

SE 3K04

Assignment 2: Part 2

DCM Design

Group 18

Dec 2, 2018

Table of Contents

1: Expand the DCM	3
2: Implement Serial Communication	3
CommIn and CommOut Contents	3
Design Principles	4
Future Changes	4
3: GUI Implementation	4
4: Testing	5
5: Appendix A: GUI Images	6

1: Expand the DCM

Expanding on the design and code setup in milestone 1, a dropdown was implemented for mode selection. Additional parameters were implemented. Provisions were made to ensure the parameters are in the appropriate ranges.

Serial communication was established between python and the device and conversion algorithms were set to encode and decode serial data from the COM port.

To plot the electrograms a math library was used. A new window was implemented to view plot data.

2: Implement Serial Communication

CommIn and CommOut Contents

Name/ (Data Type)	Size (bytes) (# of uint8 values)	Start Byte Number/Index	Values/ Respective Function
Sync (uint8)	1	0/1	1 / Signifies beginning of transmission
FnCode (uint8)	1	1/2	{1,0}/ {receive data, transmit data}
p_PacingMode (uint8)	1	2/3	{0-19} / {Off, AAT, VVT, AOO, AAI, VOO, VVI, VDD, DOO, DDI, DDD, AOOR, AAIR, VOOR, VVIR, VDDR, DOOR, DDIR, DDDR}
p_lowrateInterval (uint16)	2	3/4	1-65 535 ms/ Specifies max delay after a pace without a consecutive pace
p_vPaceAmp (uint16)	2	5/6	0-100 (sent as a double value to simulink)
p_vPaceWidth (uint16)	2	7/8	0 – p_lowrateInterval ms
p_VRP (uint16)	2	9/10	1-p_lowrateInterval ms
p_aPaceAmp (uint16)	2	11/12	0-100 % (sent as a double value to simulink)
p_aPaceWidth (uint16)	2	13/14	0 – p_lowrateInterval ms
p_ARP (uint16)	2	15/16	1-p_lowrateInterval ms
Vent_Egram(int16)	2	17/18	-500 - 500 mV
Atr_Egram(int16)	2	19/20	-500 - 500 mV
Data chkSum (uint8)	1	21/22	{0,22} / # of data bytes succsesfully sent

Design Principles

Using the Serial Receive and Serial Transmit blocks we can send and receive data to the DCM. The convention used to communicate back and fourth between the DCM and Pacemaker is established as shown in the tables above. To implement this In Simulink we used multiple charts and subsystems. Once a message is transmitted through serial communication, status updates to zero, indicating that all bytes of CommIn are filled and ready to be used. COMM_IN chart initializes all of the given information to be used in our implementation for pacing, based on the value of FnCode. FnCode tells the system what to do with the message (i.e program parameters, echo parameters to DCM etc.). Similarly, with communication to the DCM, FnCode determines whether to send egram data or the current parameters.

Future Changes

To implement more modes, we will still use the same convention to transmit and receive signals over serial communication. We can do this because we generalized the current mode byte as done in srsVVI, such that the modes can be changed without restarting the DCM or Pacemaker. When sending information about the egram, we send the outputs of the ATR_SIGNAL and VENT_SIGNAL to the DCM. However, we would need to expand our convention to send messages to the DCM to include byte space for the implementation of a legend to distinguish the egrams.

3: GUI Implementation

- a) The purpose of this module is to store specification parameters of users and store them locally. The module also provides logic and registration features. The module acts as an interface to control the device mode and parameters as well as to display data relayed by the device.
- b) The secret of this module is the way user data is accessed/stored by the software, each user has their own file of data stores in a secured location.
- c) There are 4 public functions for GUI implementation. They are login, launch, register, and parameters. The public function launch takes in the parameter “win”. The parameter win is used to launch the first window. The rest of the public functions take in the parameter button. This controls the button click input for the various GUI screens.
- d) The black box behavior of this code is as follows. The program takes in user login information or registration, which includes username and a password. The output of the program is the heart rate parameter displayed for the active user. The parameters are input and output in an understandable format and the conversion algorithm is hidden from the user.
- e) There are 2 global variables in the module, “usr” and “credentials”. The variable “usr” holds the value of the current active user. This global variable is of type string. The next global variable “credentials” is of type dictionary, which falls under arrays. This date type allows for data to be stored as a key and value, where each key is linked to a single value. This is used to hold login credentials.
- f) All of the functions in the program are public functions. All the functions are public because they are all GUI based and the user needs access to control the GUI. However, important user information, such as username and password, are stored in a local file which other functions of the code cannot access.

g) The first public function launch opens the login or the register window. The second public function login controls the login button, checks if the credentials are correct and opens the parameter window, or controls the cancel button which takes you back to the home screen.

The third public function register is responsible for registration of new users. There are two buttons that may be selected - "Register" or "Go Back". If "Register" is selected, the new user is asked to enter a username and password. Before entering the values, the functions check in the text file "credentials" (global variable) if there are 10 users registered. If max users are reached, it does not allow any more users to register. If there are less than 10 users, the functions then check the username and whether it has been repeated or not. If all these conditions are true, then the username and password are written on to the text file. This is the only function that is updating the global variable credentials, the rest are just accessing it. The fourth public function is parameters which is responsible for displaying all the parameters of the pacemaker and updating the respective ones. If the update button is pressed, the parameters will be updated, and the new value will be written into the respective user's text file. The next time the user opens the parameter window, they will see the new updated values. Furthermore, the button "Request Parameters" pulls serial data from the device and updates the Text Fields with the parameters currently stored on the device. The button "E-GRAM" displays Matplotlib graph representing 10 Ventricle and Atrium data points obtained serially from the device. If the user presses the log off button, the user will be redirected to the login window.

Design Decisions and Requirements likely to change will be the GUI interface to look more appealing to the eyes by adding images to the layout. In terms of requirements, further securing the user file accessibility will be strengthened.

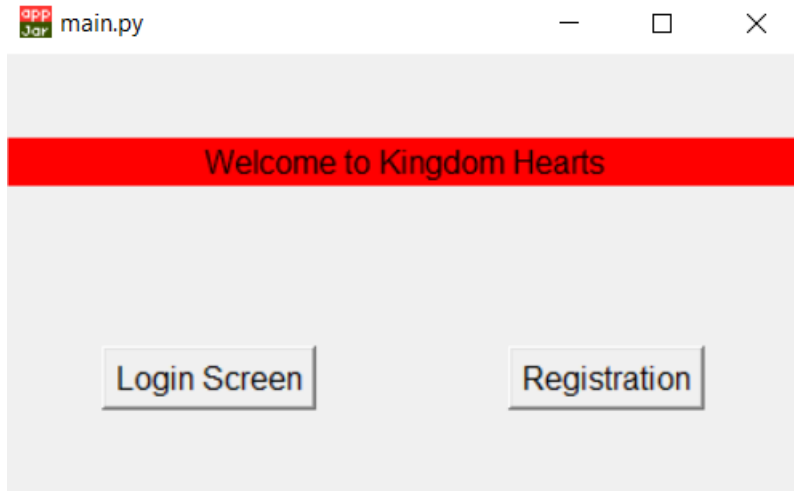
4: Testing

The following tests were used to check the durability of the program

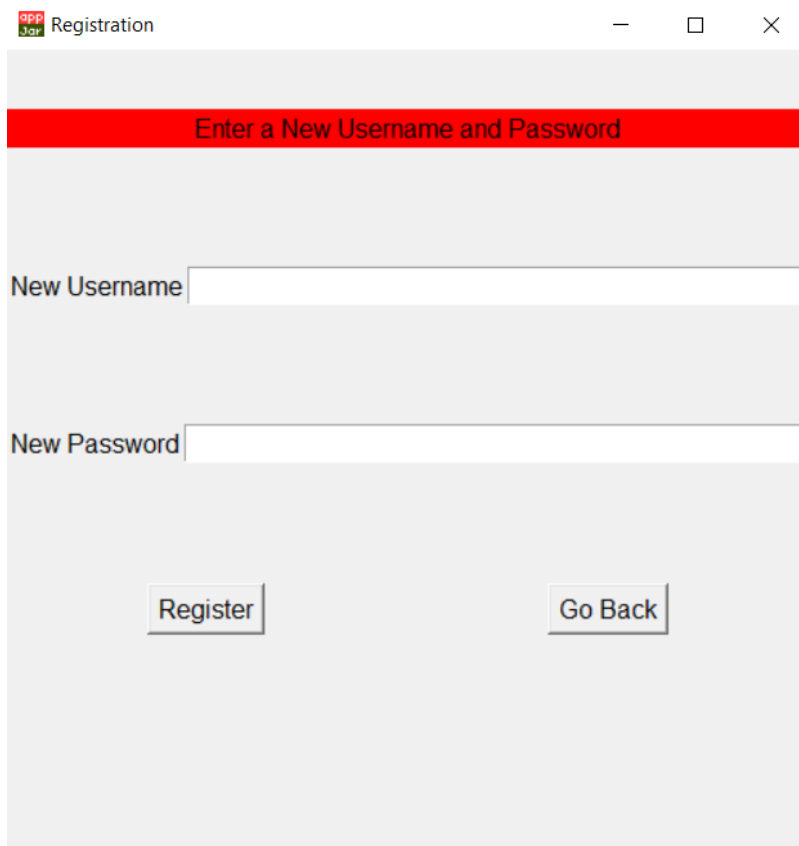
- Eleven Users were created to expect the 11th user to not be registered and display appropriate message. – Result: **Success**
- Same user names were registered to check rejection of duplicate usernames – Result: **Success**
- Login with incorrect password should display login failure with appropriate message – Result: **Success**
- Tested the buttons to go back into login and registration pages without actually logging or registering a new user – Result: **Success**
- Once exited from a GUI window, ensured windows are hidden. – Result: **Success**
- Updated user information and cross-checked the locally stored text file for updates. – Result: **Success**
- Tested serial comm by setting up "dummy" inputs from the COM ports and trying to read and change them one by one – Result: **Success**

5: Appendix A: GUI Images

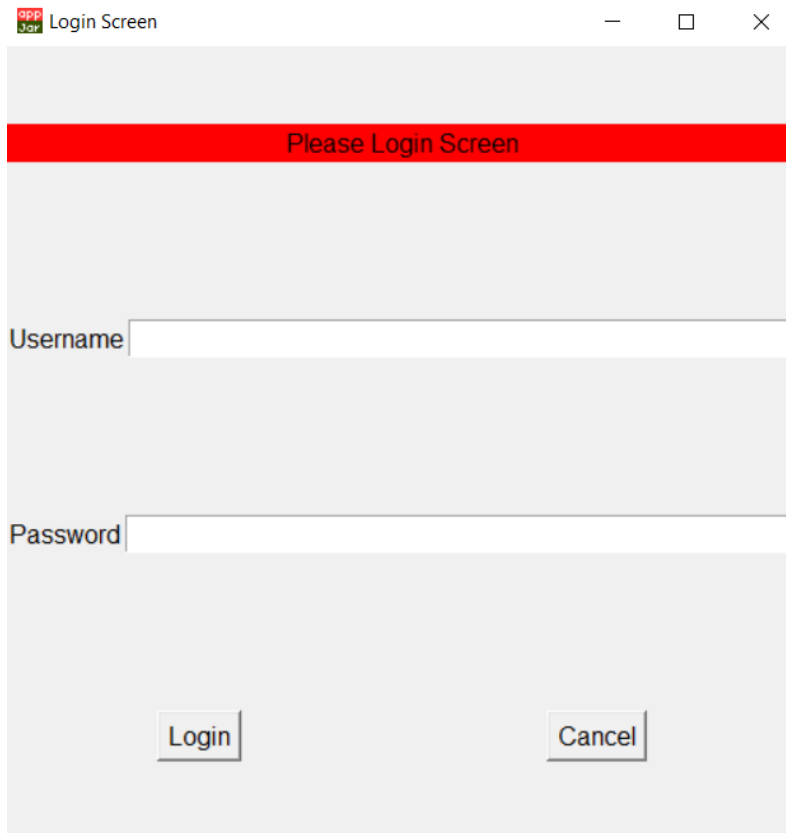
Welcome Page:



Registration Page:

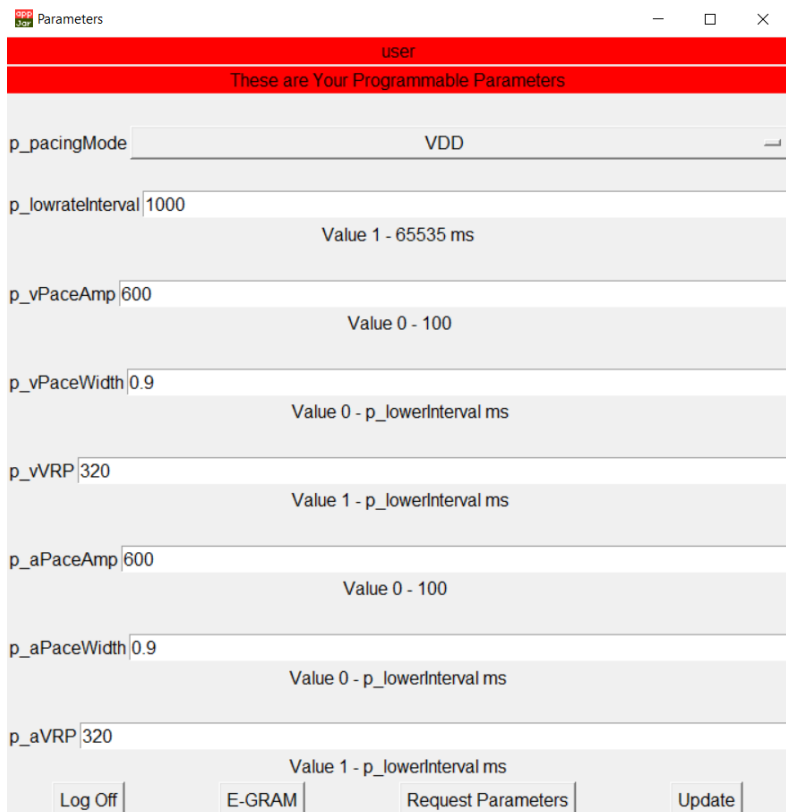


Login Page:



A screenshot of a 'Login Screen' window. The window has a title bar with a small icon and the text 'Login Screen'. Below the title bar is a red banner with the text 'Please Login Screen'. The main area contains two input fields: 'Username' and 'Password'. Below these fields are two buttons: 'Login' and 'Cancel'.

Parameters Pages:



A screenshot of a 'Parameters' window. The window has a title bar with a small icon and the text 'Parameters'. Below the title bar is a red banner with the text 'user'. Below that is another red banner with the text 'These are Your Programmable Parameters'. The main area contains several parameters, each with a value field and a range description:

- p_pacingMode**: VDD
- p_lowrateInterval**: 1000 (Value 1 - 65535 ms)
- p_vPaceAmp**: 600 (Value 0 - 100)
- p_vPaceWidth**: 0.9 (Value 0 - p_lowerInterval ms)
- p_vVRP**: 320 (Value 1 - p_lowerInterval ms)
- p_aPaceAmp**: 600 (Value 0 - 100)
- p_aPaceWidth**: 0.9 (Value 0 - p_lowerInterval ms)
- p_aVRP**: 320 (Value 1 - p_lowerInterval ms)

At the bottom of the window are four buttons: 'Log Off', 'E-GRAM', 'Request Parameters', and 'Update'.