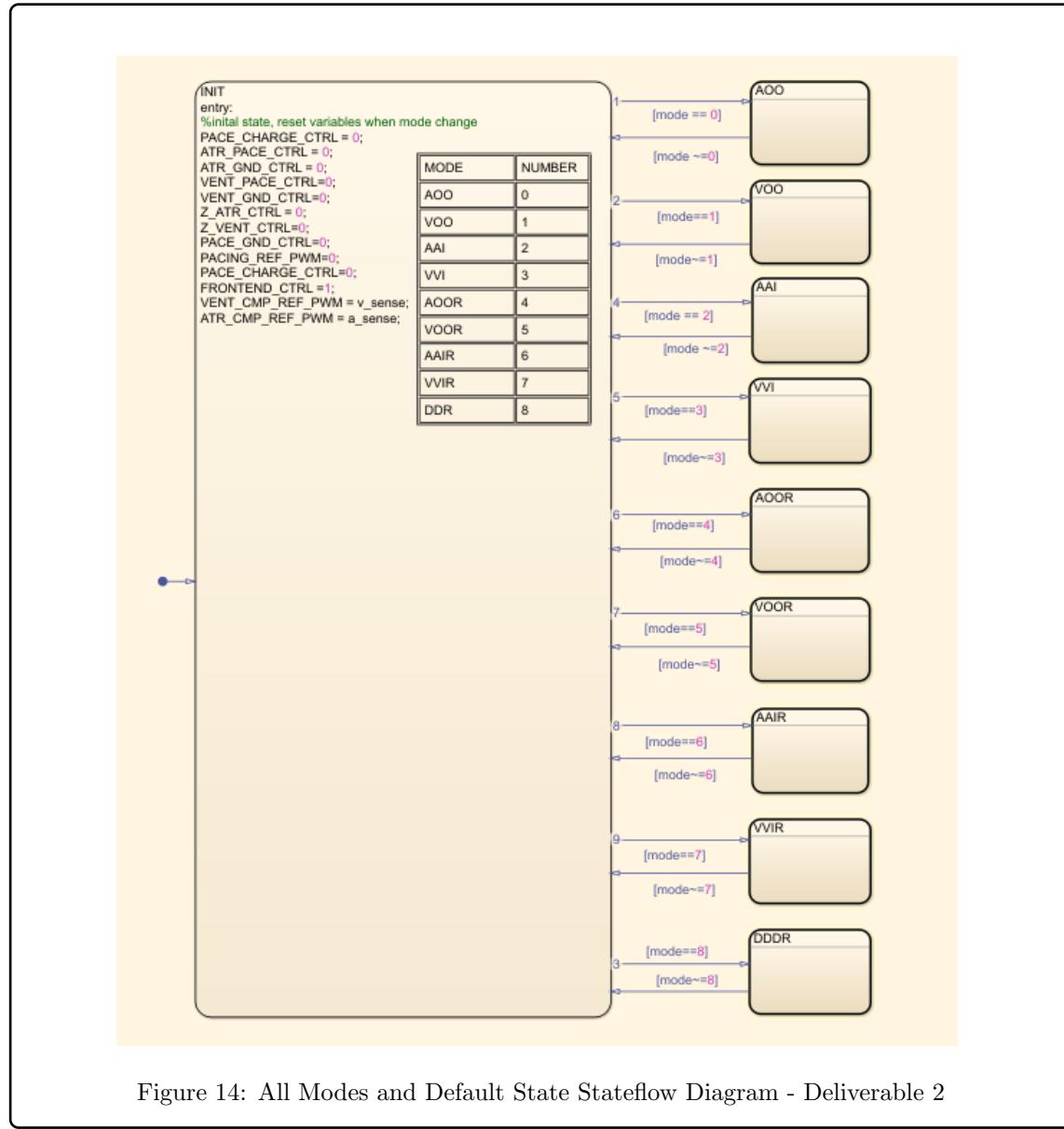
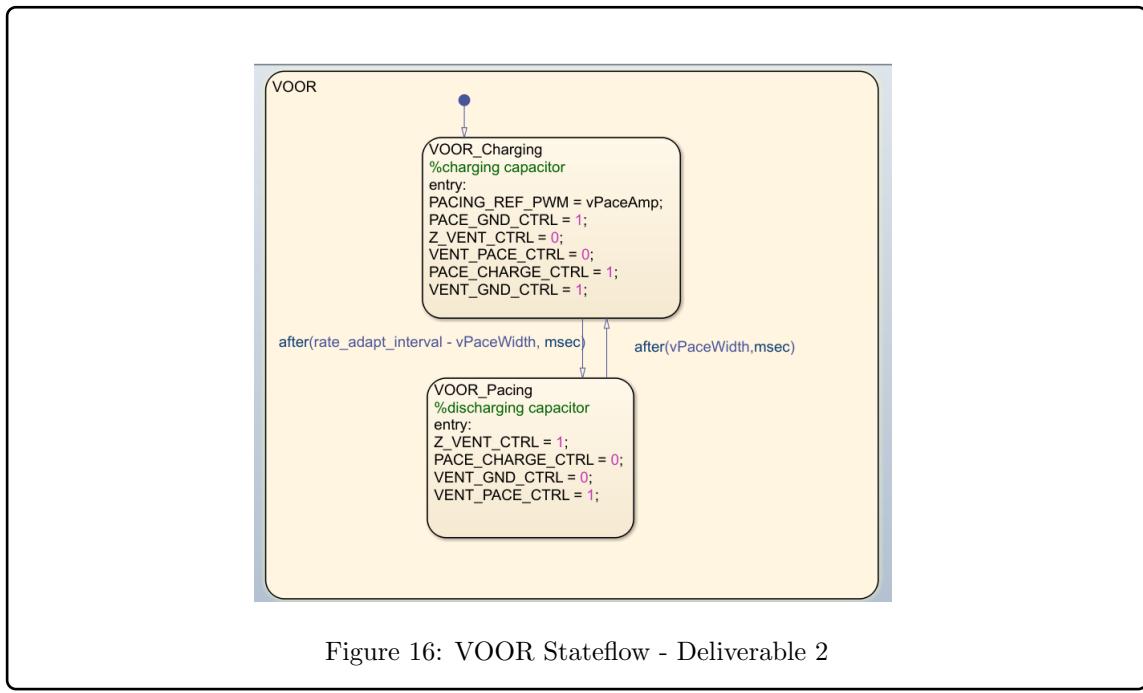
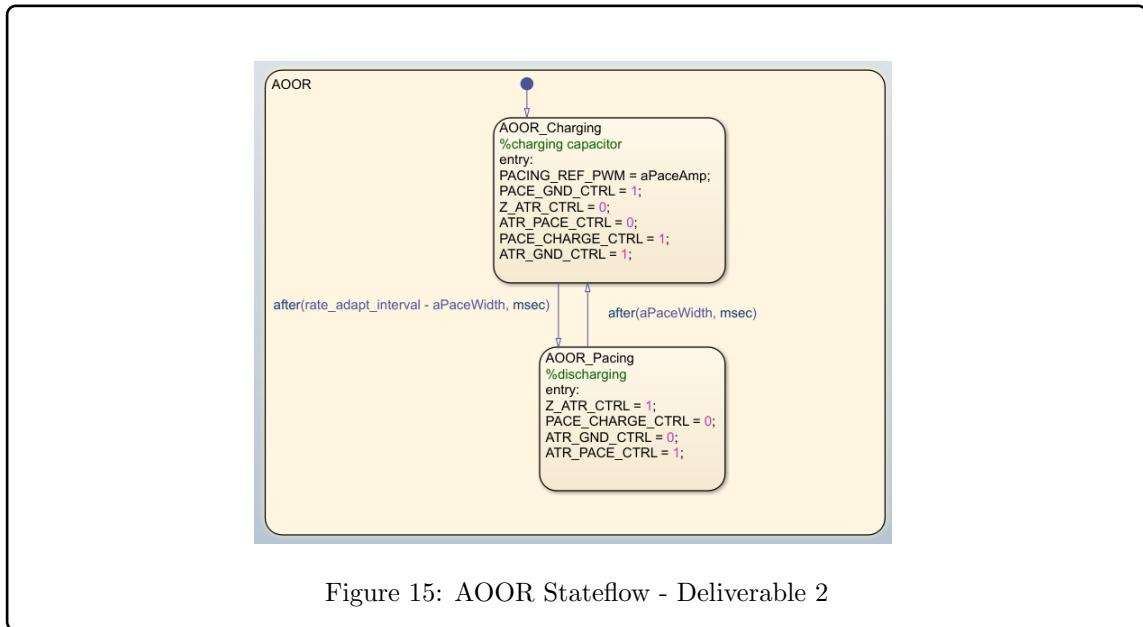


Figure 14: All Modes and Default State Stateflow Diagram - Deliverable 2

5.2.3.2 Mode Stateflow

Below are the stateflows for each rate adaptive modes:



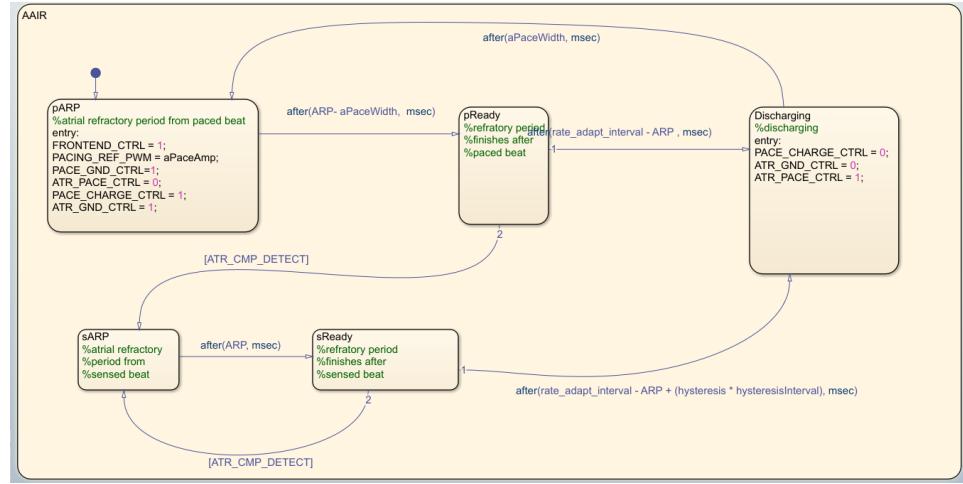


Figure 17: AAIR Stateflow - Deliverable 2

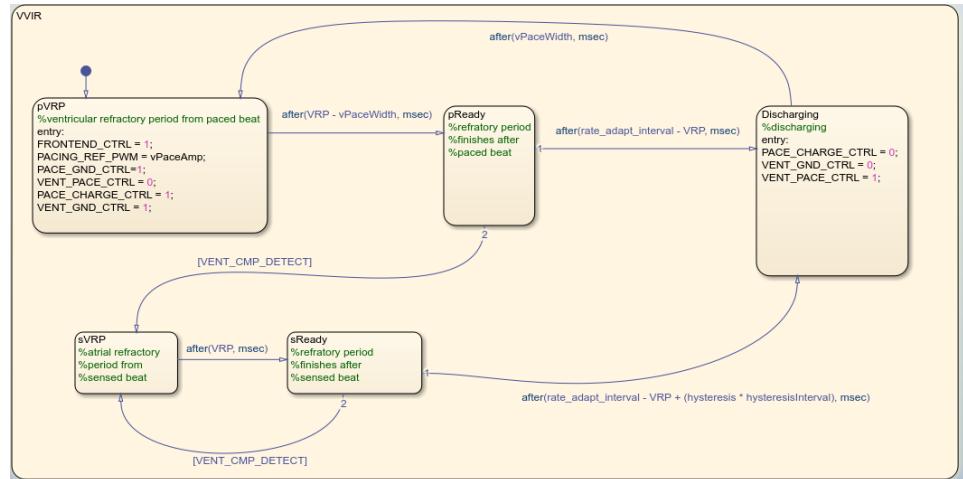


Figure 18: VVIR Stateflow - Deliverable 2

The logic of these states are similar to that of the regular modes e.g AOO, VVI except are governed by the rate_adapt_interval. This changes the effective BPM of pacing. The Simulink calculations for these modes can be seen below:

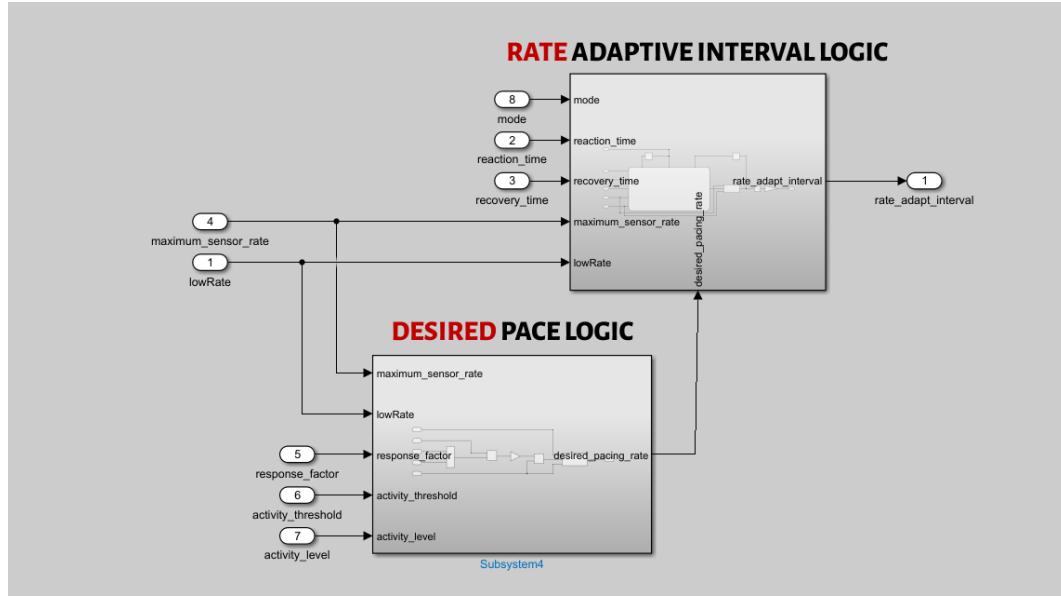


Figure 19: Rate Adaptive Logic - Deliverable 2

It takes in newly added parameters maximum sensor rate, response factor, activity threshold and activity level and outputs a rate adaptive interval value with constraints and parameters. The logic for desired pace is shown below:

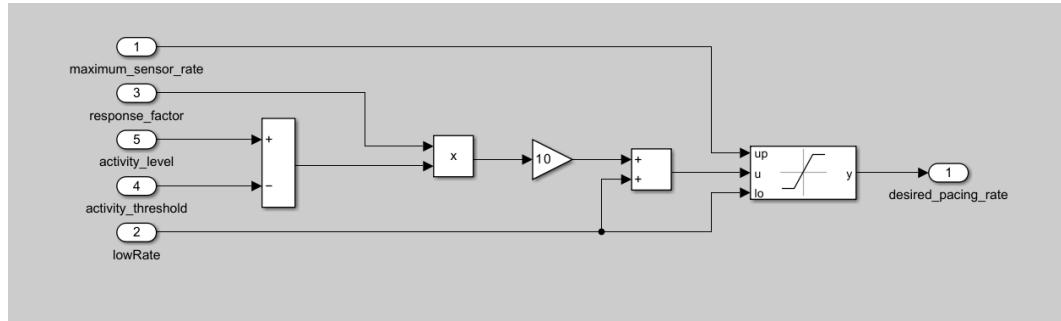


Figure 20: Desired Pace Logic - Deliverable 2

The saturation dynamic block prevents the desired pace rate from being too low, below the low rate, or being too high, above the max sensor rate. This calculates the desired pacing rate which is

fed into the rate adaptive interval block which can be seen below:

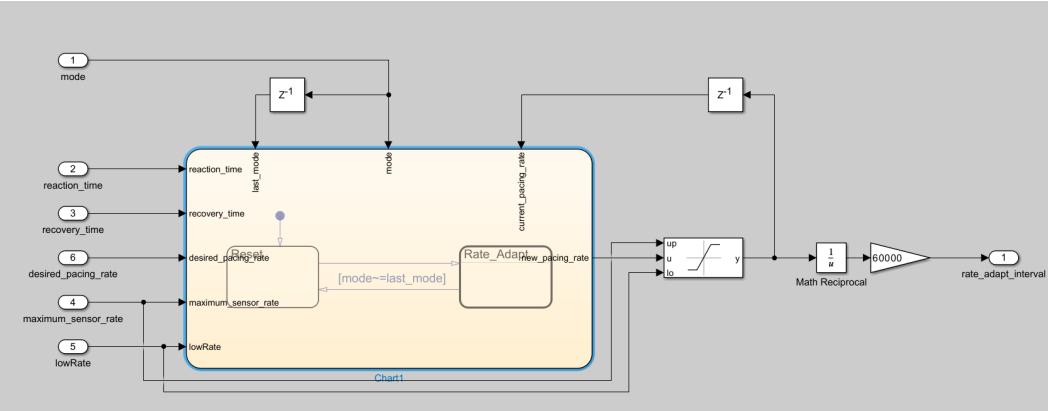


Figure 21: Rate Adaptive Interval Block - Deliverable 2

This block includes more logic within itself that resets the desired pace rates when swapping between different rate adaptive modes. The logic of which can be seen in the stateflow logic below:

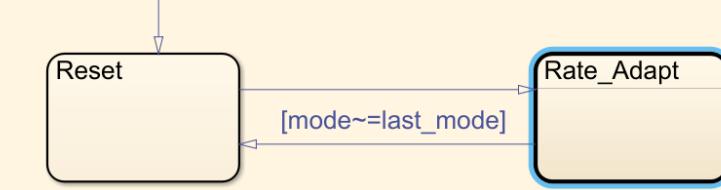


Figure 22: Stateflow for Mode Switching of Rate Adaptive Logic - Deliverable 2

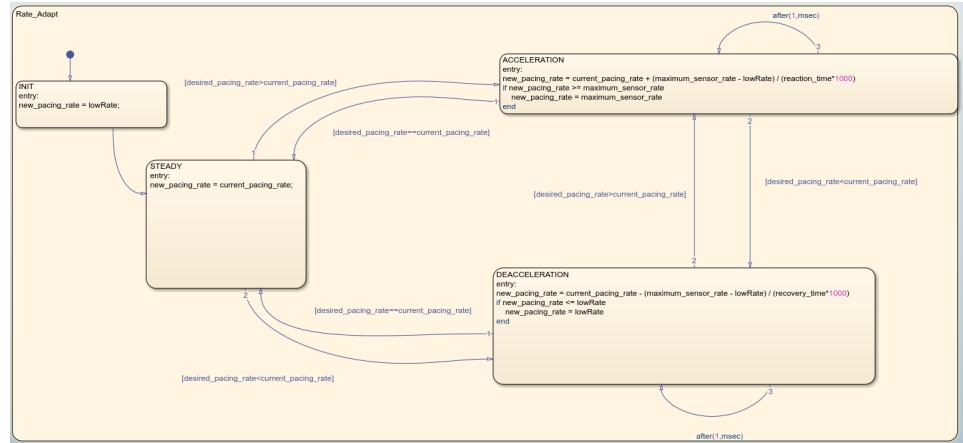


Figure 23: Rate Adaptive Mode Reset When Switching Logic - Deliverable 2

The way activity level for the logic above is measured is through the accelerometer. This is computed with the following block diagram:

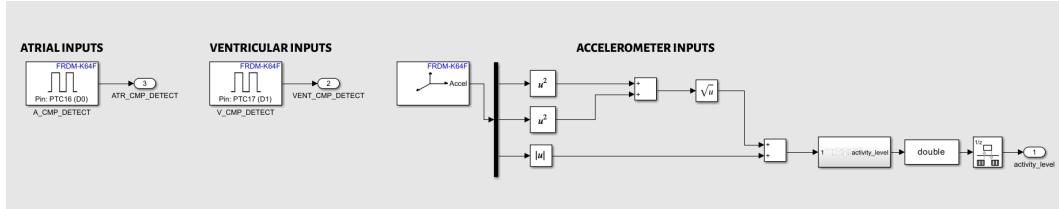


Figure 24: Accelerometer Reading Block Diagram - Deliverable 2

The activity level is calculated by finding the magnitude of x and y values first, then adding the absolute z value, normalizing these values. The reason is because we want to treat moving diagonally on the xy-plane to be the same as moving straight in x or y, since the magnitude is the same. The z-value is added for the case the patient jumps or walks up stairs.

The sampling rate of the ventricular and atrial inputs is faster, 0.001s, to produce more accurate values in AAI and VVI modes. The accelerometer sampling rate is lower and is unable to sample faster than the value shown, 0.01s.

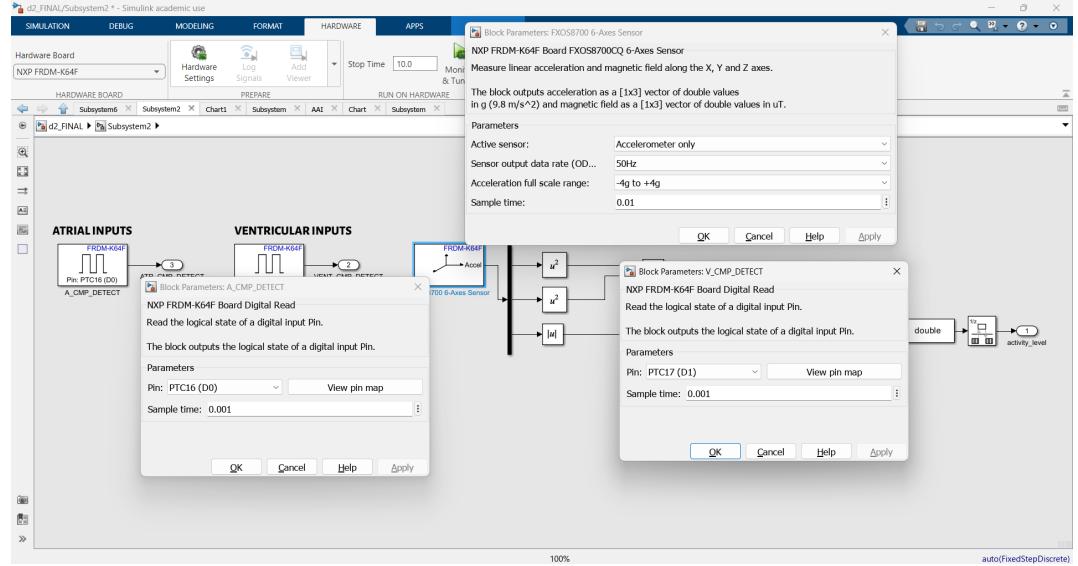


Figure 25: Accelerometer and Inputs Sampling Rates - Deliverable 2

Furthermore, this calculation uses a moving custom moving average block and the calculation of such can be seen below. This is to smooth out the signal from small deviations or noise in the measurements of acceleration.

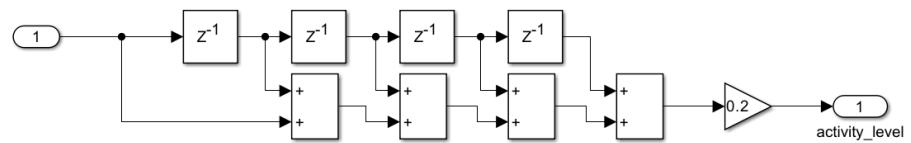


Figure 26: Custom Moving Average - Deliverable 2

5.2.4 Outputs

The outputs of the Simulink is handled in the outputs subsystem. The subsystem is shown below:

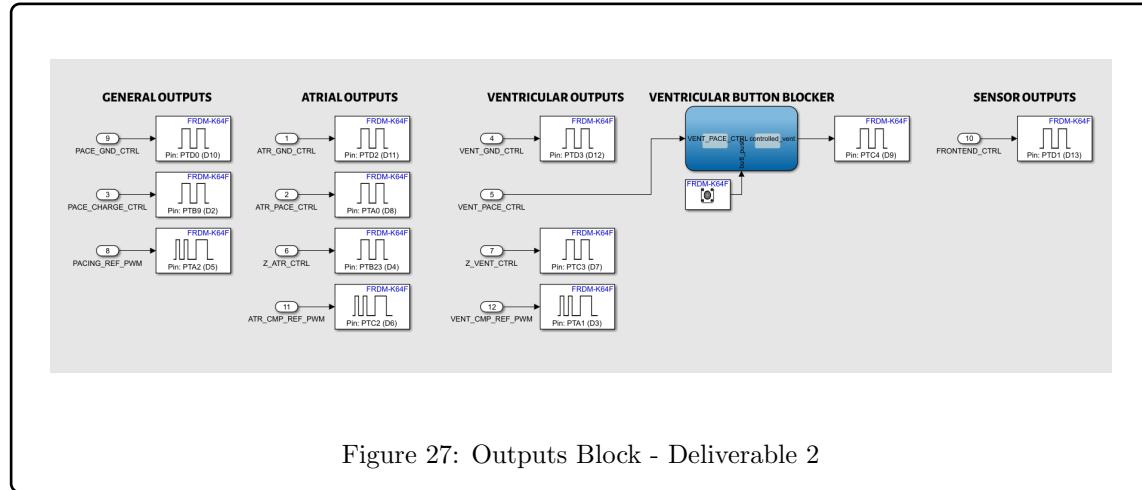


Figure 27: Outputs Block - Deliverable 2

5.2.4.1 Ventricular Inhibiting

This block includes a push button input that inhibits the ventricular pulsing. If held down, it prevents the ventricular part of the pacemaker from pacing. This is a safety feature that allows for quick emergency shutoff incase of malfunction. The logic for this block is shown below:

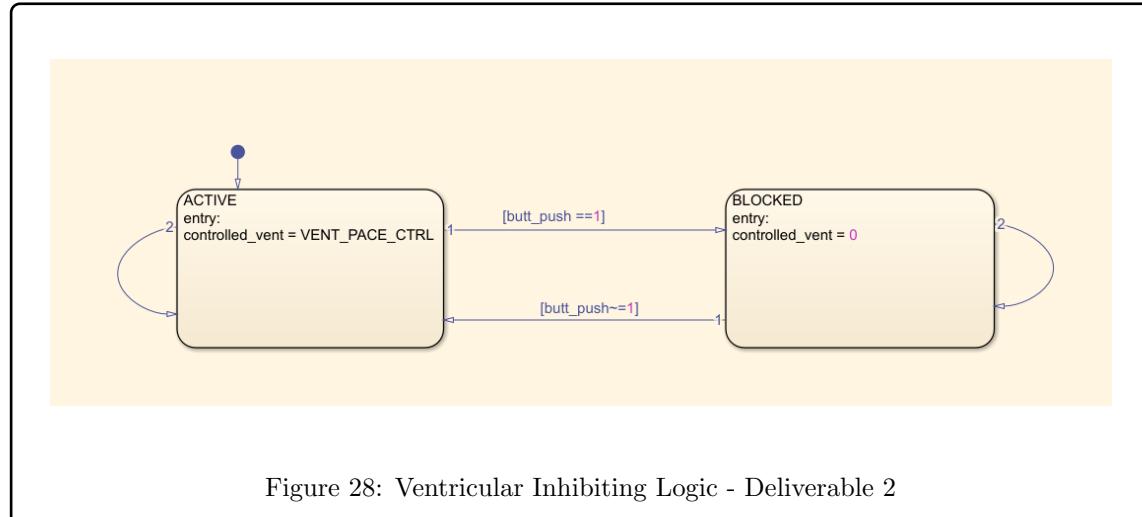


Figure 28: Ventricular Inhibiting Logic - Deliverable 2

5.3 DCM Design

The design of the DCM has 3 separate sections; the **model, view and controller**. This is to allow modularity and separation of different functionalities of the DCM from one another. For example, adding functionality can be as simple as making and including another class in another file.

5.3.1 Model

The model section handles the application's data and user logic. It does not interact directly with the interface but only manages data and enforces user authentication. Below is an explanation of the individual Python files and their functionality:

User Model

Purpose: To manage all user account information.

Functions:

- Loads and saves login information to or from the users.json file.
- Registers new users, checking for duplicates and the max user limit.
- Authenticates logins by verifying user passwords
- Provides a method to get the amount of registered users which is displayed by the DCM.

Pacing Model

Purpose: Manages pacing parameters for all users.

Functions:

- Loads and saves settings to or from pacing_settings.json
- The data is structured as a dictionary with settings as the key and value as the value.
- Saves a dictionary of parameters for a specific user and pacemaker mode
- Retrieves the saved parameters for a user and mode

Egram Model

Purpose: Manages the capture of electrogram data

Functions:

- Adds a single data point to an internal list
- Returns the list of captured data
- Clears the captured data to start a new session

5.3.2 View

The view section is what the user sees and interacts with. It is built with the customtkinter library for a more aesthetically pleasing interface. They do not have any control logic, only to display data and send user actions such as button clicks to the controller. Below are an explanation of each view:

Login View

Purpose: Provides a screen for users to login

Functions:

- Welcome class displays the login form and a register button.
- Register class displays the new user registration form and a back button.
- When a button is clicked, it collects text from the entry fields and calls a method on the controller that handles the login logic.
- The welcome view calls a function to display the current registered user count.

Main View

Purpose: Provides the main application screen after logging in

Functions:

- MainFrame is the main menu. It shows buttons for each pacing mode AOO, VOO, AAI and VVI as well as mock connection controls. It also displays the current connection status.
- DataEntry is the form for editing parameters. It displays all parameters and handles validation of data such as the input being in the correct range and is a numeric type before sending the data to the controller.

5.3.3 Controller

Controller

Purpose: The main application class that uses instances of the models and view frames.

Functions:

- It initializes itself and creates all view frames, UserModel and PacingModel. It then stores these in a dictionary.
- The navigation is handled by a show_frame function which requests a frame from the dictionary and uses tkinters library function, tkraise(), to bring it into view.
- Event handling is seperated into handling registration and saving settings. For For registration it takes the username and password from the view, calls the UserModel to verify them, then shows a message to confirm or deny entry. The saving of settings takes mode and data from the DataEntry view and passes them to the PacingModel to be saved.

Main

Purpose: Its purpose and sole function is to create an instance of the controller and run it in the main loop, starting the customtkinter application.

5.3.4 Views of DCM GUI

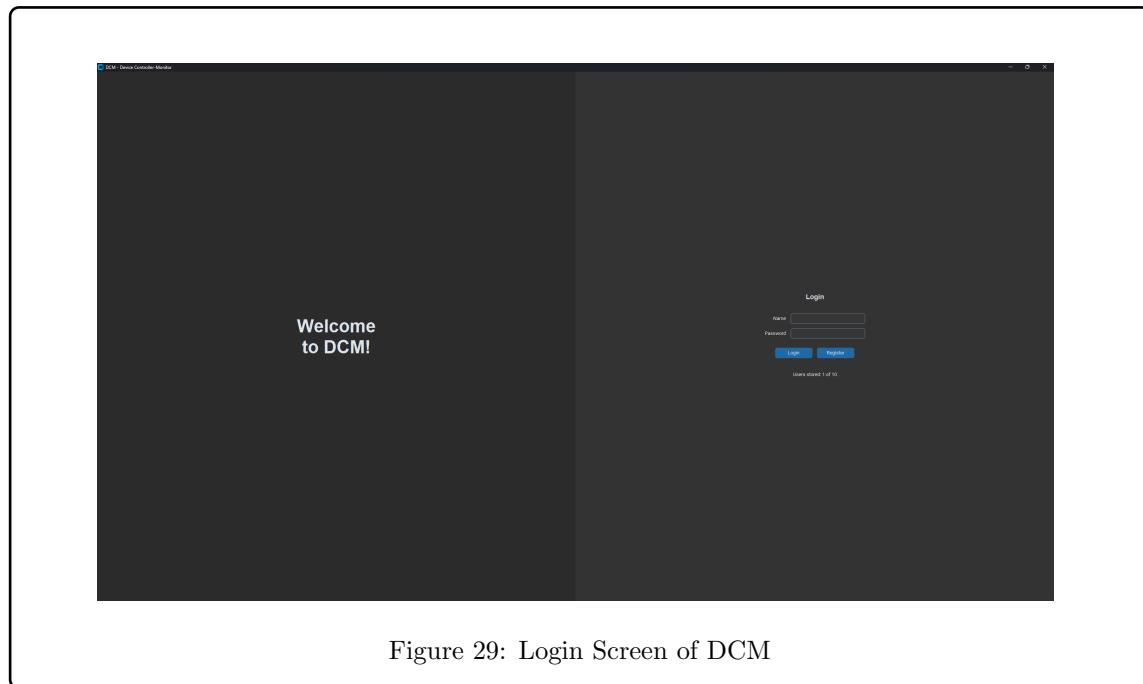


Figure 29: Login Screen of DCM

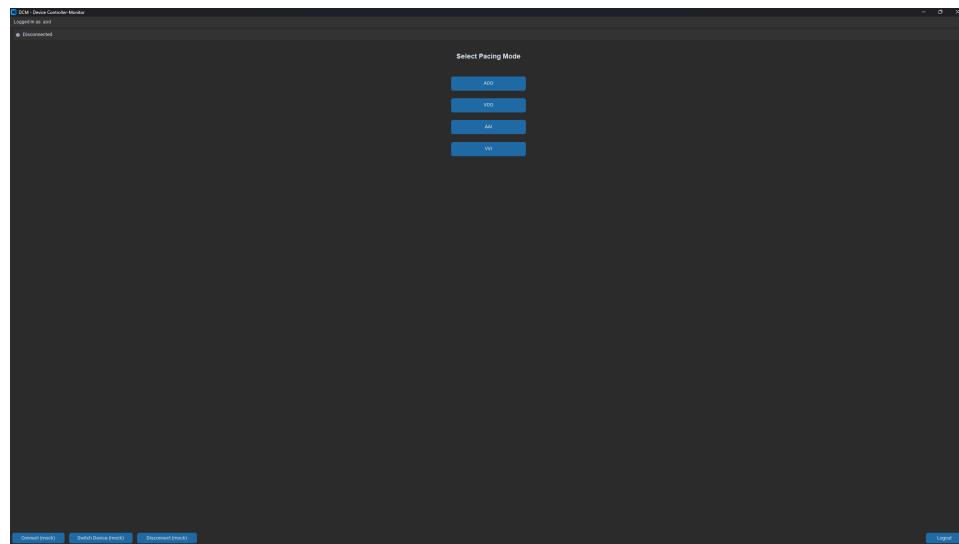


Figure 30: Main Menu of DCM

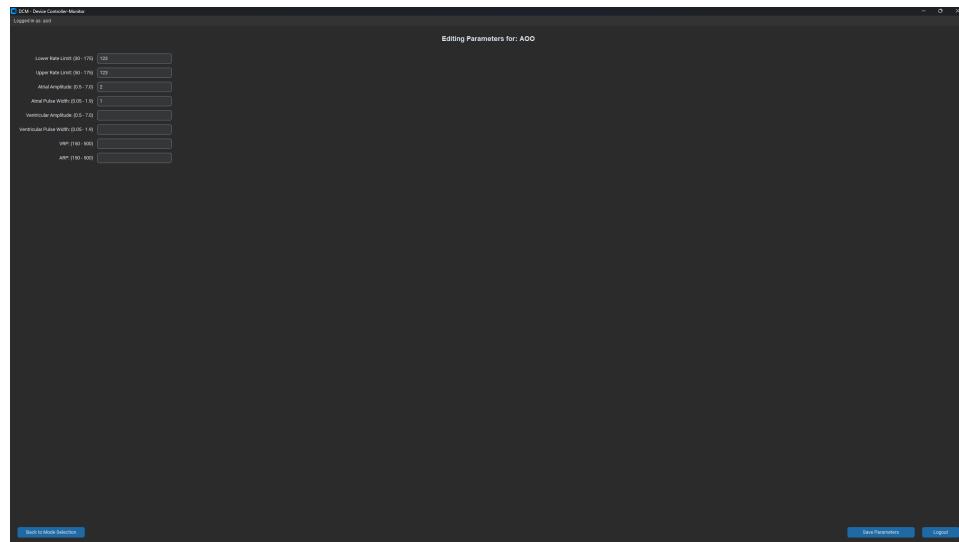
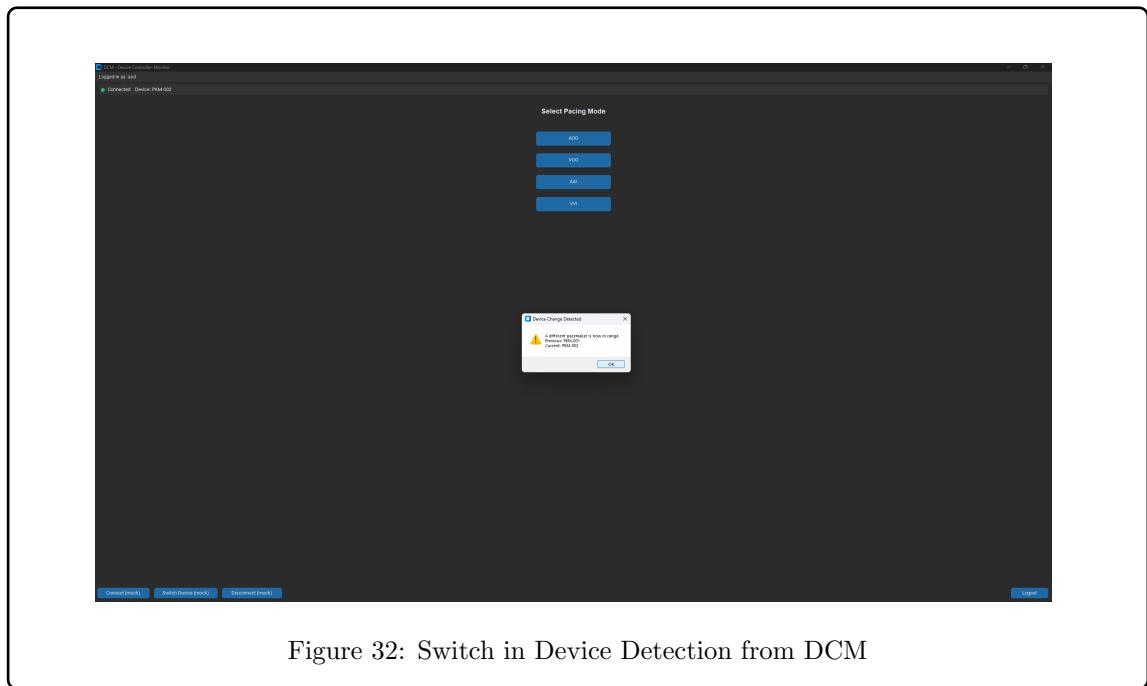


Figure 31: Data Entry View



5.4 DCM & Serial Design - Deliverable 2

The DCM was changed up in deliverable 2 to allow for serial communication and more modes to be used. This is accompanied by a new model in the structure of the DCM, serial_comms.py.

Serial_Comms

Purpose: Provides serial communication functions to the DCM through serial send and receive functions.

Functions:

- Serial port management - allows low-level hardware connection by scanning for COM ports through the get_ports function, establishes connection for the DCM by setting baud rate and other parameters, and safely disconnecting devices.
- Binary Data Structure - Uses struct library to pack and unpack data for more efficient data transfer.
- Send_params - The main function that allows for sending of parameter data to the pacemaker. It typecasts and scales data to allow for more efficient data transfer.
- Data Verification - The echo feature allows for data to be verified once sent to the pacemaker.
- Data Streaming - The EKG requires real time data streaming to be effective which is handled through this class.

5.4.1 Serial Packet Structure

With the addition of additional parameters a data conversion table is used to organize the multiple data types being sent. This can be seen in the table below. The chosen variables are due that are sent are due to the fact the increments within each range fit in a Uint8 format. This allows for the parameter data to be scaled in code to fit in a Uint8 format then scaled back and typecasted in the Simulink. This datatype choice reduces the amount of data being transferred while maintaining the same amount of fidelity. Some parameters such as activity threshold and mode are mapped to integers.

Table 10: Parameter Settings

Parameter	Min	Max	Inc	Unit	Type	Bytes
LRL	30	175	5	BPM	Uint8	1
URL	50	175	5	BPM	Uint8	1
Max Sensor Rate	50	175	5	BPM	Uint8	1
Fixed AV Delay	70	300	10	ms	Uint8	1
Dynamic AV Delay	OFF	ON	N/A	N/A	Bool	1
Sensed AV Delay Offset	0 (lowest -10)	-100	-10	ms	Uint8	1
Atrial Amplitude	0.00	5.00	1.25	Volts	Uint8	1
Ventricular Amplitude	0.00	5.00	1.25	Volts	Uint8	1
Atrial Pulse Width	0.05 0.1	- 1.9	- 0.1	ms	Uint8	1
Ventricular Pulse Width	0.05 0.1	- 1.9	- 0.1	ms	Uint8	1
Atrial Sensitivity	0.25 1	0.75 10	0.25 0.5	mV	Uint8	1
Ventricular Sensitivity	0.25 1	0.75 10	0.25 0.5	mV	Uint8	1
VRP	150	500	10	ms	Uint8	1
ARP	150	500	10	ms	Uint8	1
PVARP	150	500	10	ms	Uint8	1
PVARP Extension	0	400	50	ms	Uint8	1
Hysteresis	OFF	ON	N/A	N/A	Bool	1
Rate Smoothing	0	21	3	N/A	Uint8	1
ATR Duration	10 20 100	- 80 2000	- 20 100	cc	Uint8	1
ATR Fallback Mode	OFF	ON	N/A	N/A	Bool	1
ATR Fallback Time	1	5	1	min	Uint8	1
Activity Threshold	V-Low, Low, Med-Low, Med, Med-High, High, V-High			N/A	Uint8	1
Reaction Time	10	50	10	sec	Uint8	1
Response Factor	1	16 ₃₃	1	N/A	Uint8	1
Recovery Time	2	16	1	min	Uint8	1
Mode	OFF, DDD, VDD, DDI, DOO, AOO, AAI, VOO, VVI, AAT, VVT, DDDR, VDDR, DDIR, DOOR, AOOR, AAIR, VOOR, VVIR			N/A	Uint8	1

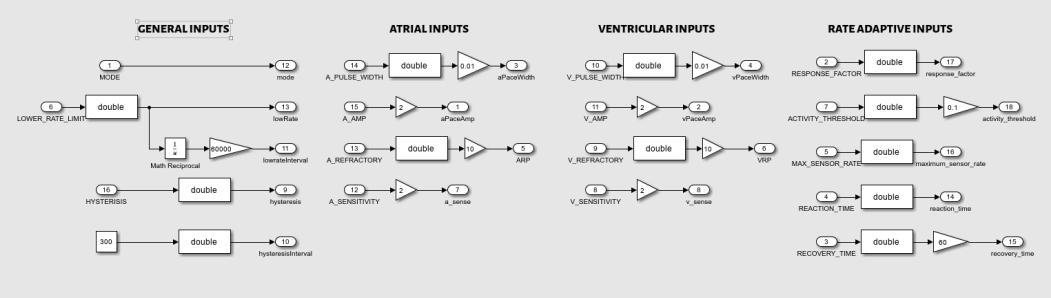


Figure 33: Conversion of Data In Simulink

Code 1: Conversion of Data in DCM

```

1  def send_params(self, params: dict):
2      if not self.ser or not self.ser.is_open: return False
3      try:
4          data = struct.pack(self.FMT_18_BYT
5                          , 0x16, 0x55,
6                          int(params.get("mode", 0)),
7                          int(params.get("a_pw", 0) * 100),
8                          int(params.get("v_pw", 0) * 100),
9                          int(params.get("lrl", 60)),
10                         int(params.get("a_amp", 0) * 10),
11                         int(params.get("v_amp", 0) * 10),
12                         int(params.get("a_ref", 0) / 10),
13                         int(params.get("v_ref", 0) / 10),
14                         int(params.get("a_sens", 0) * 10),
15                         int(params.get("v_sens", 0) * 10),
16                         int(params.get("recov", 0)),
17                         int(params.get("resp_fact", 0)),
18                         int(params.get("msr", 0)),
19                         int(params.get("act_thresh", 0)),
20                         int(params.get("react_time", 0)),
21                         int(params.get("hyst", 0)))
22
23         self.ser.write(data)
24         return True
25     except Exception: return False

```

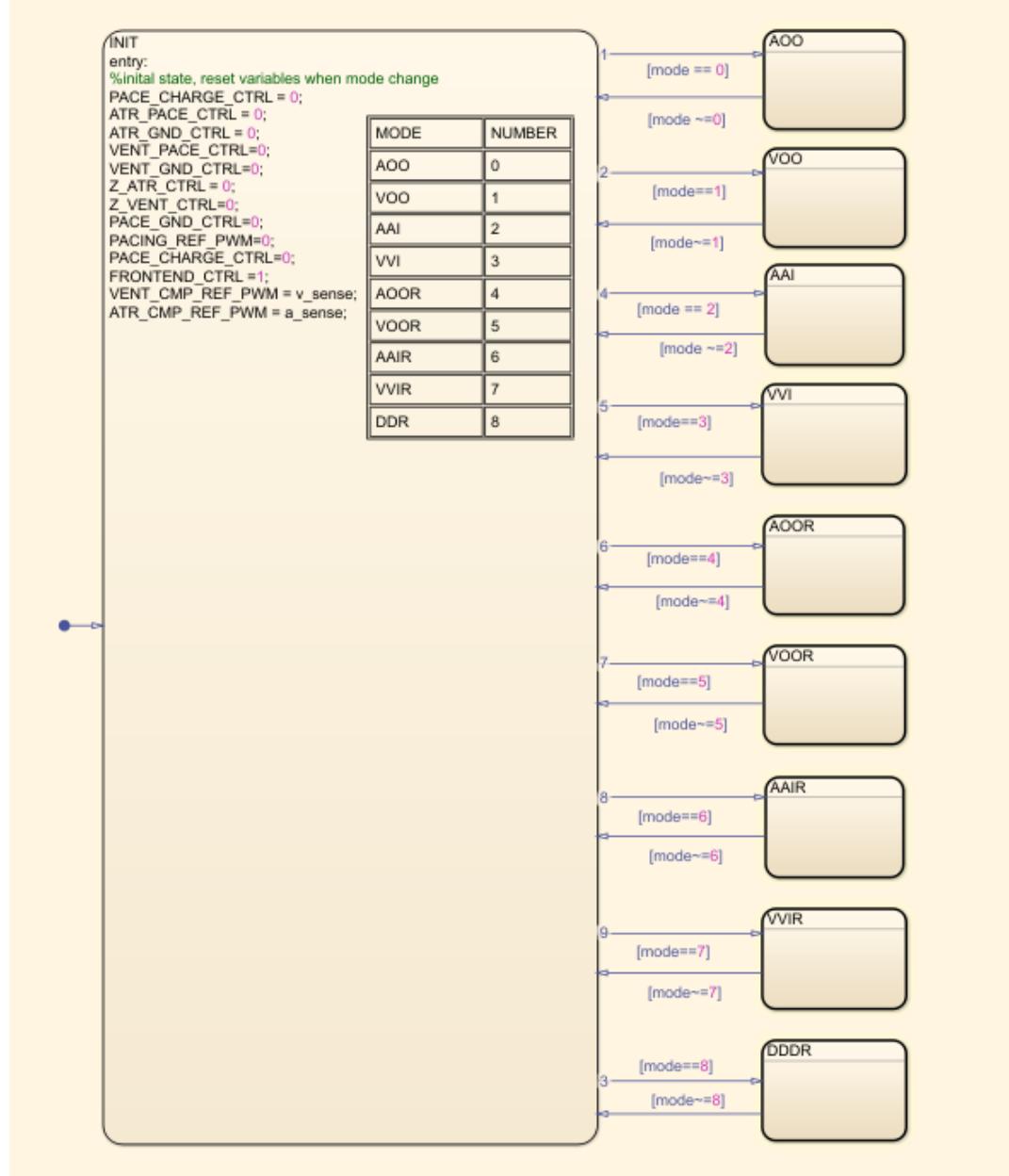


Figure 34: Mapping of Modes to Integers In Simulink

5.4.2 Main View Design & Decisions

Building on top of the previous design the main page was given plenty of new features. The main page can be seen in the screenshot below:

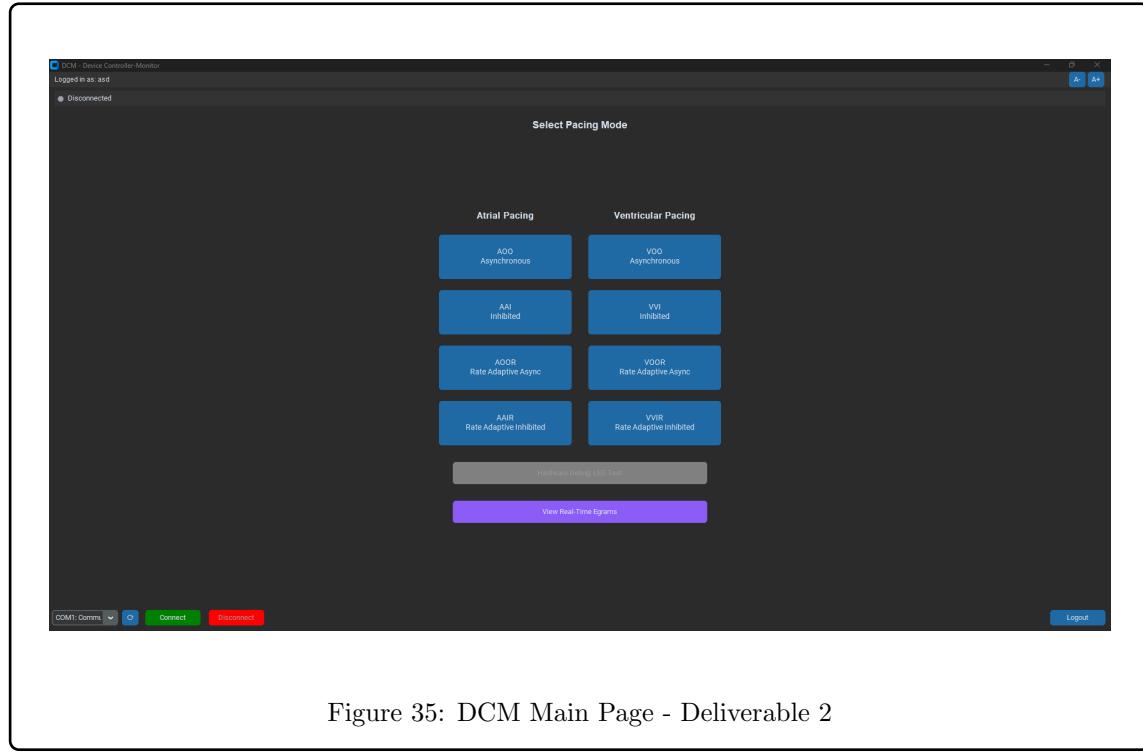


Figure 35: DCM Main Page - Deliverable 2

Changes can be seen in the top left and right corner, the bottom left corner, and the middle. The top left corner now displays COM connections with the computer rather than a mock device.

The top left features font size changing buttons, this was added for accessibility features as a previous issue was that the font size can be too small or too large when the DCM window was shrunk or expanded.

The bottom left corner now shows all COM connections on the PC, selecting the pacemaking device is typically identified as the J-Link Segger and connecting to a device that is not identified using these key words, a warning is shown as seen below:

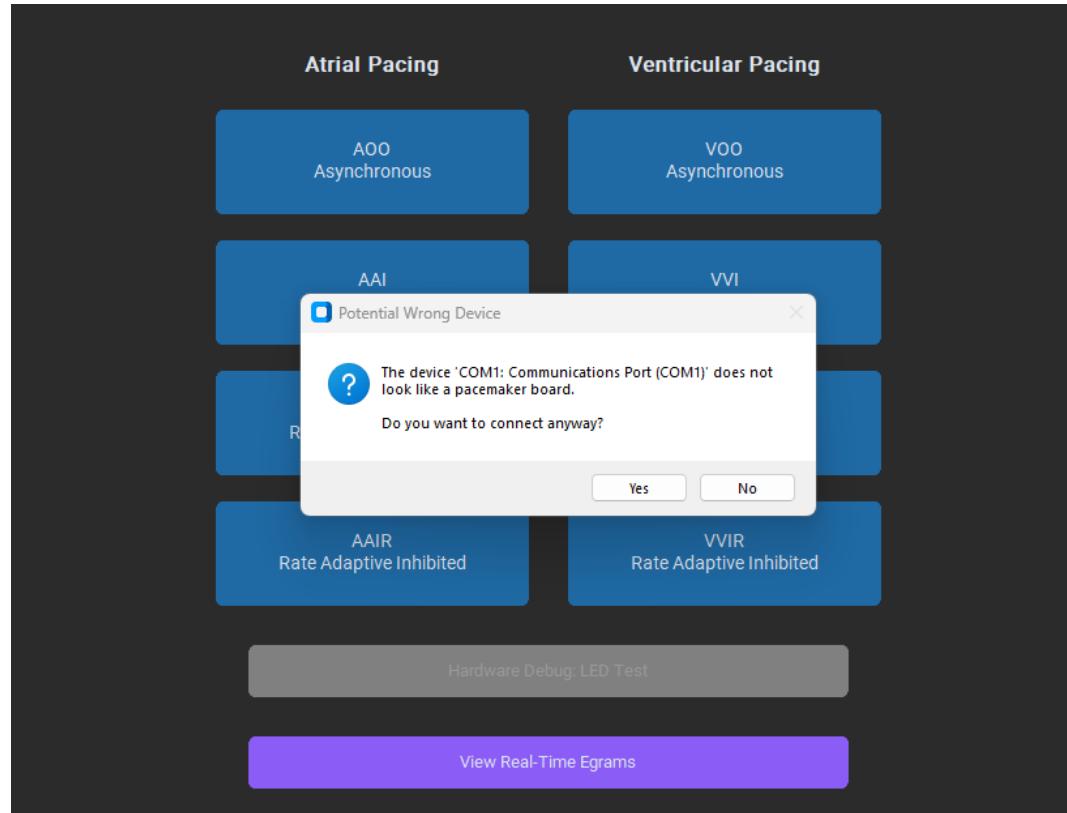


Figure 36: Warning of Potentially Wrong Device

This is done through this code snippet in the controller.py file:

Code 2: Correct Device Check

```

1 # --- SAFETY CHECK ---
2 safe_keywords = ["mbed", "OpenSDA", "NXP", "DAPLink", "JLink", "Segger"]
3
4 is_likely_safe = any(keyword.lower() in port_name_display.lower() for
5     keyword in safe_keywords)
6
7 if not is_likely_safe:
8     response = messagebox.askyesno(
9         "Potential Wrong Device",
10        f"The device '{port_name_display}' does not look like a pacemaker
11        board.\n\n"
12        "Do you want to connect anyway?"
13    )

```

Connecting anyway will not allow you to send data through as a safety feature for unrecognized devices.

If the pacemaking device is connected however, it will prompt confirming connection before establishing serial communication.

Lastly, the middle section has been expanded to both allow for expansion when font size is increased and to support all the new modes alongside the EKG view.

5.4.3 Parameter Pages Design & Decisions

The parameter page has been changed to account for all new parameters. Previously, we greyed out non-used parameters for each mode however, with the increase of parameters the view will get cluttered and more complicated to navigate. Thus, only changeable parameters for each mode is shown:

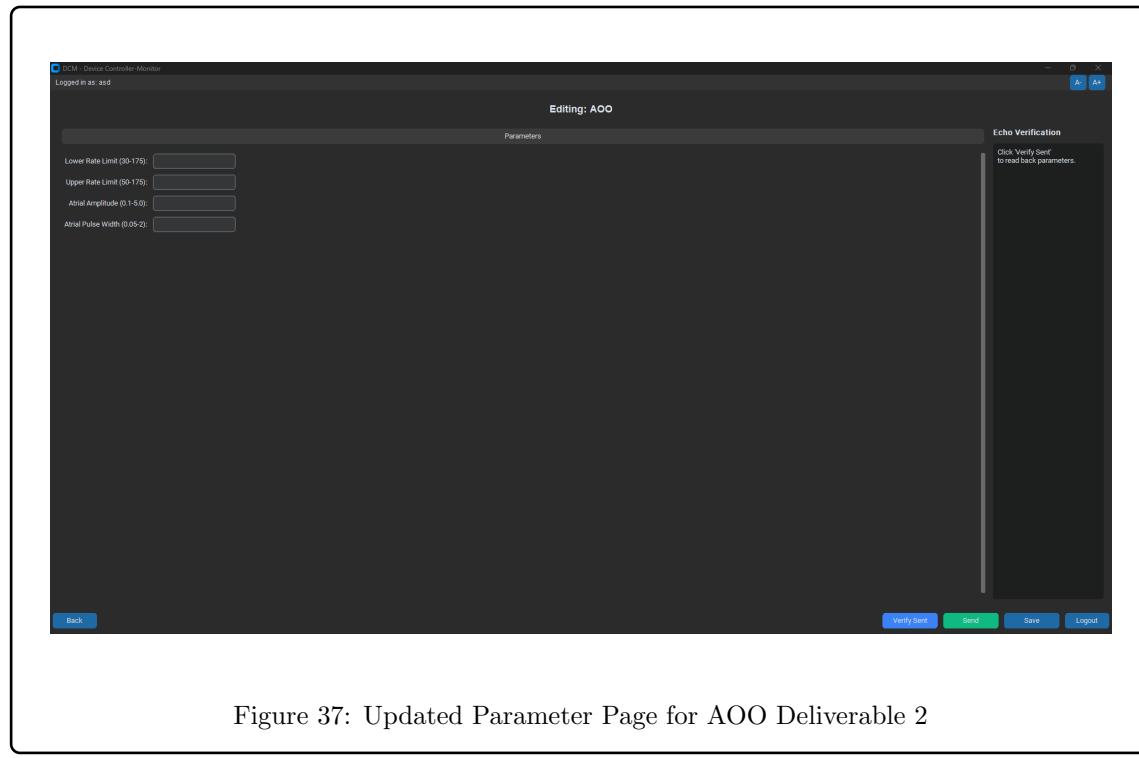


Figure 37: Updated Parameter Page for AOO Deliverable 2

This view also contains more buttons for the serial communication side of things. These features are shown on the left side of the screen with buttons verify send, send, save and the bar for echo verification.

The save button saves the parameters locally for convenience so you do not have to remember all parameters and set them all again each time you want to modify values.

The verify send sends a packet of data to the simulink that asks for the parameter values and is sent back. This is a safety feature to ensure that the right data and parameters were sent to the simulink and nothing was improperly sent or corrupted.

The send button sends a packet of data containing all the parameters to the simulink. This is in compliance with Table 11. To ensure simplicity the non-used values are just replaced with 0.0s. The simulink handling the packet unpacking is shown in below where rxdata and status are provided by the serial recieve block:

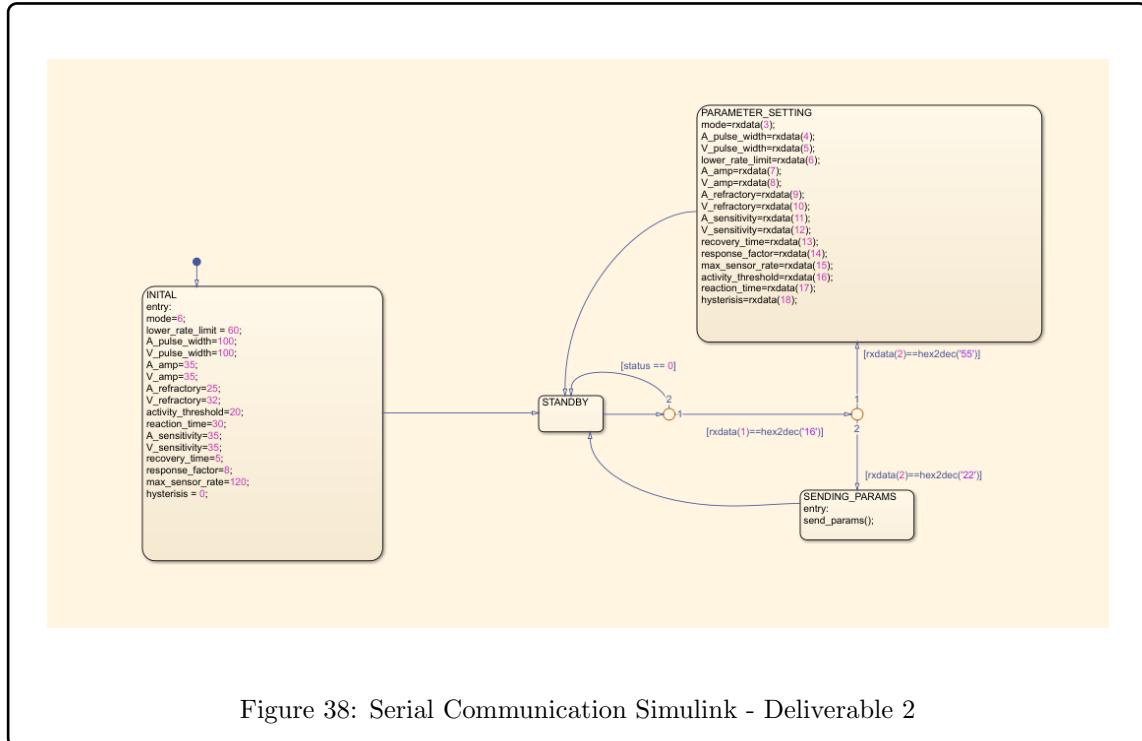


Figure 38: Serial Communication Simulink - Deliverable 2

From this, we see as well how the echo works by sending a second byte of 22 rather than 55. The send_params() function sends all parameters back by MUXing all data and sending it back via a serial transmit block.

For some pages the format is slightly different such as the rate adaptive modes. As the parameter is a word with an approximate number value a drop down menu seen below is used to select.

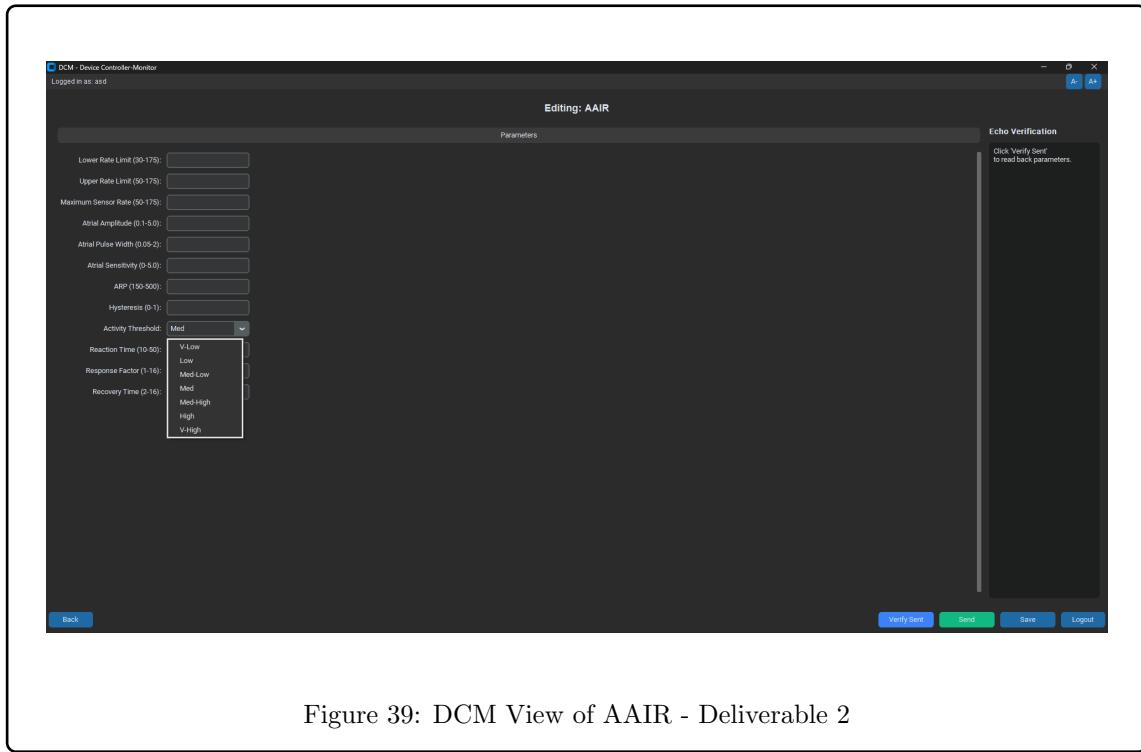


Figure 39: DCM View of AAIR - Deliverable 2

This corresponds to numerical values in the code of 1.5 at V-LOW with increments of 0.5 with higher levels of activity threshold. These values were done through testing with the use of the accelerometer as a nominal value of 1 through stationary action and a max value of 7 was found.

5.4.4 EKG View Design & Decisions

The EKG is handled through the `ekg_model.py` and `ekg_view.py`. It takes analog data from the pins relating to the atrium and ventricle. It samples the data and plots it on a graph using the `matplotlib` python library. This was used as it gave many useful features such as the ability to scroll and zoom on the EKG data. The view for the EKG can be seen below:



Figure 40: View of EKG Page

This page also allows you to isolate whether or not you want to look at just the atrial pulses, ventricular pulese or both with functions on the bottom left to zoom, scroll save and configure the subplots. The view of the EKG can be seen below with dummy data:

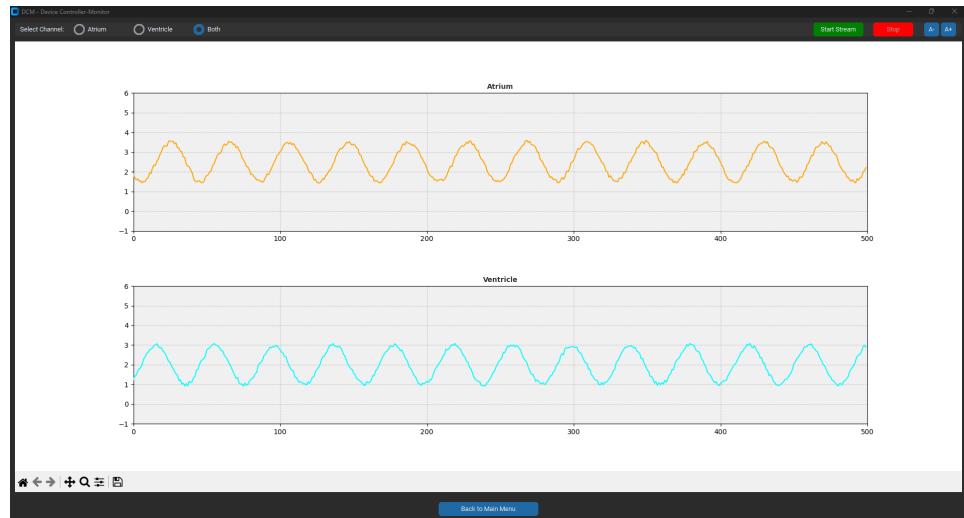


Figure 41: EKG View with Data

5.5 Assurance Cases

Some features of safety were covered in the design section of this documentation, however this section will be more comprehensive.

5.5.1 Clamping of Heart Rate For Rate Adapative Pacing

Safety Goal: For rate adaptive pacing modes, the heart rate must not be lower than the lower rate limit and the max sensor limit.

Rationale: To prevent the heart from beating too slow, lower than lower rate limit, or beating too fast, higher than max sensor limit, to ensure healthy heart operation.

Evidence: The clamp is hard locked from via how its data is processed in simulink seen in the images below. However, this can be seen live through testing as shown in the demo. The clamp is used in two spots to ensure redundancy as extra signal processing occurs.

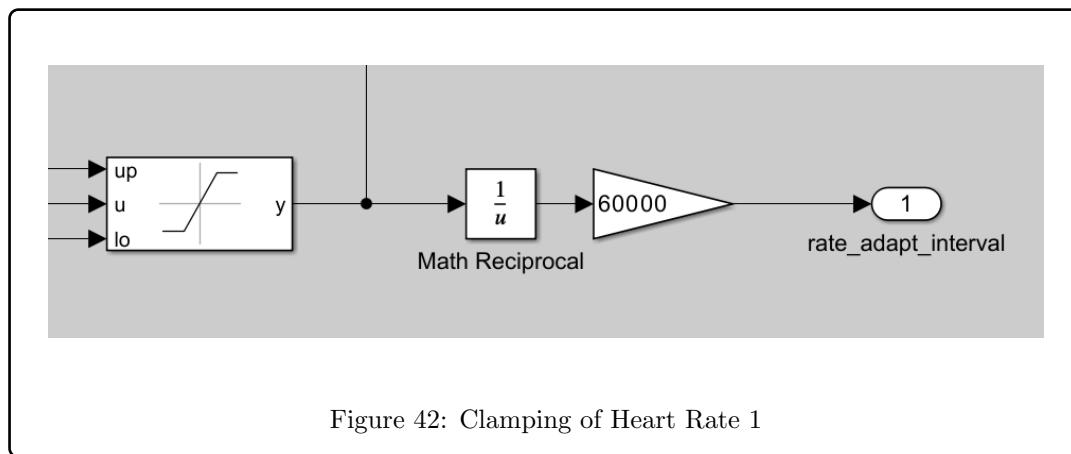


Figure 42: Clamping of Heart Rate 1

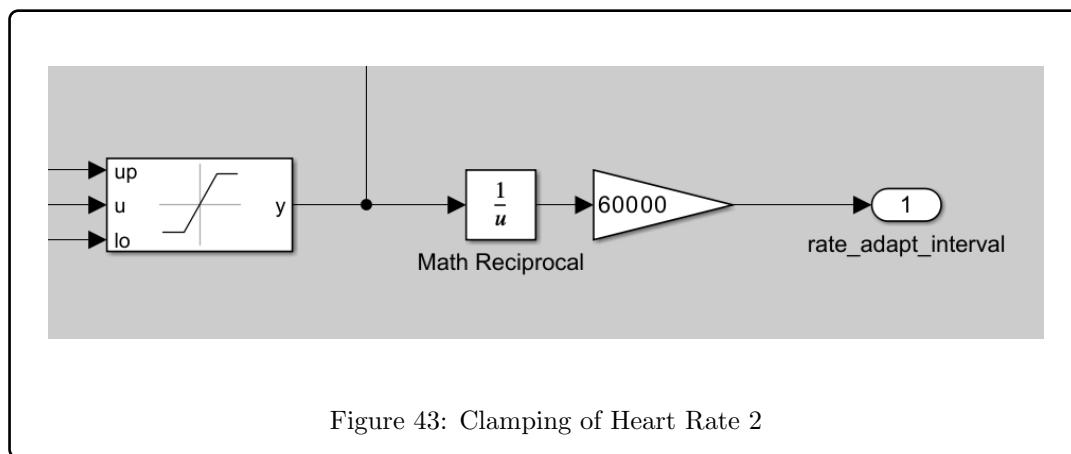


Figure 43: Clamping of Heart Rate 2

5.5.2 Echo Parameter Verification

Safety Goal: Ensuring that the data transmitted to the pacemaker by the DCM is correct.

Rationale: The ability to verify the data sent with an echo feature is imperative to ensure data was not corrupted or mismanaged when being sent to the pacemaker. Without this verification it can potentially cause issues by setting unsafe limits or incorrect parameters causing harm long term.

Evidence: The below images are of the Simulink sending the parameters back as well as the DCM sending back the sent parameters.

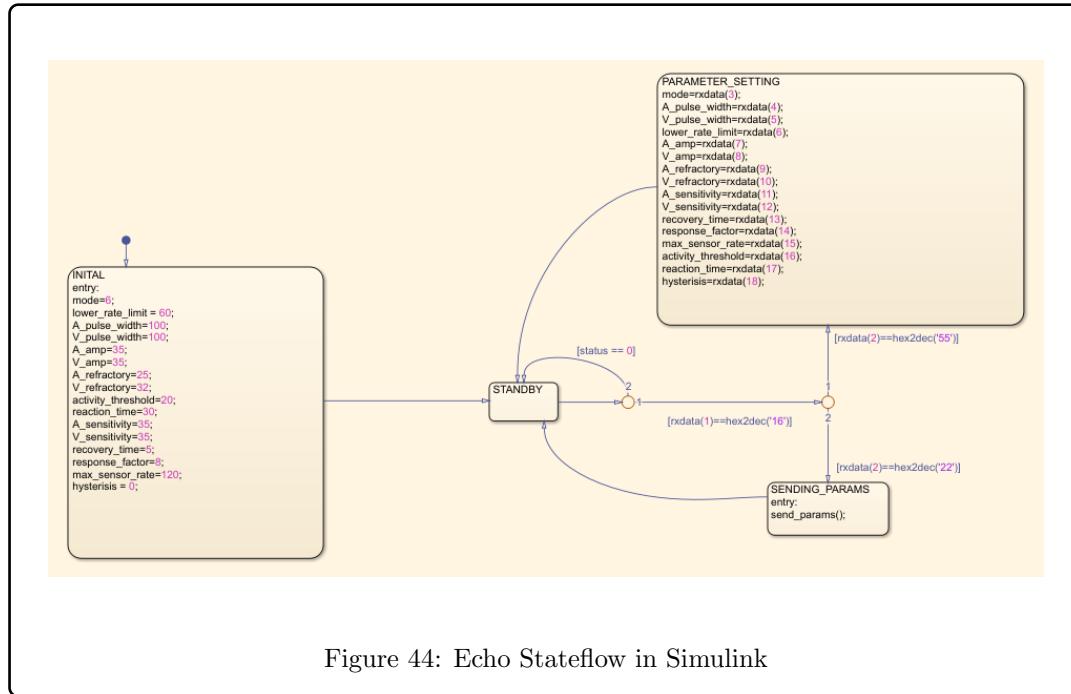


Figure 44: Echo Stateflow in Simulink

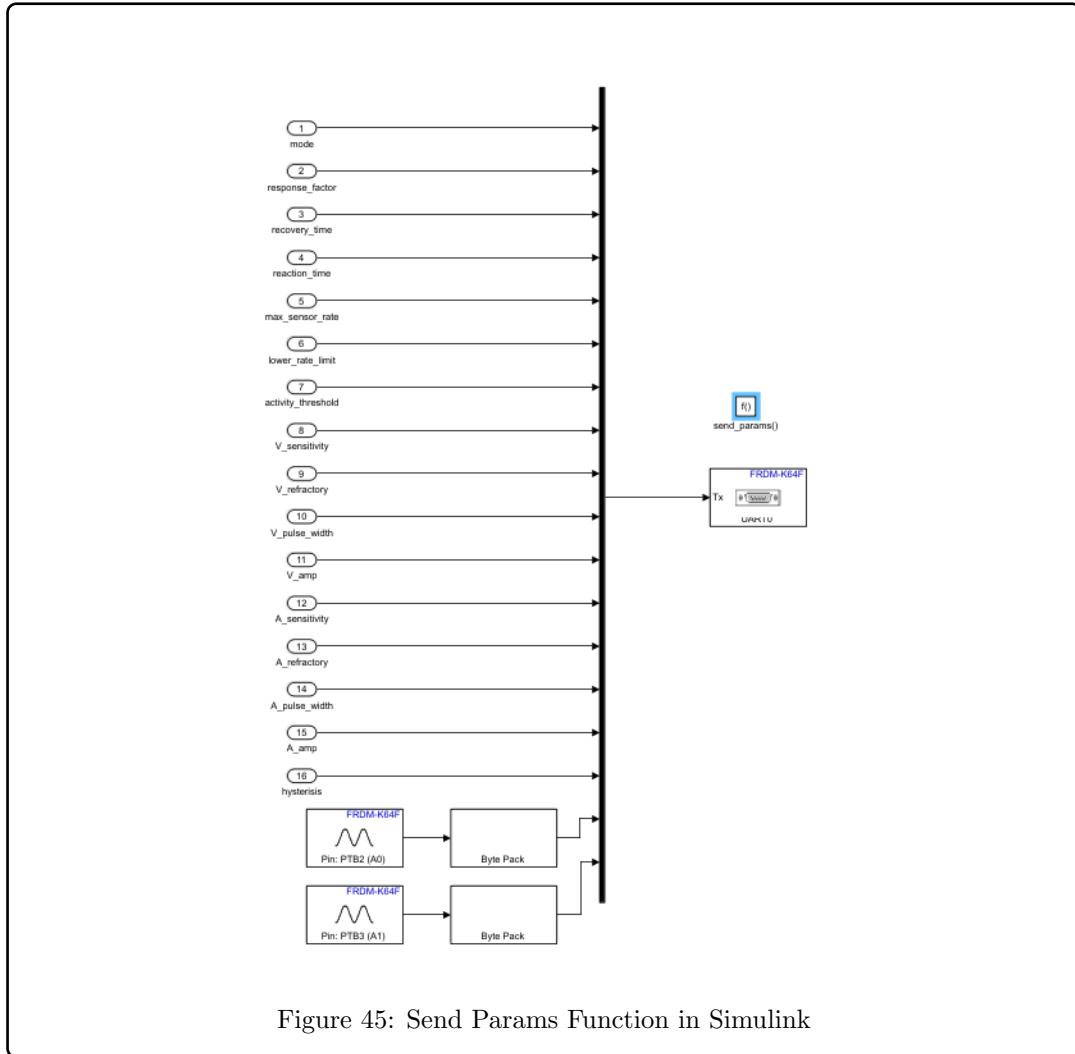


Figure 45: Send Params Function in Simulink

Code 3: Echo Function in DCM

```

1 def get_echo(self):
2     if not self.ser or not self.ser.is_open: return None
3     try:
4         self.ser.reset_input_buffer()
5         self.ser.write(struct.pack(self.FMT_11_BYTES, 0x16, 0x22,
6             0,0,0,0.0,0))
7         resp = self.ser.read(9)
8         if len(resp) != 9: return None
9         u = struct.unpack('<BBBBf', resp)
10        return {"red": u[0], "green": u[1], "blue": u[2], "switch_time": u[3], "off_time": u[4]}
11    except Exception: return None

```

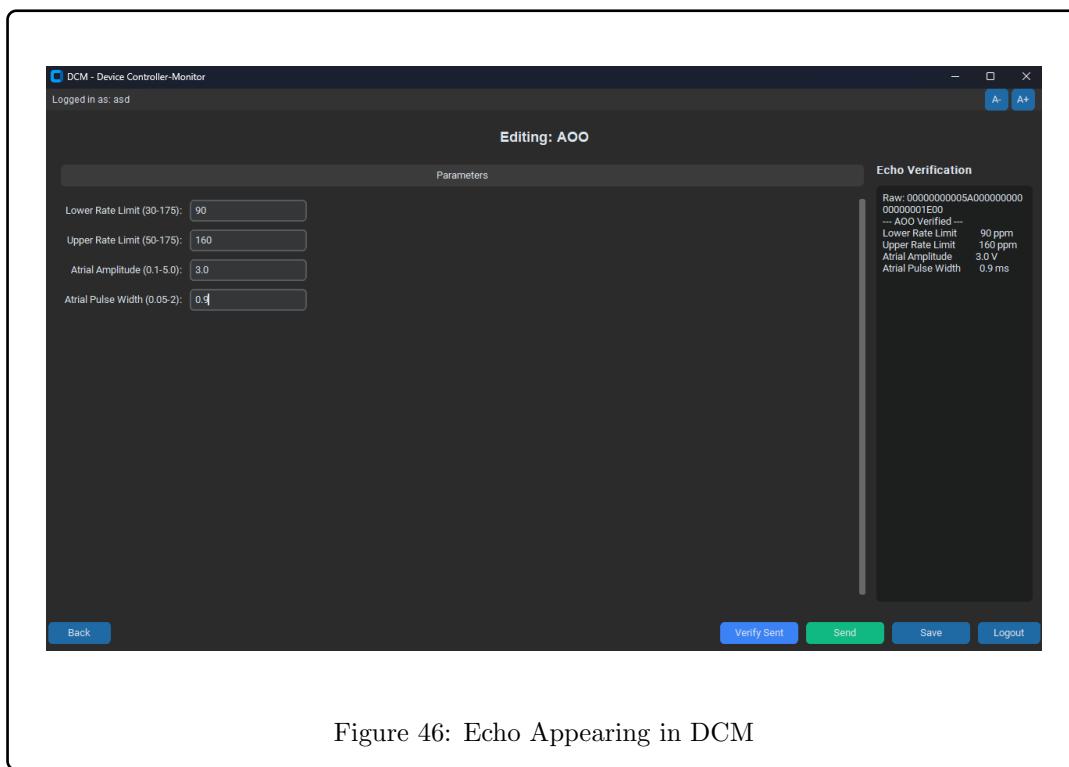


Figure 46: Echo Appearing in DCM

5.5.3 Reset Between Mode Switching

Safety Goal: Prevent unwanted data when switching between pacing modes of the pacemaker.

Rationale: When switching between modes a default state is returned to before switching to a new mode. This mode grounds all values to prevent erratic and unwanted behaviour between pacing such as switching to another mode mid pace which may cause complications

Evidence: This is soft-locked by the stateflow diagram shown below:

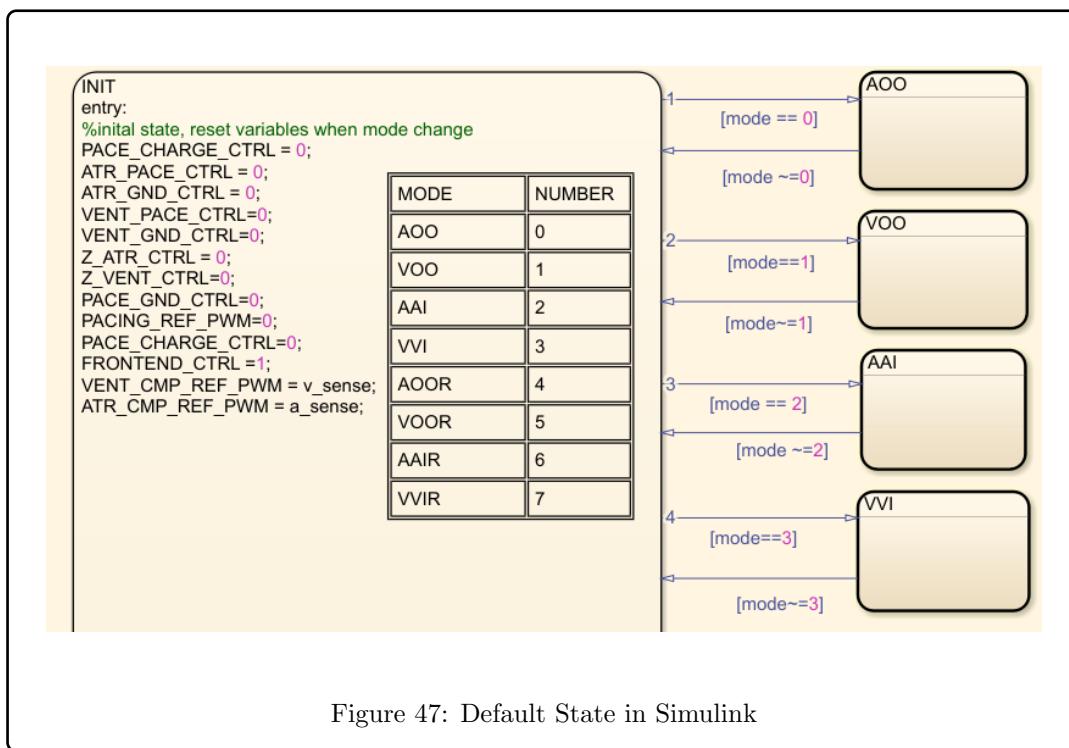


Figure 47: Default State in Simulink

5.5.4 Incorrect Device Handling

Safety Goal: When trying to use an unrecognized device the DCM will not allow serial communication and warn the user.

Rationale: To prevent the DCM from harming an unrecognized device by sending data and to warn the user that the pacemaker is currently not the device being configured.

Evidence: A warning message pops up when an unrecognized device is connected to with code to check the COM for recognizable identifiers.

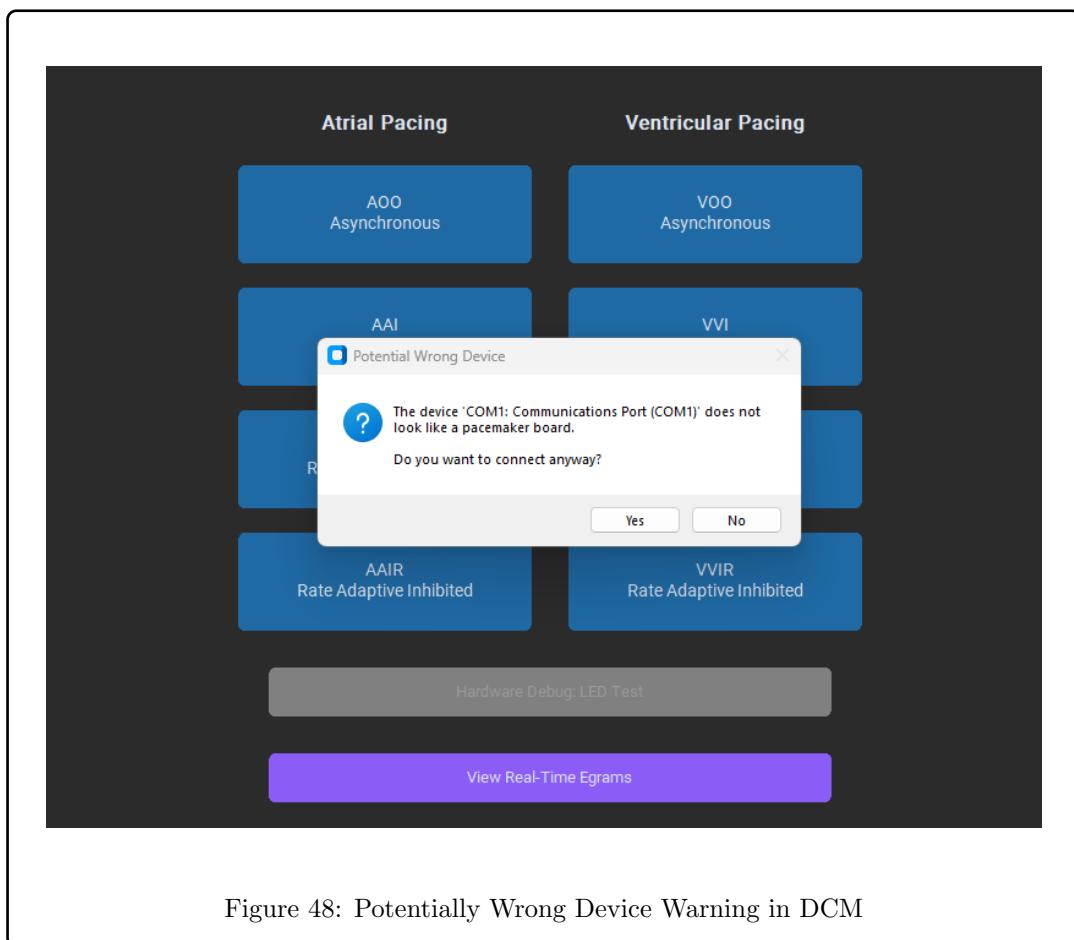


Figure 48: Potentially Wrong Device Warning in DCM

Code 4: Safety Check of Device in Code

```

1 # --- SAFETY CHECK ---
2 safe_keywords = ["mbed", "OpenSDA", "NXP", "DAPLink", "JLink",
3                  "Segger"]
4
5 is_likely_safe = any(keyword.lower() in port_name_display.lower
6                      () for keyword in safe_keywords)
7
8 if not is_likely_safe:
9     response = messagebox.askyesno(
10         "Potential Wrong Device",
11         f"The device '{port_name_display}' does not look like a
12             pacemaker board.\n\n"
13         "Do you want to connect anyway?"
14     )

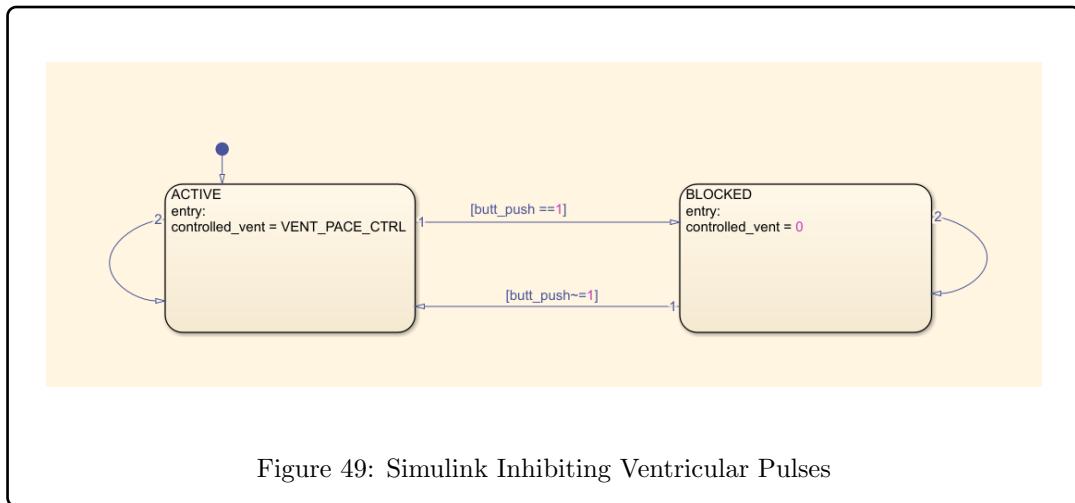
```

5.5.5 Push Button Shutoff

Safety Goal: A quick way to manual shutoff pacing of ventricular pulses.

Rationale: In case the ventricular pacing malfunctions a pushbutton to quickly stop the pacing can be used.

Evidence: Shown in demo but can be seen in the code logic



6 Requirements Potential Changes

Pacing Modes: Addition of all pacing modes such as dual chamber modes.

Strict Pacing Times: The pacing loop must execute within a strict maximum execution time. If a pulse needs to be accurate the code can not be taking time to process serial data.

Must be Optimized for Battery Life: Our pacemaker has nearly infinite power being connected to dedicated power supplies, however, pacemakers need to run off batteries for very long periods of time.

Magnet Safe: The pacemaker may experience EMI and will need to be able to pass a magnet test before being able to be deployed. EMI may cause malfunctions or failures of equipment and is thus crucial for safety critical operation.

7 Design Decision Potential Changes

Interrupt Based Approach: Using hardware based timers to detect events rather than polling.

Implementation of a Watchdog: Using a watch dog timer to detect when a failure occurs such as a looping or no pulse when one is needed and fixing accordingly. A doctor can not be watching a patient 24/7 and must be able to correct its own errors.

8 Module Description

8.1 Simulink System (Model-Based Design)

Description

The Simulink project functions as a closed-loop simulation environment designed to validate the pacemaker logic against a simulated patient heart model before hardware deployment. It consists of two primary subsystems: the **Device Model** (Pacemaker) and the **Environment Model** (Heart & Activity).

Key Sub-Modules

1. Heart & Patient Model (Environment)

- **Function:** Simulates intrinsic cardiac activity (Atrial and Ventricular signals) and patient physical activity.
- **Inputs:** User-adjustable sliders for Natural Heart Rate (bpm), pulse widths, and AV Delay.
- **Outputs:** Generates analog-like electrical signals representing the heart's intrinsic behavior to feed into the sensing channels of the pacemaker.

2. Pacemaker Logic (Device)

- **State Machine:** Implements the timing cycles and logic for standard pacing modes (AOO, VOO, AAI, VVI) and their rate-adaptive counterparts (AOOR, VOOR, AAIR, VVIR) as well as the bonus DDDR.
- **Rate Adaptation Block:** Processes simulated accelerometer input to calculate the activity level. It utilizes tunable parameters found in the testing documents: Activity Threshold, Reaction Time (time to increase rate), Response Factor, and Recovery Time (time to decrease rate).
- **Timing Control:** Manages Lower Rate Limit (LRL), Upper Rate Limit (MSR), Refractory Periods (ARP/VRP/PVARP), and Hysteresis.
- **Pulse Generation:** Outputs logical pacing triggers based on the expiration of timing cycles or lack of intrinsic sensing.

8.2 DCM Application

Description

The DCM is a desktop application built using Python and `customtkinter` that follows a Model-View-Controller (MVC) architecture. It manages user authentication, programmable device parameters, real-time data visualization, and serial communication with the pacemaker hardware (FRDM-K64F).

Architecture Breakdown

A. Controller (`controller.py`)

- **Role:** The core of the application.
- **Functionality:**
 - Initializes the main application window and manages navigation between views (Welcome, Login, Main Dashboard, Egrams).
 - Holds the global state for the active session (current user, serial connection status, font size settings).
 - Mediates data flow between the GUI Views and the Backend Models.
 - Manages the background thread for Text-to-Speech audio feedback upon connection.

B. Models (Data & Logic Layers)

- **User Model** (`models/user_model.py`):
 - Manages the local database (`users.json`).
 - Handles user registration (limit 10 users) and authentication logic.
- **Pacing Model** (`models/pacing_model.py`):
 - Manages persistence of device parameters (`pacing_settings.json`).
 - Stores and retrieves parameter sets keyed by User → Pacing Mode.
- **Serial Communication** (`models/serial_comms.py`):
 - **Driver Layer:** Wrapper around `pyserial`. Handles port scanning, connection, and disconnection.
 - **Protocol Handling:** Implements the byte-level communication protocol using `struct`.
 - * *Parameter Send:* Packs 18 parameters (Mode, LRL, MSR, Amplitudes, Pulse Widths, Rate Adaptive settings) into binary packets.
 - * *Echo/Verification:* Requests and unpacks data from the hardware to verify saved settings.
 - * *Egram Streaming:* Implements a sync-based reader that buffers incoming real-time cardiac data (Atrial/Ventricular floats) for graphing.

C. Views (GUI Layer)

- **Login Views** (`views/login_views.py`):
 - Provides the “Welcome” and “Register” screens.
 - Handles input validation for creating new users and logging in.
- **Main View** (`views/main_view.py`):
 - **Dashboard:** The primary interface for the user.

- **Mode Selection:** Allows selection of 8 pacing modes (AOO, VVI, VVIR, etc.).
- **Parameter Entry:** Dynamically generates input fields based on the selected mode. Includes strict input validation (e.g., LRL 30-175ppm, Reaction Time 10-50s) and error handling.
- **Hardware Debug:** A dedicated panel for testing LED indicators on the board via serial commands.
- **Egram View (views/egram_view.py):**
 - **Real-Time Plotting:** Uses `matplotlib` with the `TkAgg` backend to render electrograms.
 - **Performance:** Implements Blitting (drawing only changed pixels) to ensure smooth animation of high-frequency signal data.
 - **Controls:** Allows the user to Start/Stop streaming and toggle between Atrial, Ventricular, or Dual-channel views.

D. Accessibility Features

- **Global Font Scaling:** The application supports dynamic font resizing (`A+/A-` buttons) across all views to assist visually impaired users.
- **Audio Feedback:** Uses `pyttsx3` to provide auditory confirmation when the pacemaker hardware is successfully connected.

9 Testing

9.1 SimuLink Mode Testing

9.1.1 Testing of AOO

Purpose: The purpose of this test is to test basic AOO functionality.

Input Conditions: General inputs of mode = 0, AOO, and hysteresis = 0, off, and standard atrial inputs.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heart beats.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 10.

Result: Pass

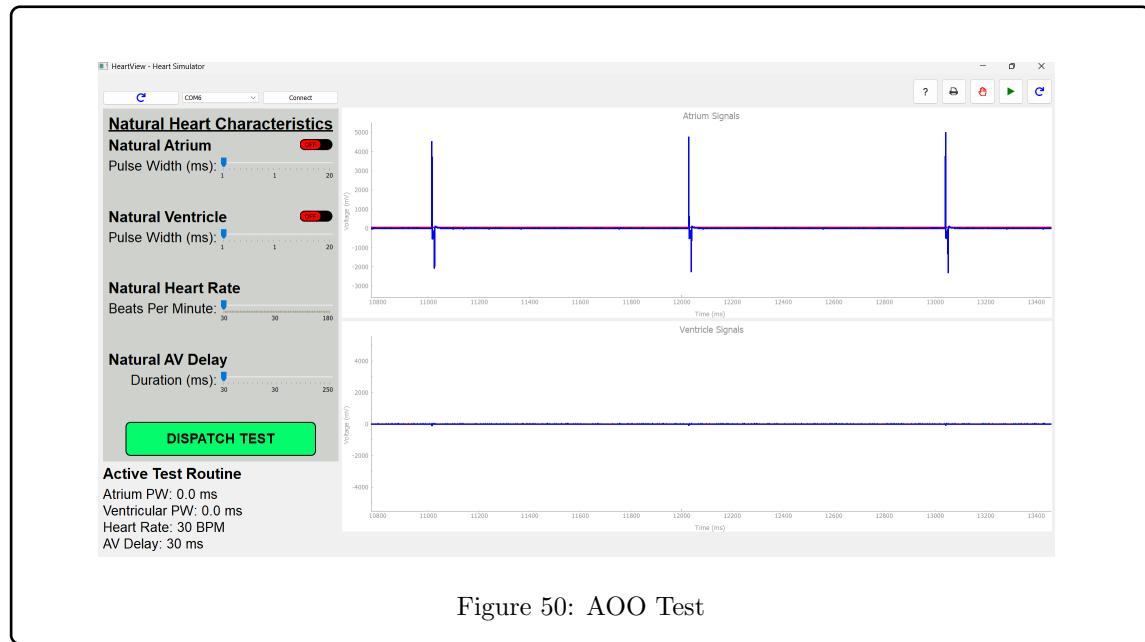


Figure 50: AOO Test

Figure 11 below shows a zoomed in view of one of the pulses in the AOO test above, Figure 10.

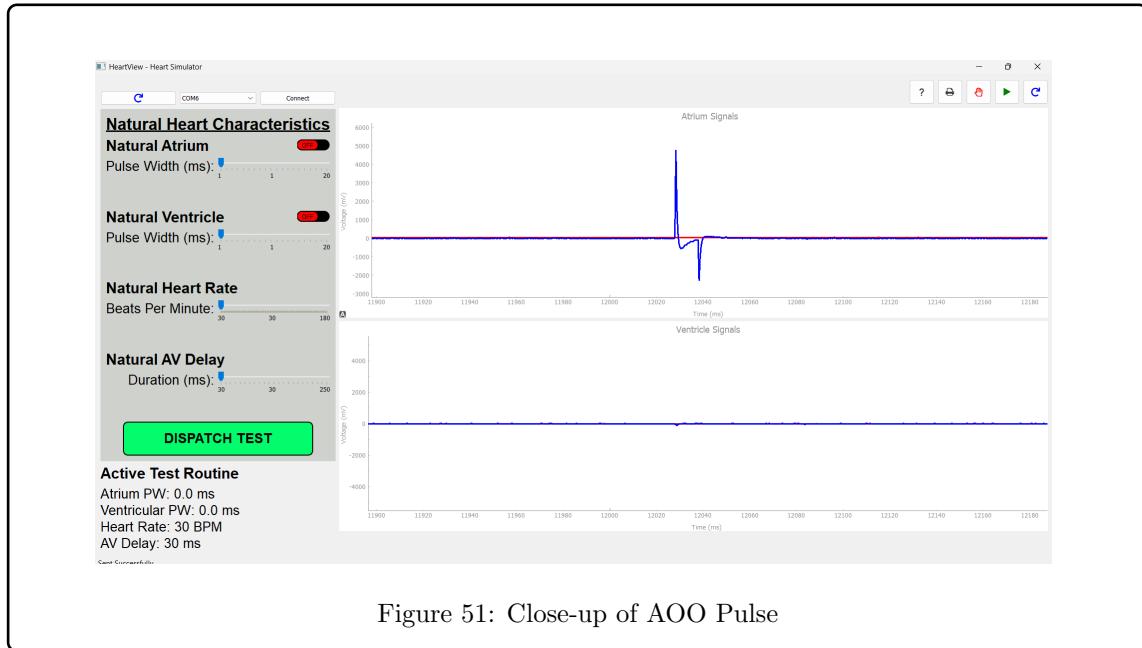


Figure 51: Close-up of AOO Pulse

9.1.2 Testing of VOO

Purpose: The purpose of this test is to test basic VOO functionality.

Input Conditions: General inputs of mode = 1, VOO, and hysteresis = 0, off, and standard ventricle inputs.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heart beats.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 12.

Result: Pass

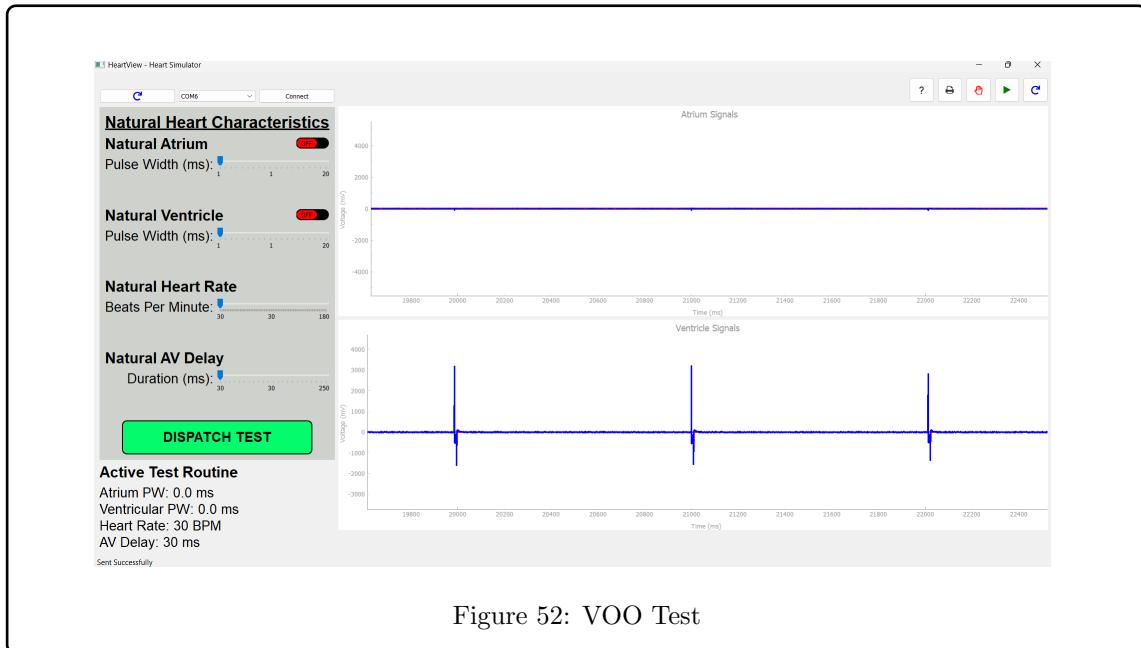


Figure 52: VOO Test

Figure 13 below shows a zoomed in view of one of the pulses in the VOO test above, Figure 12

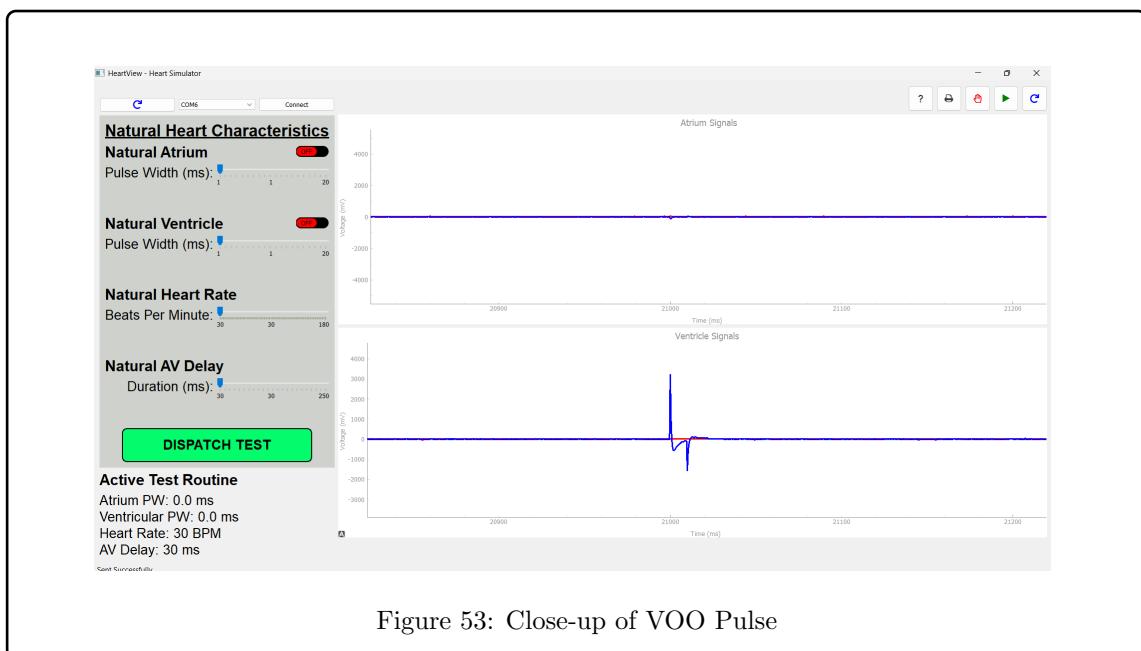


Figure 53: Close-up of VOO Pulse

9.1.3 Testing of AAI

No Natural Heart Rate

Purpose: The purpose of this test is to test basic AAI functionality with no natural heart rhythms.

Input Conditions: General inputs of mode = 2, AAI, and hysteresis = 0, off, and standard partial inputs.

Expected Output: Consistent, evenly spaced pulses in the output.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 13.

Result: Pass

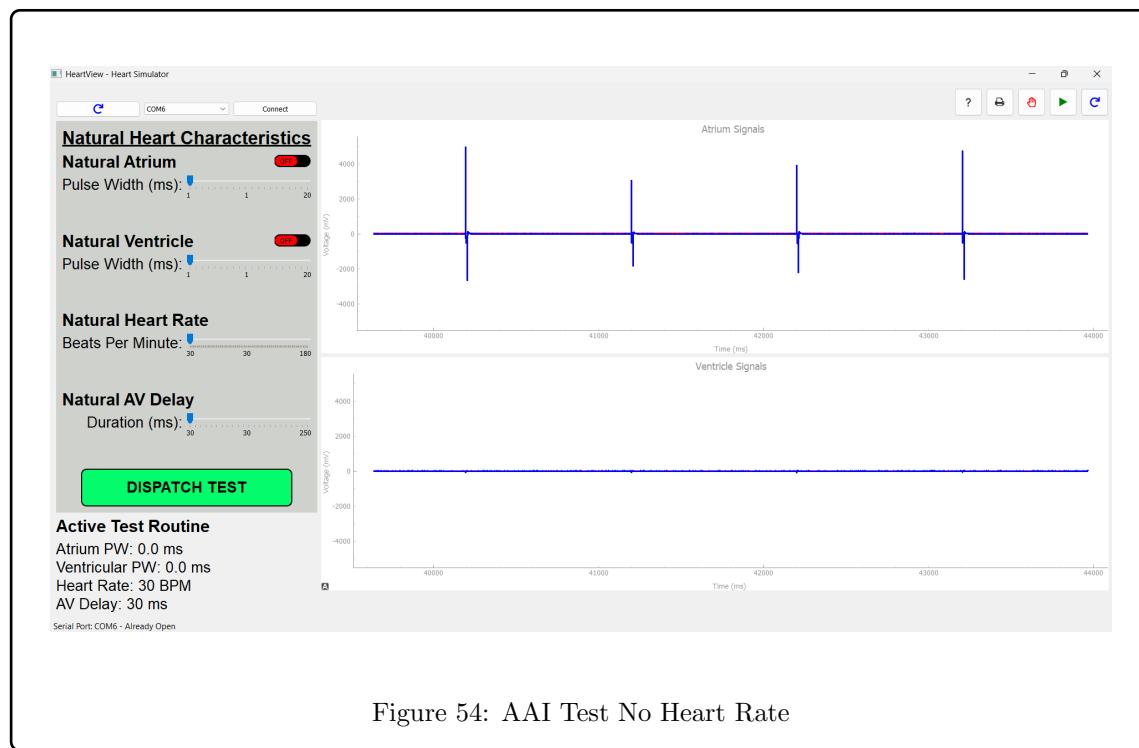


Figure 54: AAI Test No Heart Rate

Natural Heart Rate of 45 BPM

Purpose: The purpose of this test is to test basic AAI functionality at a low heart rate.

Input Conditions: General inputs of mode = 2, AAI, and hysteresis = 0, off, standard atrial inputs, and monitored atrial pulses at 45 BPM.

Expected Output: The pacemaker should pulse after the refractory period expires. An output of blue pulses right before the natural red pulses is expected.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 15.

Result: Pass

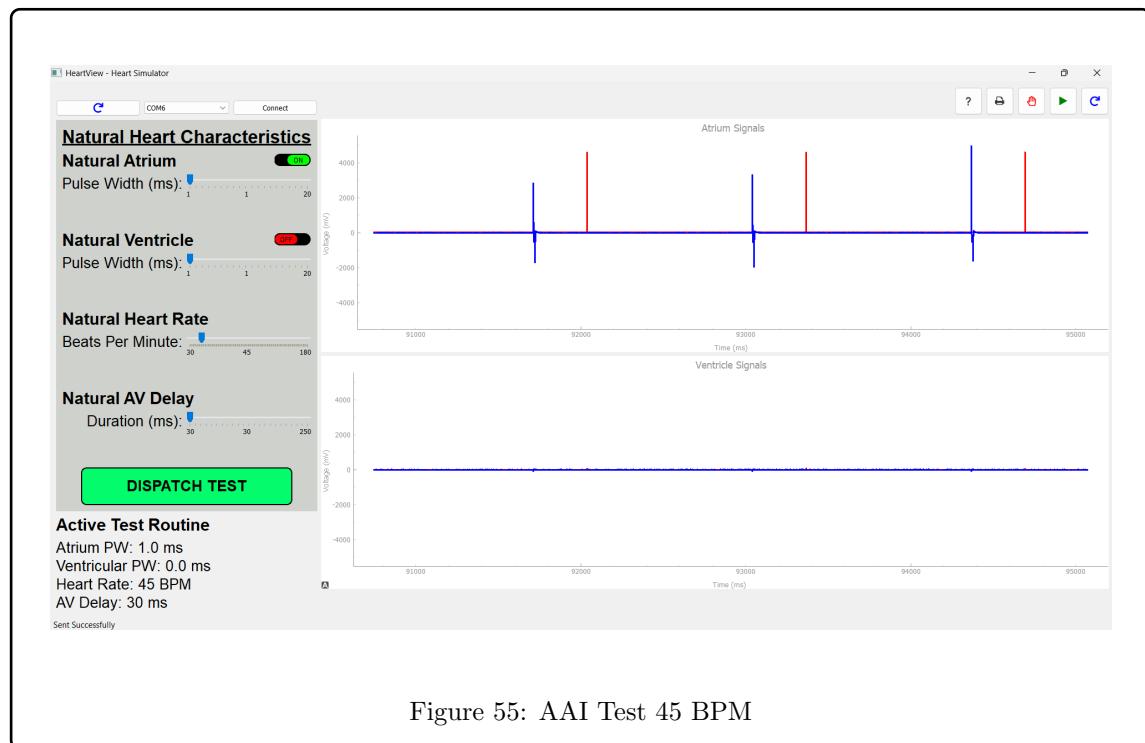


Figure 55: AAI Test 45 BPM

Natural Heart Rate of 75 BPM

Purpose: The purpose of this test is to test basic AAI functionality at a nominal heart rate.

Input Conditions: General inputs of mode = 2, AAI, and hysteresis = 0, off, standard atrial inputs, and monitored atrial pulses at 75 BPM.

Expected Output: As a nominal heart rate is being inputted, the pacemaker should not be delivering pacing pulses as the simulated heart rate is nominal and healthy.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 16.

Result: Pass

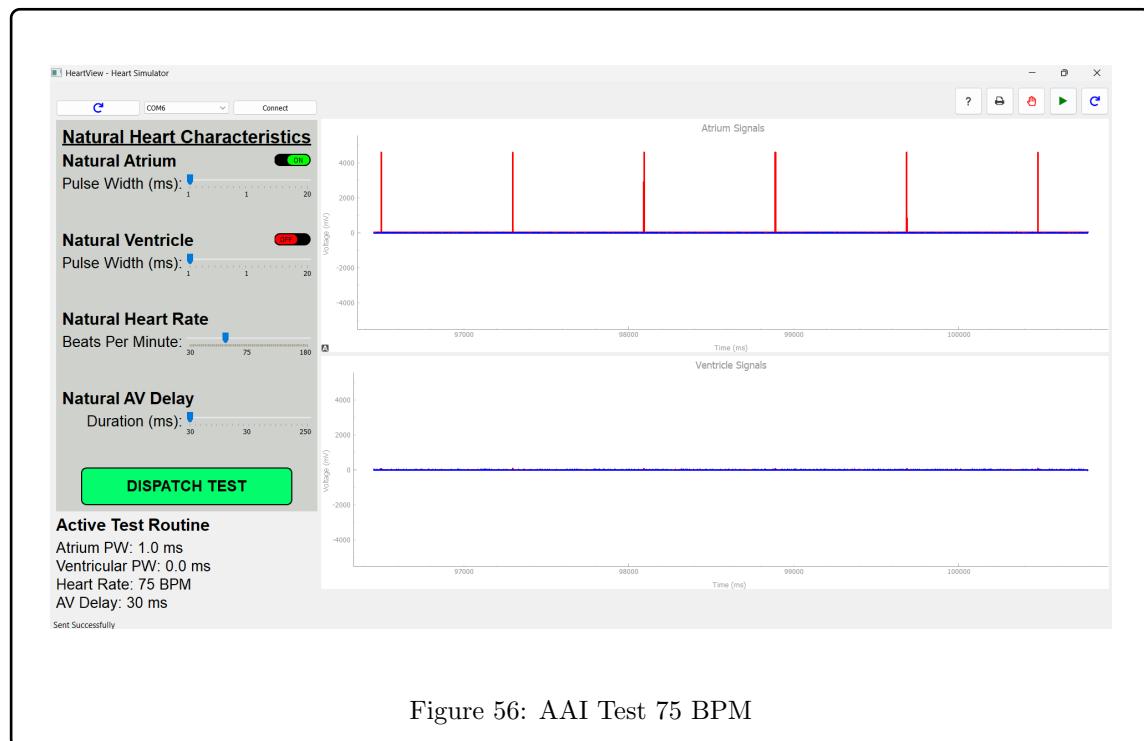


Figure 56: AAI Test 75 BPM

9.1.4 Testing of VVI

No Natural Heart Rate

Purpose: The purpose of this test is to test basic VVI functionality with no inputted heart rate.

Input Conditions: General inputs of mode = 3, VVI, and hysteresis = 0, off, standard ventricle inputs, and monitored ventricle pulses.

Expected Output: Consistent, evenly spaced pulses in the output.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 17.

Result: Pass

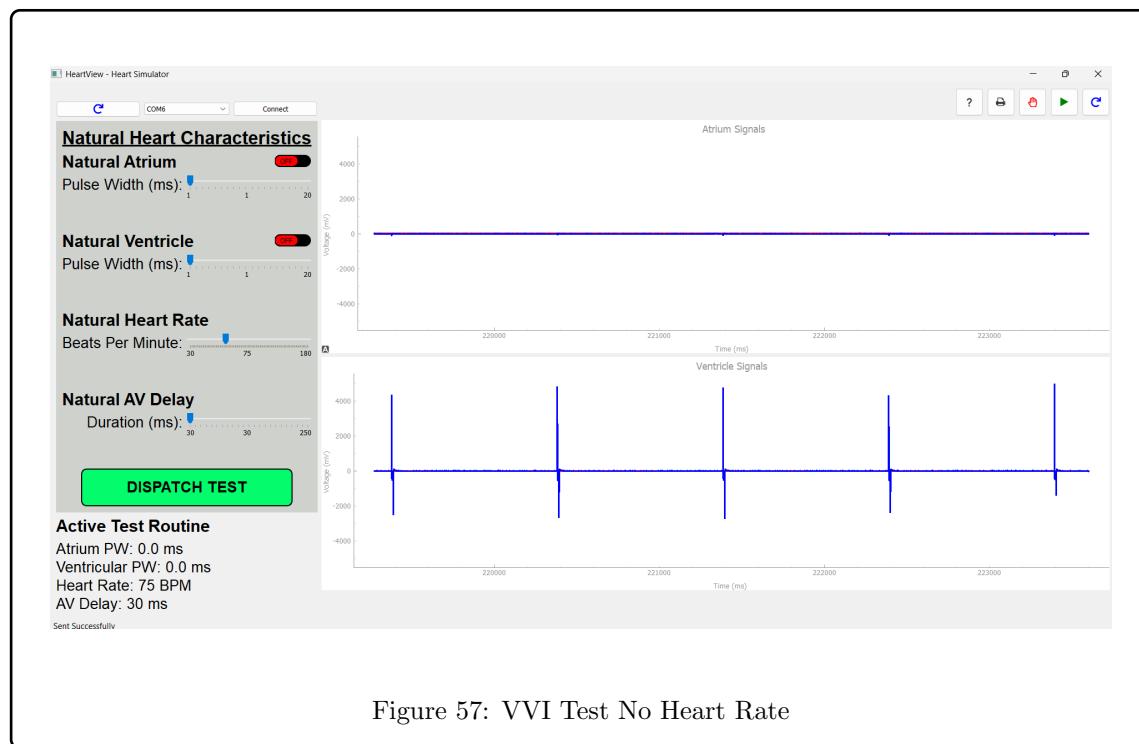


Figure 57: VVI Test No Heart Rate

Natural Heart Rate at 45 BPM

Purpose: The purpose of this test is to test basic VVI functionality with no inputted heart rate.

Input Conditions: General inputs of mode = 3, VVI, and hysteresis = 0, off, standard ventricle inputs, and monitored ventricle pulses.

Expected Output: The pacemaker should pulse after the refractory period expires. An output of blue pulses right before the natural red pulses is expected.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 18.

Result: Pass

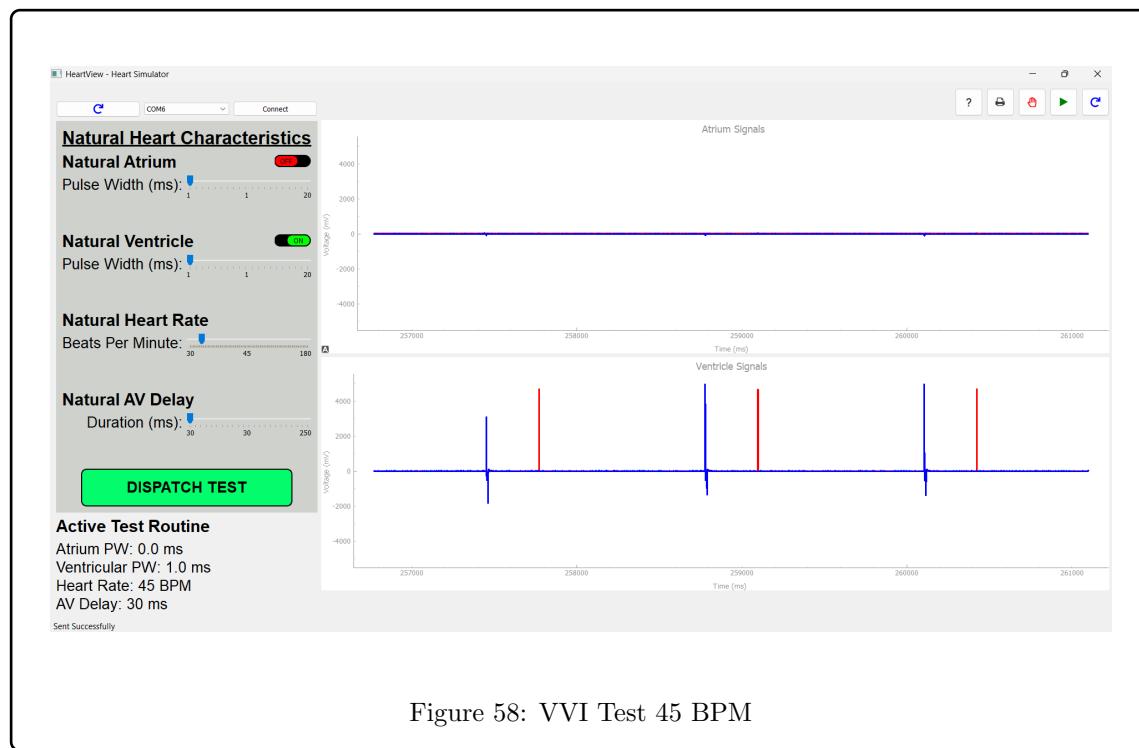


Figure 58: VVI Test 45 BPM

Natural Heart Rate at 75 BPM

Purpose: The purpose of this test is to test basic VVI functionality with no inputted heart rate.

Input Conditions: General inputs of mode = 3, VVI, and hysteresis = 0, off, standard ventricle inputs, and monitored ventricle pulses.

Expected Output: As a nominal heart rate is being inputted, the pacemaker should not be delivering pacing pulses as the simulated heart rate is nominal and healthy.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 18.

Result: Pass

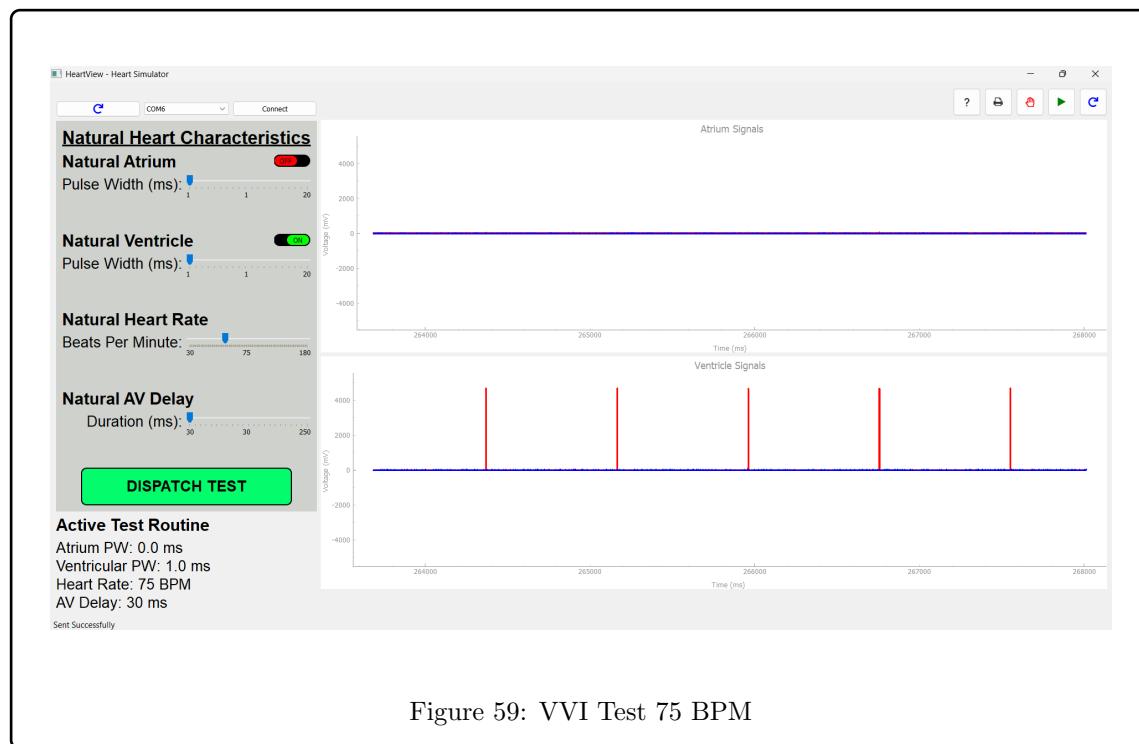


Figure 59: VVI Test 75 BPM

9.1.5 Hysteresis Testing

Hysteresis Test 1 (60 BPM)

Purpose: The purpose of this test is to test basic hysteresis mode functionality.

Input Conditions: General inputs of mode = 2, AAI, and hysteresis = 1, on, standard atrium inputs, and monitored atrial pulses at 50 BPM.

Expected Output: No output of the pacemaker.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 20.

Result: Pass

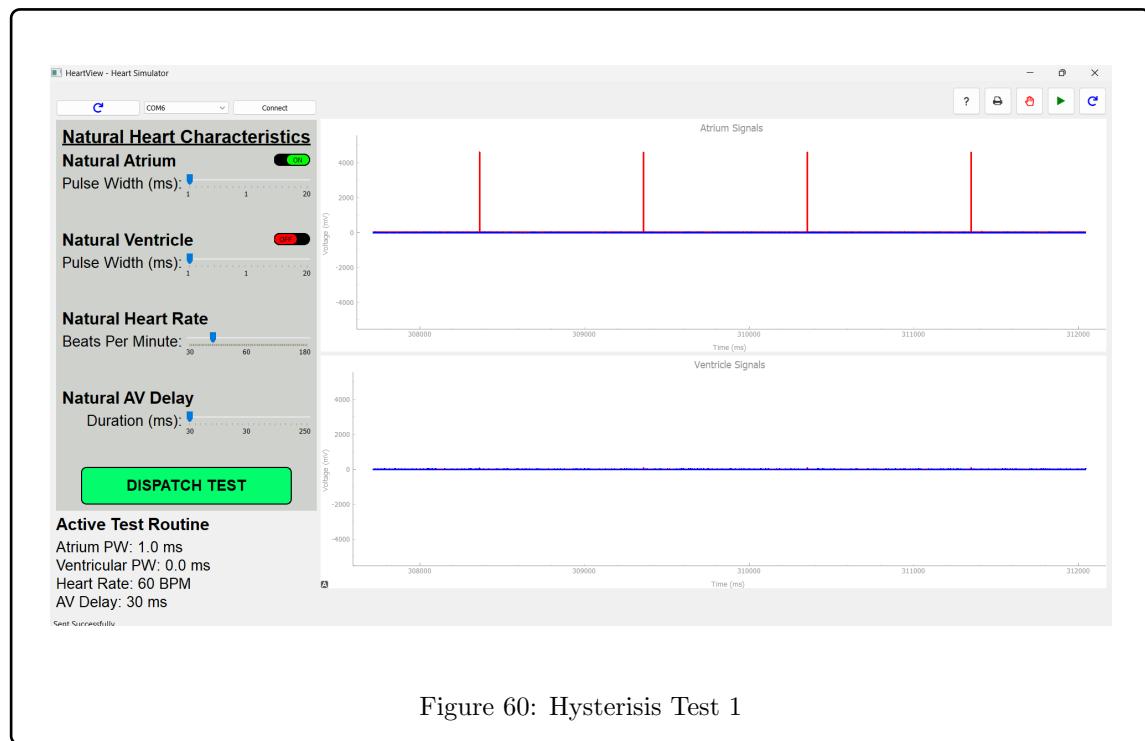


Figure 60: Hysteresis Test 1

Hysteresis Test 2 (50 BPM)

Purpose: The purpose of this test is to test basic hysteresis mode functionality.

Input Conditions: General inputs of mode = 2, AAI, and hysteresis = 1, on, standard atrium inputs, and monitored atrial pulses at 50 BPM.

Expected Output: No output of the pacemaker.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 21.

Result: Pass

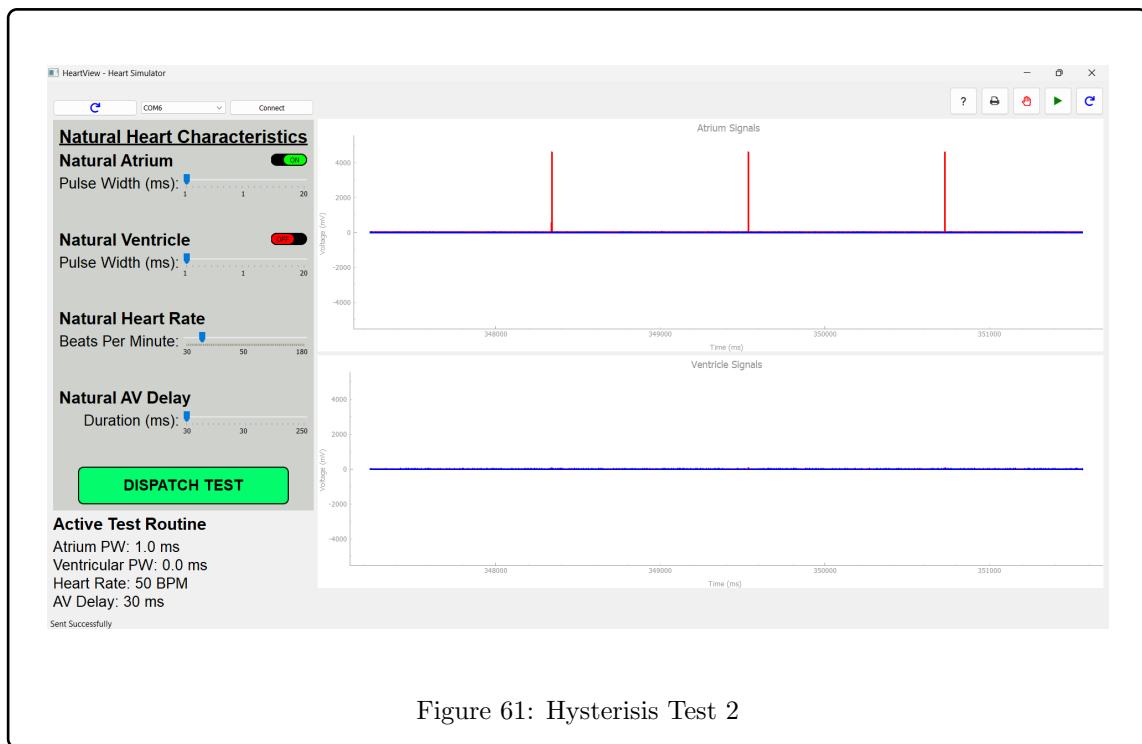


Figure 61: Hysteresis Test 2

Hysteresis Test 3 (40 BPM)

Purpose: The purpose of this test is to test basic hysteresis mode functionality.

Input Conditions: General inputs of mode = 2, AAI, and hysteresis = 1, on, standard atrium inputs, and monitored atrial pulses at 50 BPM.

Expected Output: Delayed output signals from the pacemaker.

Actual Output: Output of testing is exactly that of expected, as shown below in Figure 22.

Result: Pass

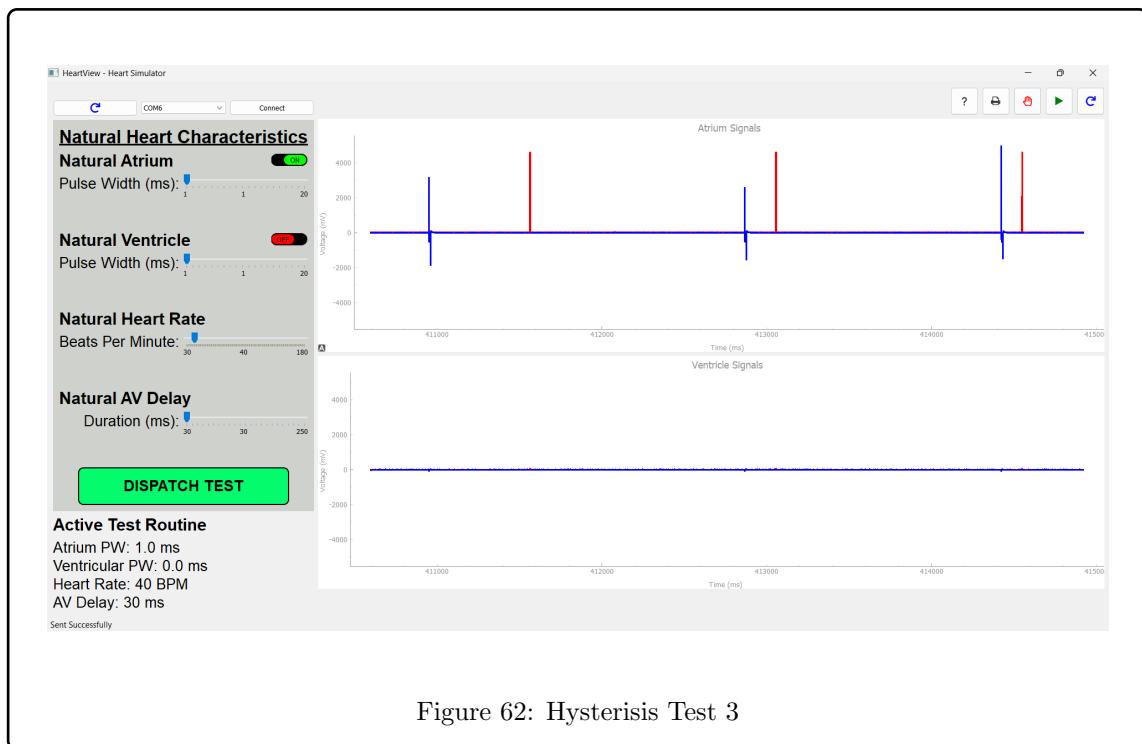


Figure 62: Hysteresis Test 3

9.2 Testing of AOOR

9.2.1 Test 1: AOOR at Rest (No Movement)

Purpose: The purpose of this test is to test basic AOOR functionality at rest.

Input Conditions: General Inputs mode = 4 (AOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats in the atrium signal.

Actual Output: Output tested exactly as expected.

Result: Pass

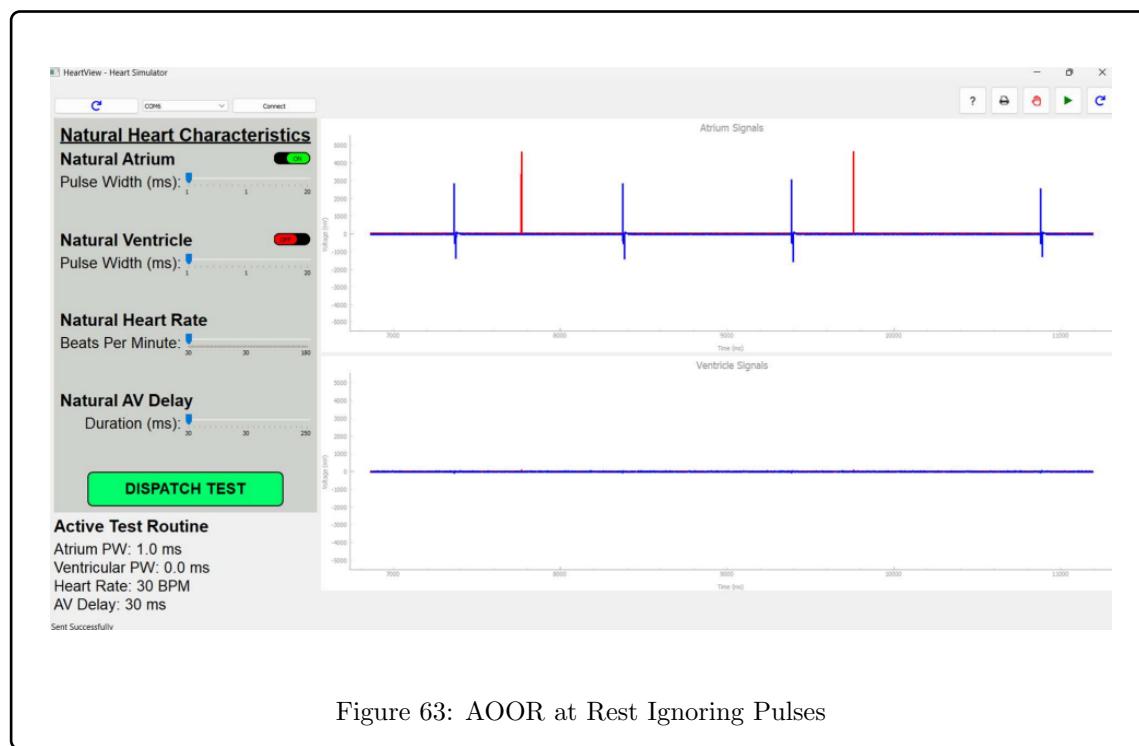
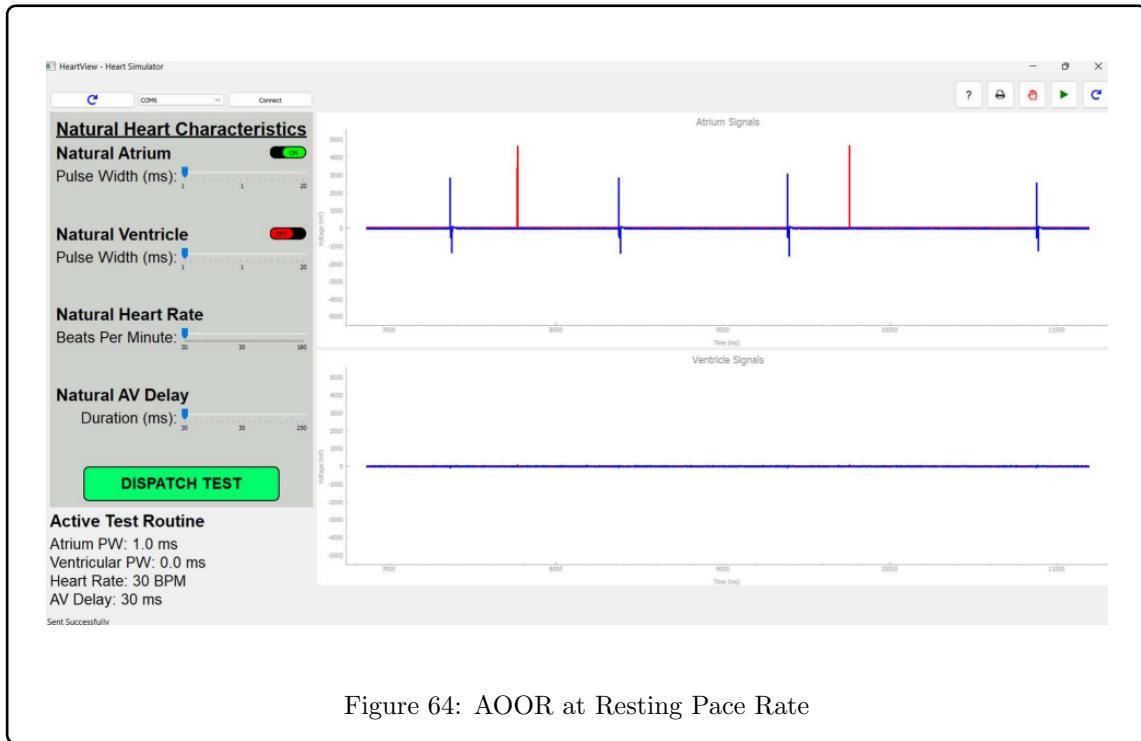


Figure 63: AOOR at Rest Ignoring Pulses



9.2.2 Test 2: AOOR with Increased Movement

Purpose: The purpose of this test is to test basic AOOR functionality while the patient has increased movement.

Input Conditions: General Inputs mode = 4 (AOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook at a medium rate for 5 seconds.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats, at a faster rate than resting in the atrium signal.

Actual Output: Output tested exactly as expected.

Result: Pass

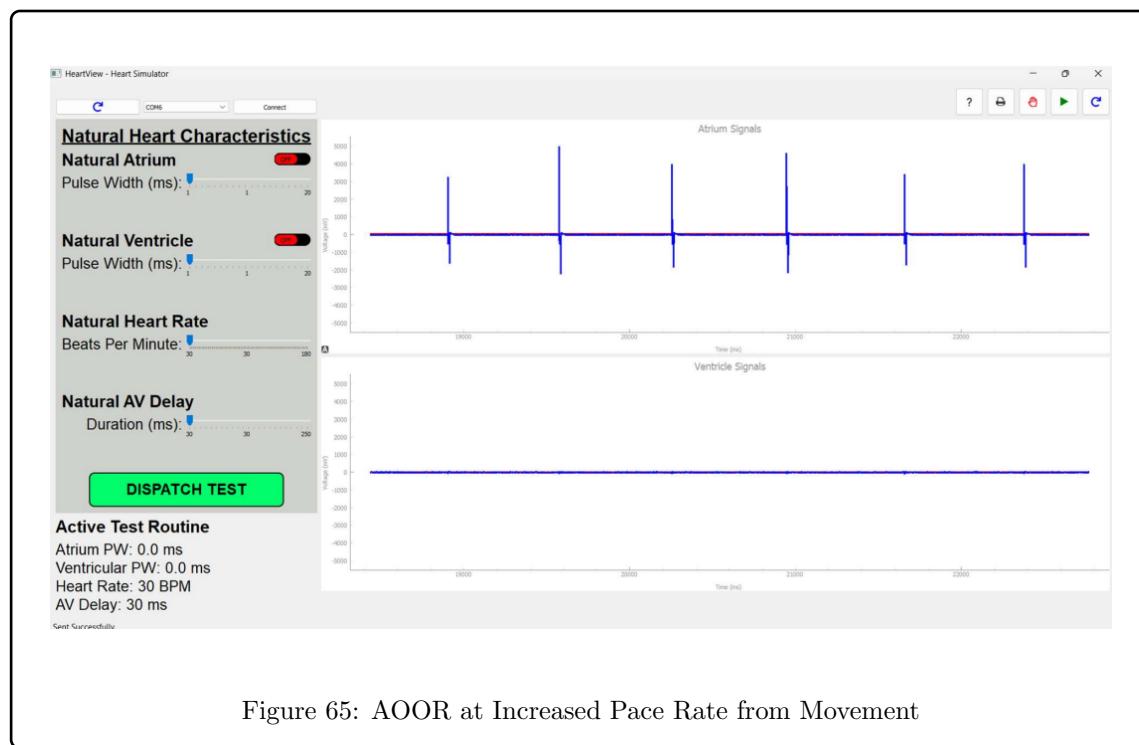


Figure 65: AOOR at Increased Pace Rate from Movement

9.2.3 Test 3: AOOR Maximum Sensor Rate

Purpose: The purpose of this test is to test basic AOOR functionality while the patient has increased movement for a prolonged period of time.

Input Conditions: General Inputs mode = 4 (AOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook vigorously for 30 seconds.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats, at the capped maximum sensor rate in the atrium signal.

Actual Output: Output tested exactly as expected.

Result: Pass

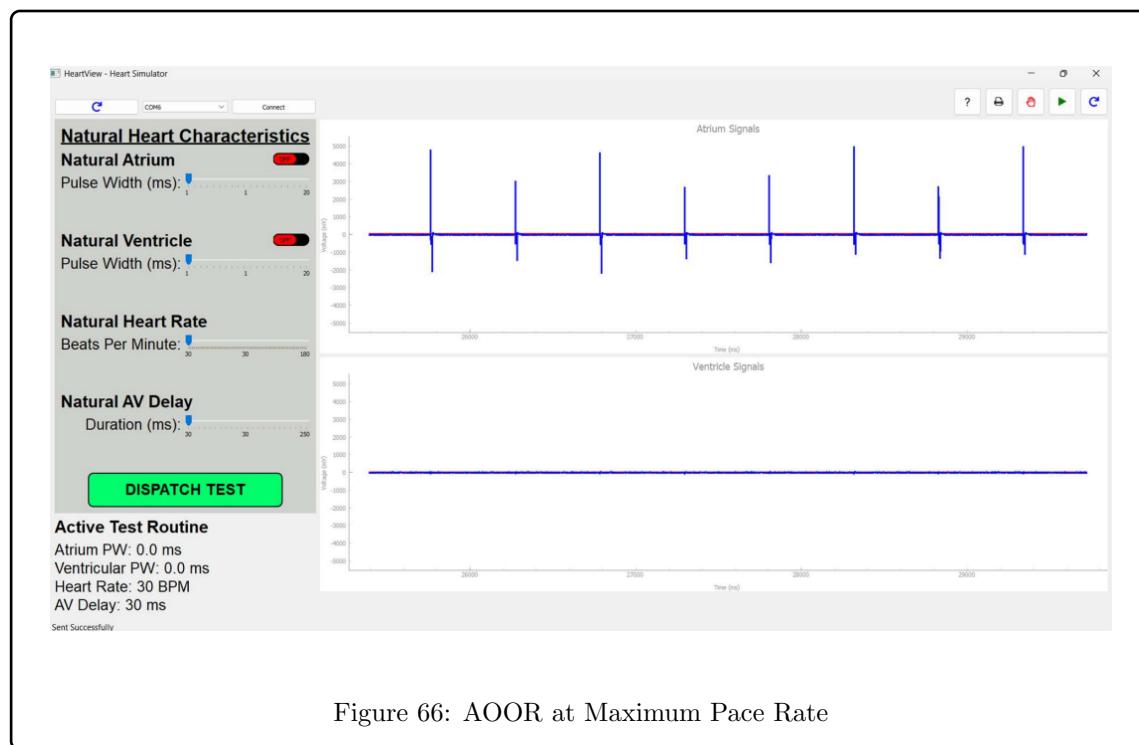


Figure 66: AOOR at Maximum Pace Rate

9.3 Testing of VOOR

9.3.1 Test 1: VOOR at Rest (No Movement)

Purpose: The purpose of this test is to test basic VOOR functionality.

Input Conditions: General Inputs mode = 5 (VOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats in the ventricular signal.

Actual Output: Output tested exactly as expected.

Result: Pass

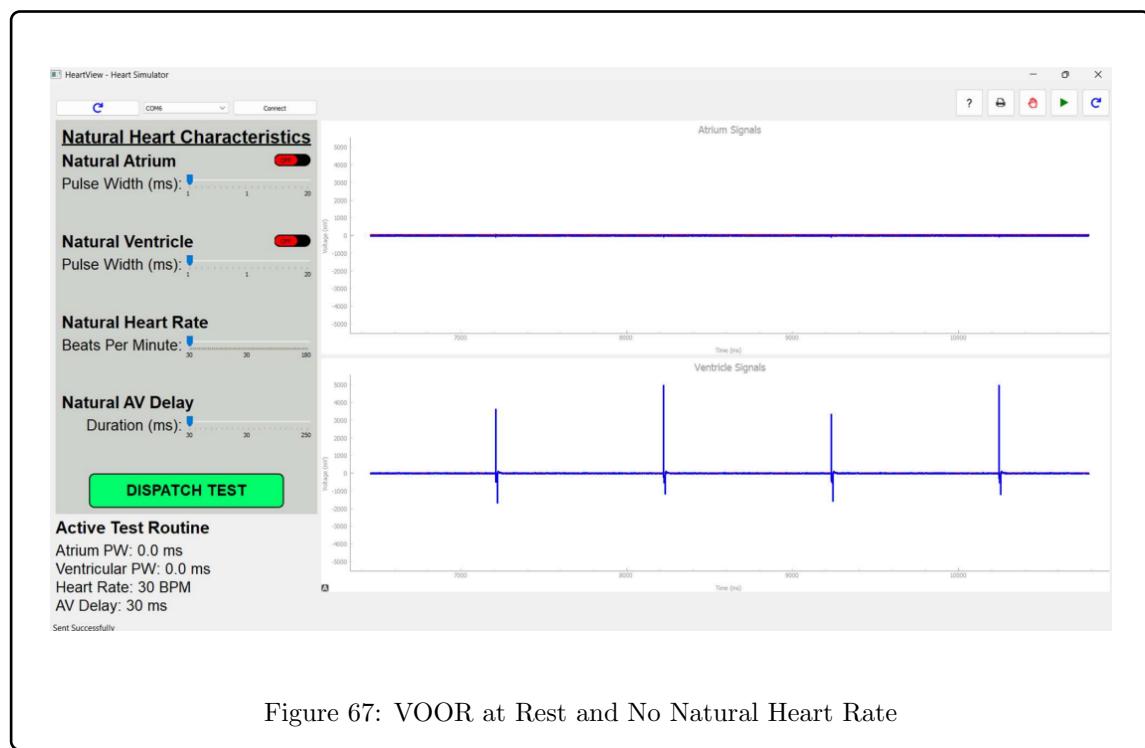


Figure 67: VOOR at Rest and No Natural Heart Rate

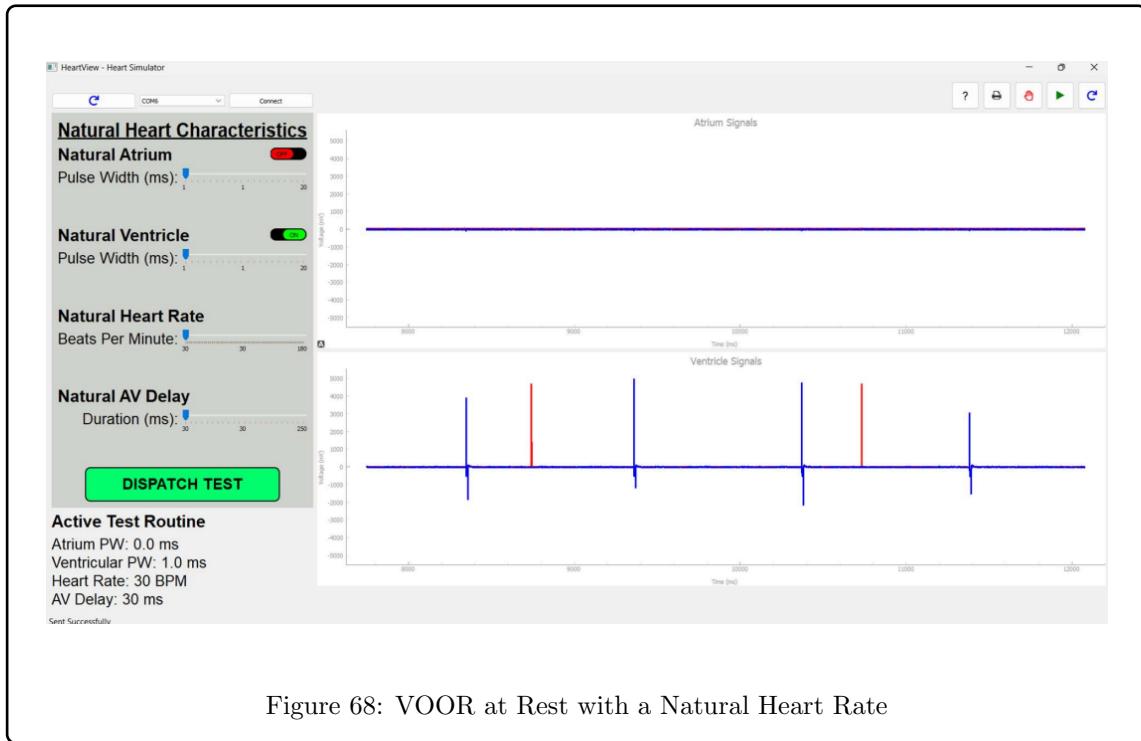


Figure 68: VOOR at Rest with a Natural Heart Rate

9.3.2 Test 2: VOOR with Increased Movement

Purpose: The purpose of this test is to test basic VOOR functionality while the patient has increased movement.

Input Conditions: General Inputs mode = 5 (VOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook at a medium rate for 5 seconds.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats in the ventricular signal, at a higher pace than resting.

Actual Output: Output tested exactly as expected.

Result: Pass

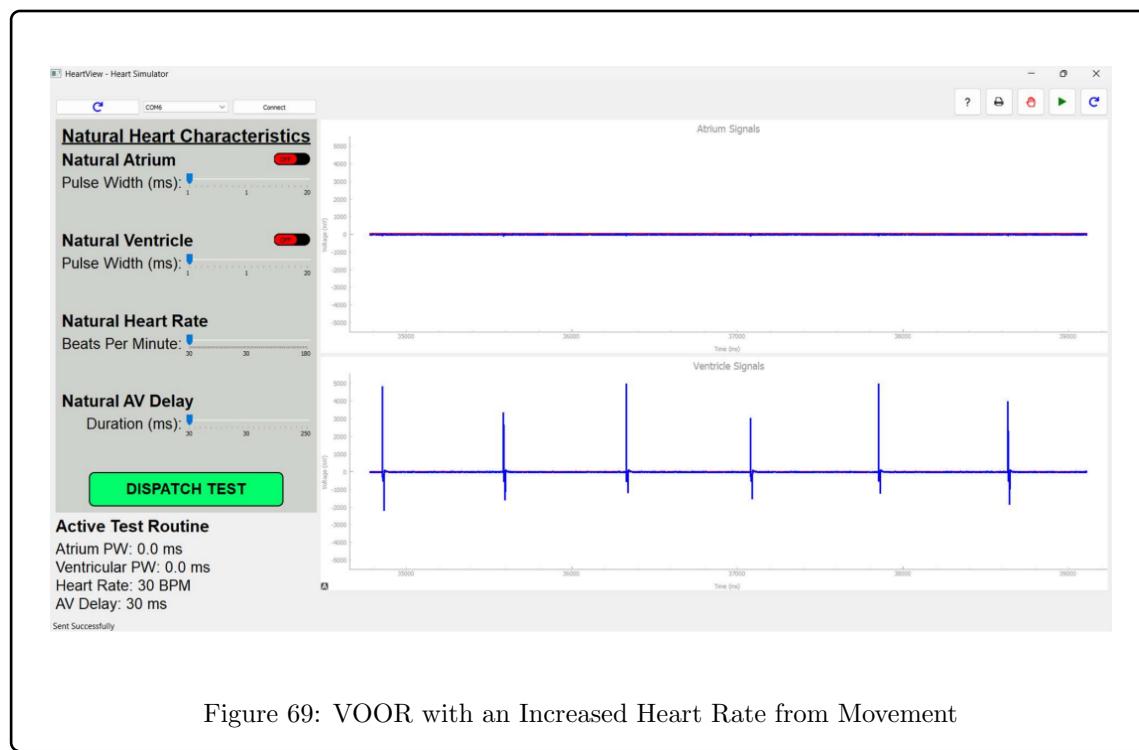


Figure 69: VOOR with an Increased Heart Rate from Movement

9.3.3 Test 3: VOOR Maximum Sensor Rate

Purpose: The purpose of this test is to test basic VOOR functionality while the patient has increased movement for prolonged period of time.

Input Conditions: General Inputs mode = 5 (VOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook vigorously for 30 seconds.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats, at the capped maximum sensor rate in the ventricular signal.

Actual Output: Output tested exactly as expected.

Result: Pass

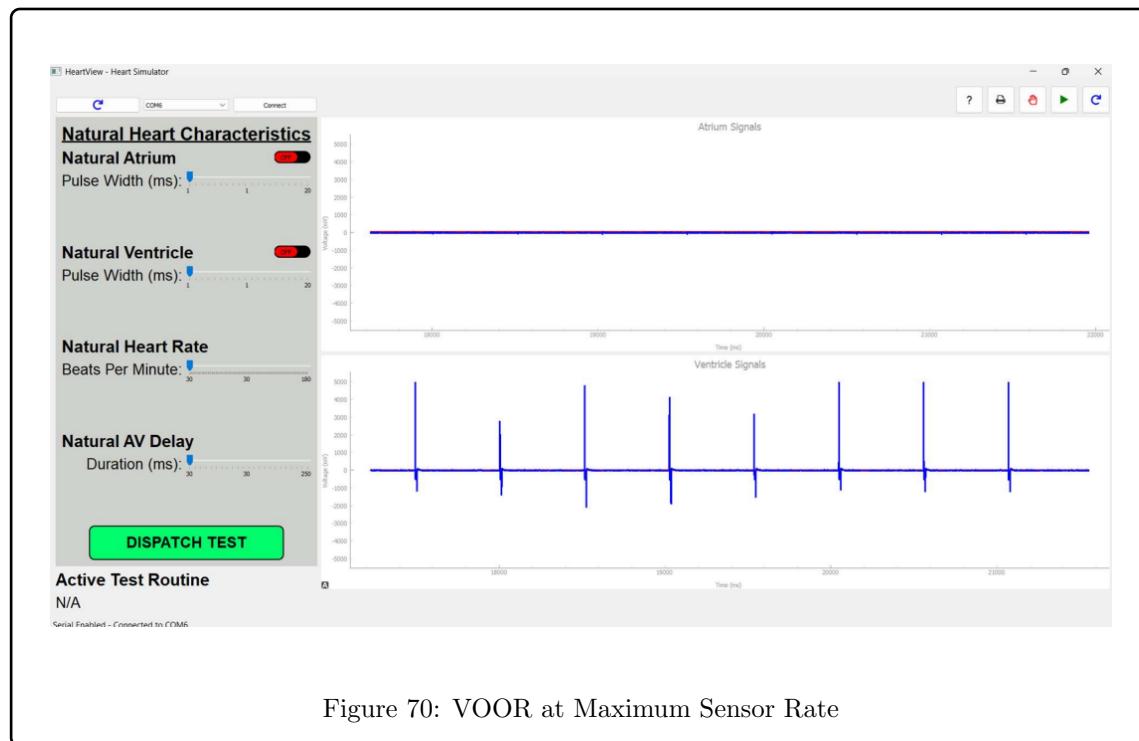


Figure 70: VOOR at Maximum Sensor Rate

9.4 Testing of AAIR

9.4.1 Test 1: AAIR at Rest (0 BPM)

Purpose: The purpose of this test is to test basic AAIR functionality at rest with no natural heart beat of 0 bpm.

Input Conditions: General Inputs mode = 6 (AAIR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Natural heart rate was set to 0 bpm.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats in the atrium signal.

Actual Output: Output tested exactly as expected.

Result: Pass

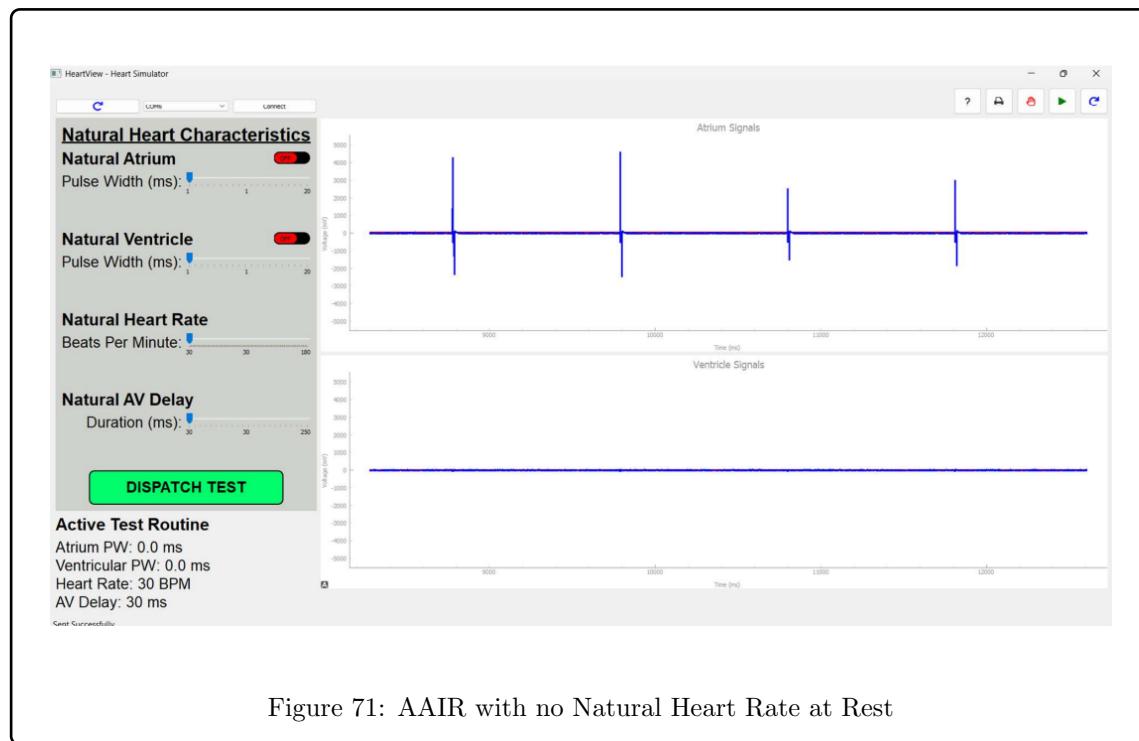


Figure 71: AAIR with no Natural Heart Rate at Rest

9.4.2 Test 2: AAIR at Rest (45 BPM)

Purpose: The purpose of this test is to test basic AAIR functionality at rest with natural heart beat at 45 bpm.

Input Conditions: General Inputs mode = 6 (AAIR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Natural heart rate set to 45 bpm.

Expected Output: The pacemaker should pulse after the refractory period expires. An output of blue pulses right before the natural red pulses is expected.

Actual Output: Output tested exactly as expected.

Result: Pass

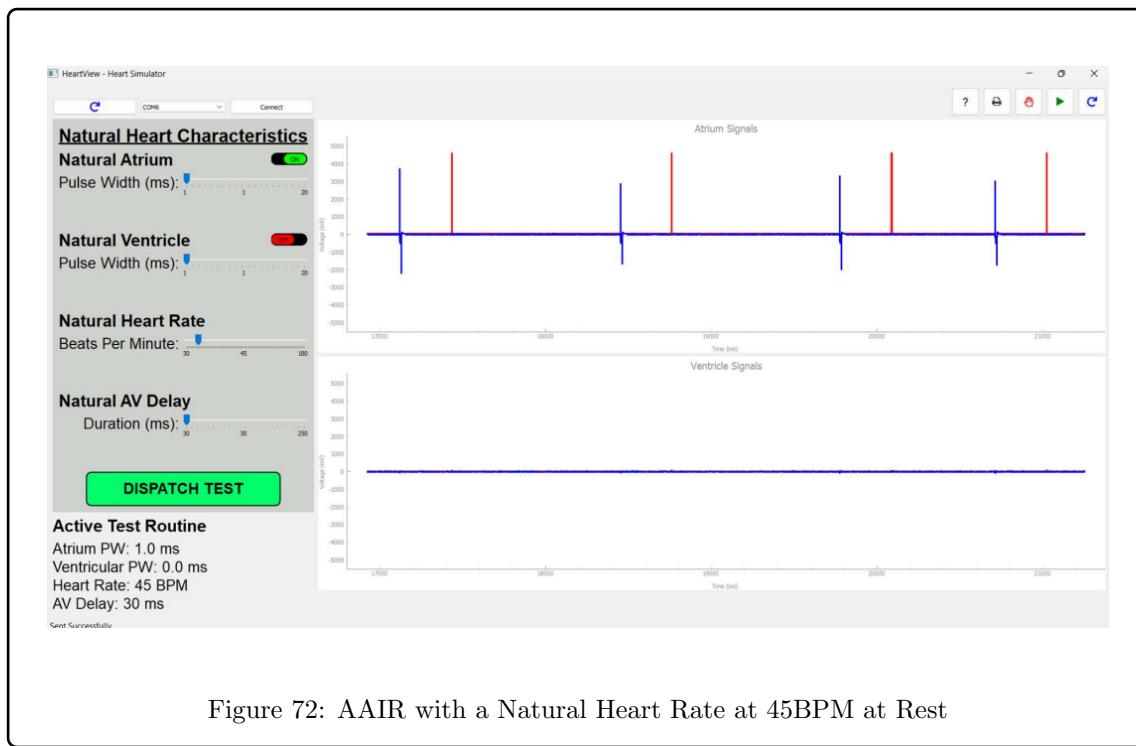


Figure 72: AAIR with a Natural Heart Rate at 45BPM at Rest

9.4.3 Test 3: AAIR at Rest (75 BPM)

Purpose: The purpose of this test is to test basic AAIR functionality at rest with natural heartbeat of 75 bpm.

Input Conditions: General Inputs mode = 6 (AAIR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Natural heart rate was set to 75 BPM.

Expected Output: As a nominal heart rate is being inputted, the pacemaker should not be delivering pacing pulses as the simulated heart rate is nominal and healthy.

Actual Output: Output tested exactly as expected.

Result: Pass

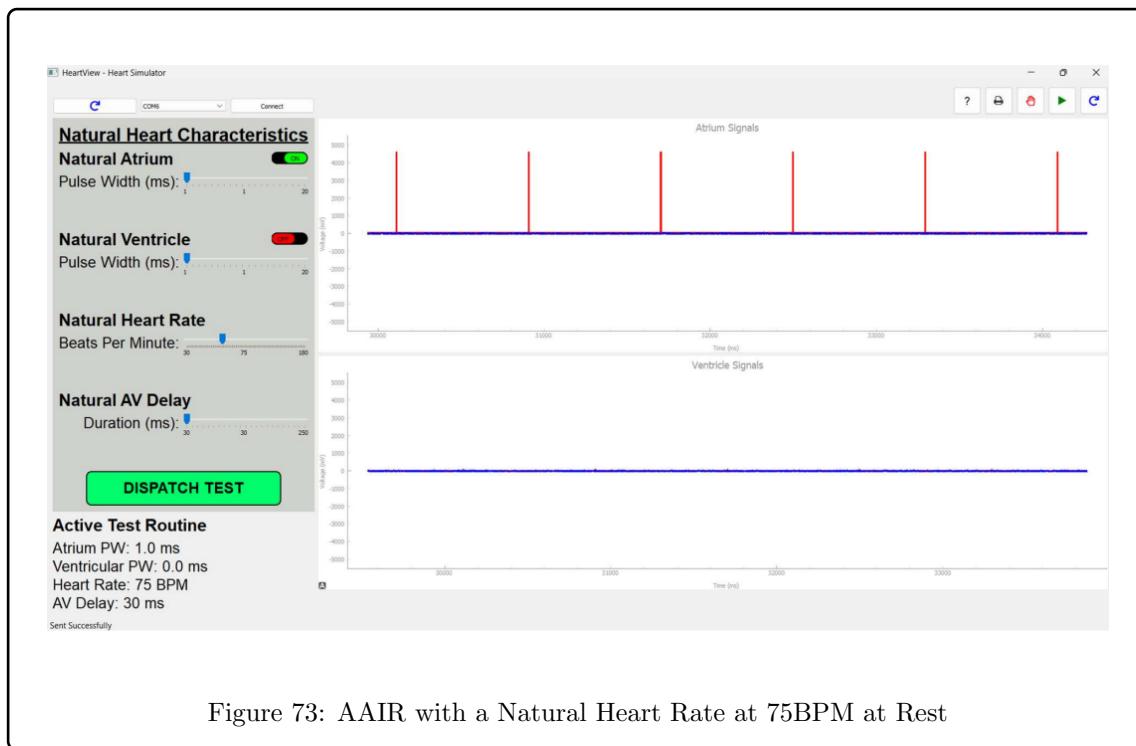


Figure 73: AAIR with a Natural Heart Rate at 75BPM at Rest

9.4.4 Test 4: AAIR with Increased Movement (45 BPM)

Purpose: The purpose of this test is to test basic AAIR functionality at increased movement with natural heartbeat of 45 bpm.

Input Conditions: General Inputs mode = 6 (AAIR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Natural heart rate was set to 45 BPM. Also the pacemaker was shaken at a medium rate for 5 seconds.

Expected Output: Since the activity level increases, the blue pulse paces need to occur faster to keep up with the increased movement. The blue pulses should have an increased bpm, while still reacting to the natural heart, pulsing when refractory period expires, in the atrium.

Actual Output: Output tested exactly as expected.

Result: Pass

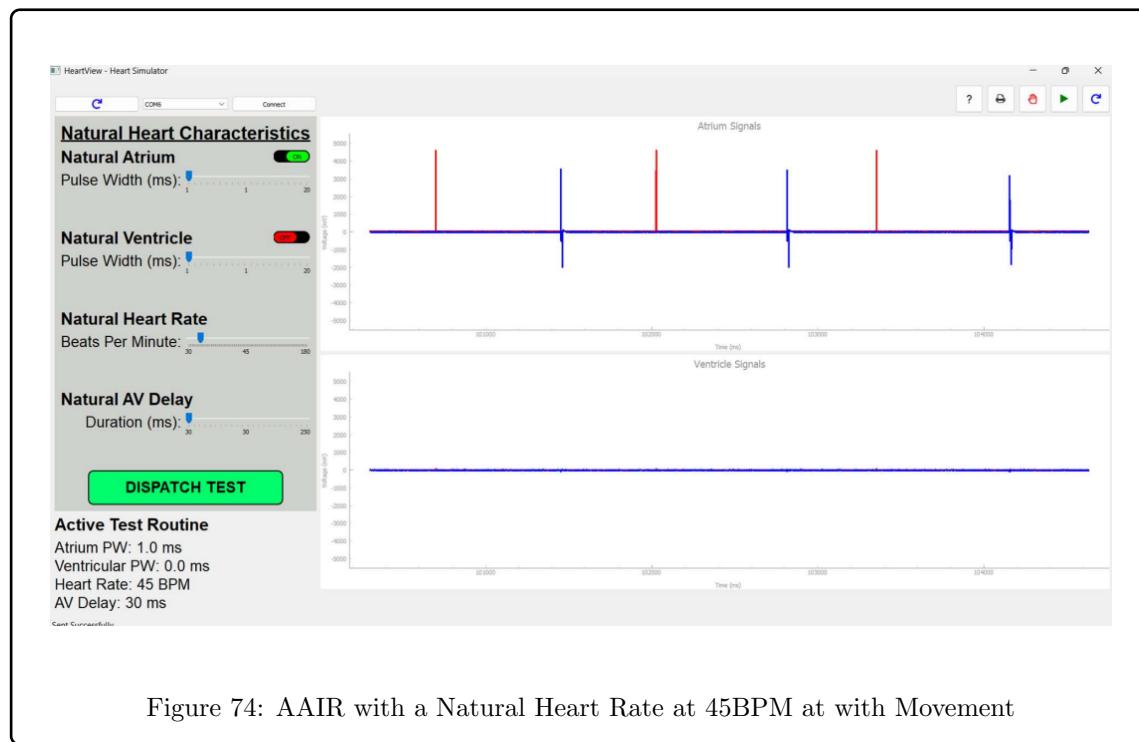


Figure 74: AAIR with a Natural Heart Rate at 45BPM at with Movement

9.4.5 Test 5: AAIR Maximum Sensor Rate (45 BPM)

Purpose: The purpose of this test is to test basic AAIR functionality at increased movement over a prolonged period with natural heartbeat of 45 bpm.

Input Conditions: General Inputs mode = 6 (AAIR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Natural heart rate was set to 45 BPM. Also the pacemaker was shook vigorously for 30 seconds.

Expected Output: The pacemaker should be pacing at the max sensor rate, of 120bpm. An output of 2 blue pulses right before the natural red pulses is expected, since the bpm is double that of resting, in the atrium.

Actual Output: Output tested exactly as expected.

Result: Pass

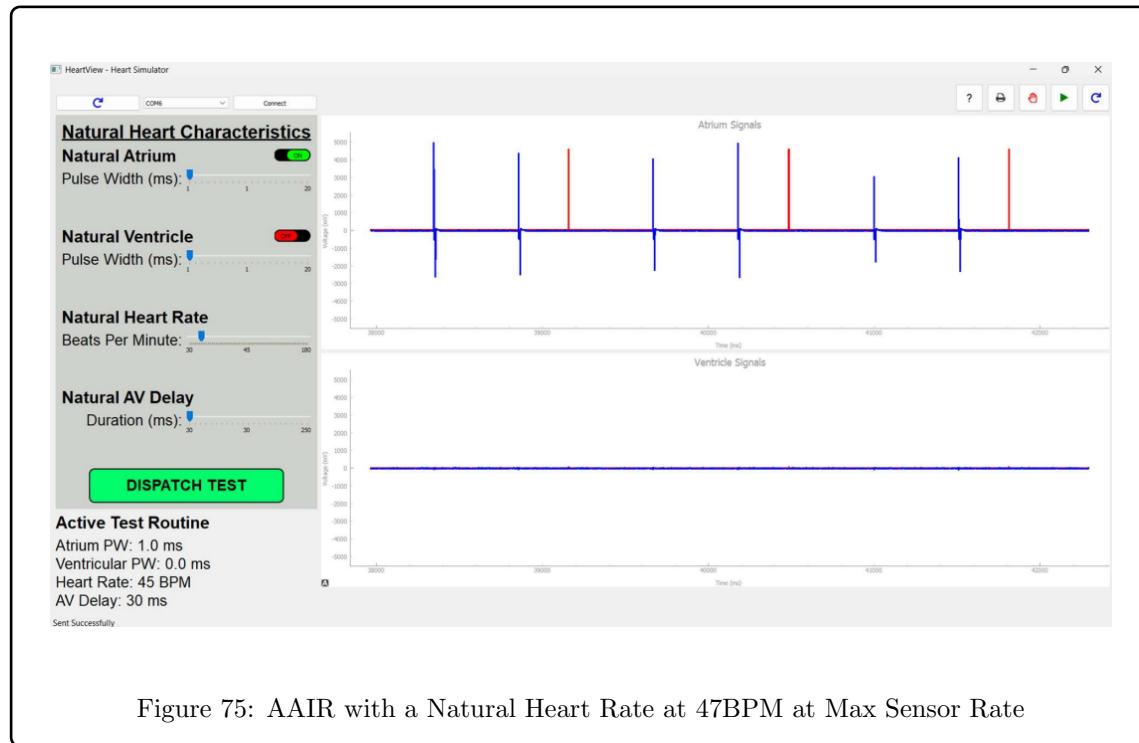


Figure 75: AAIR with a Natural Heart Rate at 47BPM at Max Sensor Rate

9.4.6 Test 6: AAIR Maximum Sensor Rate (75 BPM)

Purpose: The purpose of this test is to test basic AAIR functionality at increased movement over a prolonged period with natural heartbeat of 75 bpm.

Input Conditions: General Inputs mode = 6 (AAIR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Natural heart rate was set to 75 BPM. Also the pacemaker was shook vigorously for 30 seconds.

Expected Output: The pacemaker should pulse after the refractory period expires. An output of blue pulses right before the natural red pulses is expected in the atrium.

Actual Output: Output tested exactly as expected.

Result: Pass

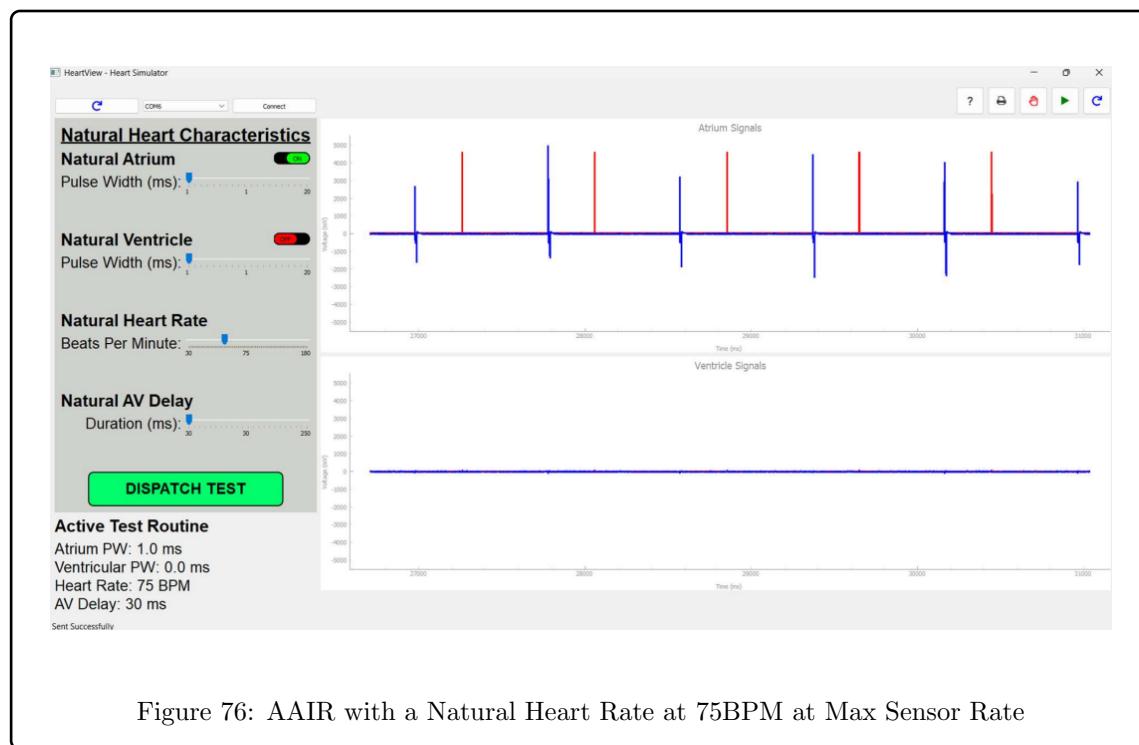


Figure 76: AAIR with a Natural Heart Rate at 75BPM at Max Sensor Rate

9.5 Testing of VVIR

9.5.1 Test 1: VVIR at Rest (0 BPM)

Purpose: The purpose of this test is to test basic VVIR functionality at rest with no natural heartbeat of 0 bpm.

Input Conditions: General Inputs mode = 7 (VVIR), hysteresis = 0 (off), standard ventricular inputs as well as standard rate adaptive inputs. Natural heart rate was set to 0 bpm.

Expected Output: Consistent, evenly spaced pulses in the output with disregard for natural heartbeats in the ventricular signal.

Actual Output: Output tested exactly as expected.

Result: Pass

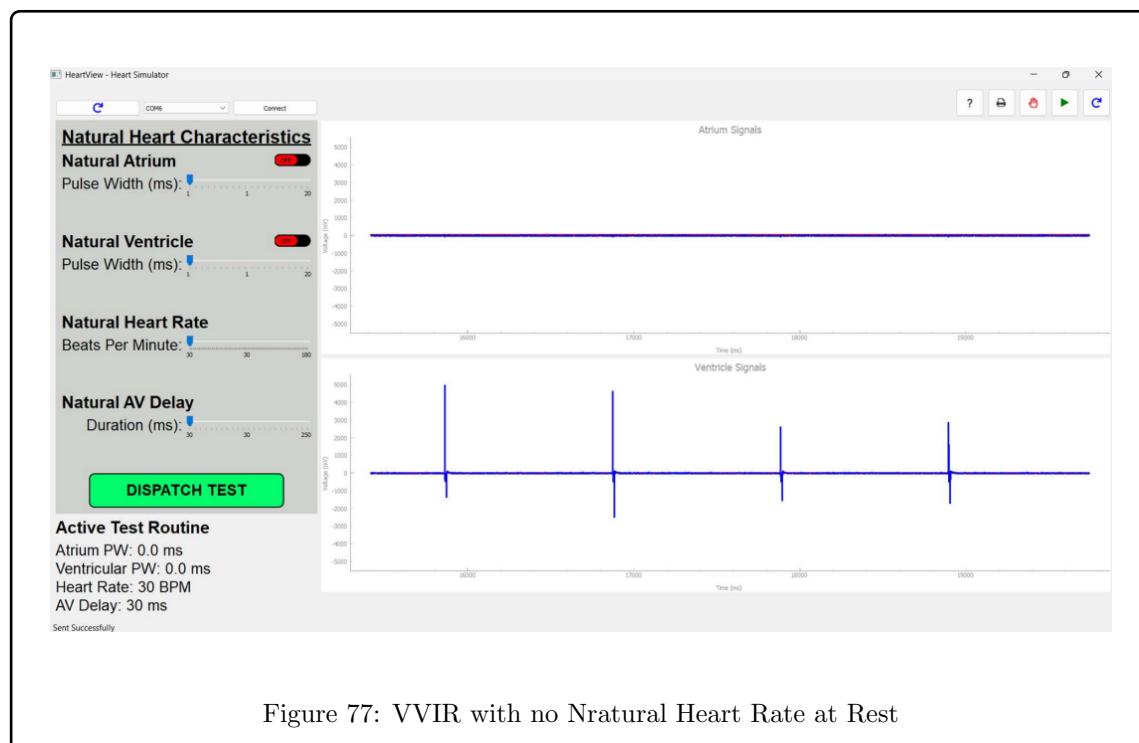


Figure 77: VVIR with no Natural Heart Rate at Rest

9.5.2 Test 2: VVIR at Rest (45 BPM)

Purpose: The purpose of this test is to test basic VVIR functionality at rest with natural heart beat at 45 bpm.

Input Conditions: General Inputs mode = 7 (VVIR), hysteresis = 0 (off), standard ventricular inputs as well as standard rate adaptive inputs. Natural heart rate set to 45 bpm.

Expected Output: The pacemaker should pulse after the refractory period expires. An output of blue pulses right before the natural red pulses is expected in the ventricular.

Actual Output: Output tested exactly as expected.

Result: Pass

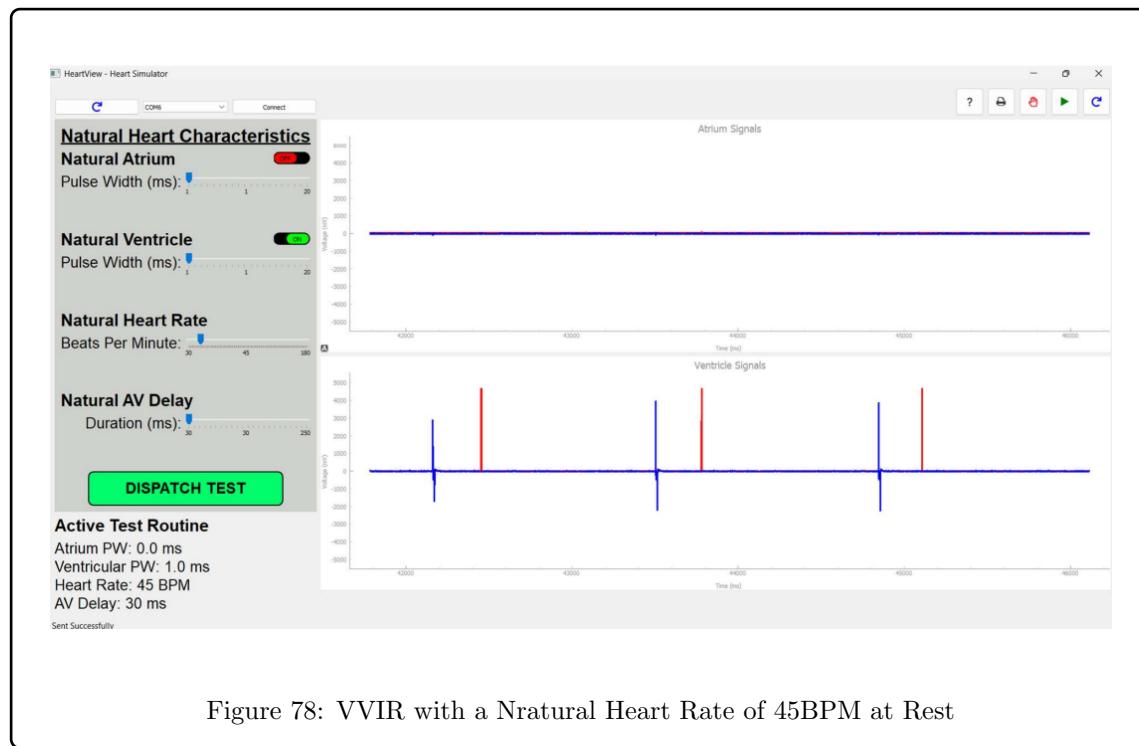


Figure 78: VVIR with a Natural Heart Rate of 45BPM at Rest

9.5.3 Test 3: VVIR at Rest (75 BPM)

Purpose: The purpose of this test is to test basic VVIR functionality at rest with natural heartbeat of 75 bpm.

Input Conditions: General Inputs mode = 7 (VVIR), hysteresis = 0 (off), standard ventricular inputs as well as standard rate adaptive inputs. Natural heart rate was set to 75 BPM.

Expected Output: As a nominal heart rate is being inputted, the pacemaker should not be delivering pacing pulses as the simulated heart rate is nominal and healthy.

Actual Output: Output tested exactly as expected.

Result: Pass

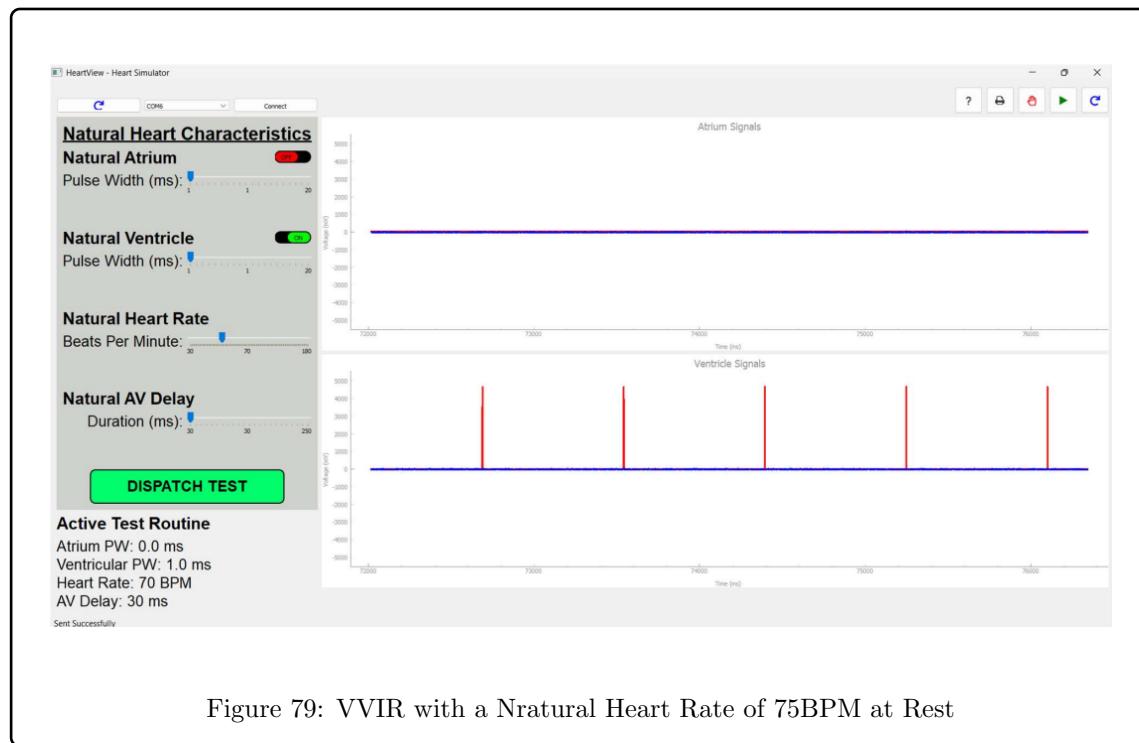


Figure 79: VVIR with a Natural Heart Rate of 75BPM at Rest

9.5.4 Test 4: VVIR with Increased Movement (45 BPM)

Purpose: The purpose of this test is to test basic VVIR functionality at increased movement with natural heartbeat of 45 bpm.

Input Conditions: General Inputs mode = 7 (VVIR), hysteresis = 0 (off), standard ventricular inputs as well as standard rate adaptive inputs. Natural heart rate was set to 45 BPM. Also the pacemaker was shaken at a medium rate for 5 seconds.

Expected Output: Since the activity level increases, the blue pulse paces need to occur faster to keep up with the increased movement. The blue pulses should have an increased bpm, while still reacting to the natural heart, pulsing when refractory period expires, in the ventricular.

Actual Output: Output tested exactly as expected.

Result: Pass

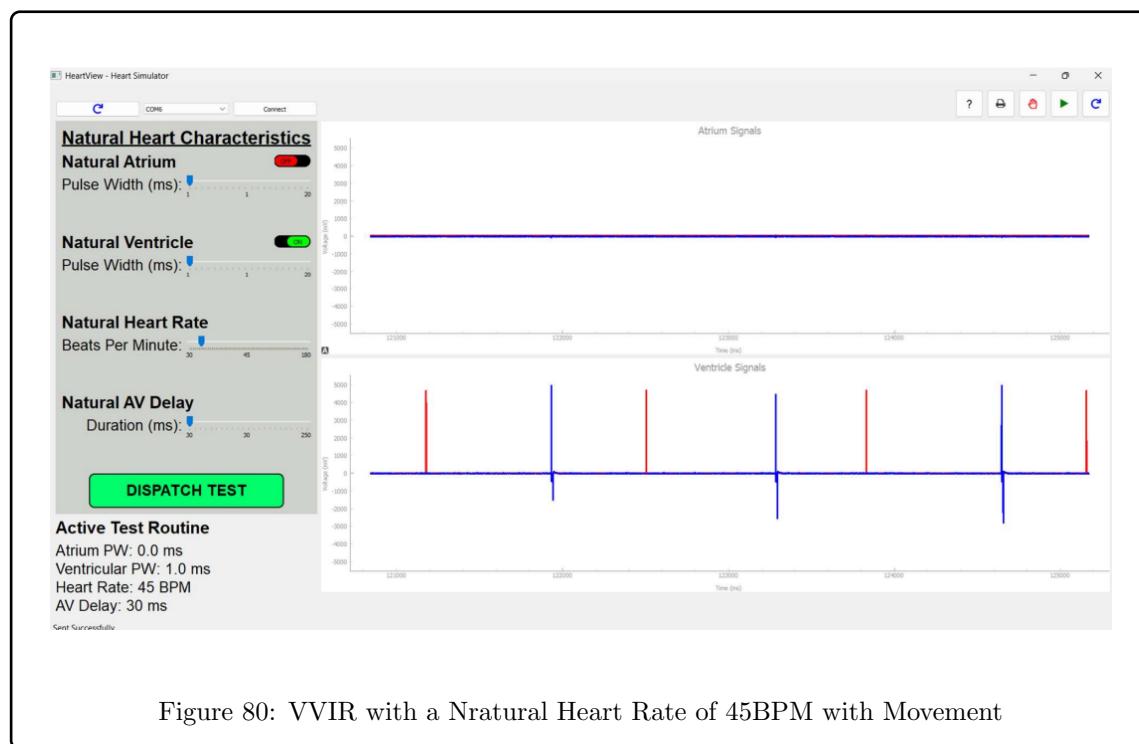


Figure 80: VVIR with a Natural Heart Rate of 45BPM with Movement

9.5.5 Test 5: VVIR Maximum Sensor Rate (45 BPM)

Purpose: The purpose of this test is to test basic VVIR functionality at increased movement over a prolonged period with natural heartbeat of 45 bpm.

Input Conditions: General Inputs mode = 7 (VVIR), hysteresis = 0 (off), standard ventricular inputs as well as standard rate adaptive inputs. Natural heart rate was set to 45 BPM. Also the pacemaker was shook vigorously for 30 seconds.

Expected Output: The pacemaker should be pacing at the max sensor rate, of 120bpm. An output of 2 blue pulses right before the natural red pulses is expected, since the bpm is double that of resting, in the ventricular.

Actual Output: Output tested exactly as expected.

Result: Pass

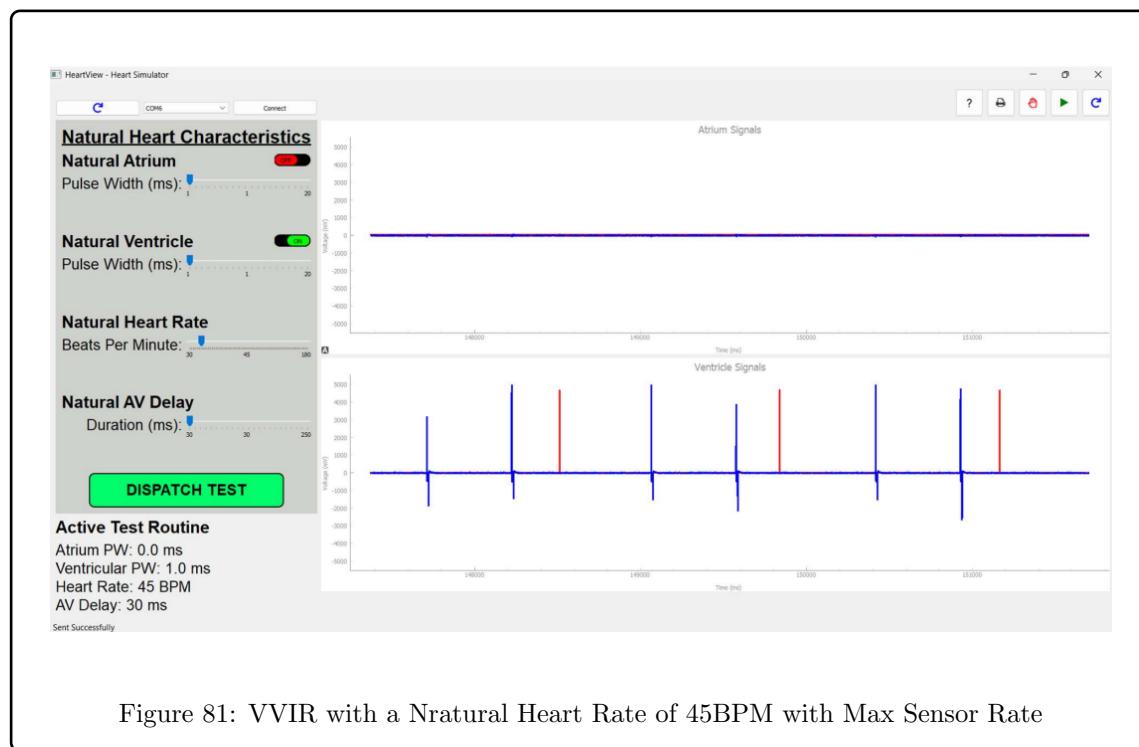


Figure 81: VVIR with a Natural Heart Rate of 45BPM with Max Sensor Rate

9.5.6 Test 6: VVIR Maximum Sensor Rate (75 BPM)

Purpose: The purpose of this test is to test basic VVIR functionality at increased movement over a prolonged period with natural heartbeat of 75 bpm.

Input Conditions: General Inputs mode = 7 (VVIR), hysteresis = 0 (off), standard ventricular inputs as well as standard rate adaptive inputs. Natural heart rate was set to 75 BPM. Also the pacemaker was shook vigorously for 30 seconds.

Expected Output: The pacemaker should pulse after the refractory period expires. An output of blue pulses right before the natural red pulses is expected in the ventricular.

Actual Output: Since the refractory period is longer for ventricular by default (320 instead of 250) the refractory period expires right before the natural pulse on occasion, which is why we see the pattern. This is expected.

Result: Pass

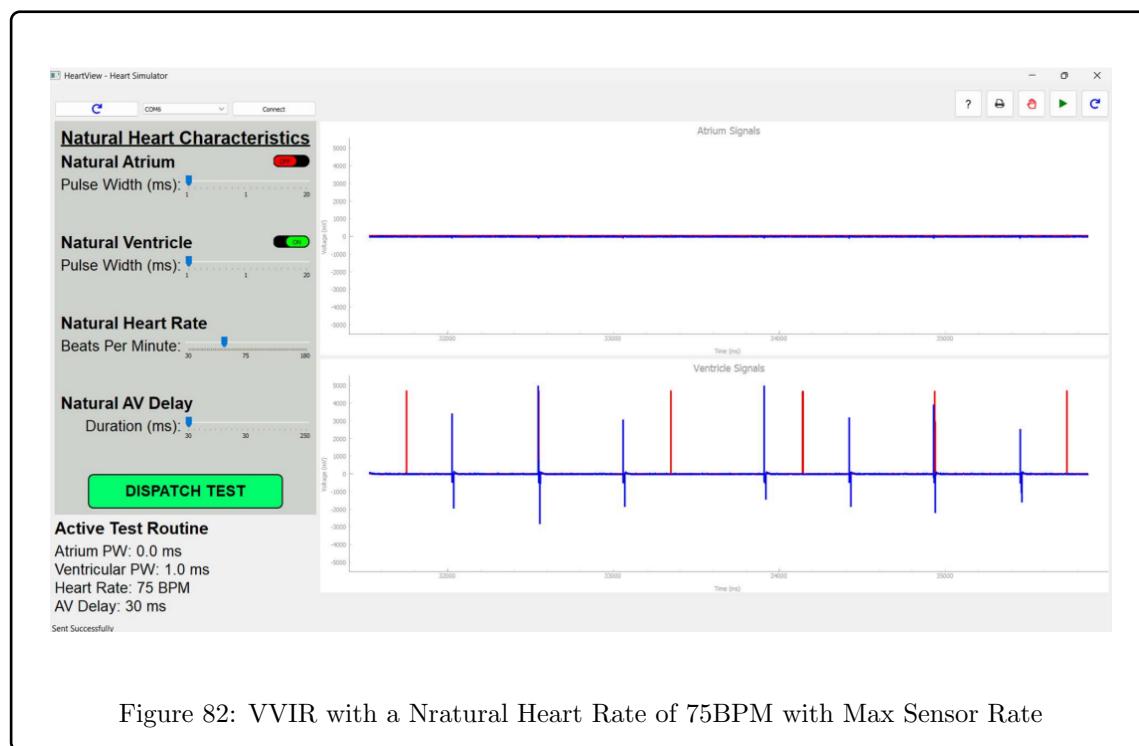


Figure 82: VVIR with a Natural Heart Rate of 75BPM with Max Sensor Rate

9.6 Rate Adaptive Tuning Testing

9.6.1 Test 1: Recovery Time Testing (30 seconds)

Purpose: The purpose of this test is to test basic AOOR functionality for the recovery time tuning.

Input Conditions: General Inputs mode = 4 (AOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook at a medium rate for 30 seconds. Recovery time was set to 0.5 (translates to 30 seconds).

Expected Output: After reaching the upper rate limit, it should take 30 seconds for the pacemaker to return to the lower rate limit.

Actual Output: Output tested exactly as expected.

Result: Pass

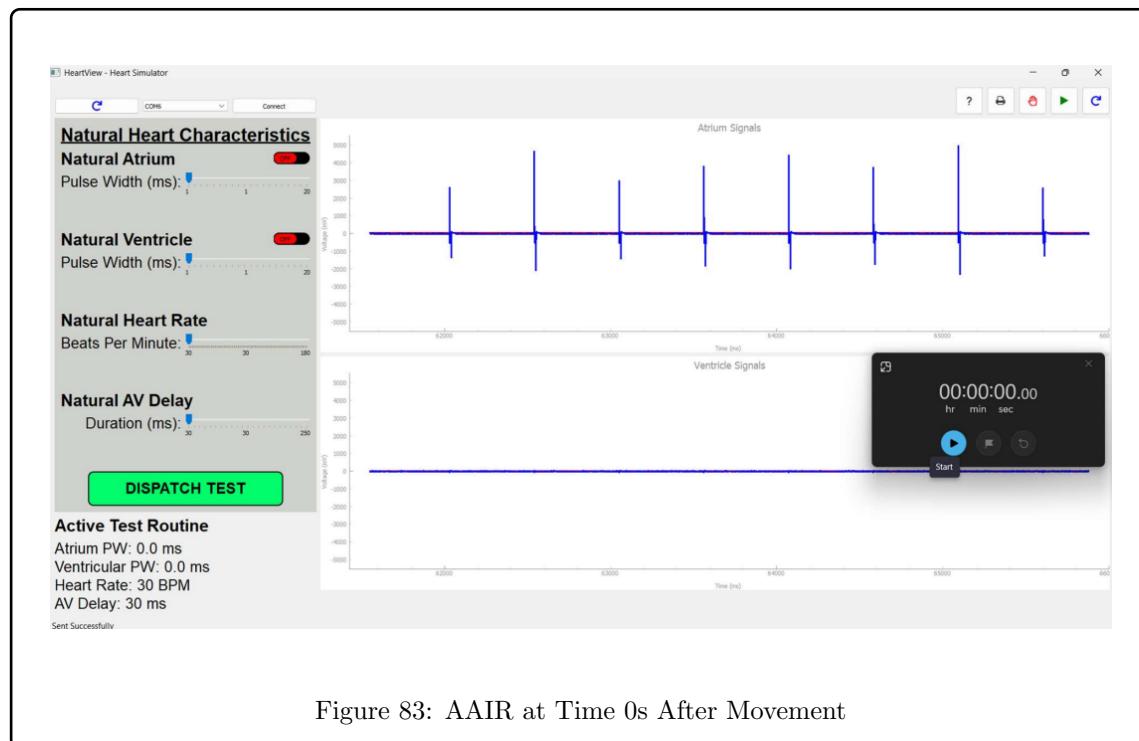


Figure 83: AAIR at Time 0s After Movement

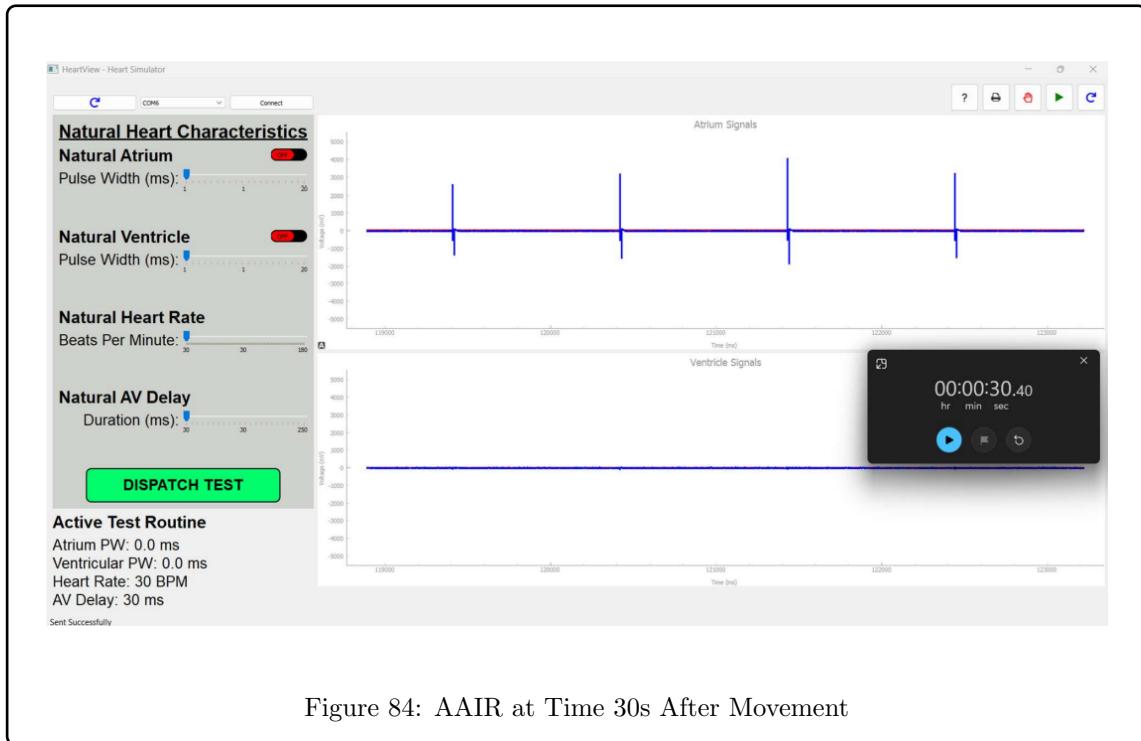


Figure 84: AAIR at Time 30s After Movement

9.6.2 Test 2: Recovery Time Testing (60 seconds)

Purpose: The purpose of this test is to test basic AOOR functionality for the recovery time tuning.

Input Conditions: General Inputs mode = 4 (AOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook at a medium rate for 30 seconds. Recovery time was set to 1 (translates to 60 seconds).

Expected Output: After reaching the upper rate limit, it should take 60 seconds for the pacemaker to return to the lower rate limit.

Actual Output: Output tested exactly as expected.

Result: Pass

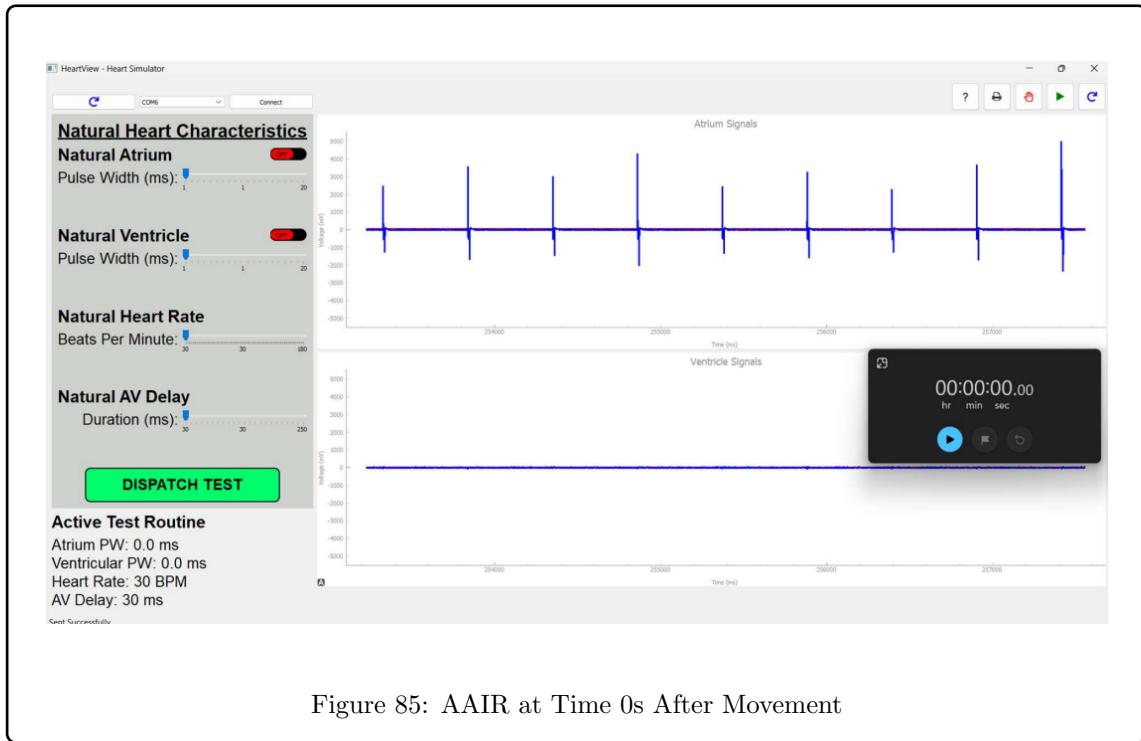


Figure 85: AAIR at Time 0s After Movement

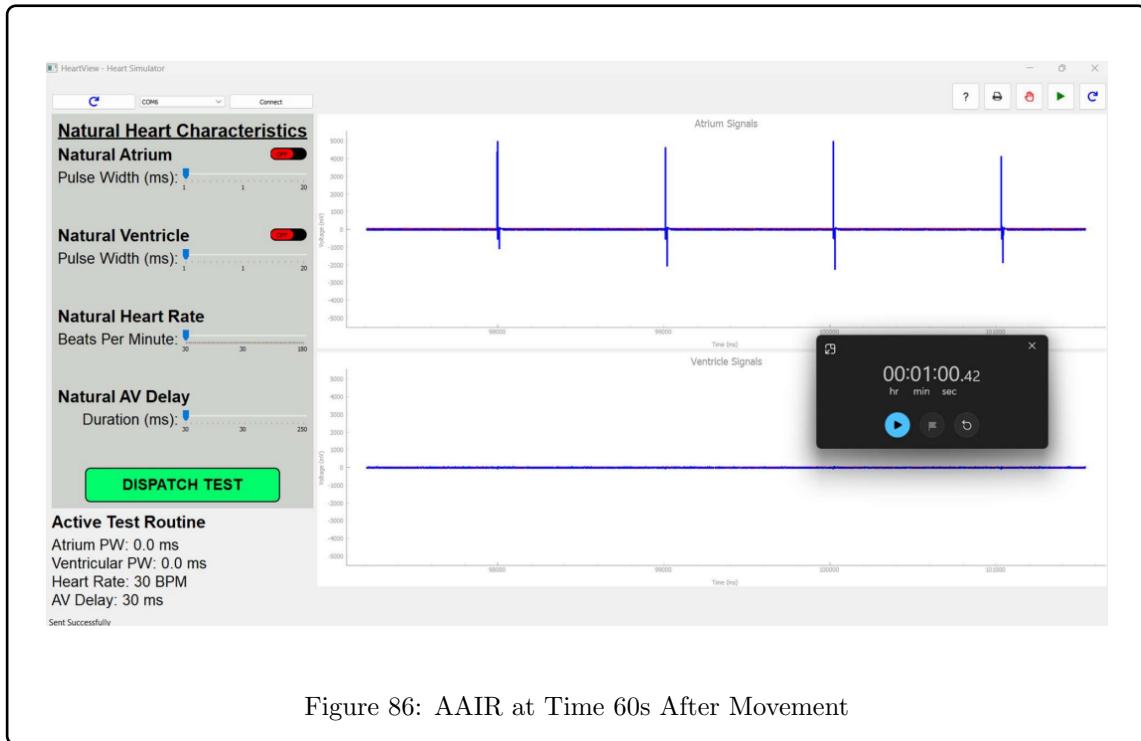


Figure 86: AAIR at Time 60s After Movement

9.6.3 Test 3: Reaction Time Testing (10 seconds)

Purpose: The purpose of this test is to test basic AOOR functionality for the reaction time tuning.

Input Conditions: General Inputs mode = 4 (AOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook at a medium rate until the maximum sensor rate. Recovery time was set to 10 seconds.

Expected Output: The pacemaker should reach the upper rate limit in 10 seconds.

Actual Output: Output tested exactly as expected.

Result: Pass

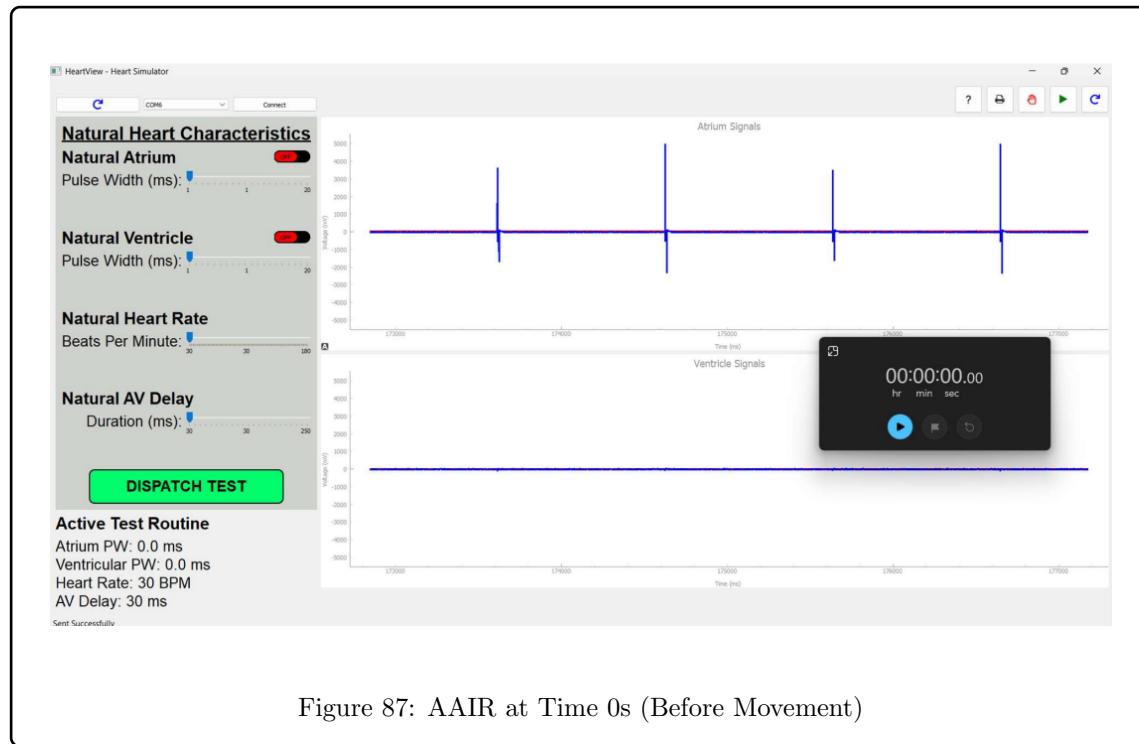


Figure 87: AAIR at Time 0s (Before Movement)

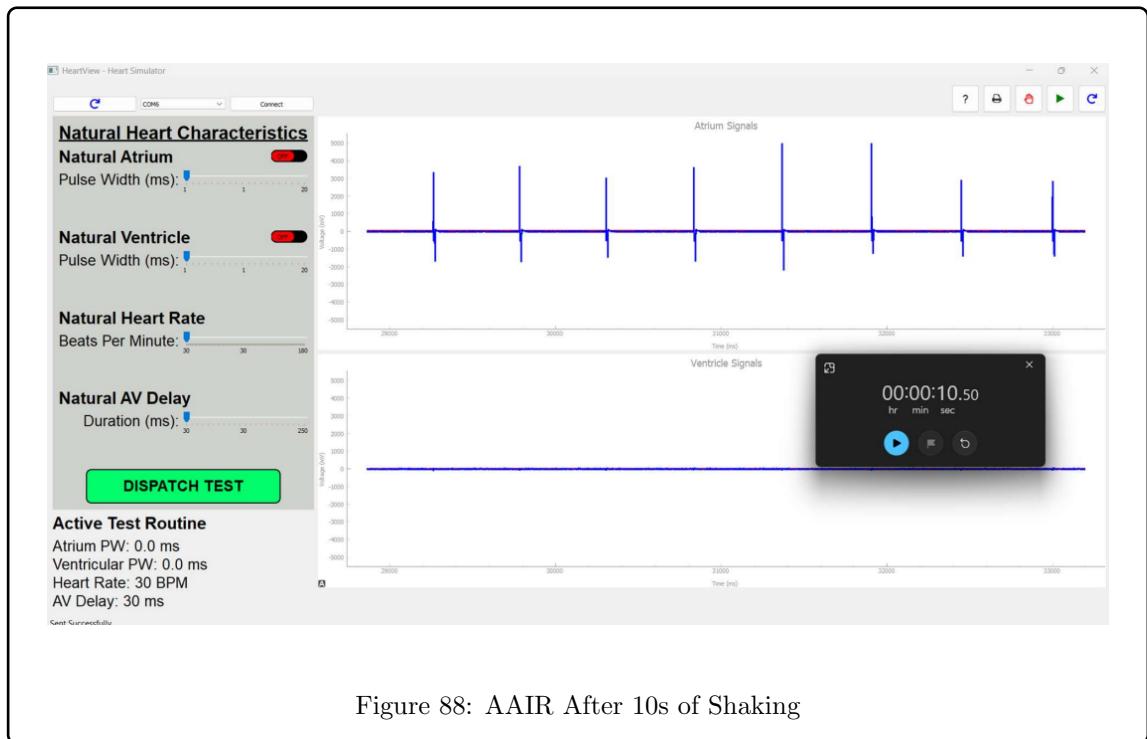


Figure 88: AAIR After 10s of Shaking

9.6.4 Test 4: Reaction Time Testing (30 seconds)

Purpose: The purpose of this test is to test basic AOOR functionality for the reaction time tuning.

Input Conditions: General Inputs mode = 4 (AOOR), hysteresis = 0 (off), standard atrial inputs as well as standard rate adaptive inputs. Also the pacemaker was shook at a medium rate until the maximum sensor rate. Recovery time was set to 30 seconds.

Expected Output: The pacemaker should reach the upper rate limit in 30 seconds.

Actual Output: Output tested exactly as expected.

Result: Pass

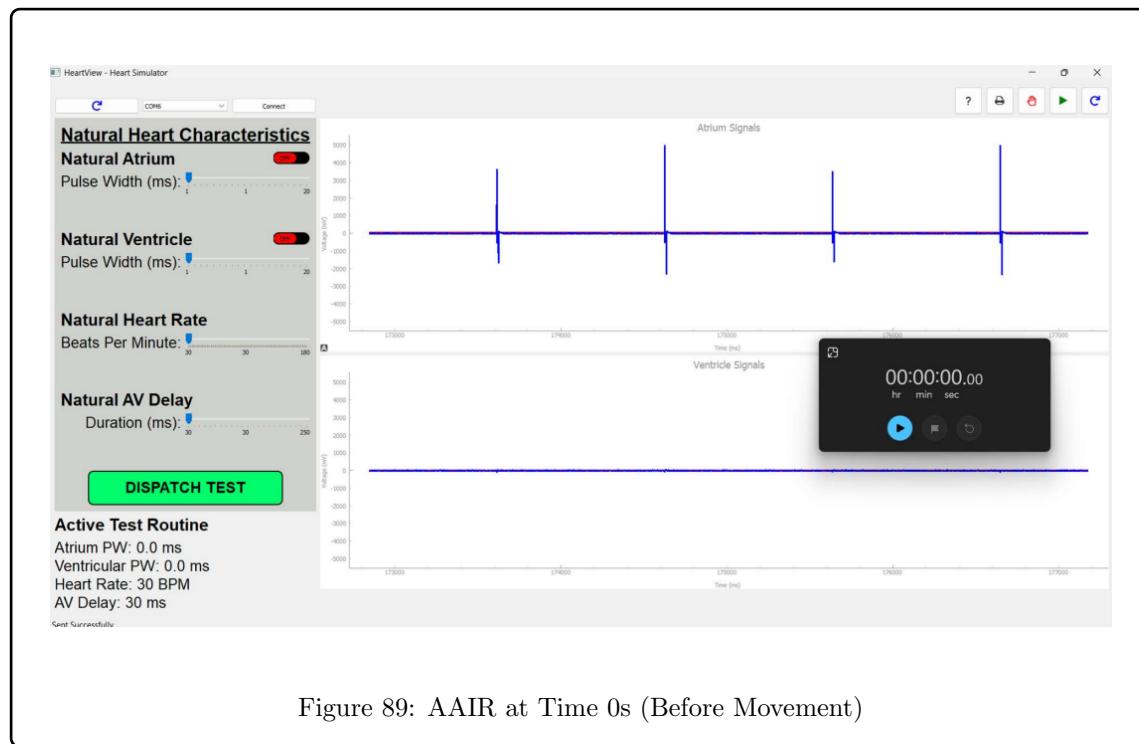


Figure 89: AAIR at Time 0s (Before Movement)

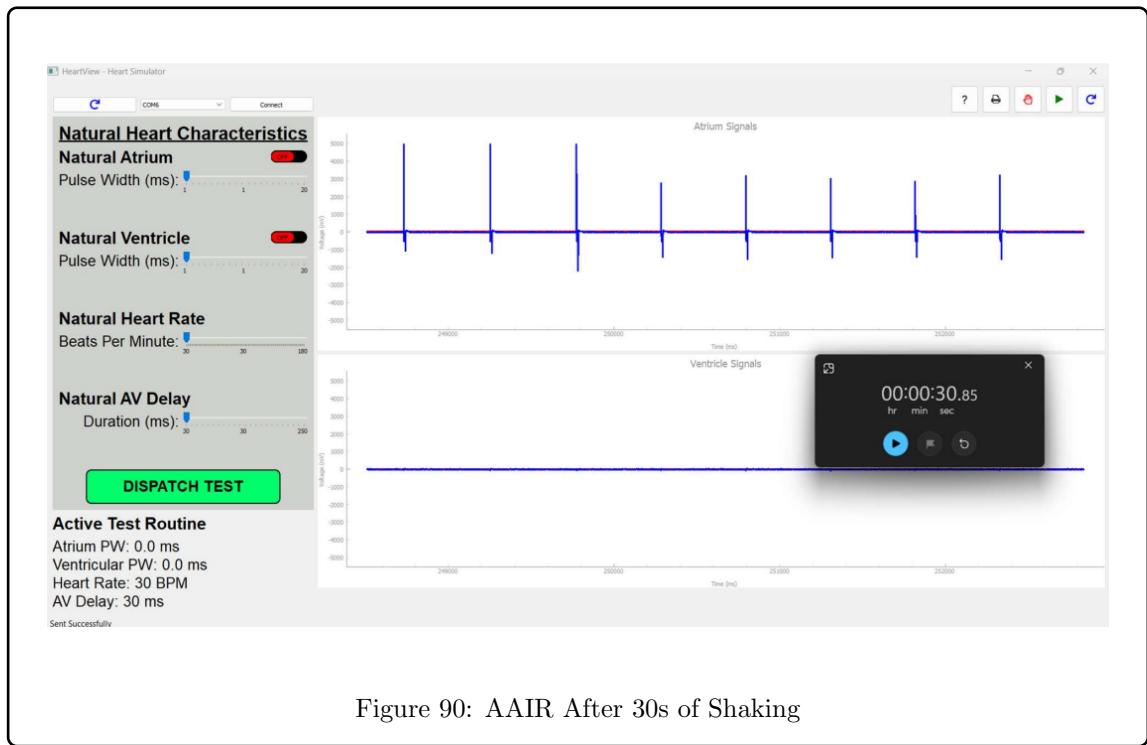


Figure 90: AAIR After 30s of Shaking

9.7 DCM Testing

9.7.1 Login and Registration

Purpose: The purpose of this test is to verify correct storage of newly registered user data and allow login.

Input Conditions: A random username and password. This will be used again in the login screen to access the DCM.

Expected Output: A window should pop up notifying the user an account has been registered. The DCM controls should be accessible after the user logs in.

Actual Output: Dialogue is shown and the file is updated to include new user data. The user is then brought to the

Result: Pass



Figure 91: Registration Test



Figure 92: Registration Result

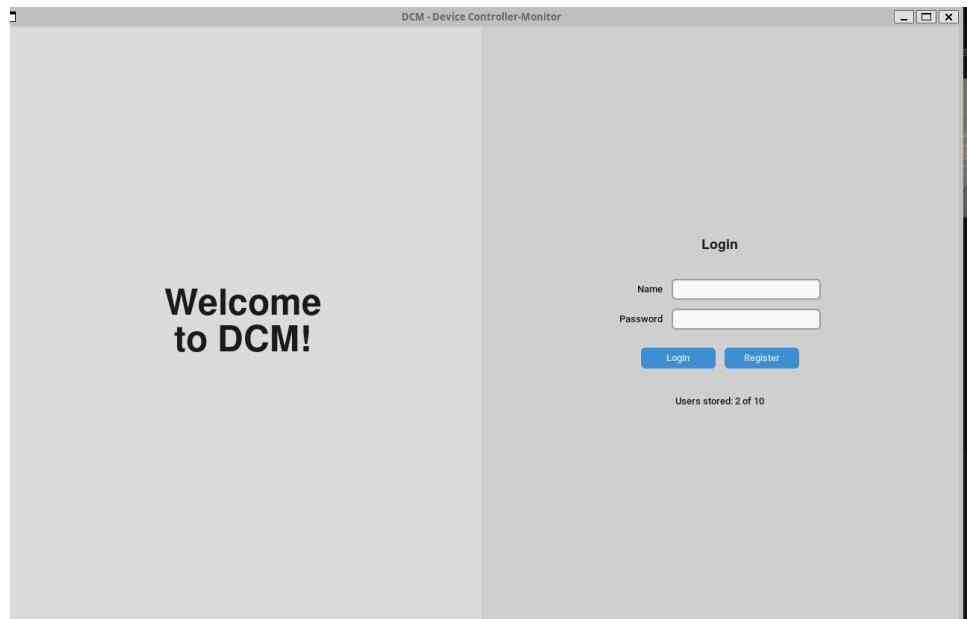


Figure 93: Login Test

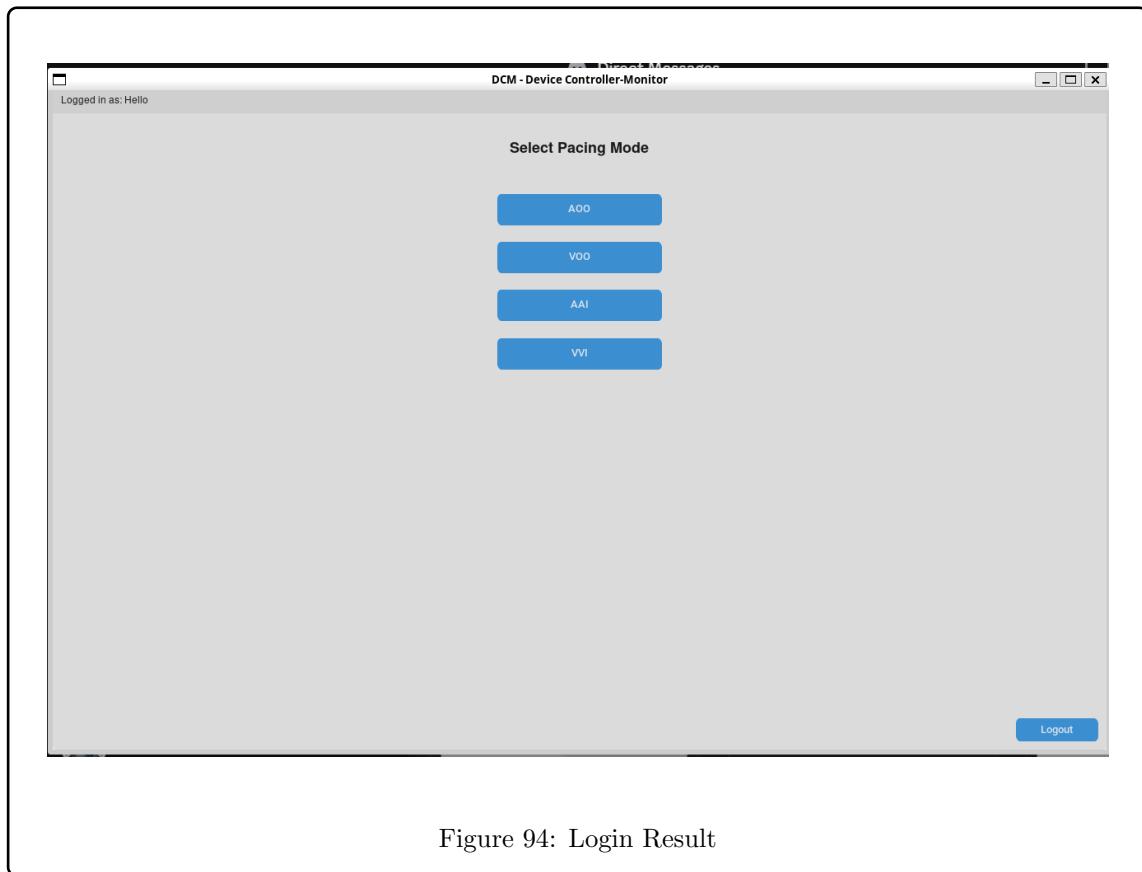


Figure 94: Login Result

9.7.2 Parameter Input Validation

Purpose: To enforce numeric types within an allowed range and to ensure the upper rate interval is greater than lower rate interval.

Input Conditions: Entering a non-numeric, an out of range number and an upper rate interval that is greater than lower rate interval in parameter settings.

Expected Output: Invalid input dialogue is shown and parameter changes are not saved.

Actual Output:

Result: Pass

The below figures show the general parameter page with some values inputted as well as the results of putting invalid values into parameter page.

The screenshot shows the DCM Parameter Page for AOO mode. The page has a dark background with white text and input fields. At the top left, it says "Logged in as: test". In the center, it says "Editing Parameters for: AOO". On the left, there is a vertical list of parameters with their current values:

- Lower Rate Limit (20-175): 122
- Upper Rate Limit (20-175): 122
- Atrial Amplitude (0.5-7.0): 2
- Atrial Pulse Width (0.05-1.0): 1
- Ventricular Amplitude (0.5-7.0):
- Ventricular Pulse Width (0.05-1.0):
- VPR (100-500):
- ARP (100-500):

At the bottom left is a "Back to Mode Selection" button. At the bottom right are "Save Parameters" and "Logout" buttons.

Figure 95: DCM Parameter Page with Values Filled

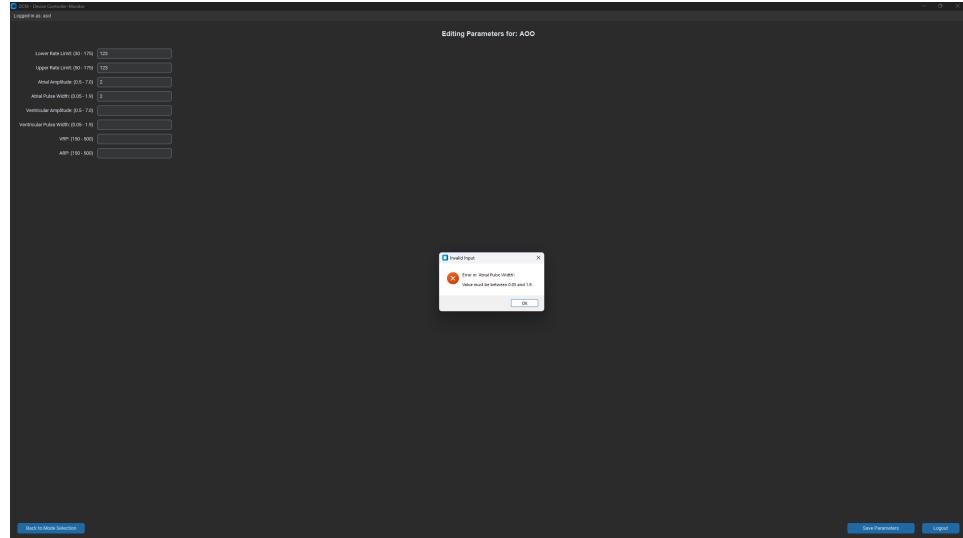


Figure 96: DCM Parameter Error When Number Inputted is Out of Range

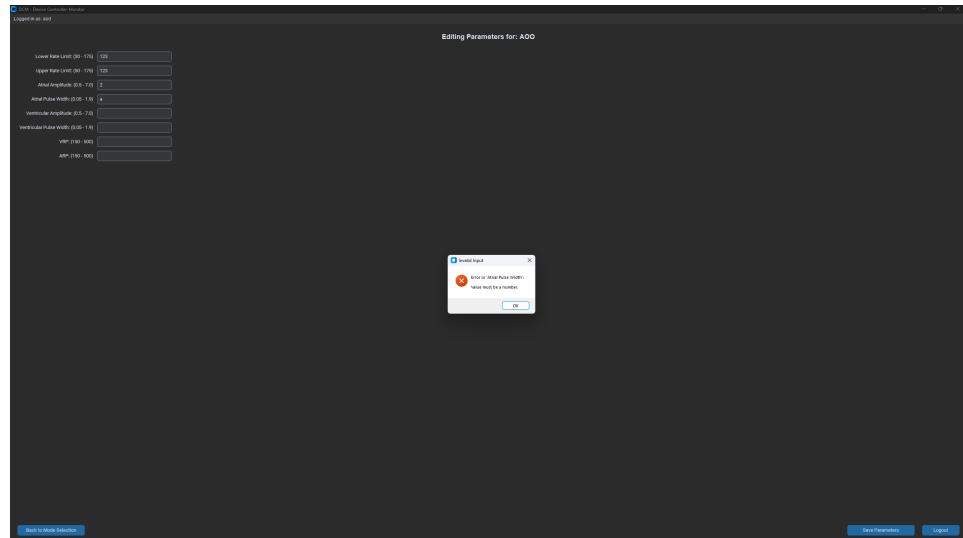


Figure 97: DCM Parameter Error When Non-Number is Inputted

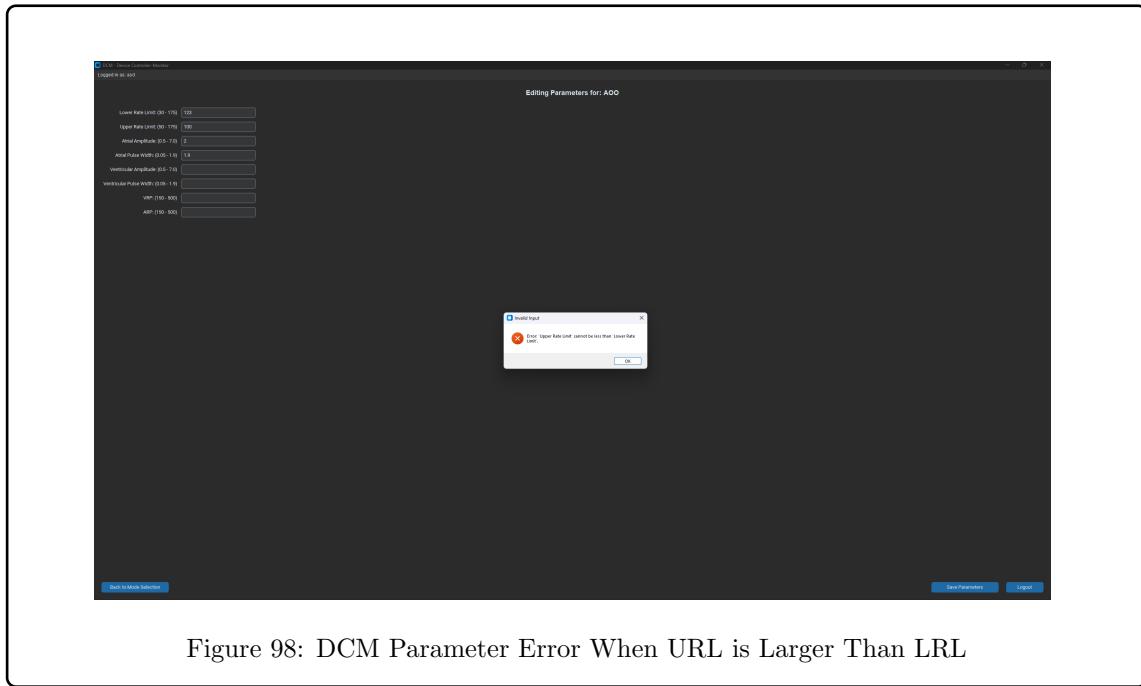


Figure 98: DCM Parameter Error When URL is Larger Than LRL

9.7.3 Mode Selection and Data Retrieval

Purpose: To test data storage, ensuring proper saving of user data

Input Conditions: Registering account, logging in, and saving parameters.

Expected Output: User data is now found in the associated JSON files.

Actual Output: Registered user data and parameters are found in their respective JSON files.

Result: Pass

This test was done in conjunction to previous tests except with a different user registered. A user "asd" with password "asd" was used for faster log ins. The following images are of the JSON files and the saved parameters from the previous test.

```
3K04-Project > DCM > models > {} pacing_settings.json > ...
1  {
2    "asd": {
3      "A00": {
4        "Lower Rate Limit": "123",
5        "Upper Rate Limit": "123",
6        "Atrial Amplitude": "2",
7        "Atrial Pulse Width": "1.9"
8      }
9    }
10 }
```

Figure 99: Stored Parameter File

```
3K04-Project > DCM > models > {} users.json > ...
1  {
2    "asd": "asd"
3  }
```

Figure 100: Stored Users File

9.7.4 Font Size Accessibility Controls

Purpose: To verify that font size controls improve accessibility by allowing users to enlarge or reduce text for clearer visibility.

Input Conditions: The user presses either **A+** to increase font size or **A-** to decrease font size.

Expected Output: All visible text elements update to a larger or smaller size immediately, including labels, buttons, and plot titles. The layout remains stable.

Actual Output: Pressing **A+** increased font size across all interface elements. Pressing **A-** reduced size consistently. All components updated correctly and remained properly aligned.

Result: Pass

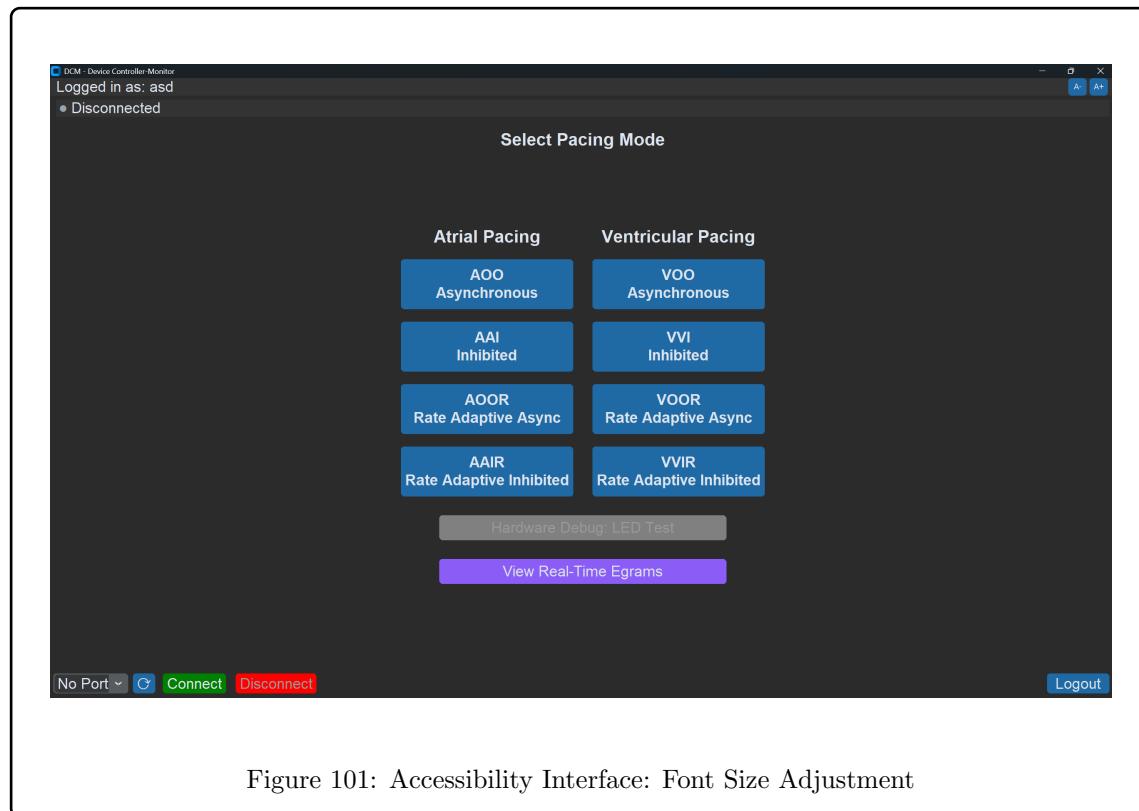


Figure 101: Accessibility Interface: Font Size Adjustment

9.7.5 Manual Entry of “OFF” in Programmable Field

Purpose: To confirm that typing “OFF” into a programmable parameter field is recognized as a valid symbolic value and updates the internal model correctly.

Input Conditions: The user manually types “OFF” into a programmable text entry field, such as amplitude, sensitivity, or refractory period.

Expected Output: The system should accept “OFF” as a valid entry, convert it to the correct internal value (such as zero or the OFF code), and display “OFF” in the interface without errors.

Actual Output: Typing “OFF” was accepted. The field saved the OFF state, the internal parameter was updated correctly, and the value was transmitted in the outgoing packet with the proper OFF code.

Result: Pass

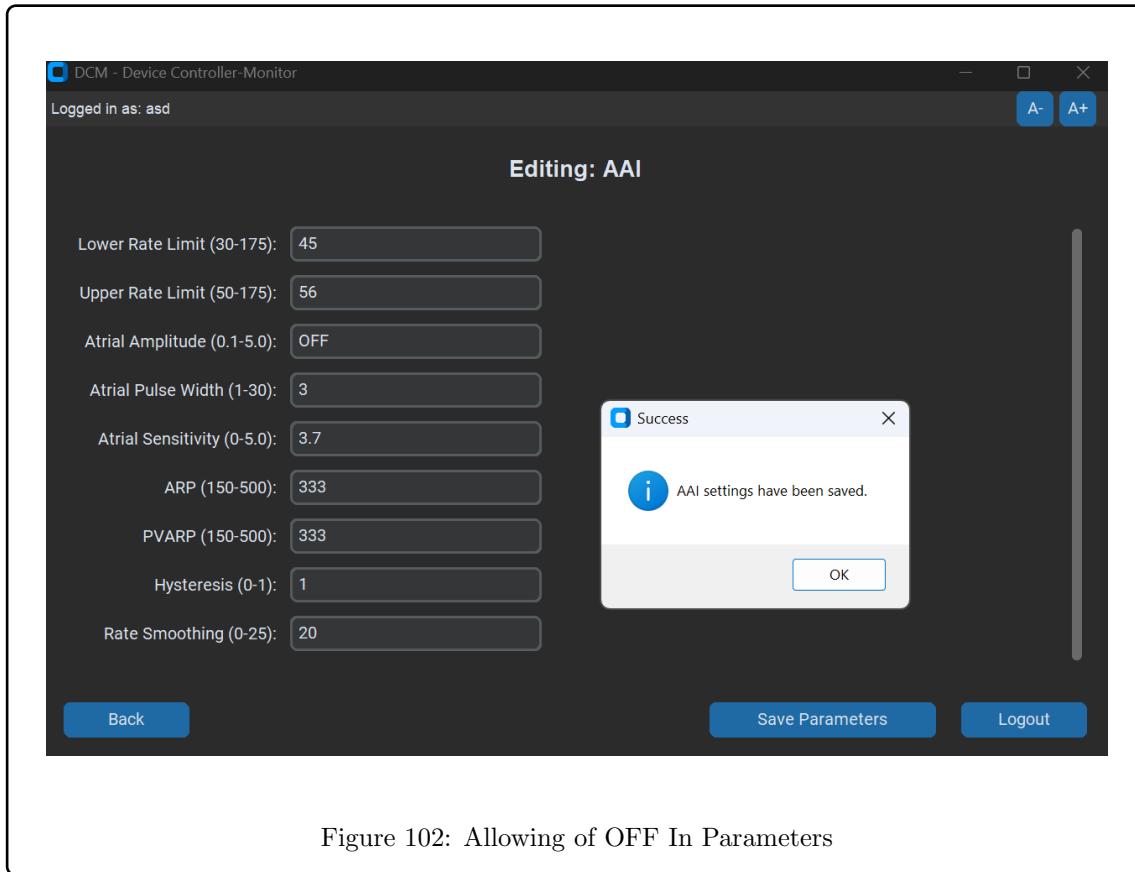


Figure 102: Allowing of OFF In Parameters

9.7.6 Error When Incrementing Values Not by 0.1

Purpose: To confirm that the system rejects values that are not valid increments of 0.1 and provides a clear error message.

Input Conditions: The user types a value such as 2.37 or 0.15 into a field that only allows increments of 0.1.

Expected Output: The system should detect the invalid step size, prevent saving the value, highlight the field, and show an error message explaining that only 0.1 increments are allowed.

Actual Output: The system displayed an error, rejected the invalid value, and restored the previous valid value. No incorrect data was saved or transmitted.

Result: Pass

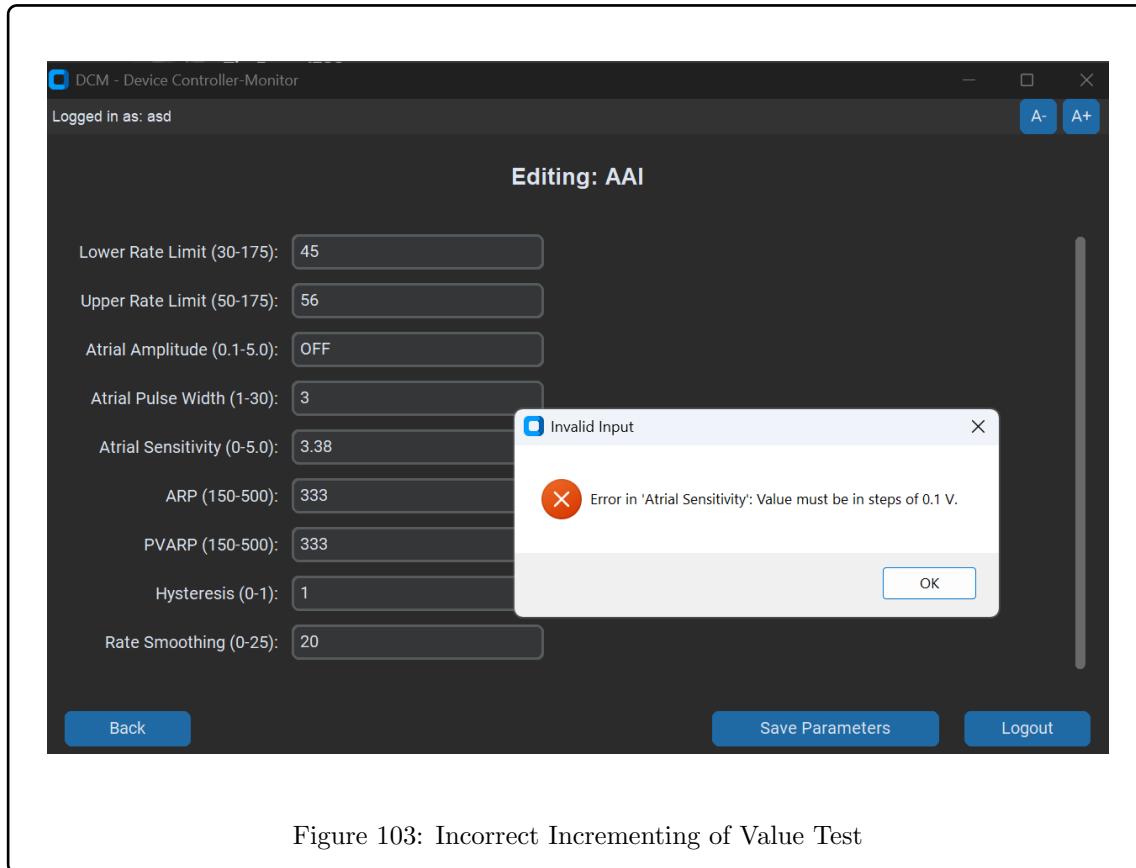


Figure 103: Incorrect Incrementing of Value Test

9.7.7 Amplitude Range 0.1–5.0 V

Purpose: Verify that the A or V Pulse Amplitude field rejects values outside the allowed range of 0.1 V to 5.0 V.

Input Conditions: User types the following values into the amplitude field and presses Save:

- Test A: 5.5
- Test B: 0
- Test C: -0.1

Expected Output: For each invalid value the system should:

- Show an error message stating that amplitude must be between 0.1 V and 5.0 V.
- Prevent the value from being saved.

- Keep or restore the previous valid value.

Actual Output: For 5.5, 0, and -0.1, the DCM displayed an error message that the amplitude must be within 0.1–5.0 V, did not save the invalid entry, and left the previous valid amplitude in the field. No incorrect data was written to `pacing_settings.json`.

Result: Pass

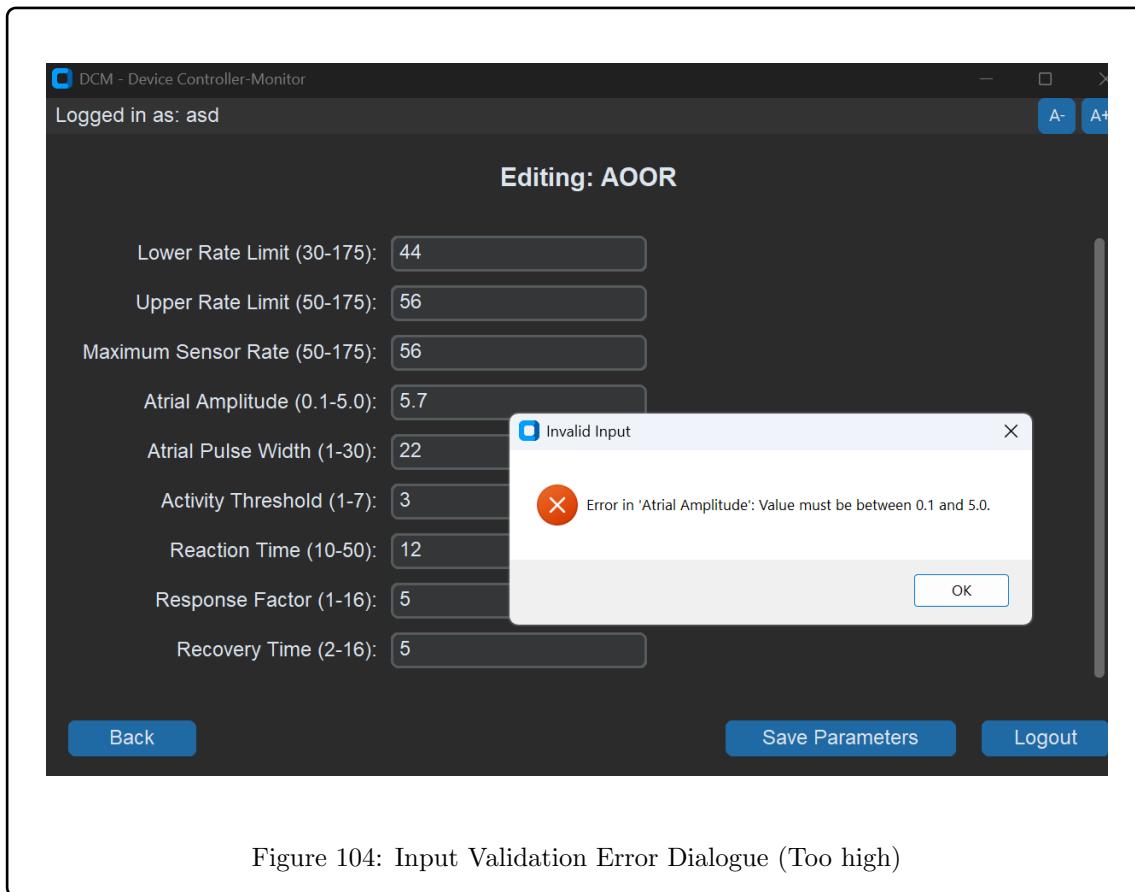
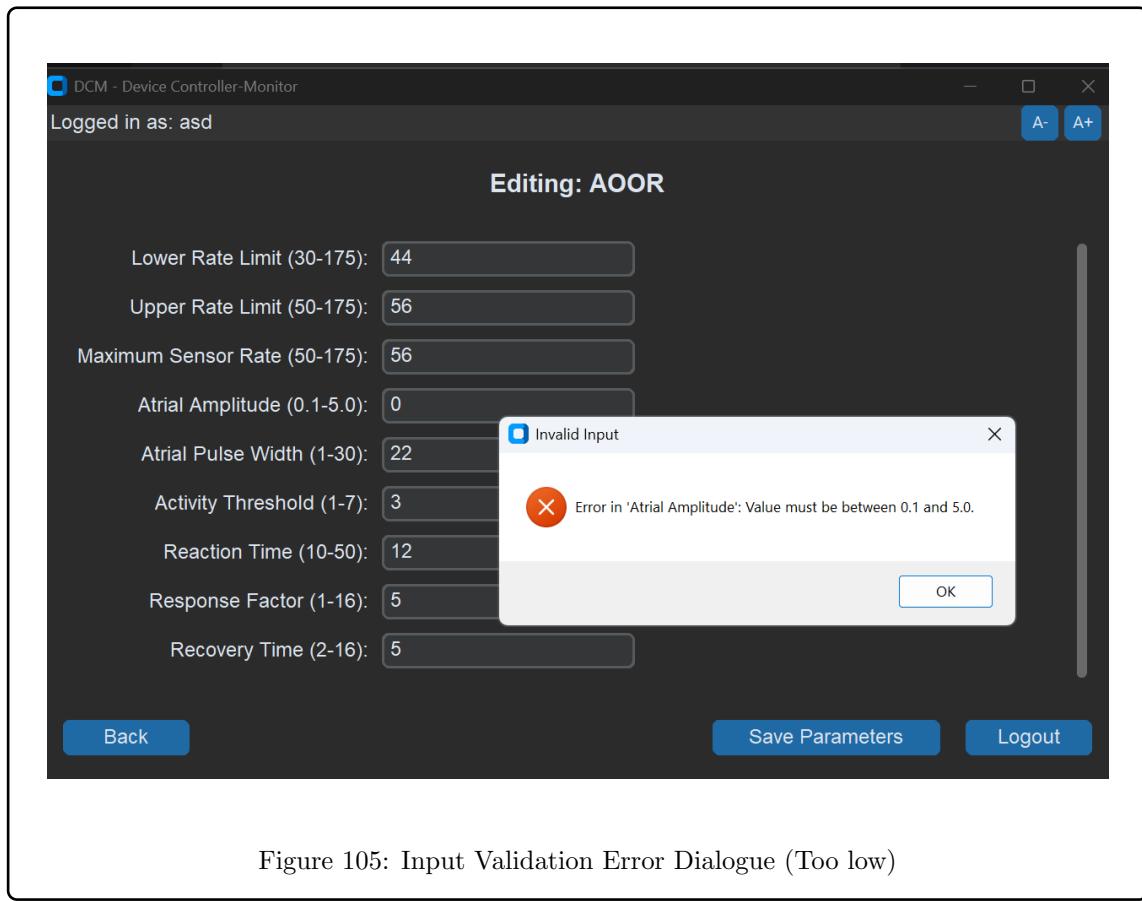


Figure 104: Input Validation Error Dialogue (Too high)



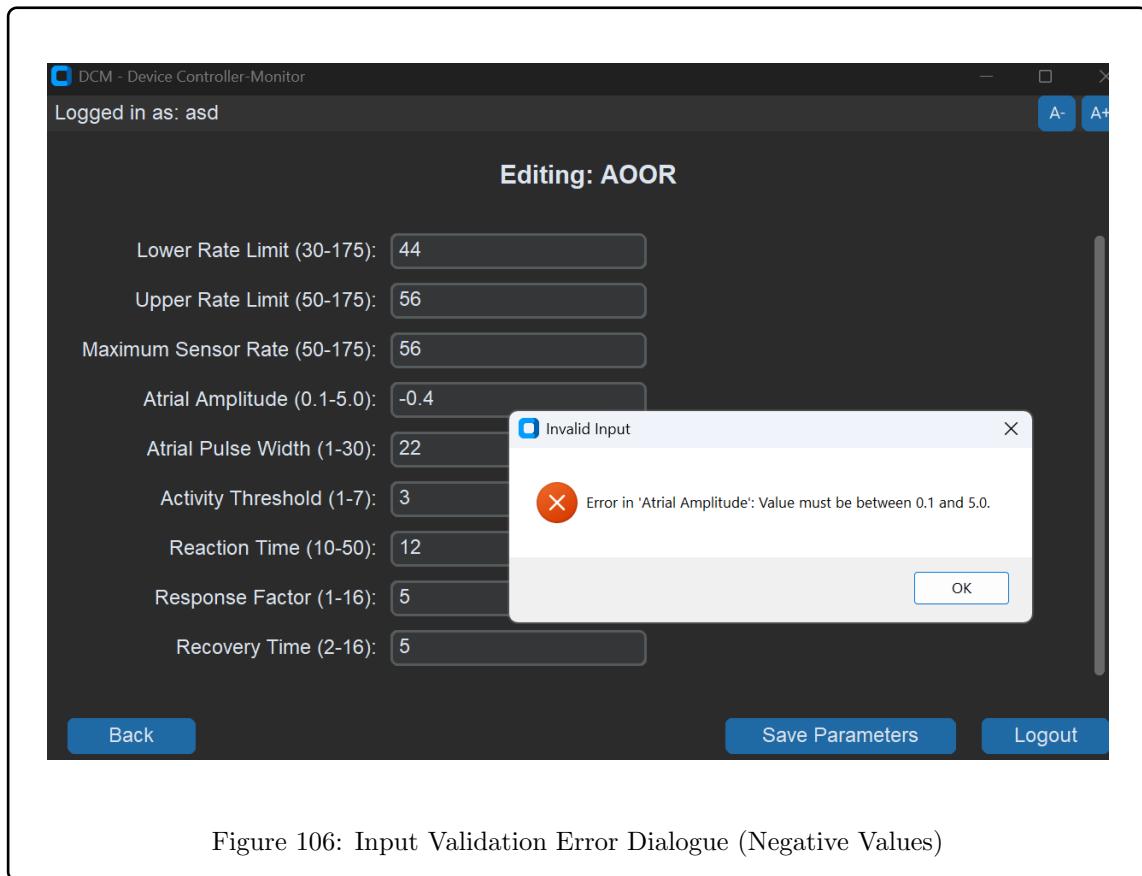


Figure 106: Input Validation Error Dialogue (Negative Values)

10 GenAI Usage

We used a Generative AI assistant to support development of the DCM. It provided starter boilerplate for a Tkinter app with a welcome screen, registration and login, JSON storage capped at ten users, which we then adapted and tested. We also used it to clarify Python functions and libraries such as Tkinter, JSON, etc. and to troubleshoot installing tkinter. We had AI to clarify comments within the code as well. All design decisions, requirements, and validation were done by our team, and we reviewed and verified all AI outputs before inclusion. Furthermore, AI was used as a search engine for debugging of both Simulink and DCM issues.