SFWRENG 3K04: Software Development

Instructor: Dr. Alan Wassyng

Assignment 2

Group **4**:

Yilong Wang– wangy761 – 400246235

Bofeng Cao – caob16 – 400271840

Hairong Bao – baoh10 – 400237309

Chinghung Tan – tanc24 – 400244643

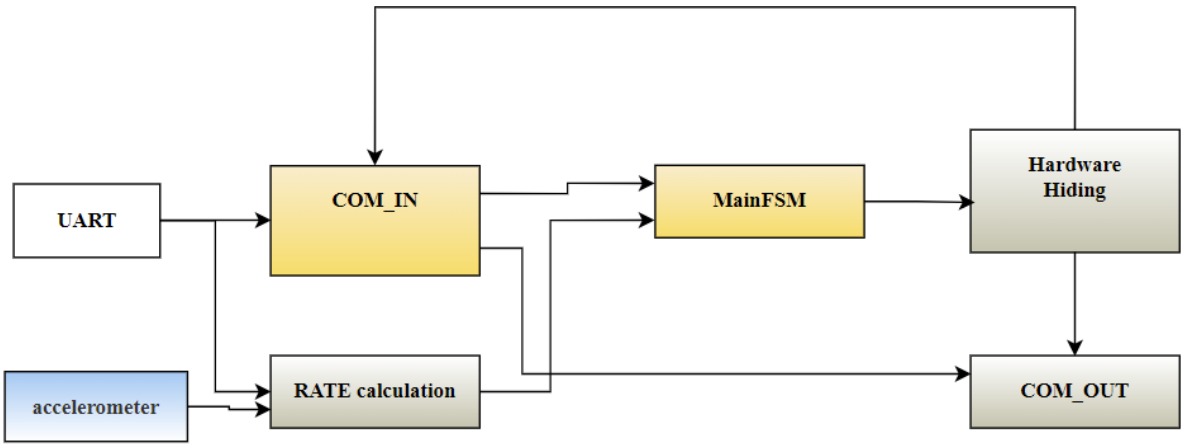
Yuhan Cui – cuiy52 – 400240238

Zhaohan Wang – wangz394 – 400188640

Table of Contents

Part 1: Pacemaker Design

- *Overview of the design:*



overview on the state flow

UART: Behave as a communication port with DCM and receive parameters used in a pacemaker.

COM_IN: Initialize parameters with default values and change the values when receiving parameters from DCM/ giving data to DCM. Using '55' to change the parameters received from DCM; Using '22' to send *Param* data to DCM; Using '47' to send *Egram* data to DCM.

Rate calculation: Collect data from three directions and calculate the resultant acceleration.

MainFSM: Use a flow chart to display the transition and constraints between different modes and to show the steps to complete a whole cycle. In this state, we use different numbers to go to different modes, such as 0 for AOO, 1 for AAI, 2 for AOOR and etc.

Hardware Hiding: This subsystem contains all the components to communicate with the pacemaker, such as LEDs, pushbuttons, and VENT/ATR controllers. Also, it can detect the signals by *ATR/VENT_CMP_DETECT*.

COM_OUT: Generate *Param* data and *Egram* data and send them back to DCM through UART when needed.

Details in subsystem:

- **COM_IN**

In this COM_IN subsystem, the parameters are set to the default values and also receive data from the DCM.

Inputs:

rxdata, status

Outputs:

PACING_MODE, ATR_AMPULITUDE, VENT_AMPULITUDE, ATR_WIDTH, VENT_WIDTH, AV_DELAY, ARP, VRP, Threshold, LRL, URL, ReactionTime, ResponseFactor, RecoverTime, MaxSensorRate, change

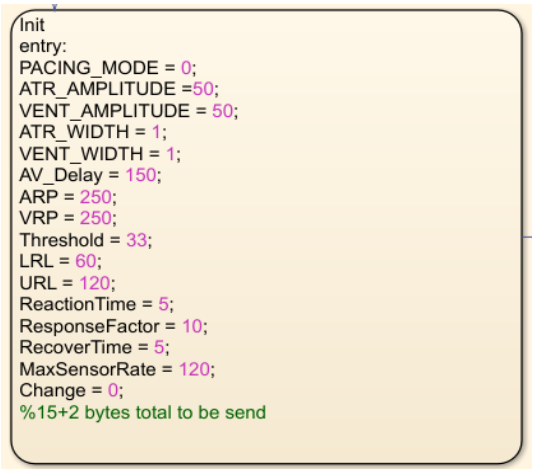
1. The INITIAL state sets the parameters with default values (shown in State Configurations)
2. Then goes into STANDBY state, waits for DCM input communication.
3. Check if the first Byte of rxdata is 0x16. If it is, then there are three options for next states, SET_PARAM, ECHO_PARAM, and ECHO_EGRAM, determined by the second Byte of rxdata, 0x55, 0x22, and 0x47 respectively.

Table 1: State Transition Table of COM_IN subsystem

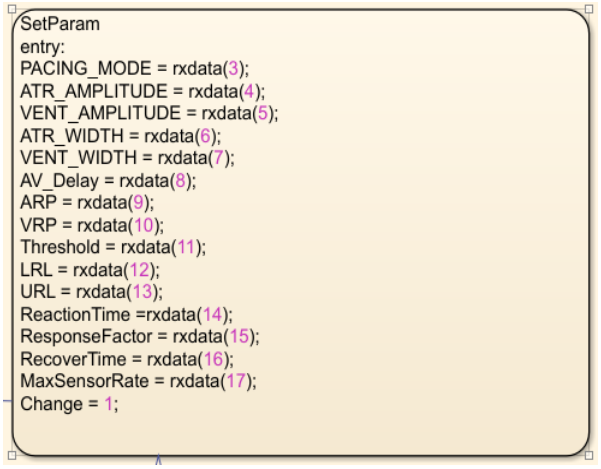
Current state	Event		Next state
INITIAL	-		STANDBY
STANDBY	Status == 0 rxdata(1) == 0x16	rxdata(2) == 0x55	SET_PARAM
		rxdata(2) == 0x22	ECHO_PARAM
		rxdata(2) == 0x47	ECHO_EGRAM
		NOT(rxdata(2) == 0x55 or 0x22 or 0x47	-
	NOT(status == 0 and rxdata(1) == 0x16)		
SET_PARAM	NOT(status == 0 and rxdata(1) == 0x16)		STANDBY
ECHO_PARAM	-		STANDBY
ECHO_EGRAM	After (5000,m)		STANDBY

State Configurations:

Init:

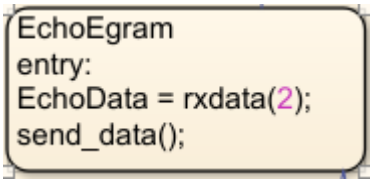


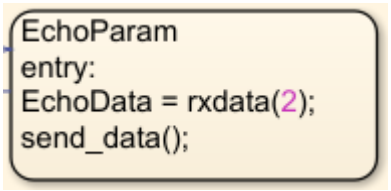
SetParam:



EchoEgram:

EchoParam:





EchoData is used to determine whether we need to send back the current params or the egram data to the DCM.

- **COM_OUT**

This subsystem is a send_data() global function which is called by other subsystems.

Inputs:

> PACING_MODE, ATR_AMPULITUDE, VENT_AMPULITUDE, ATR_WIDTH, VENT_WIDTH, AV_DELAY, ARP, VRP, Threshold, LRL, URL, ReactionTime, ResponseFactor, RecoverTime, MaxSensorRate, change

Outputs:

> EchoParam, EchoEgram

Two multiplexers and a switch are used in this subsystem to combine the data into packets.

- The first MUX is used to pack the transmit data for programmable params received from the COM_IN subsystem.
- The second MUX is used to pack the transmit data for egram data received from the Hardware Hiding subsystem (ATR_SIGNAL, and VENT_SIGNAL).
- The switch checks the value of EchoData to decide which output the function should transmit to send_data().

State Configurations:

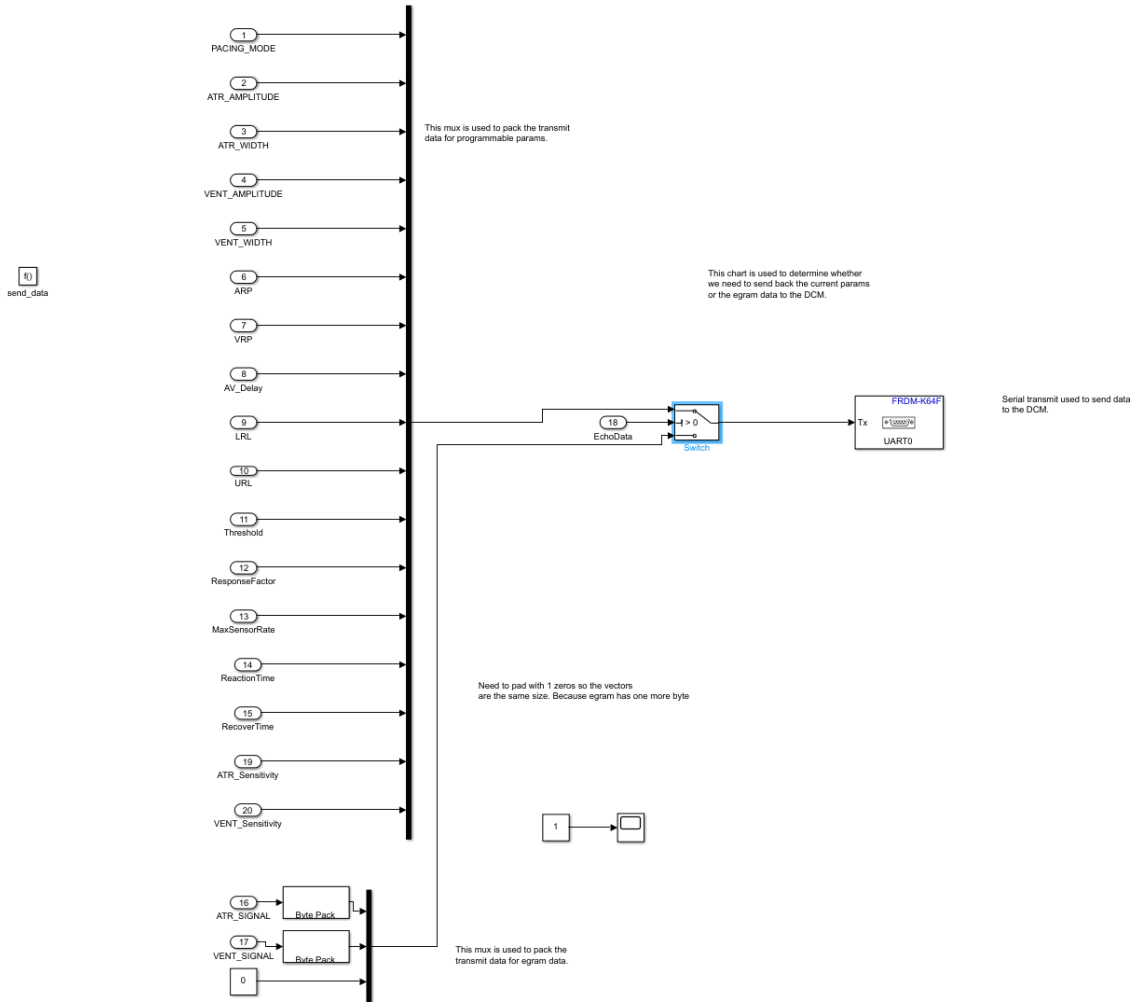


Table 2: Condition Table of COM_OUT subsystem

EchoData == 0x22	send_data() ← EchoParam
EchoData == 0x47	send_data() ← EchoEgram
NOT(EchoData == 0x22 or 0x47)	—

- **MainFSM:**

This subsystem controls all the pacing modes required and the hardware circuitry.

Inputs:

ATR_CMP_DETECT, VENT_CMP_DETECT, Threshold, PACING_MODE, ATR_AMPLITUDE, ATR_WIDTH, VENT_AMPLITUDE, VENT_WIDTH, ARP, VRP, Rate, Change, AV_Delay, LRL. PUSH_BUTTON

Outputs:

PACE_CHARGE_CTRL, Z_ATR_CTRL, PACING_REF_PWN, Z_VENT_CTRL, ATR_PACE_CTRL, VENT_PACE_CTRL, PACE_GND,CTRL, VENT_GND_CTRL, FRONTEND_CTRL

State configuration:

Atrium_ Charging

To discharge the Blocking Capacitor, we need to ground C21 and to make the current flow back to C22.

Pin	Setting
PACE_CHARGE_CTRL	HIGH (True)
VENT_PACE_CTRL	LOW (False)
Z_VENT_CTRL	LOW (False)
ATR_PACE_CTRL	LOW (False)
ATR_GND_CTRL	HIGH (True)
VENT_GND_CTRL	LOW (False)

Atrium Pacing:

After charging the C22 capacitor, the system may move to the pacing state after ATR_DELAY.

Pin	Setting
PACE_CHARGE_CTRL	LOW (False)
PACE_GND_CTRL	HIGH (True)
ATR_PACE_CTRL	HIGH (True)

ATR_GND_CTRL	LOW (False)
Z_ATR_CTRL	LOW (False)
Z_VENT_CTRL	LOW (False)
VENT_GND_CTRL	LOW (False)
VENT_PACE_CTRL	LOW (False)
BLUE_LED	HIGH (True)

Ventricular Charging:

The charging and discharging session of VOO is very similar to AOO, however, few changes have been made, shown in **BLUE**.

Pin	Setting
PACE_CHARGE_CTRL	HIGH (True)
VENT_PACE_CTRL	LOW (False)
Z_VENT_CTRL	LOW (False)
ATR_PACE_CTRL	LOW (False)
ATR_GND_CTRL	LOW (False)
VENT_GND_CTRL	HIGH (True)

Ventricular Pacing:

After charging the C22 capacitor, the system may move to the pacing state after ATR_DELAY.

The Pacing part of VOO is very similar to AOO, however, few changes have been made, shown in ***BLUE***.

Pin	Setting
PACE_CHARGE_CTRL	LOW (False)
PACE_GND_CTRL	HIGH (True)
ATR_PACE_CTRL	LOW (False)
ATR_GND_CTRL	LOW (False)
Z_ATR_CTRL	LOW (False)

Z_VENT_CTRL	LOW (False)
VENT_GND_CTRL	LOW (False)
VENT_PACE_CTRL	HIGH (True)
RED_LED	HIGH (True)

Pace will stop and move to charge the Primary Capacitor (C22) and discharge the Blocking Capacitor (C21) after ATR_WIDTH.

Different modes will determine how to switch these states.

1. **AOO**

Current state	**condition**	**next state**
A_Charging	after(60000/LRL-ATR_WIDTH,msec)	A_Pacing
A_Pacing	after(ATR_WIDTH,msec)	A_Charging

2. **AAI**

Current state	**condition**	**next state**
A_Charging	after(60000/LRL-ATR_WIDTH-ARP,msec)	A_Pacing
	after(ARP,msec)&& ATR_CMP_DETECT == true || PUSH_BUTTON ==true	A_Charging
A_Pacing	after(ATR_WIDTH,msec)	

3. **AOOR**

Current state	**condition**	**next state**
A_Charging	after(60000/Rate-ATR_WIDTH,msec)	A_Pacing
A_Pacing	after(ATR_WIDTH,msec)	A_Charging

4. **AAIR**

Current state	condition	next state
A_Charging	after(60000/Rate-ATR_WIDTH-ARP,msec)	A_Pacing
	after(ARP,msec)&& ATR_CMP_DETECT == true || PUSH_BUTTON ==true	A_Charging
A_Pacing	after(ATR_WIDTH,msec)	

5. **VOO**

Current state	condition	next state
V_Charging	after(60000/LRL-VENT_WIDTH,msec)	V_Pacing
V_Pacing	after(VENT_WIDTH,msec)	V_Charging

6. **VVI**

Current state	condition	next state
V_Charging	after(60000/LRL-VENT_WIDTH-ARP,msec)	V_Pacing
	after(ARP,msec)&& VENT_CMP_DETECT == true || PUSH_BUTTON ==true	V_Charging
V_Pacing	after(VENT_WIDTH,msec)	

7. VOOR

Current state	condition	next state
V_Charging	after(60000/Rate-VENT_WIDTH,msec)	V_Pacing
V_Pacing	after(VENT_WIDTH,msec)	V_Charging

8. **VVIR**

Current state	condition	next state

V_Charging	after(60000/Rate-VENT_WIDTH-ARP,msec)	V_Pacing
	after(ARP,msec)&& VENT_CMP_DETECT == true || PUSH_BUTTON ==true	V_Charging
V_Pacing	after(VENT_WIDTH,msec)	

9. DOO

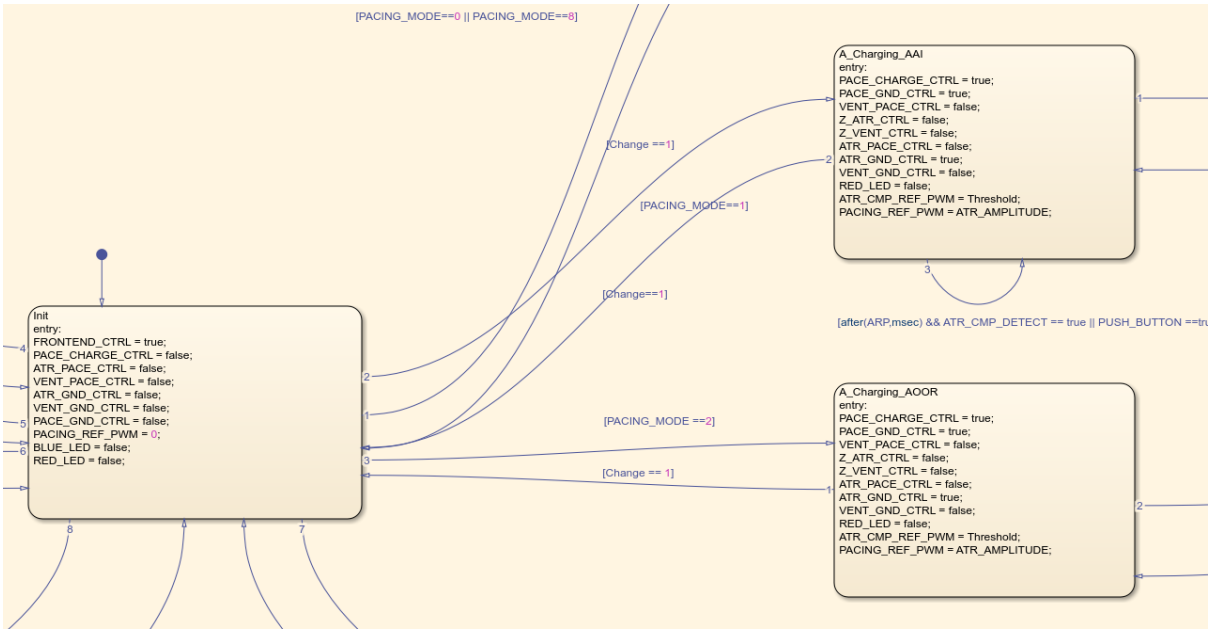
Current state	condition	Next state
A_Charging	after(60/LRL*1000-VENT_WIDTH-AV_Delay,msec)	A_Pacing
A_Pacing	after(ATR_WIDTH,msec)	V_Charging
V_Charging	after(AV_Delay-ATR_WIDTH,msec)	V_Pacing
V_Pacing	after(VENT_WIDTH,msec)	A_Charging

10.DOOR

Current state	condition	Next state
A_Charging	after(60/Rate*1000-VENT_WIDTH-AV_Delay,msec)	A_Pacing
A_Pacing	after(ATR_WIDTH,msec)	V_Charging
V_Charging	after(AV_Delay-ATR_WIDTH,msec)	V_Pacing
V_Pacing	after(VENT_WIDTH,msec)	A_Charging

Dynamically changing mode

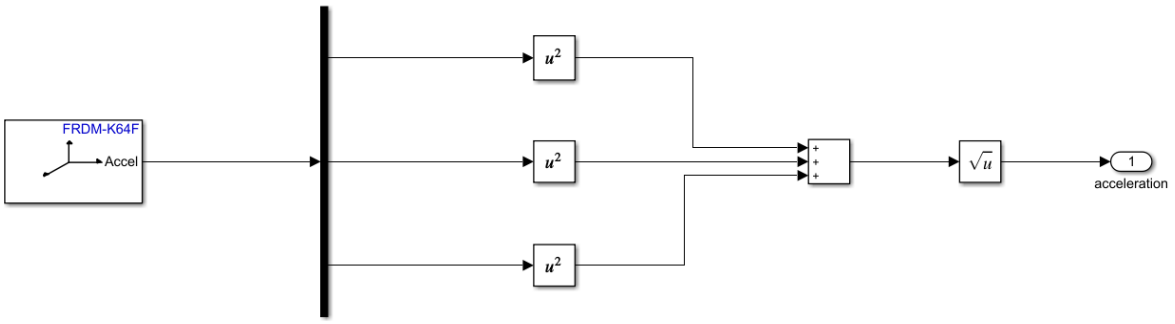
Between the Init state and each Charging state, there is a condition that if [Chang == 1], it will move back to the Init and will go to another mode.



The variable Change is set in the COM_IN, When DCM sends the data and sets the parameter, the Change will be 1. This indicates DCM changes the mode. So it will dynamically change the state without restarting the device.

- **Rate calculation**

Accelerometer:



The accelerometer vector data sensed is passed through a demux to get 3 dimensional acceleration vectors. The acceleration values are then computed by these 3 dimensional vectors using the root mean square.

Rate calculation:

This subsystem is for calculating the rate modulation.

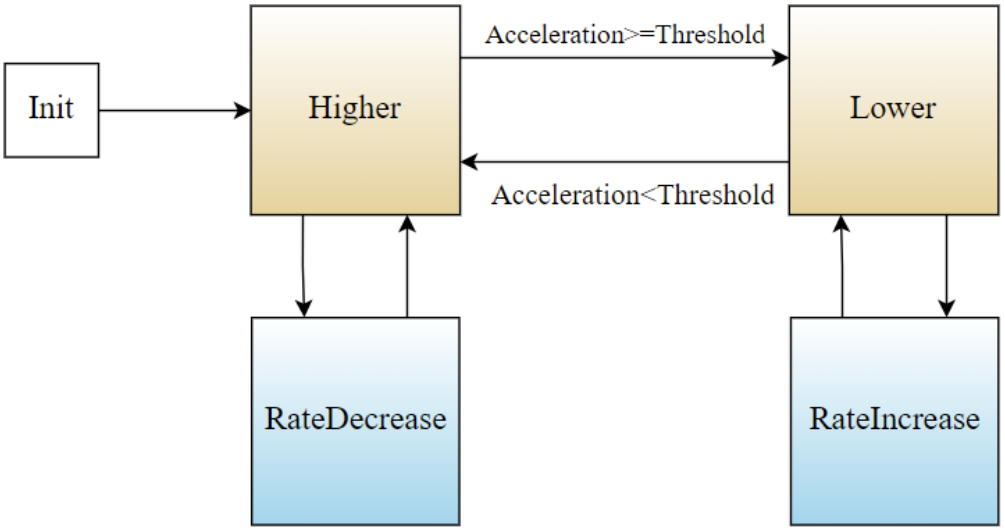
Inputs:

Threshold, ResponseFactor, MaxSensorRate, ReactionTime, RecoverTime, Acceleration, LRL

Outputs:

Rate

The initial state sets the rate value to LRL. Then checks if the Acceleration is higher or lower than the Threshold value.



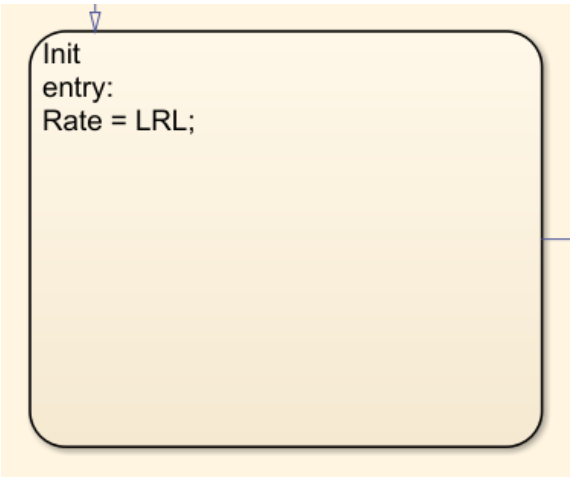
State Transition Table for Rate Modulation

Current State	Event	Next State
Init	—	Higher
Higher	Acceleration >= Threshold	Lower
	after(1000, msec)	RateDecrease

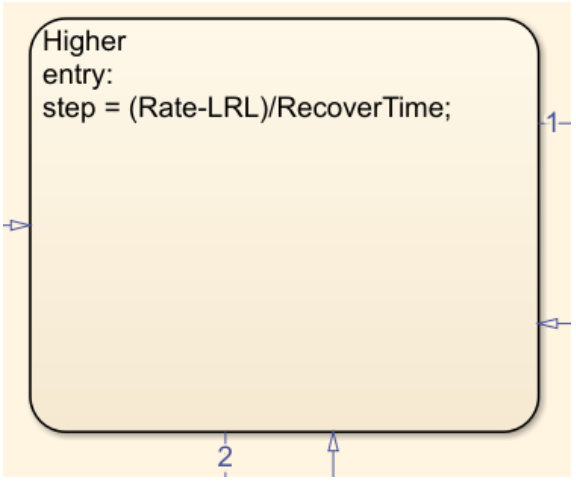
Lower	Acceleration < Threshold	Higher
	after(1000, msec)	RateIncrease
RateDecrease	—	Higher
RateIncrease	—	Lower

State Configurations:

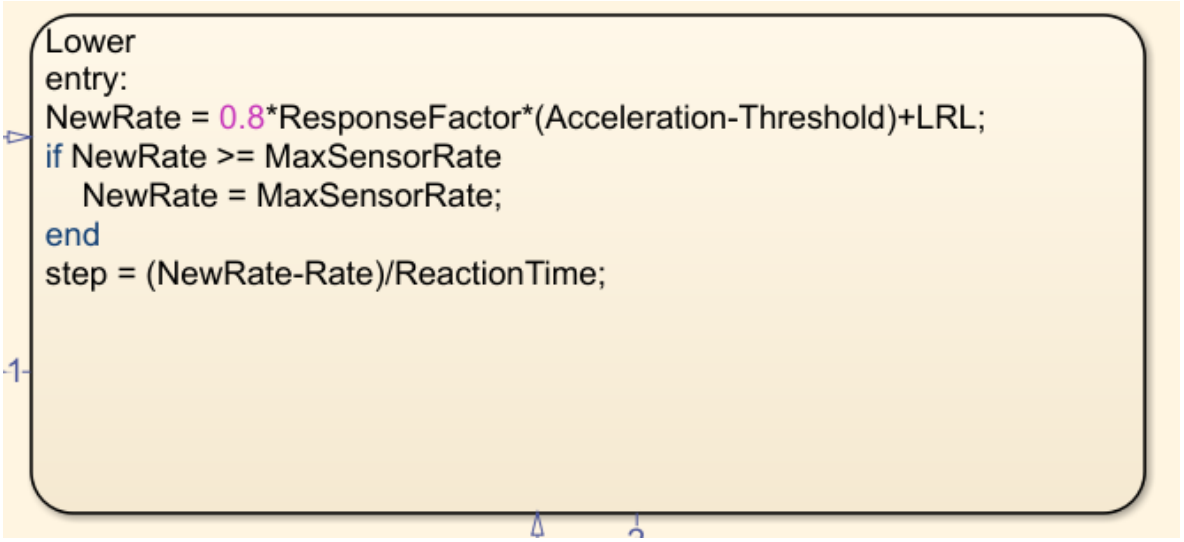
Init:



Higher:

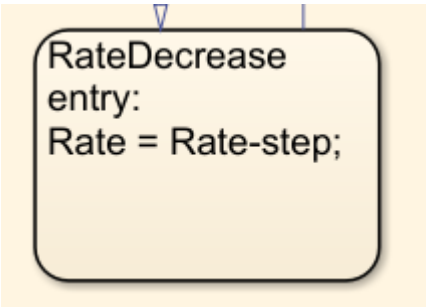


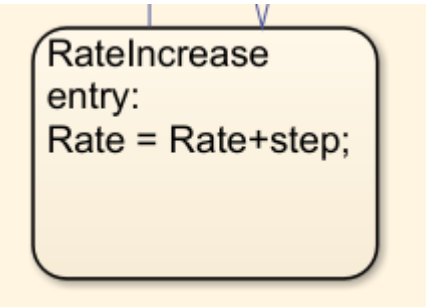
Lower:



RateDecrease:

RateIncrease:





- **Hardware Hiding**

This subsystem contains blocks for reading and writing from the pins of the FRDM-K64F microcontroller.

Inputs:

> PACE_CHARGE_CTRL, VENT_CMP_REF_PWM, Z_ATR_CTRL, PACING_REF_PWM, ATR_CMP_REF_PWM, Z_VENT_CTRL, ATR_PACE_CTRL, VENT_PACE_CTRL, PACE_GND_CTRL, ATR_GND_CTRL, VENT_GND_CTRL, FRONTEND_CTRL, RED_LED, BLUE_LED

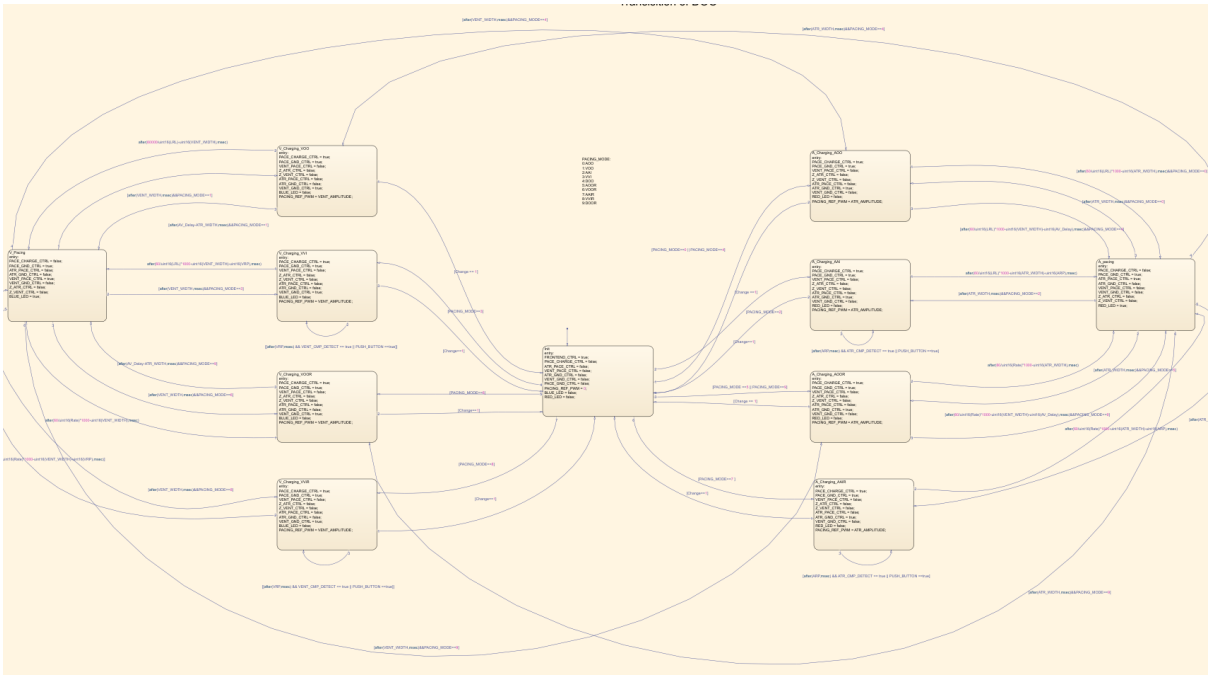
Outputs:

> ATR_CMP_DETECT, VENT_CMP_DETECT, PUSH_BUTTON, ATR_SIGNAL, VENT_SIGNAL.

Control Variables	Data Type	Pin
PACE_CHARGE_CTRL	Boolean	D2
VENT_CMP_REF_PWM	uint8	D3
Z_ATR_CTRL	Boolean	D4
PACING_REF_PWM	uint8	D5
ATR_CMP_REF_PWM	uint8	D6

Z_VENT_CTRL	Boolean	D7
ATR_PACE_CTRL	Boolean	D8
VENT_PACE_CTRL	Boolean	D9
PACE_GND_CTRL	Boolean	D10
ATR_GND_CTRL	Boolean	D11
VENT_GND_CTRL	Boolean	D12
FRONTEND_CTRL	Boolean	D13
RED_LED	Boolean	RED_LED
BLUE_LED	Boolean	BLUE_LED
ATR_CMP_DETECT	Boolean	D0
VENT_CMP_DETECT	Boolean	D1
PUSH_BUTTON	Boolean	PUSH BOTTOM
ATR_SIGNAL	double	A0
VENT_SIGNAL	double	A1

likely changes



1) In the MainFSM state, the charging states of four types of ventricle/atrium are almost the same, we could make the interface more concise by reducing and recycling repeated states, which makes it easier to read and debug.

2) We can add more interactive buttons/LEDs to find errors. For example, let the three LEDs flash at different frequencies in duifferent states, which will help us quickly recognize which stage has a problem and fix it faster.

Part 2: DCM Design

General

The main purpose for this part to is to create main application module with respect to other components of DCM that create the connection and data transmission between different modules

Requirement Description

expand more required modes for DCM

build the serial communication to connect the DCM and Pacemaker part in order to create data collection and transmission.

according to the document srsVVI rev to store and set the data meeting the requirements and limitations for Pacemaker.

the user can choose to display what they want for egram data (ventricles, atria, or both), that the DCM recessive the egram data due to the serial communication.

Parameter	Programmable Values	Increment	Nominal	Tolerance
A or V Pulse Amplitude Regulated	Off, 0.1-5.0V	0.1V	5V	± 12%
A or V Pulse Width	1-30 ms	1 ms	1 ms	1 ms
A or V Sensitivity	0-5V	0.1V	-	±2%

Parameter	AAT	VVT	AOO	AAI	VOO	VVI	VDD	DOO	DDI	DDD	AOOR	AAIR	VOOR	VVIR	VDDR	DOOR	DDIR	DDDR
Lower Rate Limit	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Upper Rate Limit	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Maximum Sensor Rate											X	X	X	X	X	X	X	X
Fixed AV Delay							X	X	X	X					X	X	X	X
Dynamic AV Delay							X			X					X			X
Sensed AV Delay Offset										X								X
Atrial Amplitude	X		X	X				X	X	X	X	X				X	X	X
Ventricular Amplitude		X			X	X	X	X	X	X			X	X	X	X	X	X
Atrial Pulse Width	X		X	X				X	X	X	X	X				X	X	X
Ventricular Pulse Width		X			X	X	X	X	X	X			X	X	X	X	X	X
Atrial Sensitivity	X			X					X	X		X					X	X
Ventricular Sensitivity		X				X	X		X	X				X	X		X	X
VRP		X				X	X		X	X				X	X		X	X
ARP	X			X					X	X		X					X	X
PVARP	X			X					X	X		X					X	X
PVARP Extension							X			X					X			X
Hysteresis				X		X				X		X		X				X
Rate Smoothing				X		X	X			X		X		X	X			X
ATR Duration							X			X					X			X
ATR Fallback Mode							X			X					X			X
ATR Fallback Time							X			X					X			X
Activity Threshold											X	X	X	X	X	X	X	X
Reaction Time											X	X	X	X	X	X	X	X
Response Factor											X	X	X	X	X	X	X	X
Recovery Time											X	X	X	X	X	X	X	X

Table 6: Programmable Parameters for Bradycardia Therapy Modes

the limitation of the parameter

A Programmable Parameters

Parameter	Programmable Values	Increment	Nominal	Tolerance
Mode	Off DDD VDD DDI DOO AOO AAI VOO VVI AAT VVT DDDR VDDR DDIR DOOR AOOR AAIR VOOR VVIR	—	DDD	—
Lower Rate Limit	30-50 ppm 50-90 ppm 90-175 ppm	5 ppm 1 ppm 5 ppm	60 ppm	±8 ms
Upper Rate Limit	50-175 ppm	5 ppm	120 ppm	±8 ms
Maximum Sensor Rate	50-175 ppm	5 ppm	120 ppm	±4ms
Fixed AV Delay	70-300 ms	10 ms	150 ms	±8 ms
Dynamic AV Delay	Off, On	—	Off	—
Minimum Dynamic AV Delay	30-100 ms	10 ms	50 ms	
Sensed AV Delay Offset	Off, -10 to -100 ms	-10 ms	Off	±1 ms
A or V Pulse Amplitude Regulated	Off, 0.5-3.2V 3.5-7.0 V	0.1V 0.5V	3.5V	±12%
A or V Pulse Amplitude Unregulated	Off, 1.25, 2.5, 3.75, 5.0V	—	3.75V	—
A or V Pulse Width	0.05 ms 0.1-1.9 ms	— 0.1 ms	0.4 ms	0.2 ms
A or V Sensitivity	0.25, 0.5, 0.75 1.0-10 mV	— 0.5 mV	A-0.75 mV V-2.5 mV	±20%
Ventricular Refractory Period	150-500 ms	10 ms	320 ms	±8 ms
Atrial Refractory Period	150-500 ms	10 ms	250 ms	±8 ms
PVARP	150-500 ms	10 ms	250 ms	±8 ms
PVARP Extension	Off, 50-400 ms	50 ms	Off	±8 ms
Hysteresis Rate Limit	Off or same choices as LRL	—	Off	±8 ms
Rate Smoothing	Off, 3, 6, 9, 12, 15, 18, 21, 25%	—	Off	±1%
ATR Mode	On, Off	—	Off	—
ATR Duration	10 cardiac cycles 20-80 cc 100-2000 cc	— 20 cc 100 cc	20 cc	±1 cc
ATR Fallback Time	1-5 min	1 min	1 min	±1 cc
Ventricular Blanking	30-60 ms	10 ms	40 ms	—
Activity Threshold	V-Low, Low, Med-Low, Med, Med-High, High, V-High	—	Med	—
Reaction Time	10-50 sec	10 sec	30 sec	±3 sec
Response Factor	1-16	1	8	—
Recovery Time	2-16 min	1 min	5 min	±30 sec

Table 7: Programmable Parameters

The project parameters follow the limitation that the number of the input of each parameter cannot be written out of the boundary with respect to the Programmable Parameter table which verifies in the test cases that can only enter the number within the boundary of parameter.

Likely changes

1. Update and beautify the interface.
2. Write the program more neatly.
3. Comment more clear during programming.
4. Write the program structure more standardized.

Function Descriptions

❖ Login

Module: **Login & Sign-up window**

Description: Create a login window and allow users to register and login as an existing user. Auto form a sample data set for new user

A welcome screen that contains “sign-in” and “sign-up” parts for users.

public function:

Function Name	Parameters	Function Descriptions
main_window()	——	communicate with the pacemaker to distinguish which ports do pacemaker connect with the computer
sample_para()	_	Write format of the parameter of AOO","VOO","AAI","VVI","DOO","AOOR","VOOR","AAIR","VVIR","DOOR separately in parameters.txt file
get_information()	-	Get user’s input for login info
sign_up_get_info()	-	Get information from v_usr and v_pwd
sign_up()	-	get() data from name and pwd

registerUser(uName, p1, p2)	Input new username, new password and confirmed password	Register new users by checking the repeated usernames and if two typed passwords match.
loginGo(uName, Pass)	Get input of the username and password	Check the condition for going to the Pacemaker window
sign_up()	-	Create a sign-up window for new users

Black Box Behaviour

Function Name	Parameters	Description
registerUser(uName, p1, p2)	New username, new password and confirmed password	Creates new users in the database and stores the data in the database. Returns nothing
loginGo(uName, Pass)	Username, password	Check whether the input user name and password exist and match in the database. If the input is valid, go to the Pacemaker window. If not, pop up a warning window.

Global Variables

Variable Name	Parameters
name	username for the user
pwd	password that the user set
window	the window interface level
v_usr	value stored for username
v_pwd	value stored for password

main window

Module: Pacemaker Main Screen

Description: Create a window for the pacemaker that can choose to connect and disconnect the device. Display the basic function for

the pacemaker. Mode selection is available when the port is connected

public function

Function Name	Parameters	Function Descriptions
pacemaker()	-	Create a window for first interface include the main buttons for AOO","VOO","AAI","VVI","DOO","AOOR","VOOR","AAIR","VVIR","DOOR the connect situation, user name, current device, and port name for the connection
connexion()	-	check the situation of connection, if the connection is success or not
sign_out()	-	sign out the current user, go back to the login window
connect()	-	transmit the connection situation between pacemaker and computer
disconnect()	-	Disconnect between the pacemaker and computer

Black Box Behaviour

Function Name	**Input variables**	**Description**
data_up_date()	Based on the different pacing mode	Save data into the database. Return nothing.

module internal design

Global Variables

Variable Name	Description
window_pc	level of interface window for pacemaker screen
connect_bt	state for connection(active or not)
refresh	state for connection (active or not)
drop_COM	state of port(active or not)
clicked_com	store the selected port
ser	serial for the device

Pacemaker Screen:

Module: Pacemaker functions

Description: Basic pacemaker functions that allow users to set different modes and display the egram and user stored data. Limits are set for each Spinbox.

To display the device status and different mode options

Function Name	`Parameters`	Description
verify()	name	Display currently stored user data
updateDataGet()	name	Locate user position in parameter.txt
parameter_AOO()	name	If the child window is connected to the pacemaker window's AOO button which enters the value of lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_AOO_get_info()		It is the definition inside parameter_AOO(). Get entered value for lower rate limit, Upper Rate Limit, Atrial

		Amplitude, and Atrial Pulse Width.
data_update_aoo()	name,v_lrl, v_url, v_aa, v_apw current username value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width	Inside the parameter_AOO() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width inside the parameter.txt file
parameter_VOO()	name	If the child window is connected to the pacemaker window's VOO button which enters the value of lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_VOO_get_ info()	——	It is the definition inside parameter_VOO(). Get entered value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
data_update_voo()	name,v_lrl, v_url, v_aa, v_apw current username value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width	Inside the parameter_VOO() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width inside the parameter.txt file
parameter_AAI()	name	If the child window is connected to the pacemaker window's AAI button which enters the value of lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_AAI_get_ info()	——	It is the definition inside parameter_AAI(). Get entered value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.

data_update_aai()	(name, v_lrl, v_url, v_va, v_vpw,v_s,v_vrp,v_hy,v_rs) current username value for lower rate limit,Upper Rate Limit, Atrial Amplitude, and Atrial Pulse, Width, VRP,Hysteresis,Rate Smoothing	Inside the parameter_AAI() which can save value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width, VRP, Hysteresis, Rate Smoothing inside the parameter.txt file
parameter_VVI()	name	It is the definition inside parameter_VVI(). Get entered value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_VVI_get_info()	——	Inside the parameter_AOO() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width inside the parameter.txt file
data_up_vii()	(name, v_lrl, v_url, v_va, v_vpw,v_s,v_vrp,v_hy,v_rs) current username value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse, Width, VRP, Hysteresis, Rate Smoothing	Inside the parameter_VVI() which can save value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse, Width, VRP, Hysteresis, Rate Smoothing inside the parameter.txt file
parameter_DOO()	name	If the child window is connected to the pacemaker window's DOO() button which enters the value of lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_DOO_get_info()		It is the definition inside parameter_DOO(). Get entered value for lower rate limit, Upper Rate Limit, Atrial

		Amplitude, and Atrial Pulse Width.
data_update_doo()	name,lrl, url,maxSensorRate, entryed_aa, apw,activityThreshold, reactionTime, responseFactor,recoveryTime	Inside the parameter_DOO() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width inside the parameter.txt file
parameter_AOOR()	name	If the child window is connected to the pacemaker window's AOOR() button which enters the value of lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_AOOR_get_info()		It is the definition inside parameter_AOOR(). Get entered value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
data_upadate_Aoor()	name,lrl, url,maxSensorRate, va, vpw,activityThreshold, reactionTime,entryed_responseFactor,entryed_recoveryTime	Inside the parameter_AOOR() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width inside the parameter.txt file
parameter_AAIR()	name	If the child window is connected to the pacemaker window's AAIR() button which enters the value of lower rate limit, Upper Rate
parameter_AAIR_get_info()		It is the definition inside parameter_AAIR(). Get entered value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
data_upadate_AAIR()	name,lrl, url,maxSensorRate, va, vpw,activityThreshold, reactionTime,entryed_responseFactor,entryed_recoveryTime	Inside the parameter_AAIR() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse

		Width inside the parameter.txt file
parameter_VVIR()	name	If the child window is connected to the pacemaker window's VVIR() button which enters the value of lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_VVIR_get_info()		It is the definition inside parameter_VVIR(). Get entered value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
data_upadate_VVIR()	name,lrl, url,maxSensorRate, va, vpw,activityThreshold, reactionTime,entryed_responseFactor,entryed_recoveryTime	Inside the parameter_VVIR() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width inside the parameter.txt file
parameter_DOOR()	name	If the child window is connected to the pacemaker window's DOOR() button which enters the value of lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
parameter_DOOR_get_info()		It is the definition inside parameter_DOOR(). Get entered value for lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width.
data_upadate_DOOR()	name,lrl, url,maxSensorRate, va, vpw,activityThreshold, reactionTime,entryed_responseFactor,entryed_recoveryTime	Inside the parameter_DOOR() which can save lower rate limit, Upper Rate Limit, Atrial Amplitude, and Atrial Pulse Width inside the parameter.txt file

global variable

Variable Name	Description
window_pc	level of interface window for pacemaker parameter screen
connect_bt	level of this window for checking the connection state
refresh	state for connection active or not
drop_COM	state of port(active or not)
clicked_com	state of port(active or not)

Serial Communication

Module: Serial Communication

Description: Serial communication between the DCM and Pacemaker. Allow the user to pass the set data to the pacemaker and read data out from the Pacemaker

Public Function

Function Name	Parameters	Function Descriptions
inputPass()	name	ready-to-be-used data. Construct the stored data with the corresponding user into a list for further usage
serialC()	-	Start the serial communication, pack and unpack the data, determine the mode that selected to pass data, set ready-to-be-used data to the corresponding variables
parameterReset()		Reset all the parameters that entered
checkDeviceInfo()		check the information of device pacemaker

connect()		serial communication connect between pacemaker and computer
disconnect()		disconnect pacemaker and computer
grab_data()		get data
animate()	i	Function used to graph out collected data
GRAPH_Mode_Modifier()		A function used to consistently graph natural pacing
egram()		create egram according to the param setting

module internal design

global variable

Variable Name	Description
myListGet	A list that contains the current user's info
pacing_mode	A variable that indicates the mode selected
ser	serial.Serial()

egram

Function Name	Parameters	Function Descriptions
egram	-	display the egram

Private function

Function Name	Parameters
__name__ == '__main__'	module that is being run is the main

	program

Important library

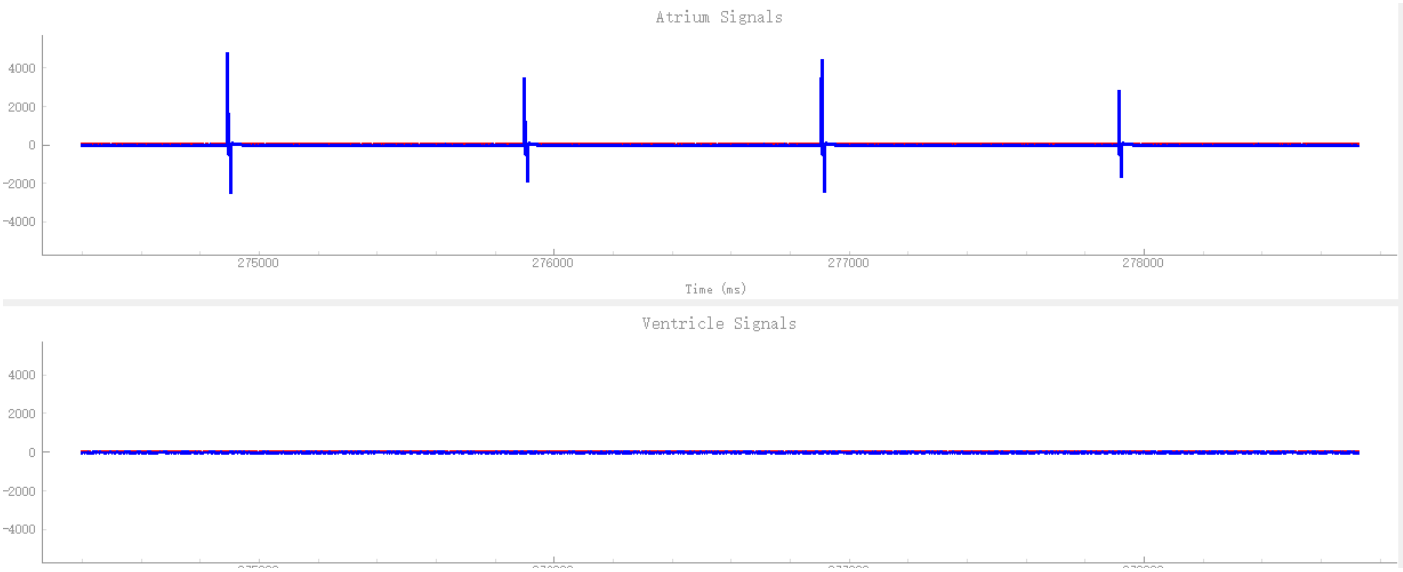
Tkinter library

Tkinter is a standard GUI library to create an interface that is convenient to combine the python interface and GUI toolkit application. It applies an object-oriented interface to the Tk GUI toolkit.

Part 3: TESTCASE

Simulink Testcases

1. **AOO**



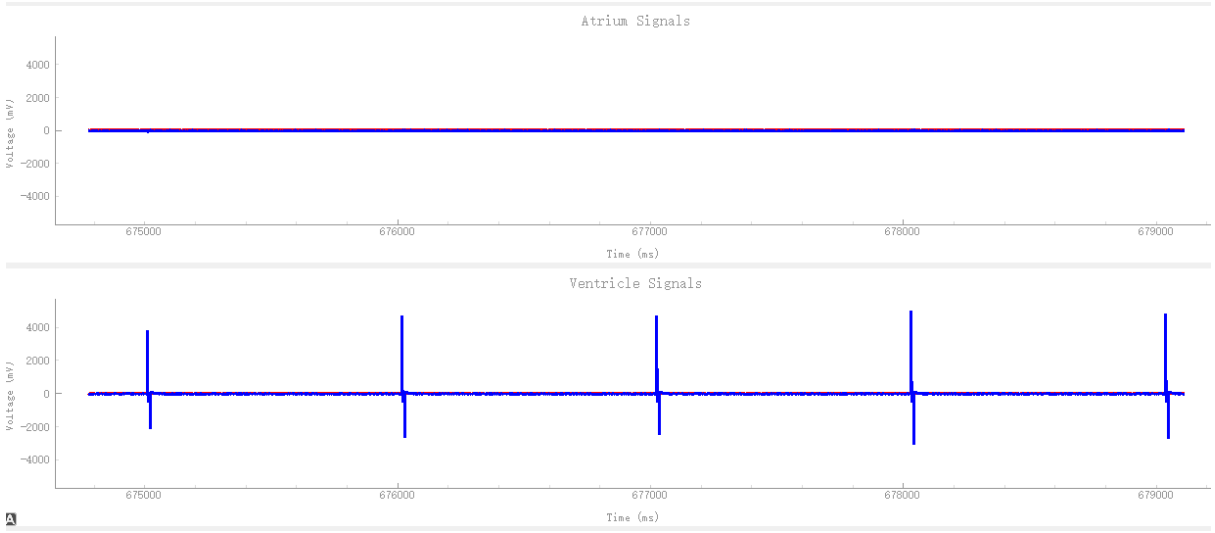
input LRL = 60

expect BPM = 60

AV_Delay =120 ms

Actual atrium BPM = 60

2. **VOO**



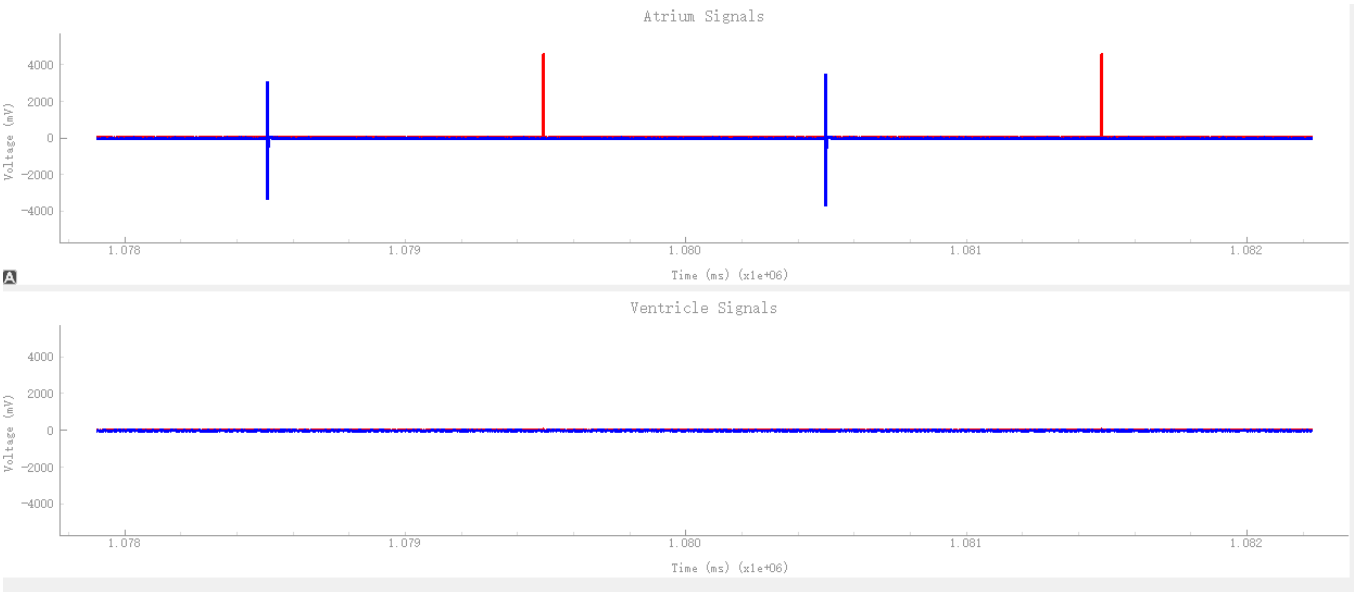
input LRL = 60

expect BPM = 60

AV_Delay =120 ms

Actual Ventricle BPM = 60

3. AAI



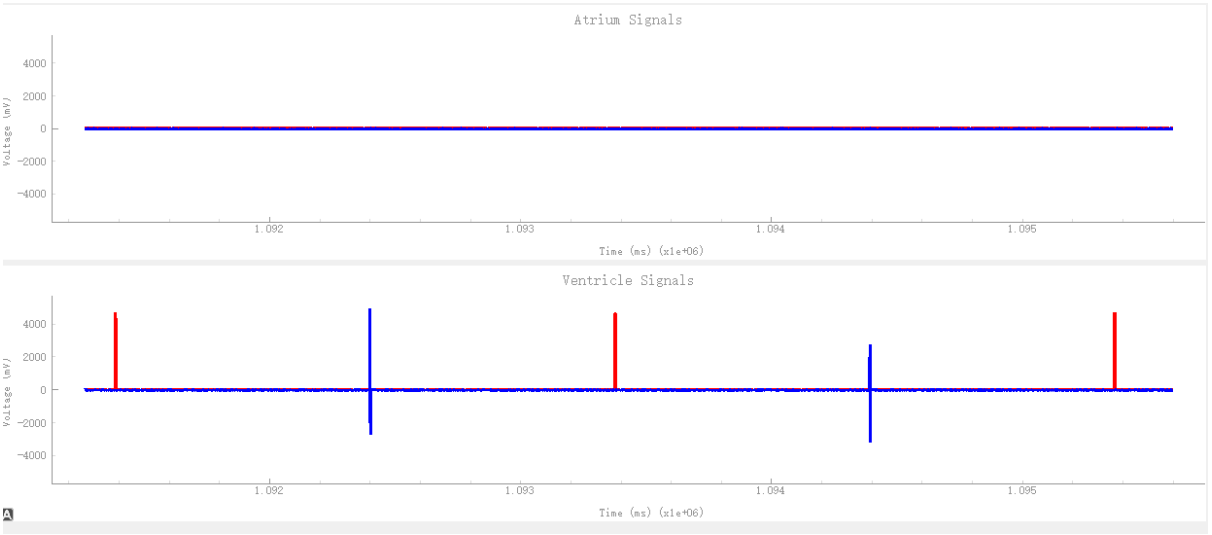
input LRL = 60

expect BPM = 60

AV_Delay =120 ms

Actual Atrium BPM = 60

4. **VVI**



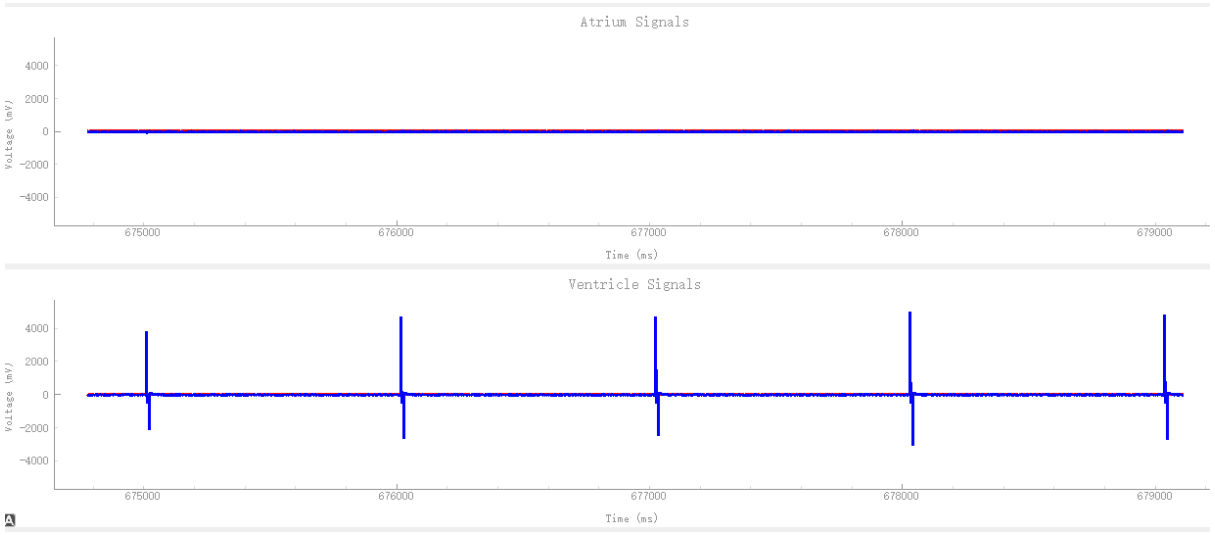
input LRL = 60

expect BPM = 60

AV_Delay =120 ms

Actual Ventricle BPM = 60

5. **DOO**



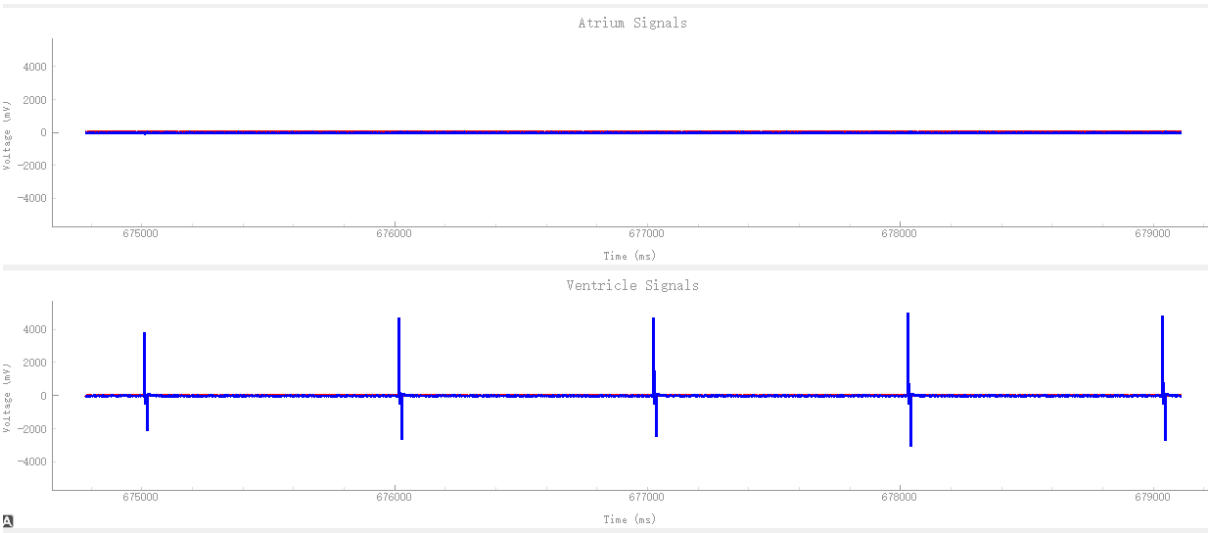
input LRL = 60

expect BPM = 60

AV_Delay =120 ms

Actual BPM = 60

6. **AOOR**



input LRL = 60 and shaking the pacemaker

expect BPM > 60

AV_Delay =120 ms

Actual BPM > 60

Test	expect output	actual output	result
AOO	atruim BPM = 60	atruim BPM = 60	Pass
VOO	Ventricle BPM = 60	Ventricle BPM = 60	Pass
AAI	atruim BPM = 60	atruim BPM = 60	Pass
VVI	Ventricle BPM = 60	Ventricle BPM = 60	Pass
DOO	both BPM = 60	Both BPM = 60	Pass
AOOR	BPM>60	BPM>60	Pass

DCM Testcases

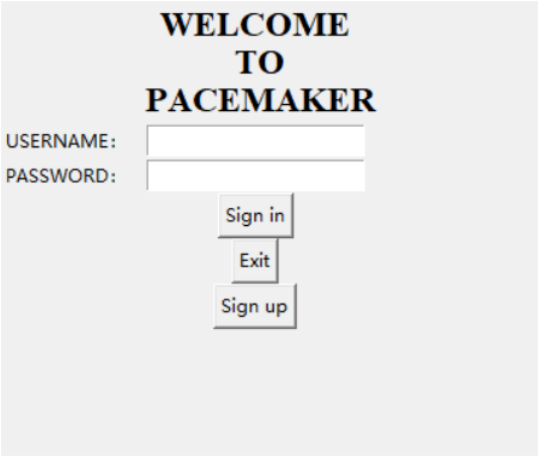
Sign up testing

Test case 1:for login window screen

input: run the code

expected output: display the login window screen

result: pass

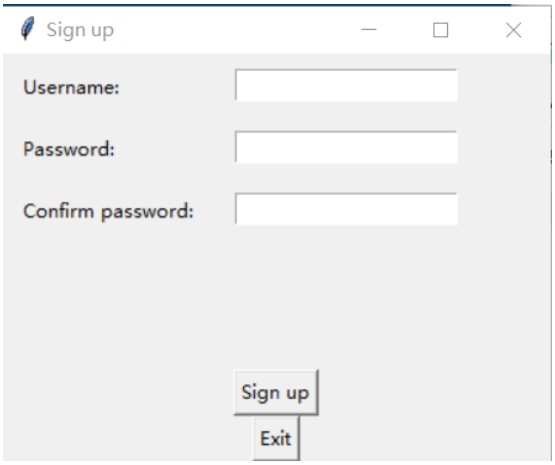


Test case 2:for signup screen

input: press sign up button

expected output: display for signup screen

result: pass



Test case 3:

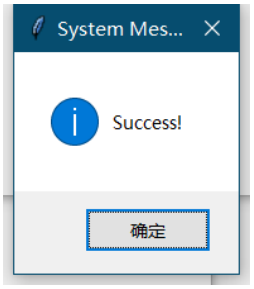
Input

username	abcd
password	1234
confirm password	1234

expected output:success!

Output: success

result: pass



Test case 4:

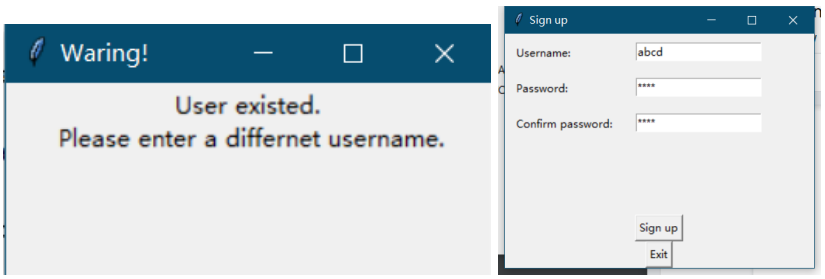
input sign up same username and password again

username	abcd
password	1234
confirm password	1234

expected output: warning , User existed. please enter a different username.

Output: warning User existed. please enter a different username.

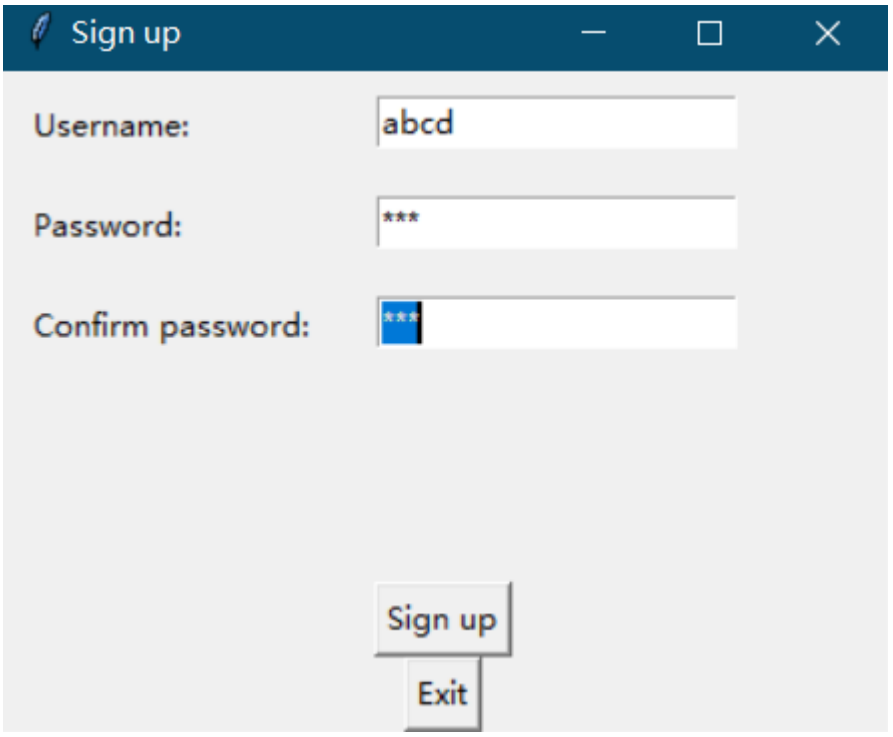
result: pass



Test case 5:

Input sign up the same user name but a different password

username	abcd
password	123
confirm password	123



Expected Output:warning: User existed. Please enter a different username.

Output: warning User existed. please enter a different username.

result: pass

Test case 6 :

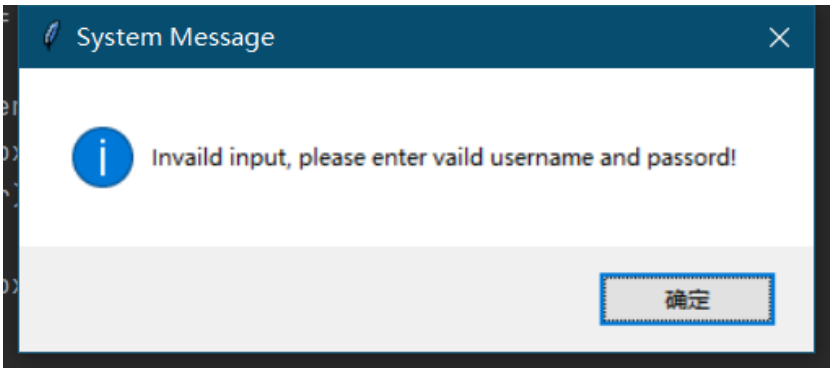
input:

username	
password	
confirm password	

Expected Output: Warning for “Invalid input, please enter valid username and password!”

Output: Warning for “Invalid input, please enter valid username and password!”

result: pass



Testcase 7

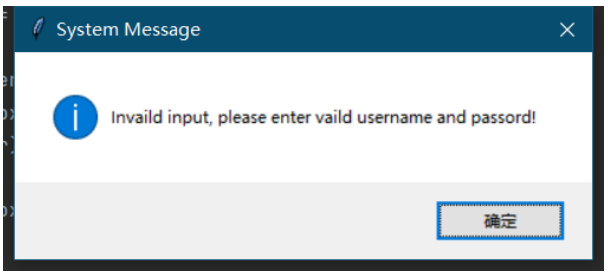
input:

username	
password	123e
confirm password	123

expected output:Warning for “Invalid input, please enter valid username and password!”

Output: Warning for “Invalid input, please enter valid username and password!”

result: pass



Testcase 8

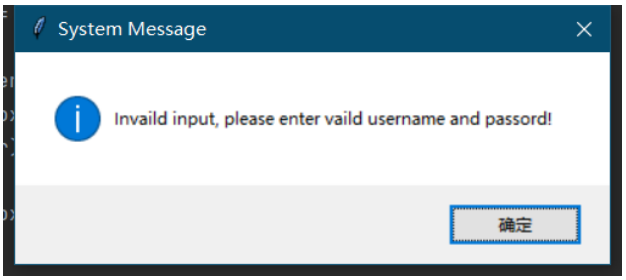
input:

username	
password	1234a
confirm password	

expected output:Warning for “Invalid input, please enter valid username and password!”

Output: Warning for “Invalid input, please enter valid username and password!”

result: pass



Testcase 9

Input: Signup with a different user name but the same password as before

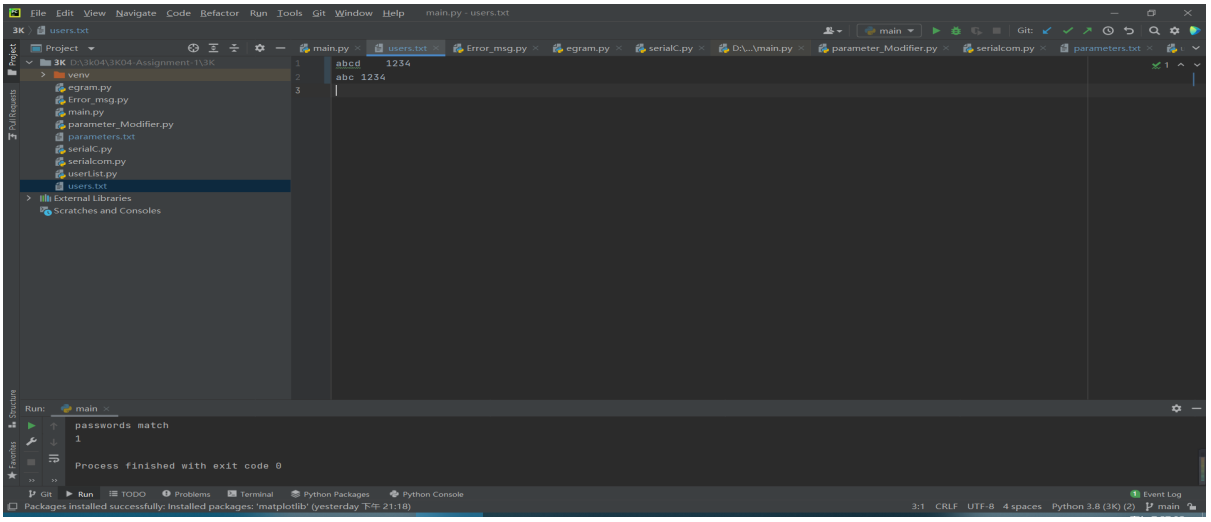
username	abc
password	1234
confirm password	1234

expected output: “Success” message window

Output: “Success” message window

result: pass

After Testcase 1-9 the output for users.txt:



Testcase 10

input:

username	a
password	123
confirm password	123

Expected Output: “Success” message window

Output: “Success” message window

result: pass

Testcase 11

input

username	b
password	12
confirm password	12

expected output: “Success” message window

Output: “Success” message window

result: pass

Testcase 12

input:

username	123
password	abc1
confirm password	abc1

expected output: “Success” message window

Output: “Success” message window

result: pass

Module: Pacemaker Main Screen

Testcase 13

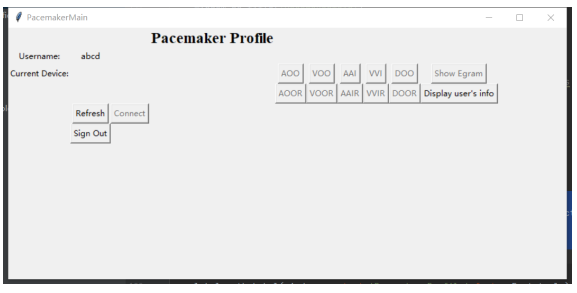
Input and then press sign in

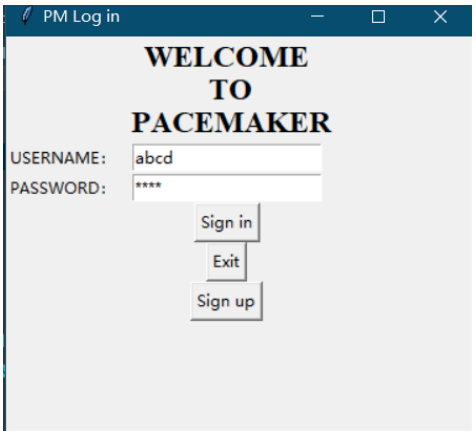
username	abcb
password	1234

expected output:Success to enter pacemaker main screen(Pacemaker Profile)

Output: Success to enter pacemaker main screen(Pacemaker Profile)

result: pass

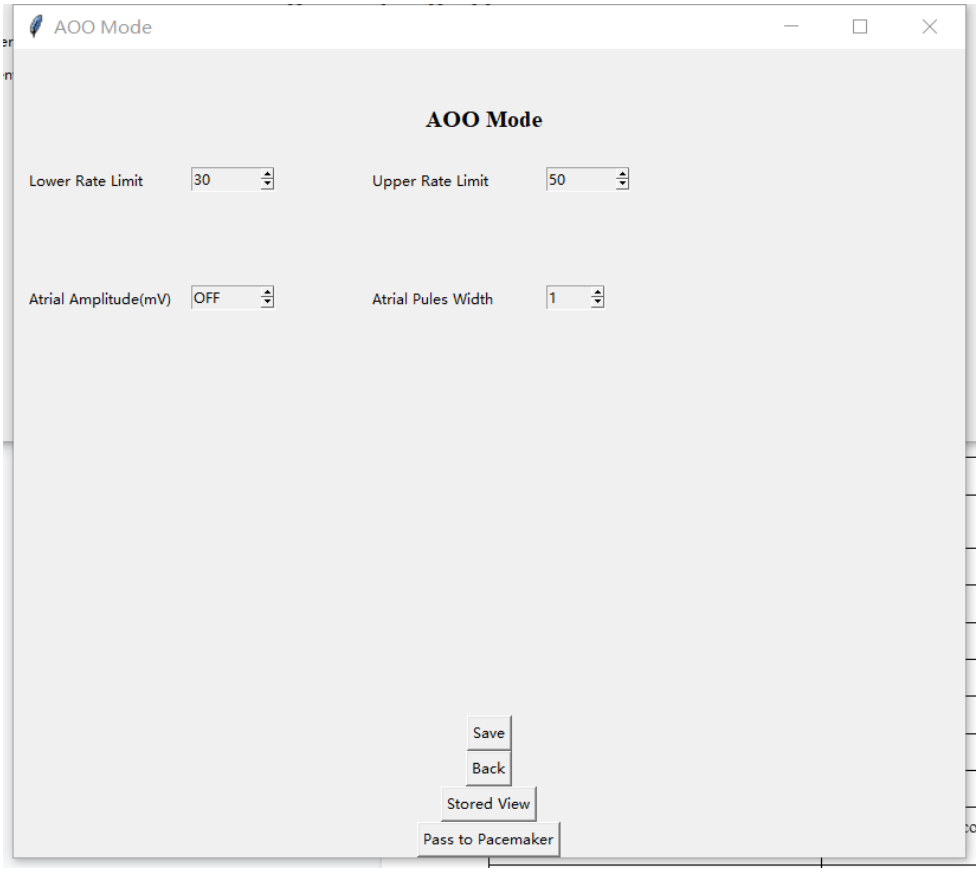




Module: Pacemaker functions

Testcase 14

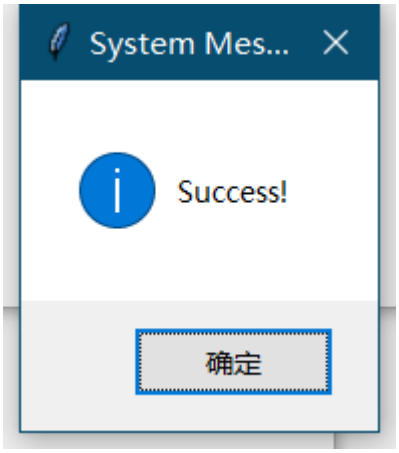
Input: The lower limit of the AOO data mode(An entry box that can only set a certain range)



Expected output: “success” window

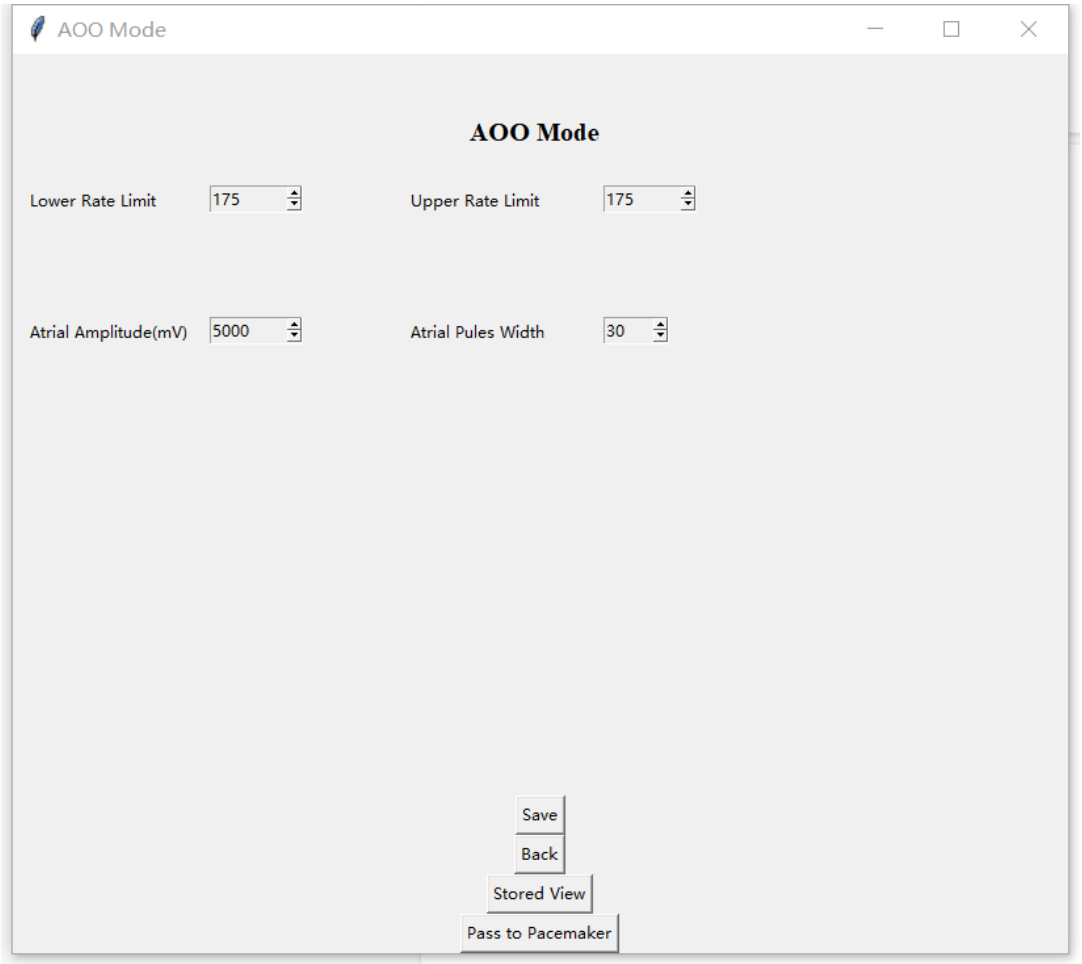
Output: “success” window

result: pass



Testcase 15

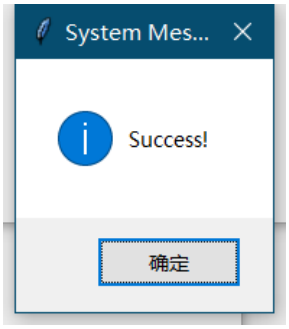
Input: The upper limit of the AOO mode(An entry box that can only set a certain range)



Expected output: “success” window

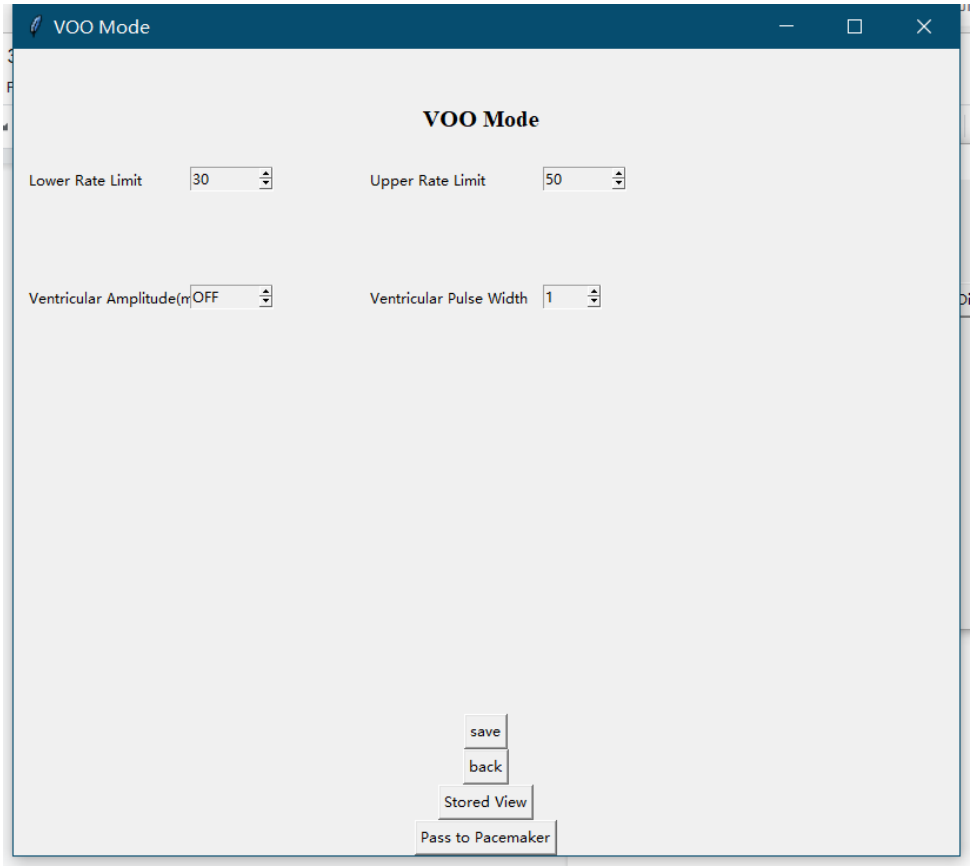
Output: “success” window

result: pass



Testcase 16

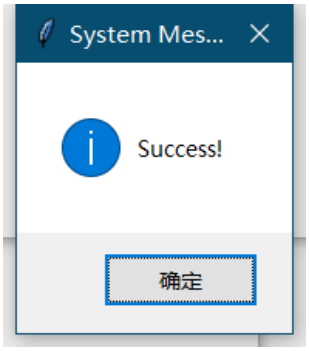
Input: The lower limit of VOO Mode (An entry box that can only set a certain range)



Expected output: “success” window

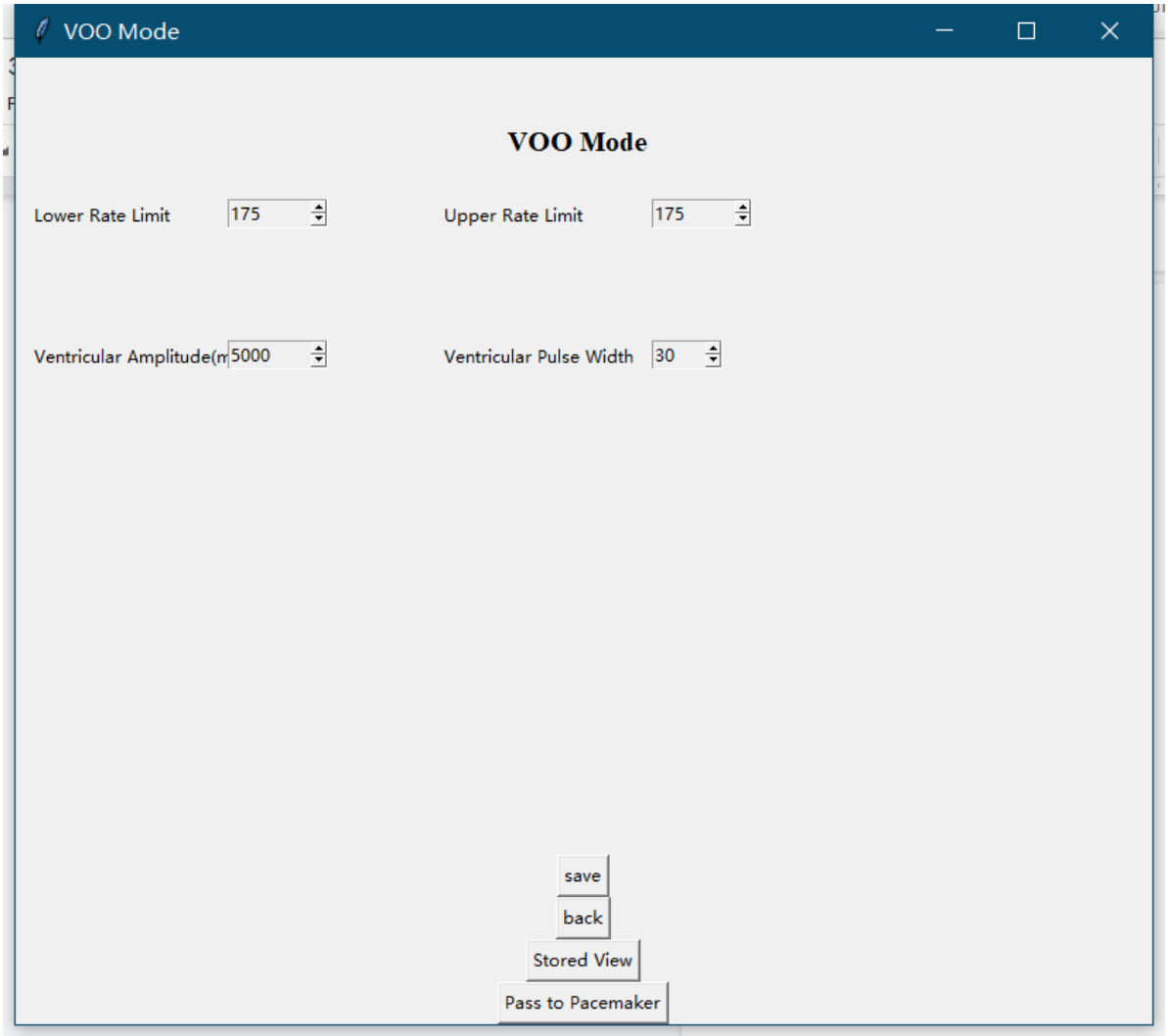
Output: “success” window

result: pass



Testcase 17:

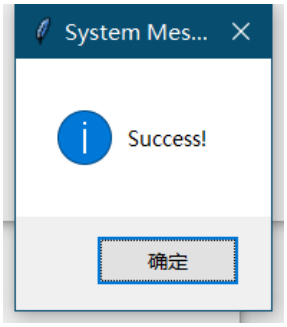
Input: The upper limit of VOO Mode (An entry box that can only set a certain range)



Expected output: “success” window

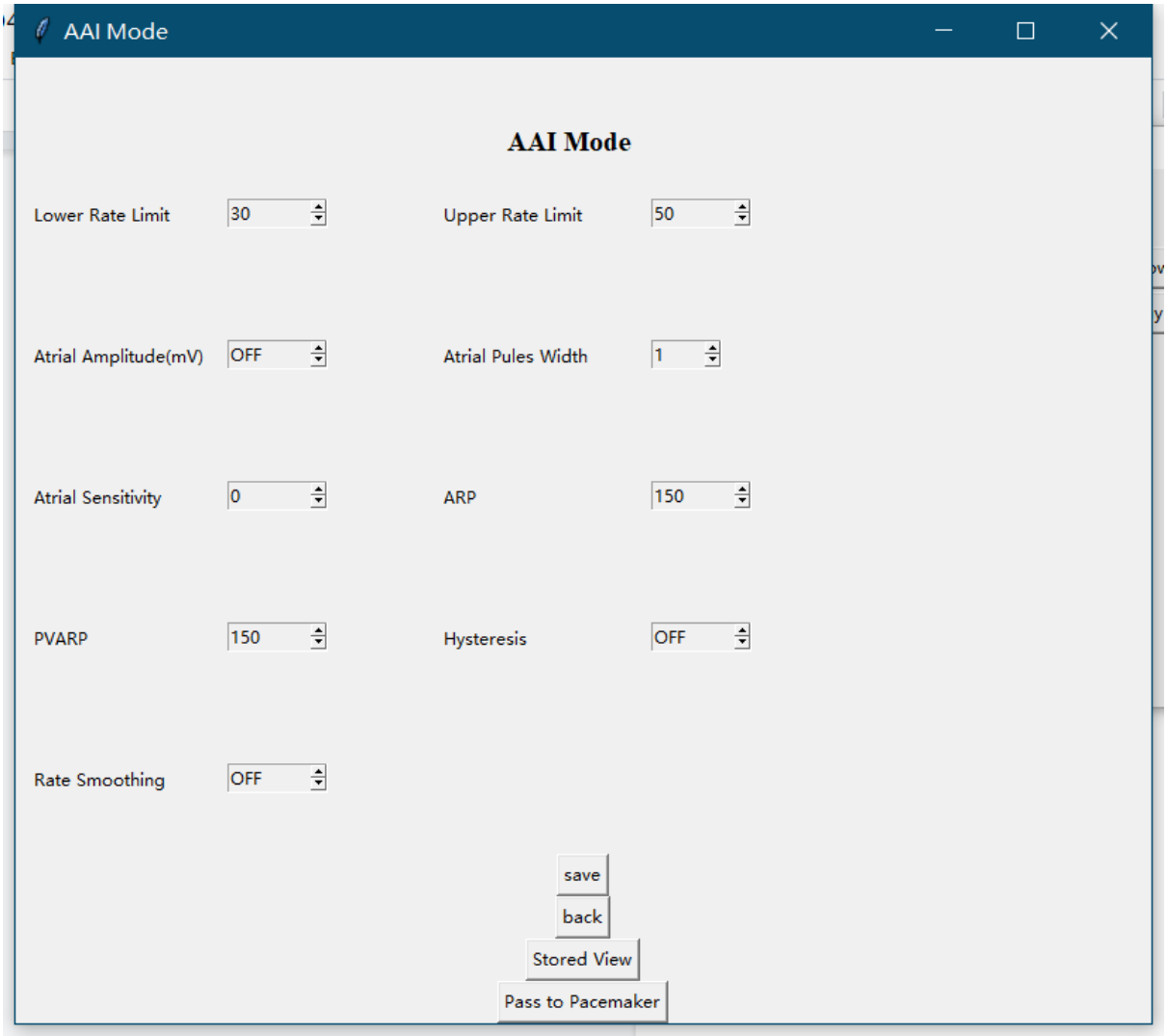
Output: “success” window

result: pass



Testcase 18:

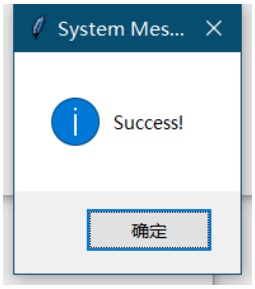
Input: Lower rate limit of AAI Mode (An entry box that can only set a certain range)



Expected output: “success” window

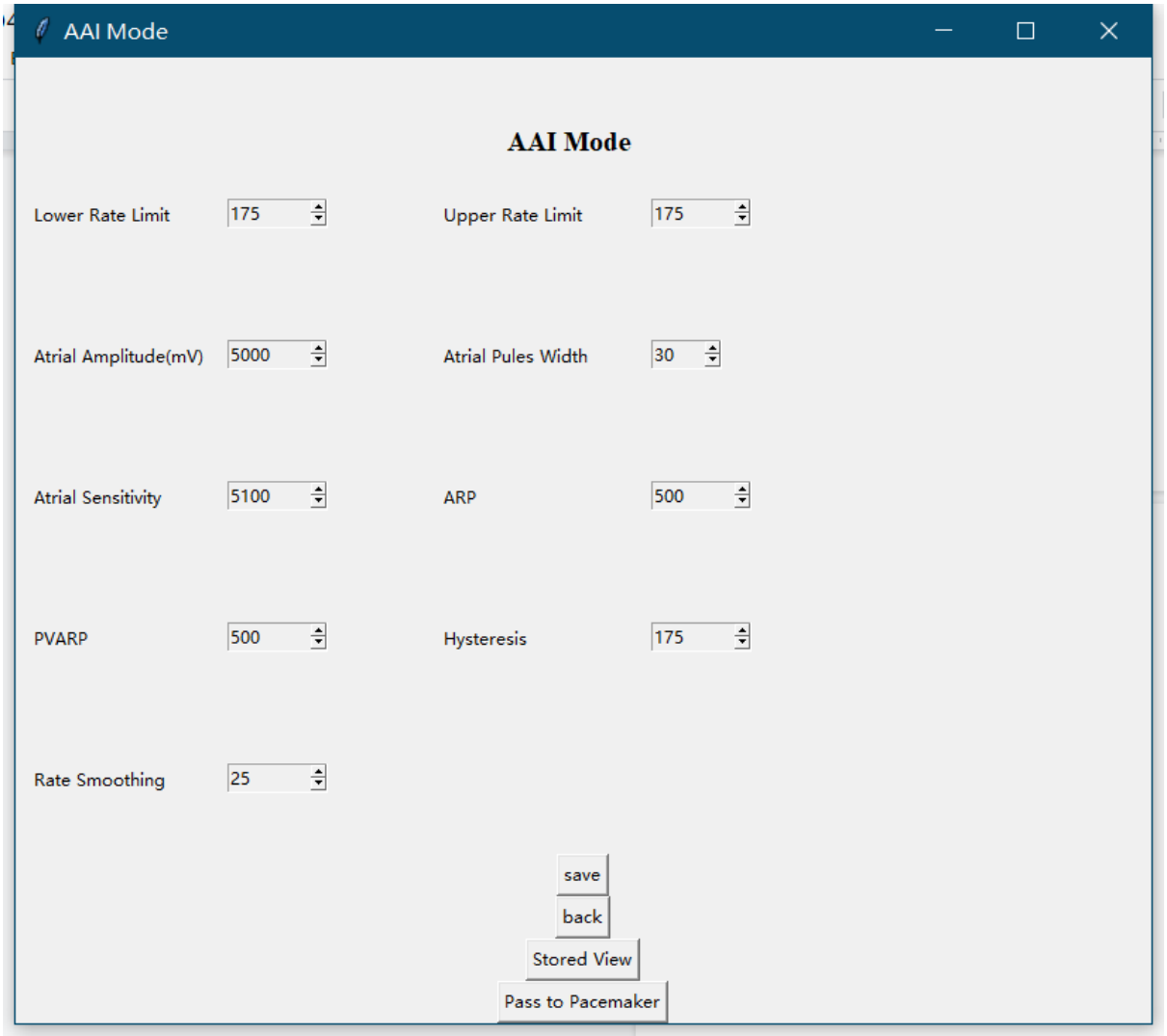
Output: “success” window

result: pass



Testcase 19:

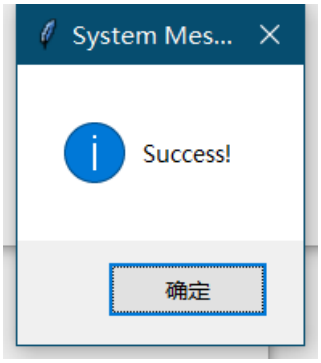
Input: Upper rate limit of AAI Mode (An entry box that can only set a certain range)



Expected output: “success” window

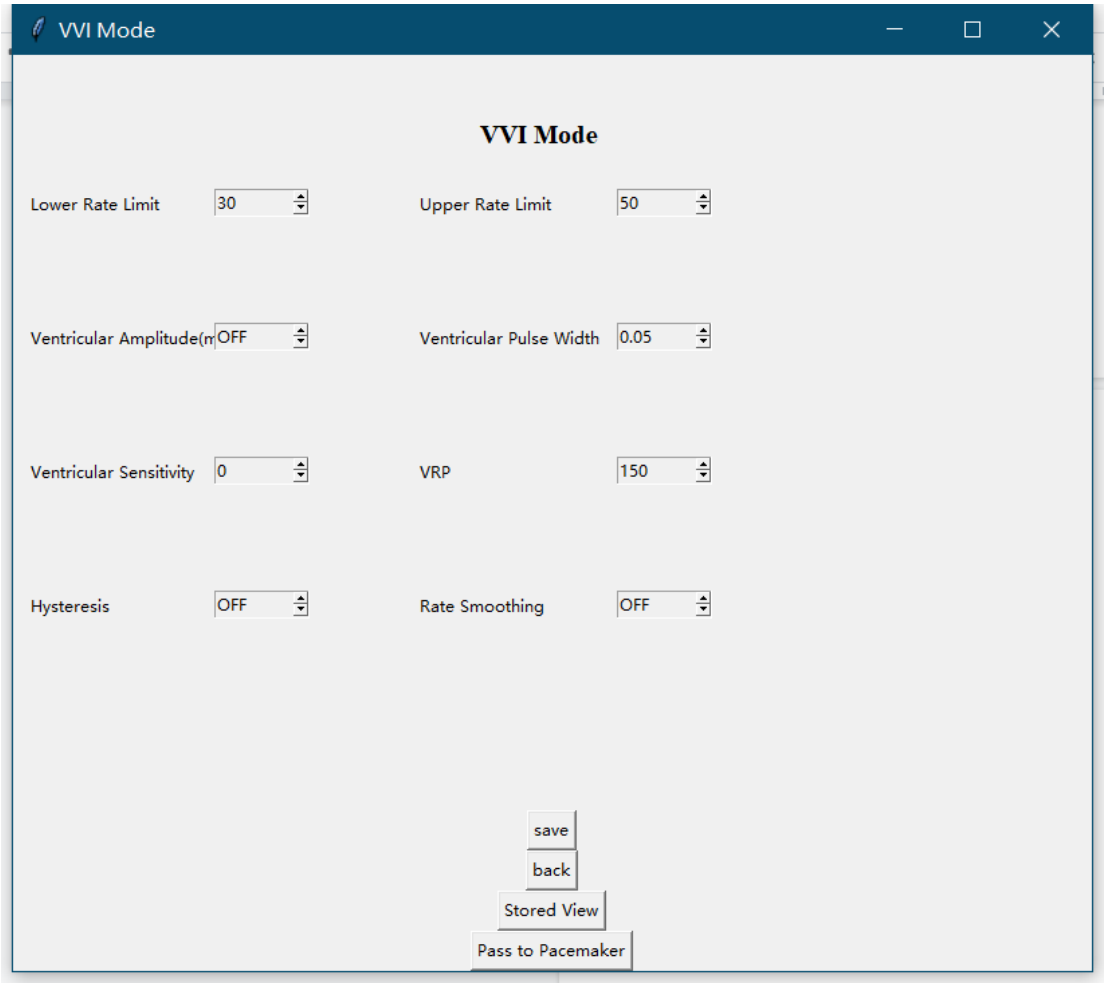
Output: “success” window

result: pass



Testcase 20:

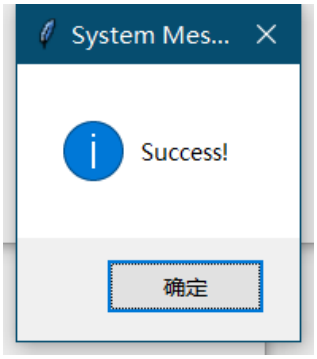
Input: Lower limit of VVI Mode (An entry box that can only set a certain range)



Expected output: “success” window

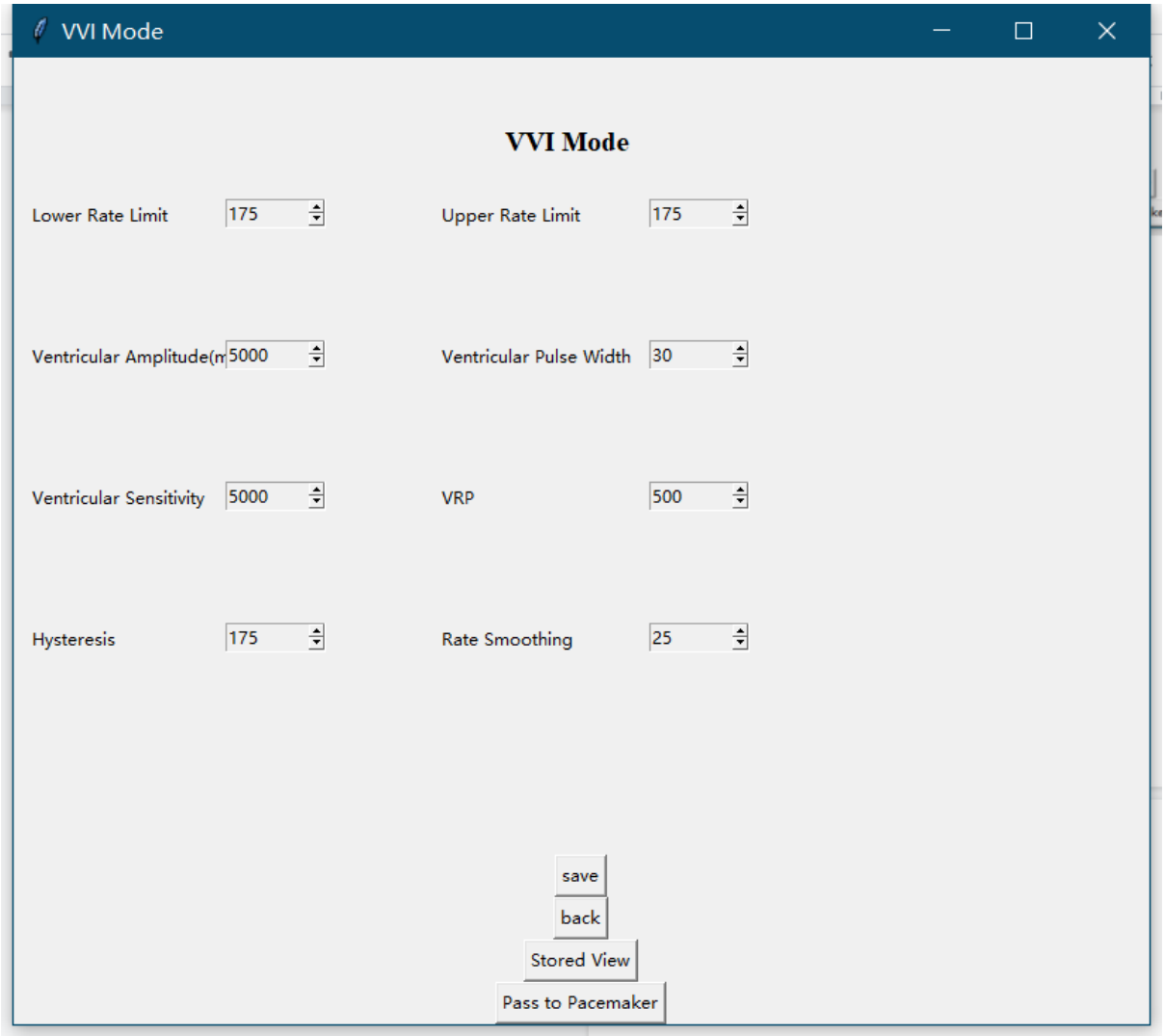
Output: “success” window

result: pass



Testcase 21:

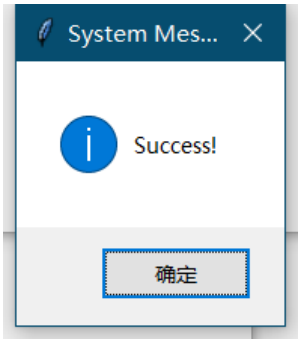
Input: upper limit of VVI Mode (An entry box that can be only set within a certain range)



Expected output: “success” window

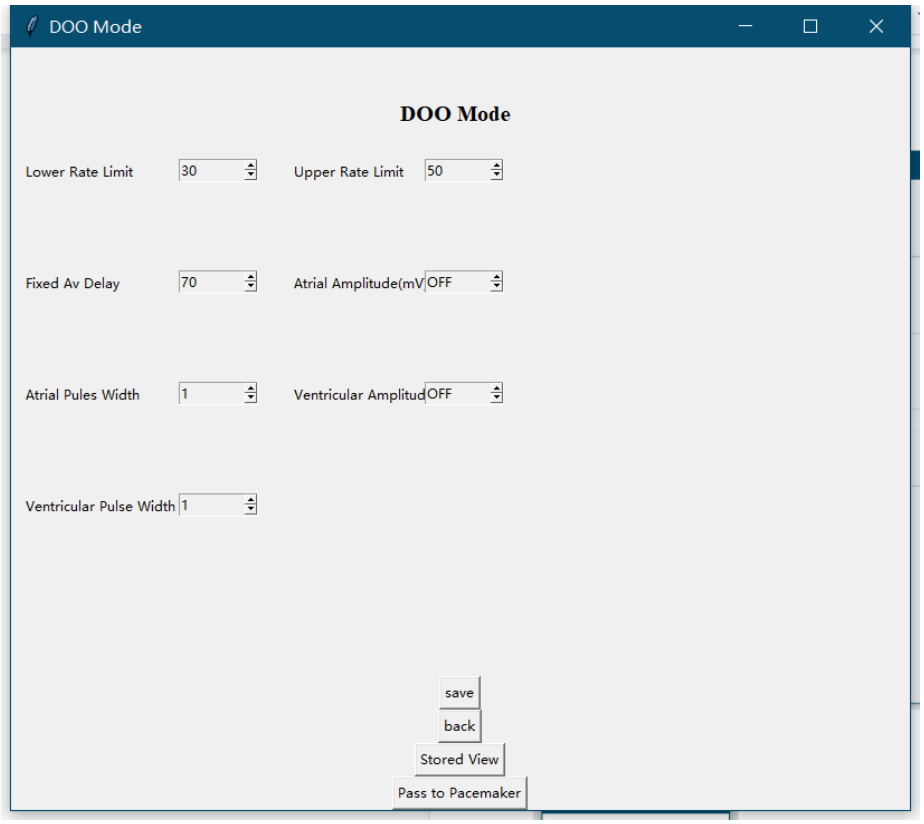
Output: “success” window

result: pass



Testcase 22:

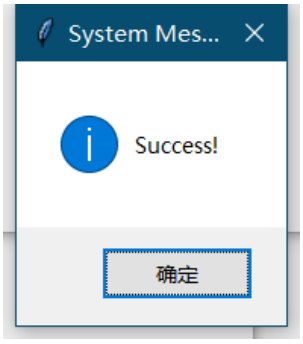
Input: lower limit of DOO Mode (An entry box that can be only set within a certain range)



Expected output: “success” window

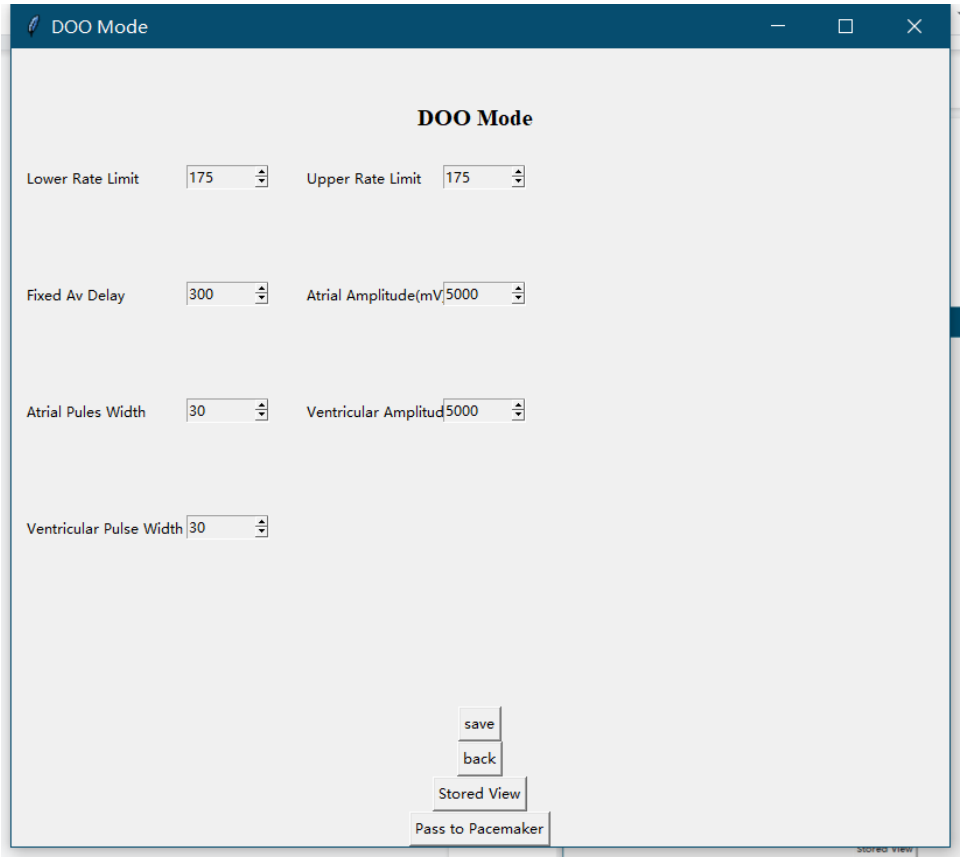
Output: “success” window

result: pass



Testcase 23:

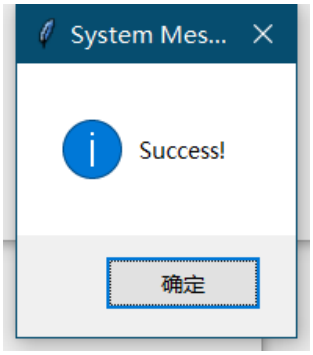
Input: Upper limit for DOO Mode (An entry box that can be only set within a certain range)



Expected output: “success” window

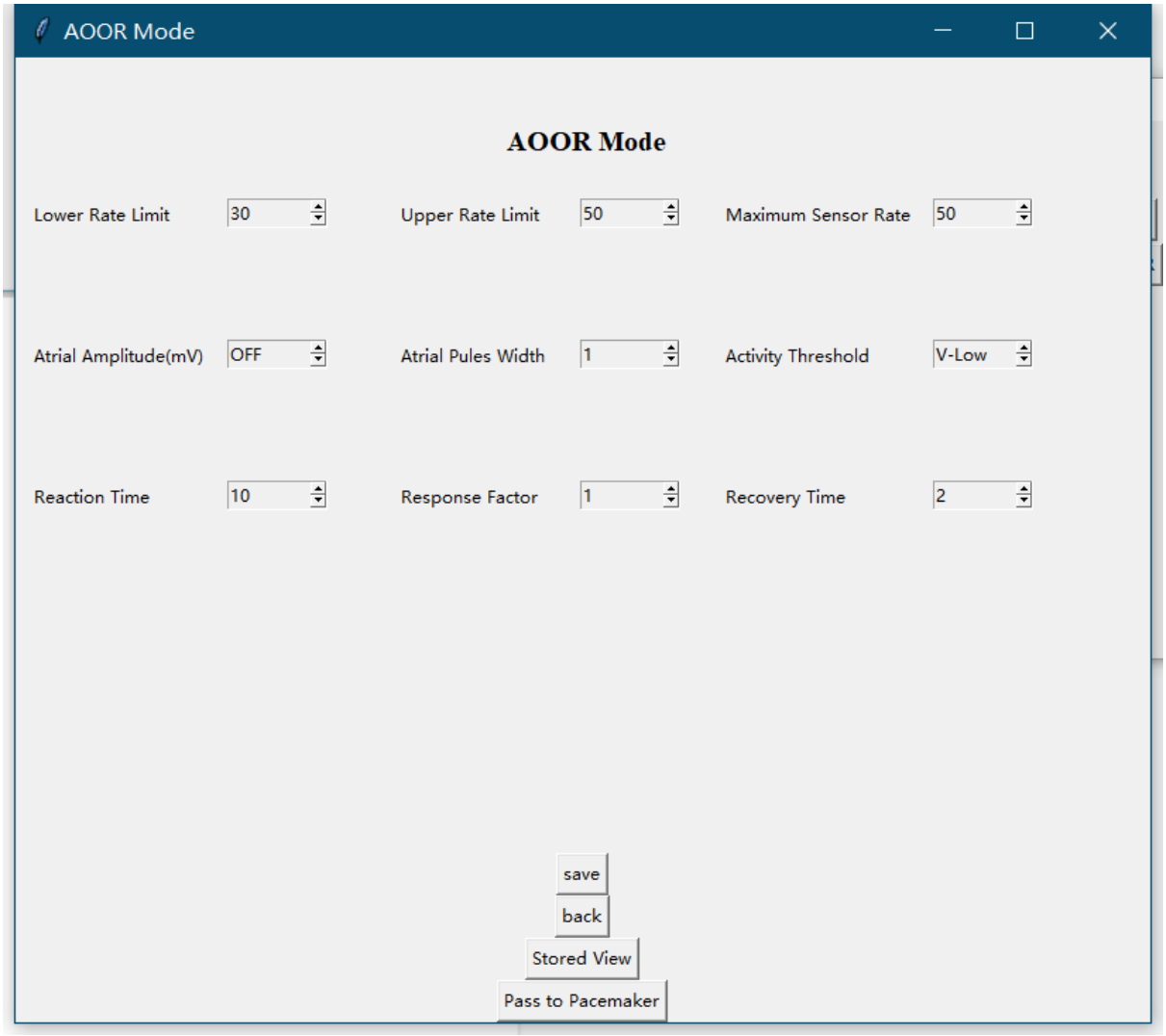
Output: “success” window

result: pass



Testcase 24:

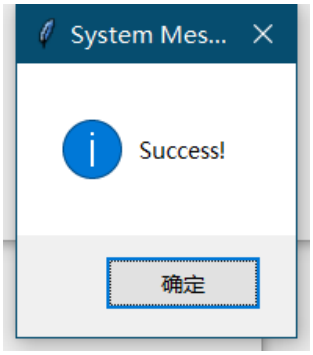
input:lower limit of AOOR Mode (An entry box that can be only set within a certain range)



Expected output: “success” window

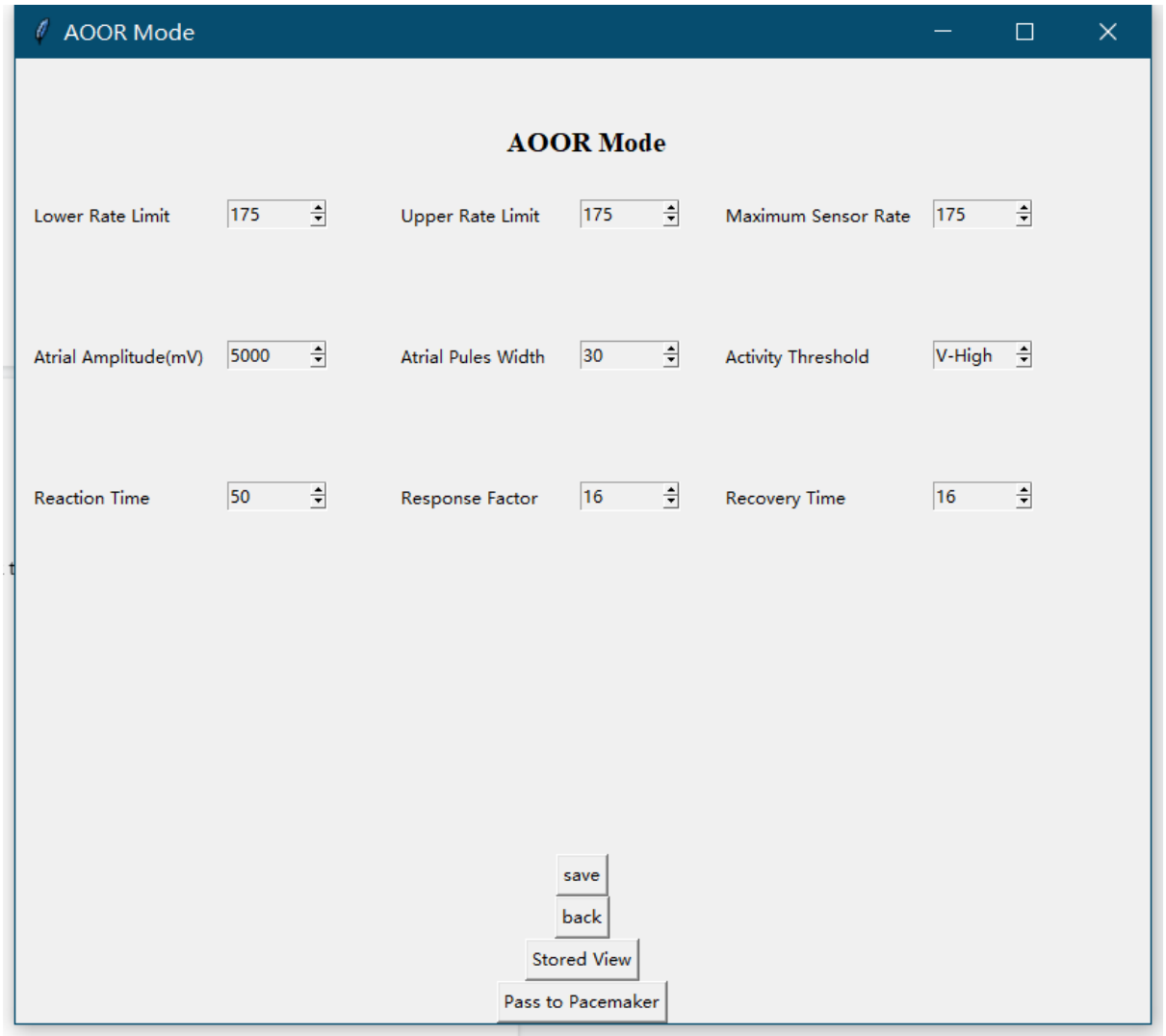
Output: “success” window

result: pass



Testcase25:

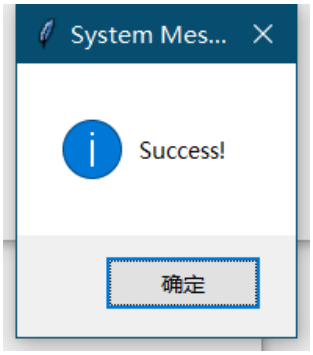
input:upper limit of AOOR Mode (An entry box that can be only set within a certain range)



Expected output: “success” window

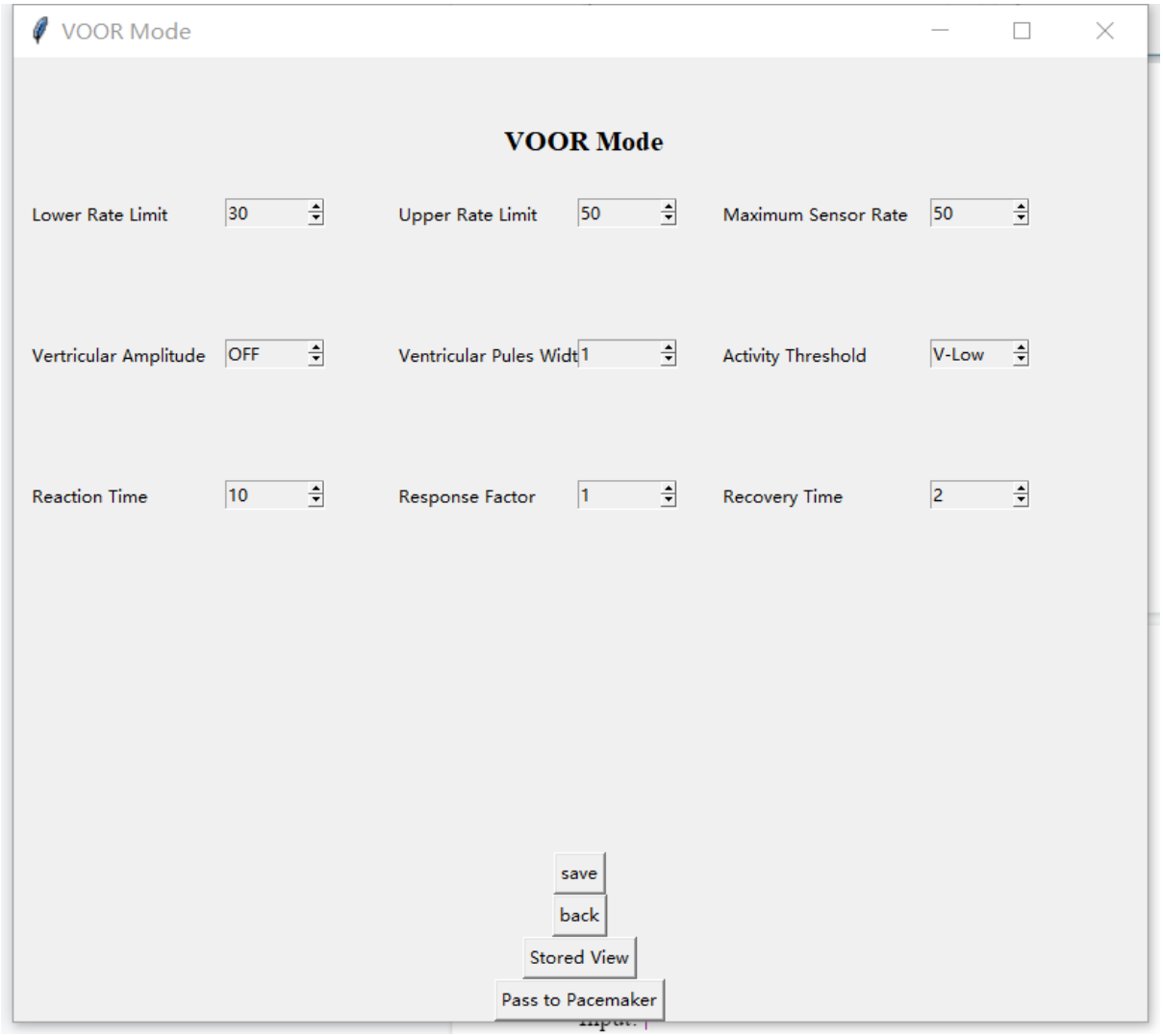
Output: “success” window

result: pass



Testcase26:

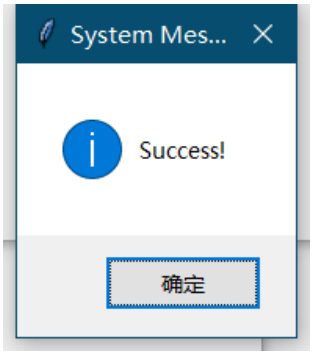
input: lower limit of VOOR Mode(An entry box that can be only set within a certain range)



Expected output: “success” window

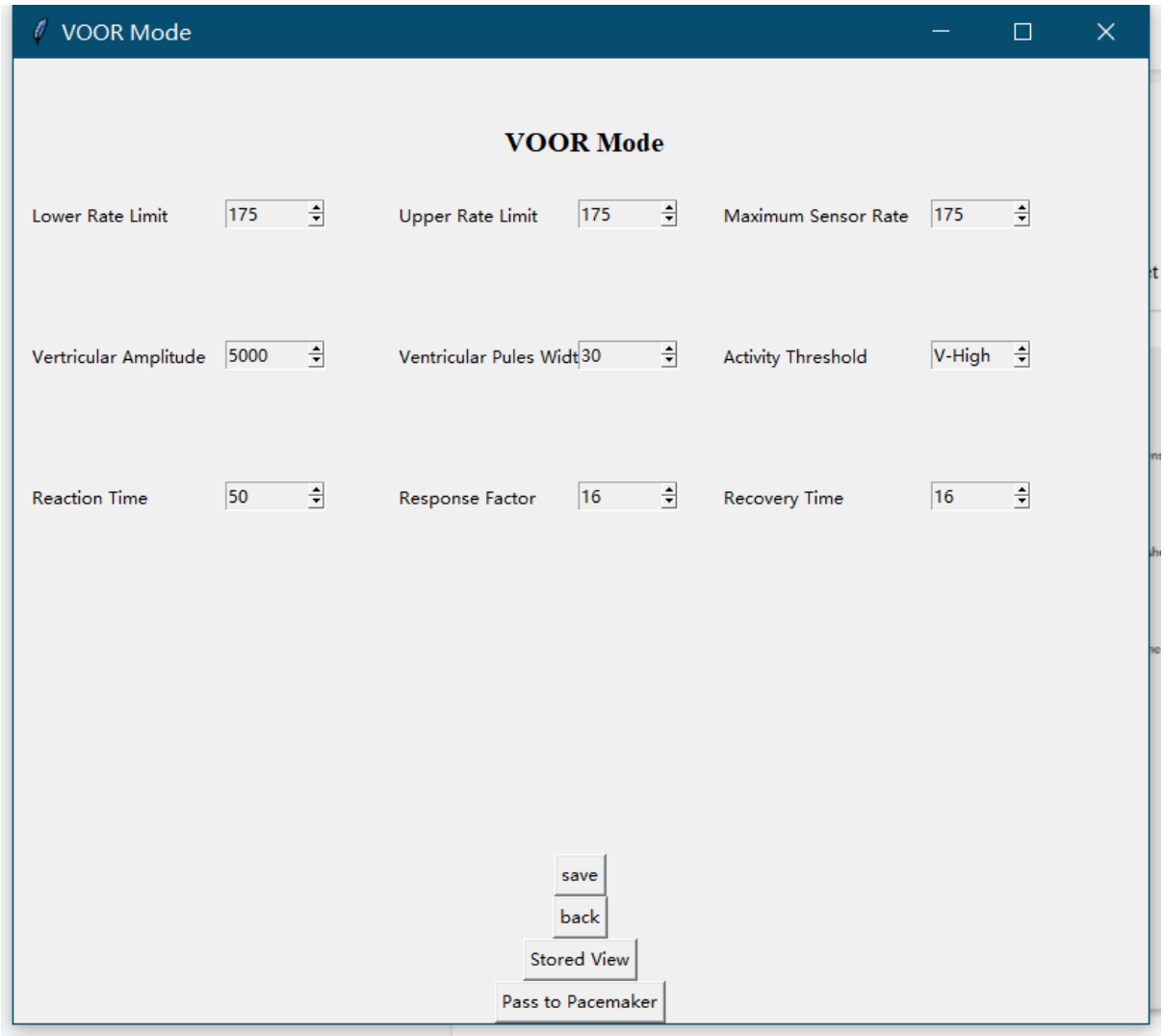
Output: “success” window

result: pass



Testcase27:

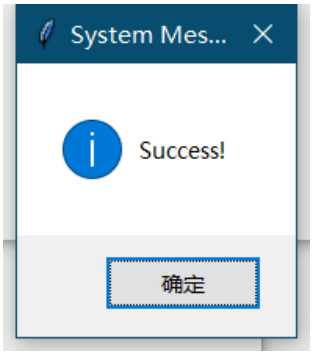
input: upper limit of VOOR Mode(An entry box that can be only set within a certain range)



Expected output: “success” window

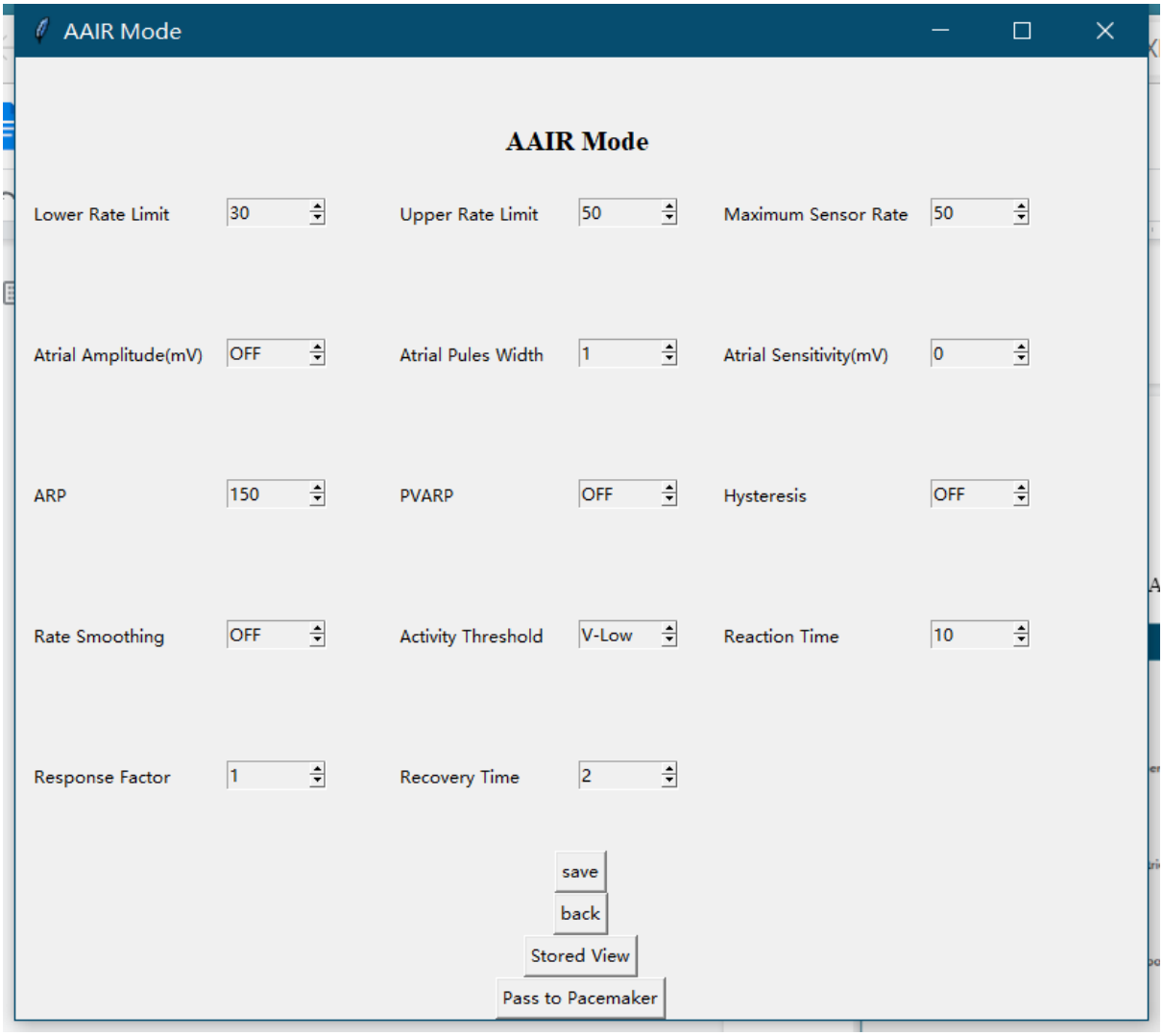
Output: “success” window

result: pass



Testcase28:

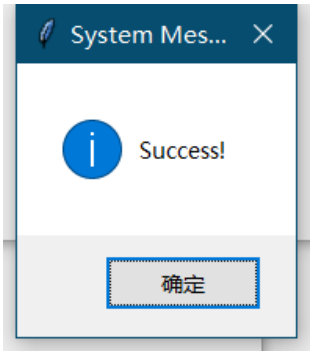
input: lower limit of VOOR Mode(An entry box that can be only set within a certain range):



Expected output: “success” window

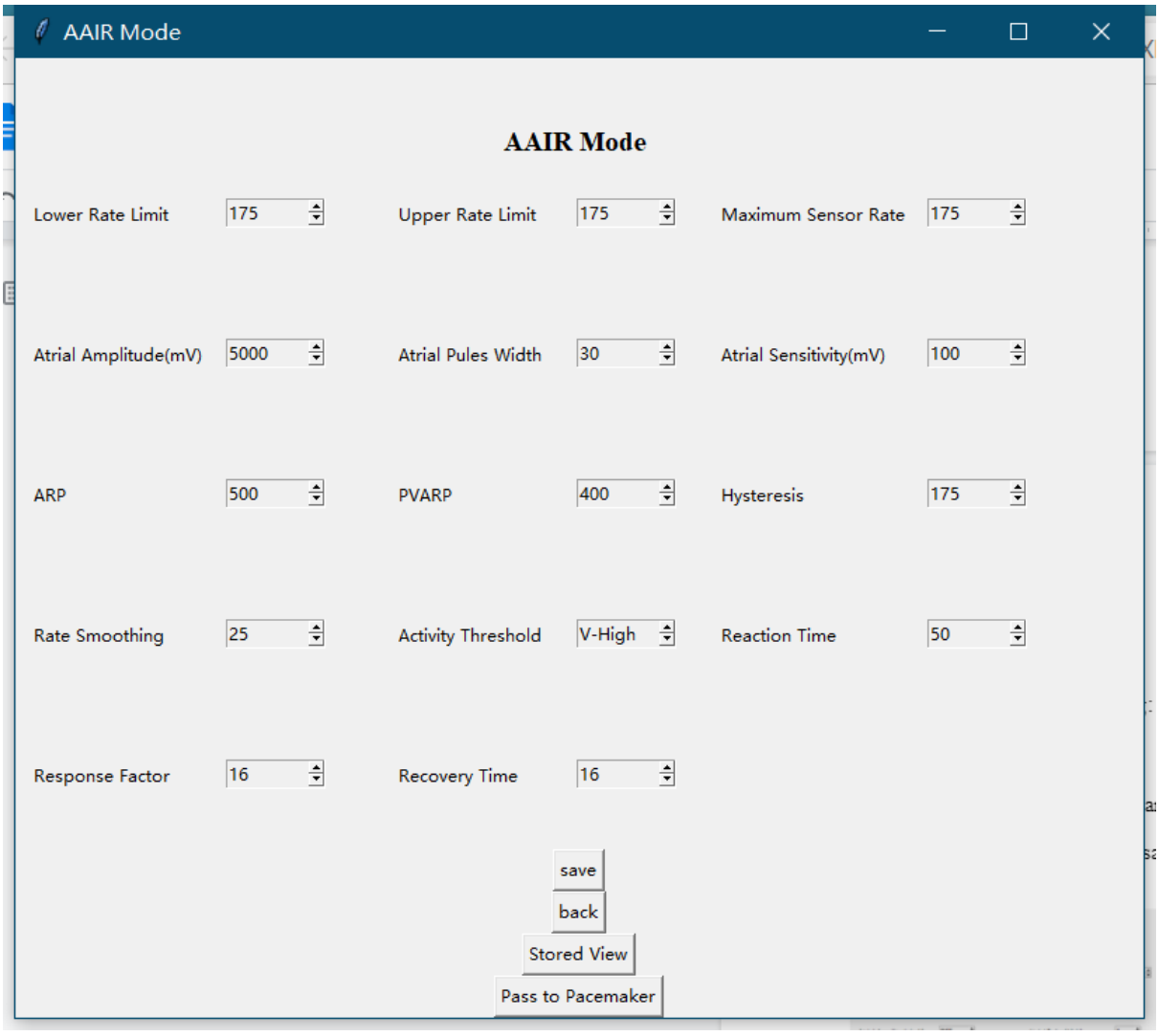
Output: “success” window

result: pass



Testcase29

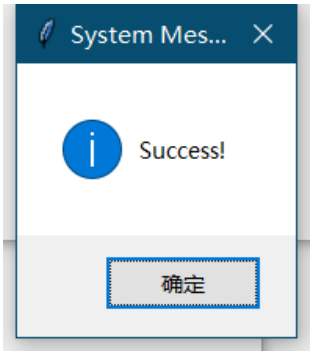
input: upper limit of VOOR Mode(An entry box that can be only set within a certain range):



Expected output: “success” window

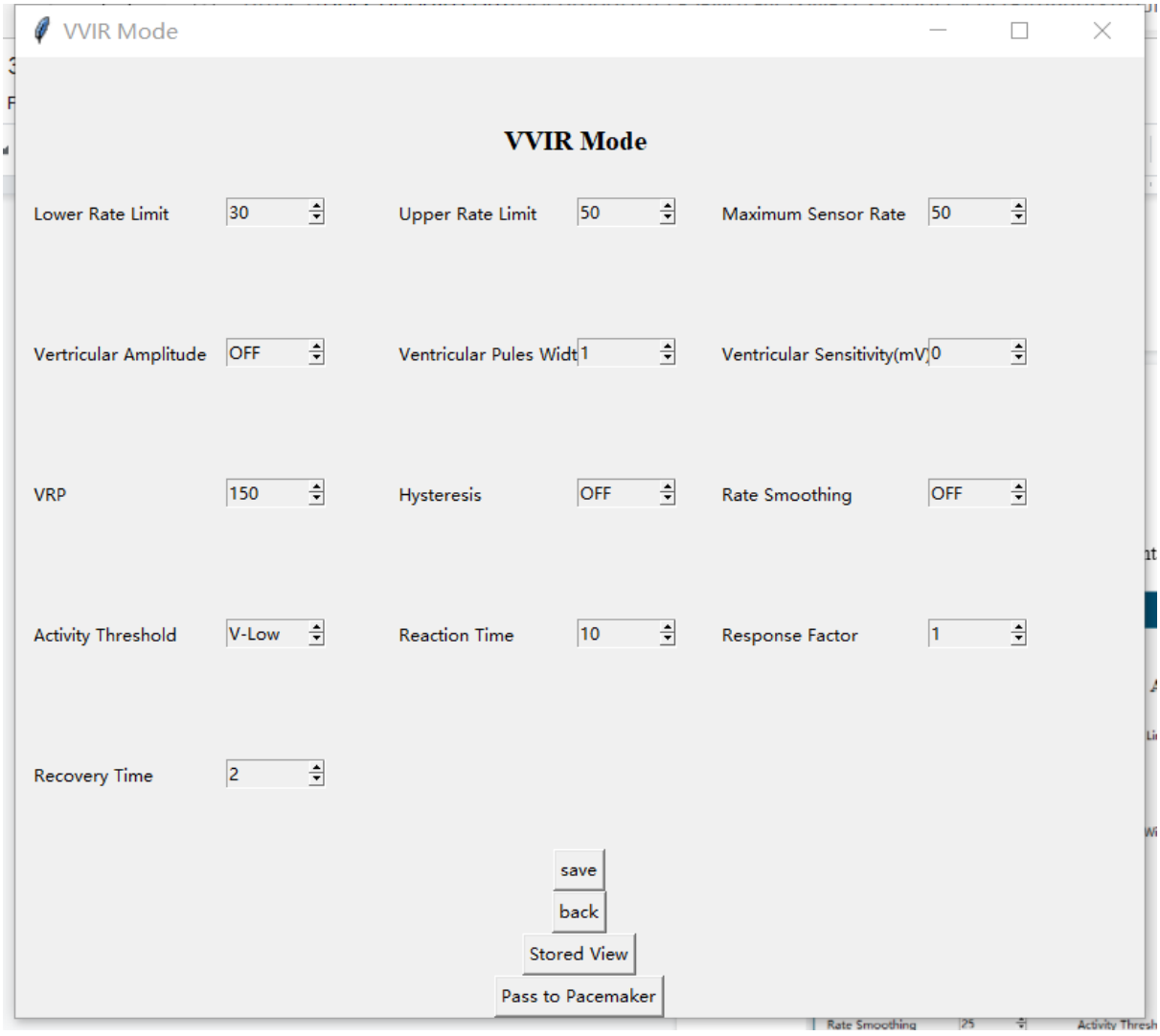
Output: “success” window

result: pass



Testcase30

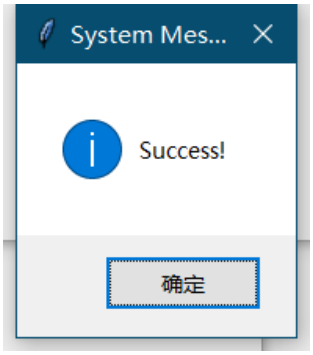
input: lower limit of VVIR Mode (An entry box that can be only set within a certain range):



Expected output: “success” window

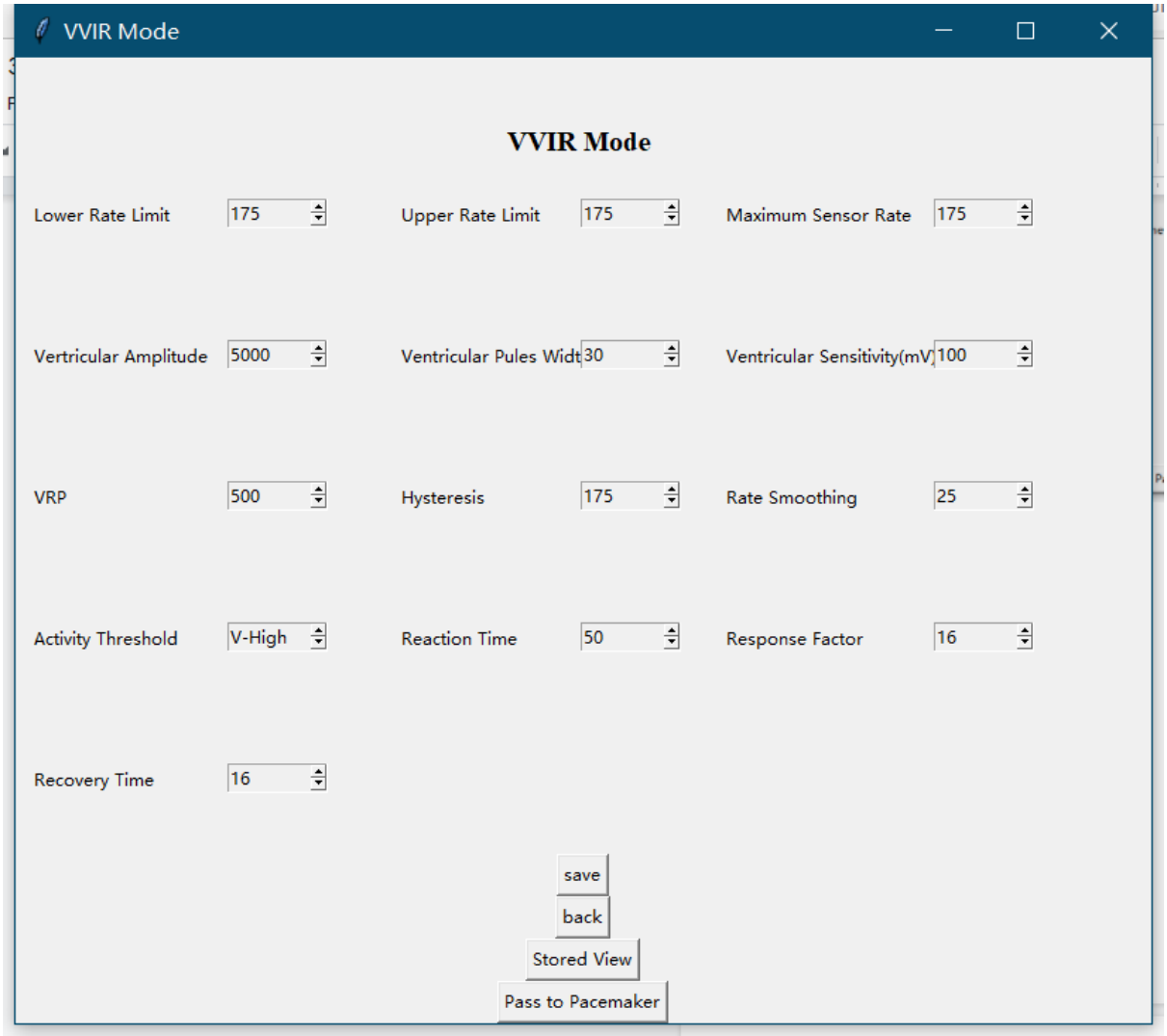
Output: “success” window

result: pass



Testcase 31:

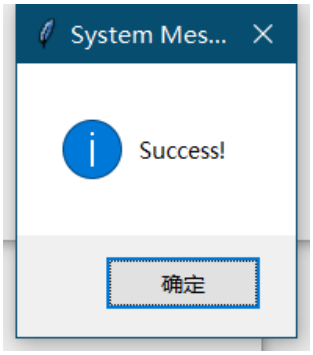
input: upper limit of VVIR Mode(An entry box that can be only set within a certain range):



Expected output: “success” window

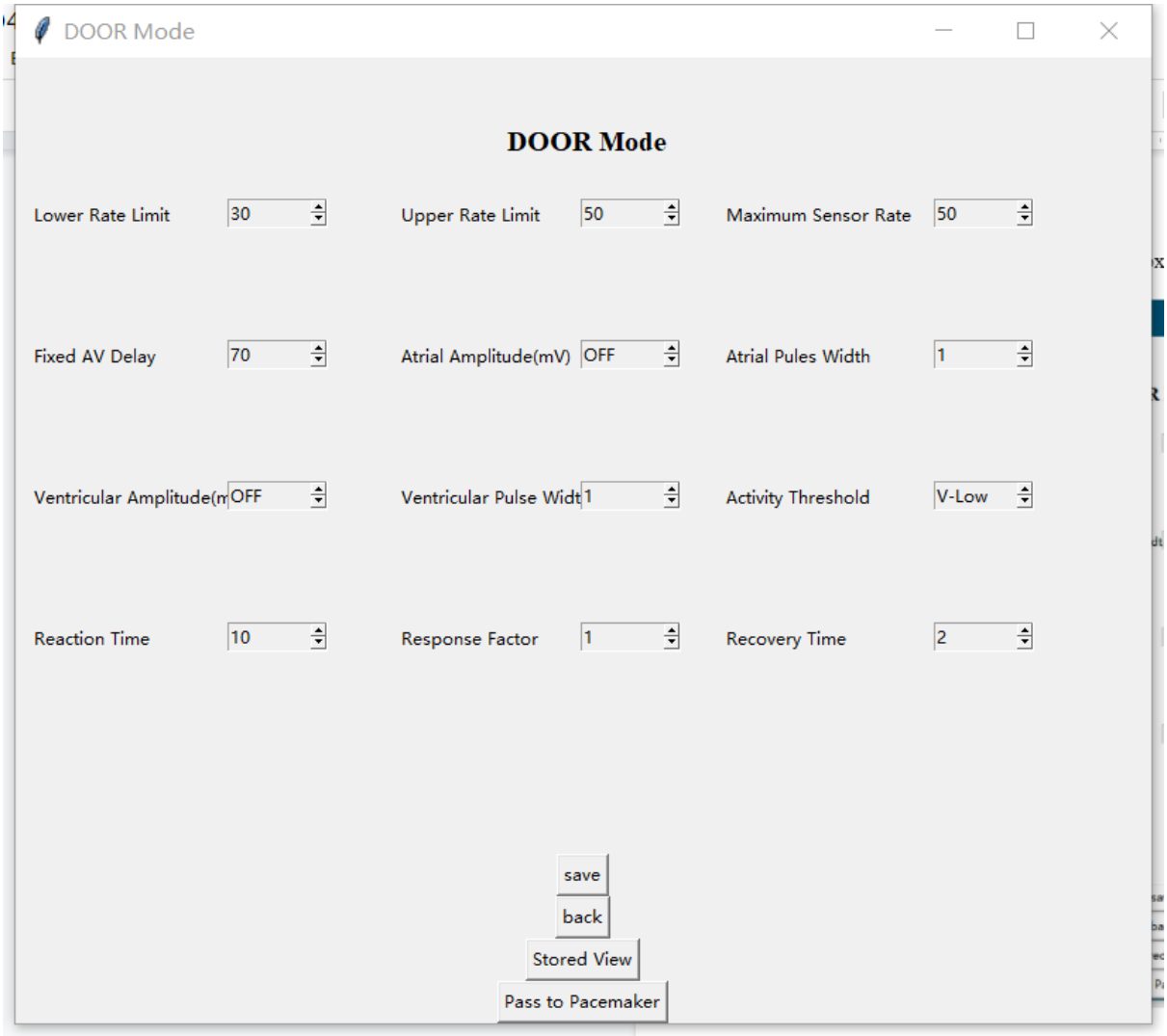
Output: “success” window

result: pass



Testcase32:

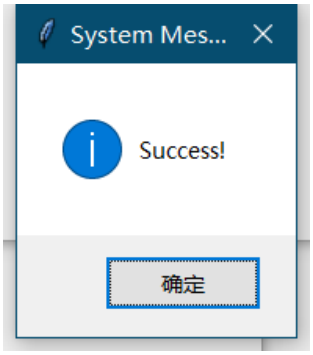
input: lower limit of DOOR Mode(An entry box that can be only set within a certain range)



Expected output: “success” window

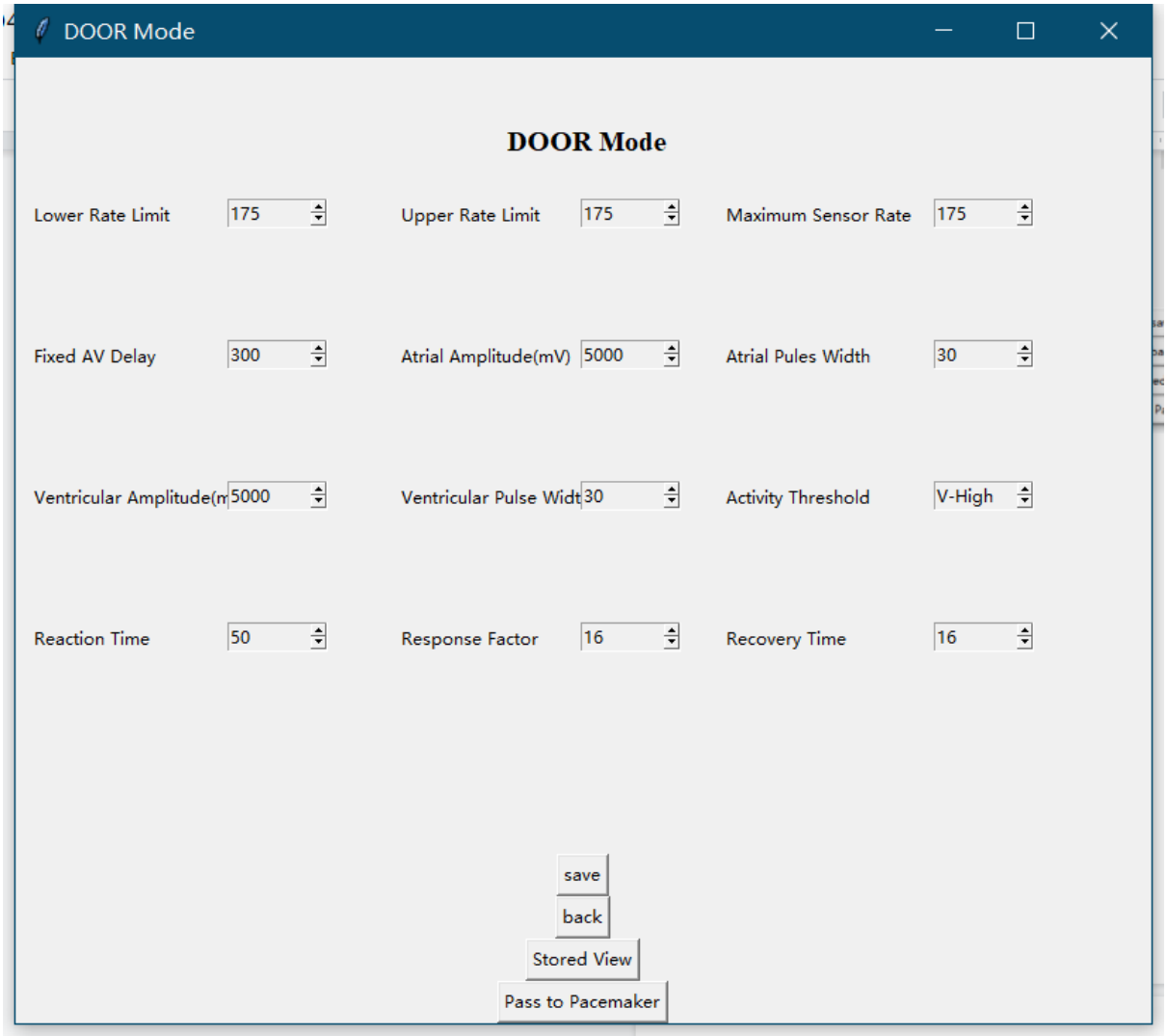
Output: “success” window

result: pass



Testcase33:

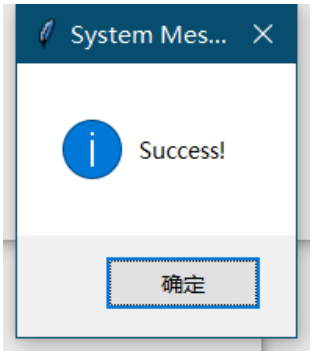
input: upper limit of DOOR Mode(An entry box that can be only set within a certain range):



Expected output: “success” window

Output: “success” window

result: pass



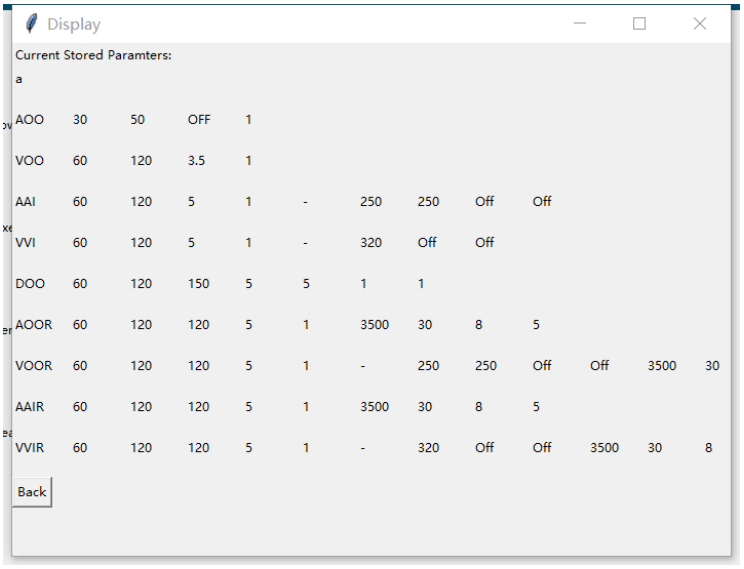
Testcase34:

input: press stored View

expected output: the window include current stored parameters

output:the window include current stored parameters

result: pass



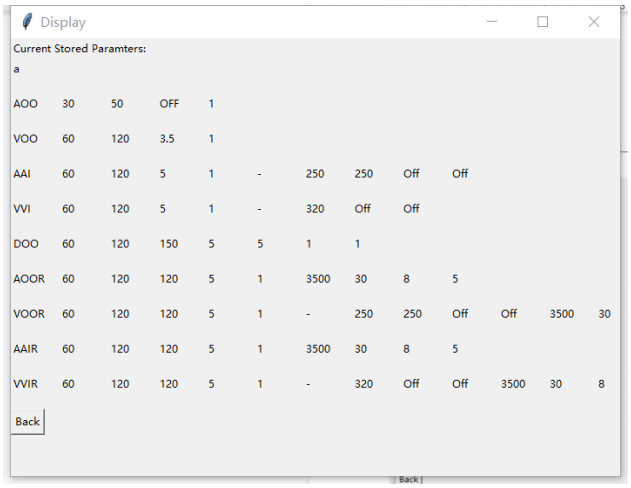
Testcase35:

input: press button “ display current info”

expected output:the window include current stored parameters

output:the window include current stored parameters

result: pass



Module for Serial Communication Testing:

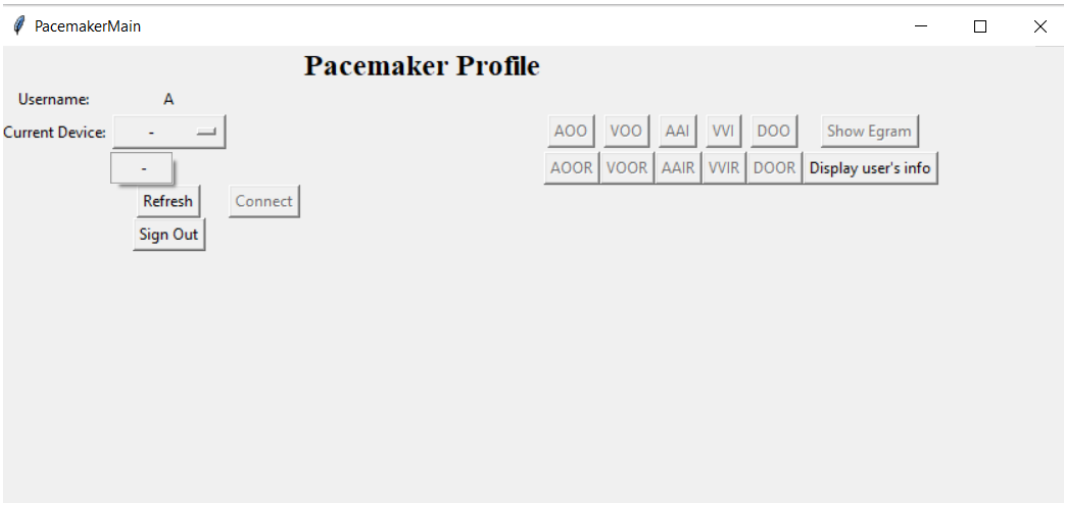
Testcase 36:

Input:No port connected

Expected output: No selection available for the current device drop-down menu

Output: No selection available for the current device drop-down menu

result: pass



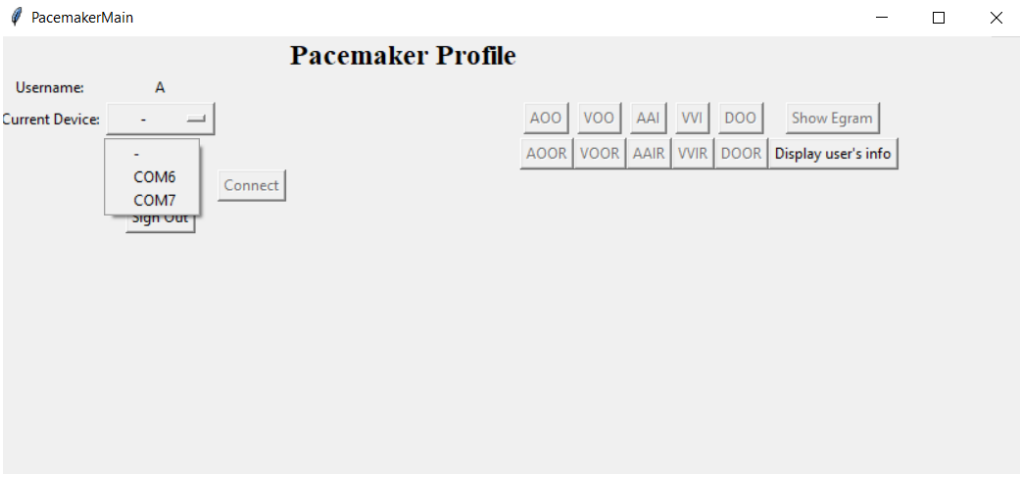
Testcase 37:

Input: Ports connected

Expected Output:Port selection available for the current device drop-down menu

Output: Port selection available for the current device drop-down menu

result: pass



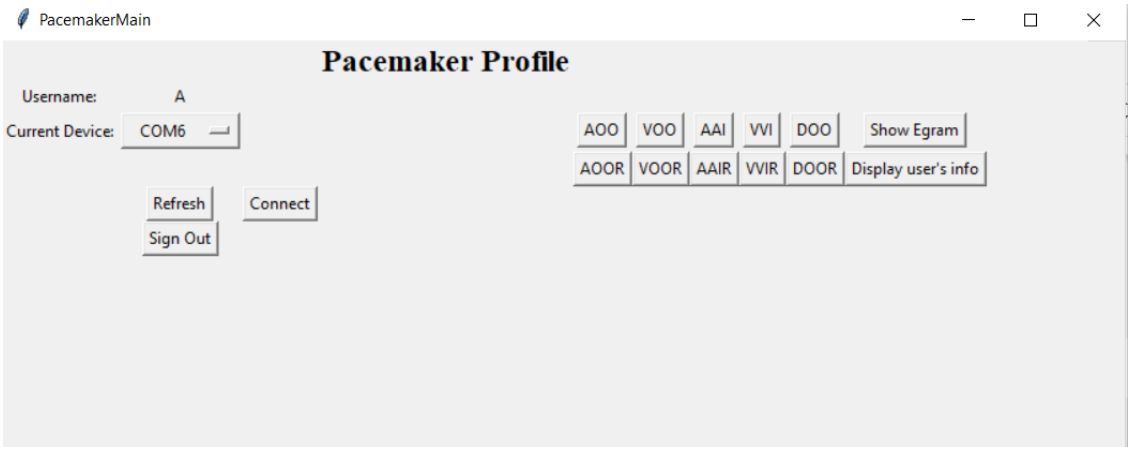
Testcase 38:

Input: Ports selected

Expected Output: All the buttons state are active

Output: All the buttons state are active

result: pass



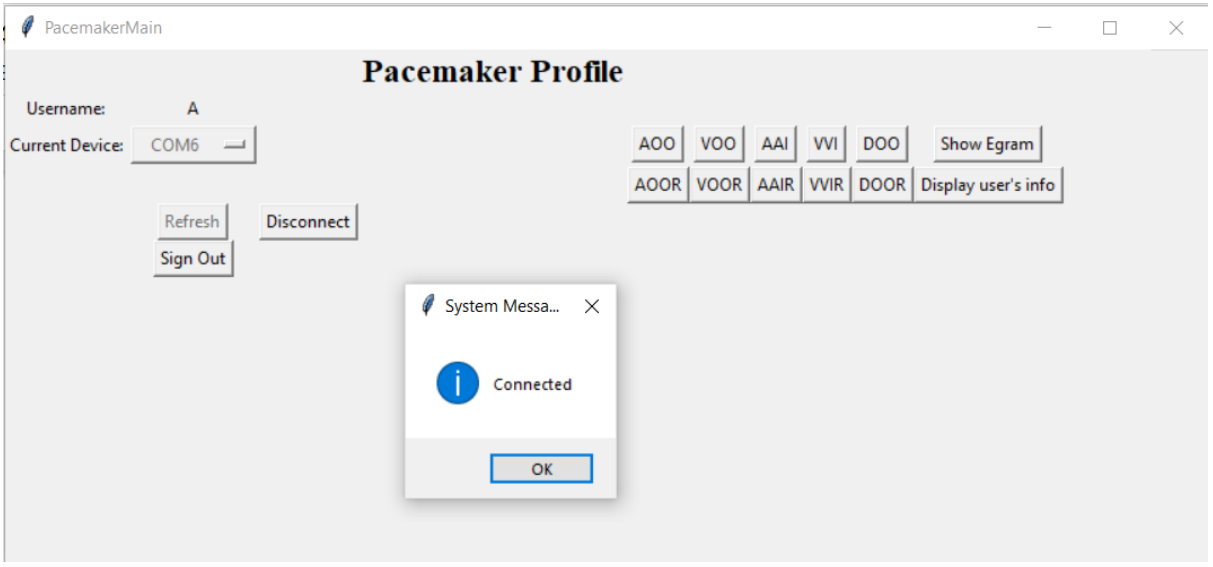
Testcase 39:

Input: Connect button clicked

Expected Output: “Connected” message window. ‘Current Device’ and ‘Refresh’ button state disabled

Output: “Connected” message window. ‘Current Device’ and ‘Refresh’ button state disabled

result: pass



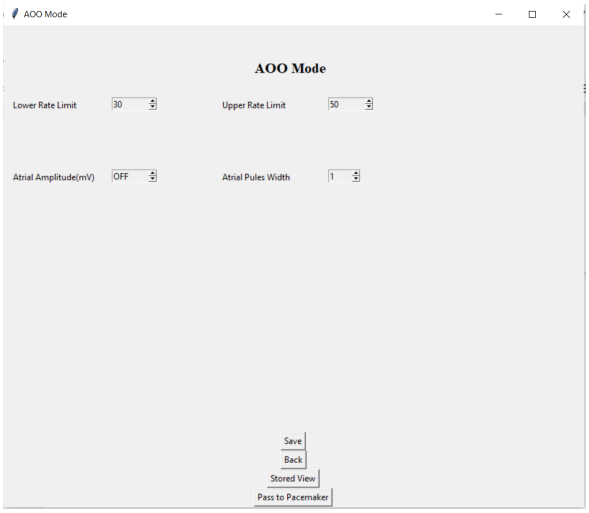
Testcase 40:

Input: AOO Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



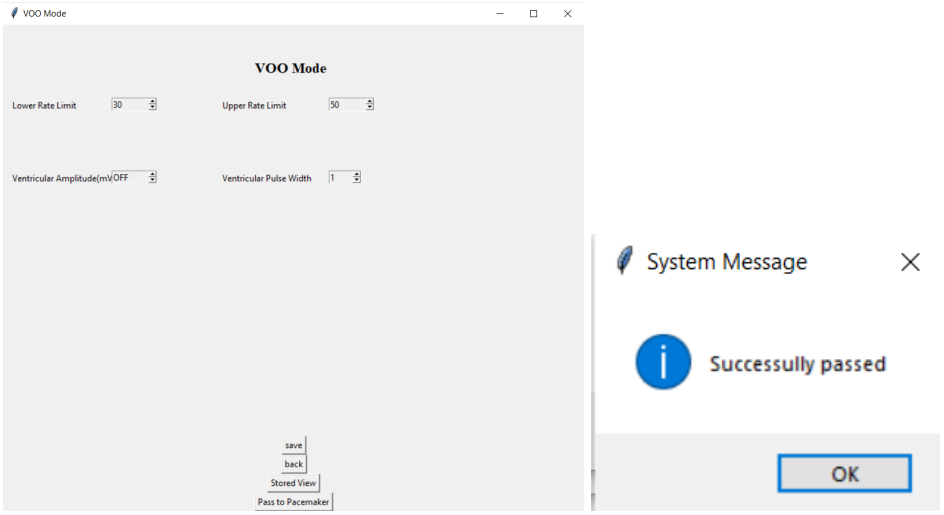
Testcase 41:

Input: VOO Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



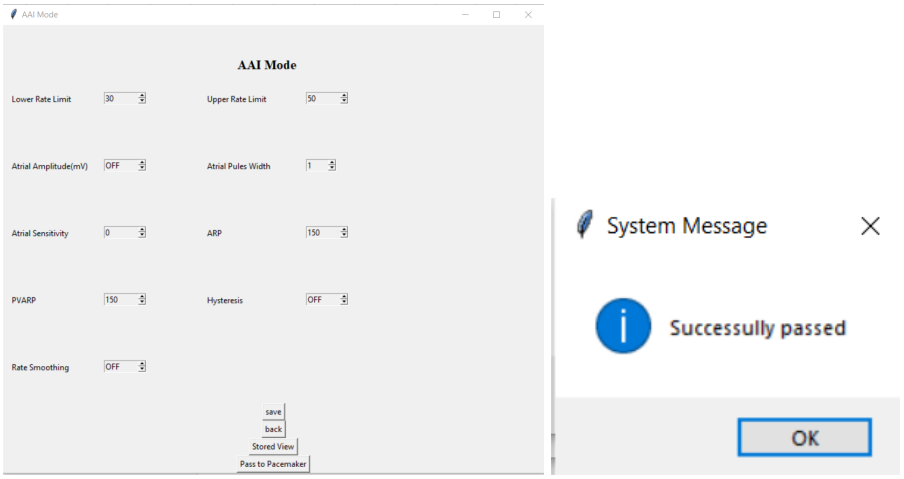
Testcase 42:

Input: AAI Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



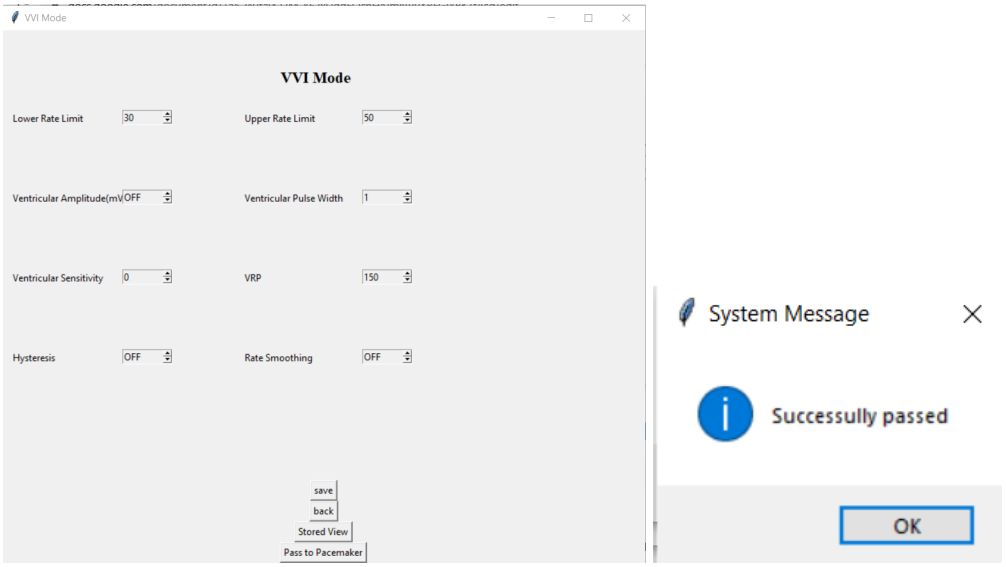
Testcase 43:

Input: AAI Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



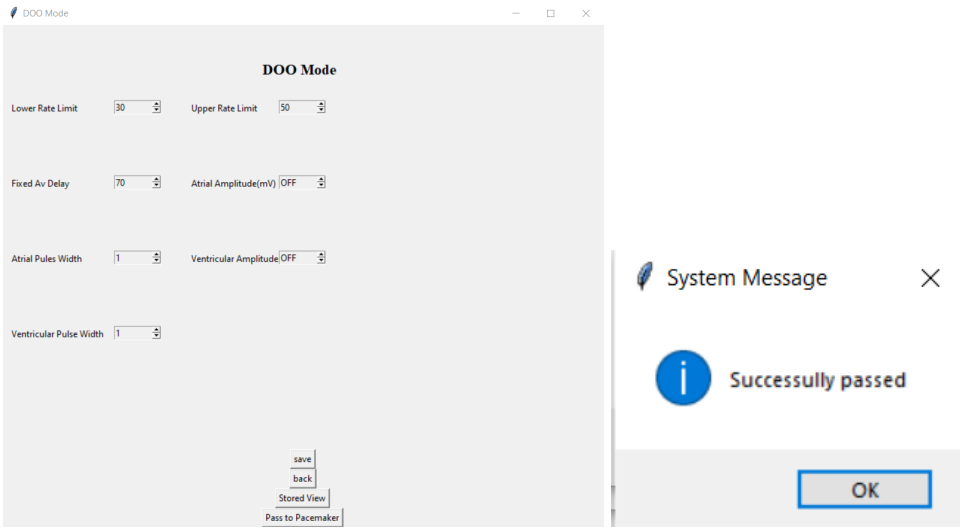
Testcase 44:

Input: DOO Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



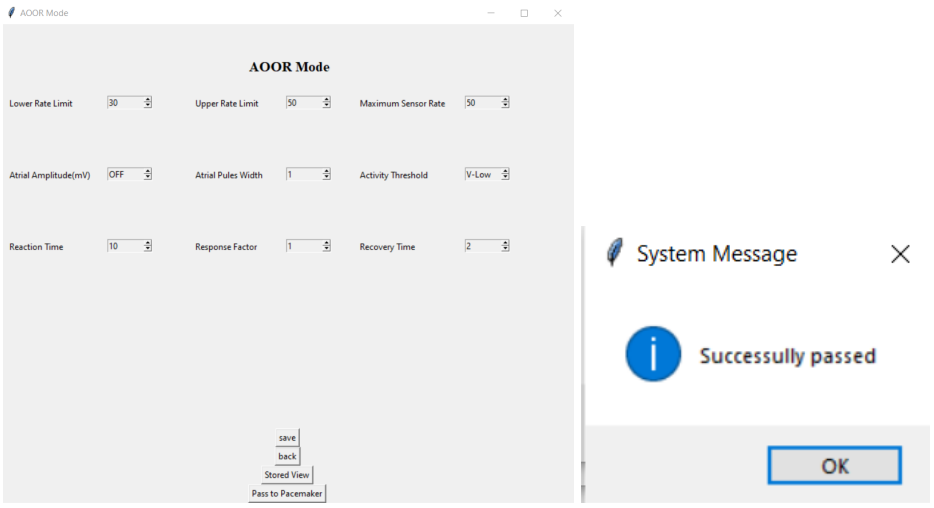
Testcase 45:

Input: AOOR Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



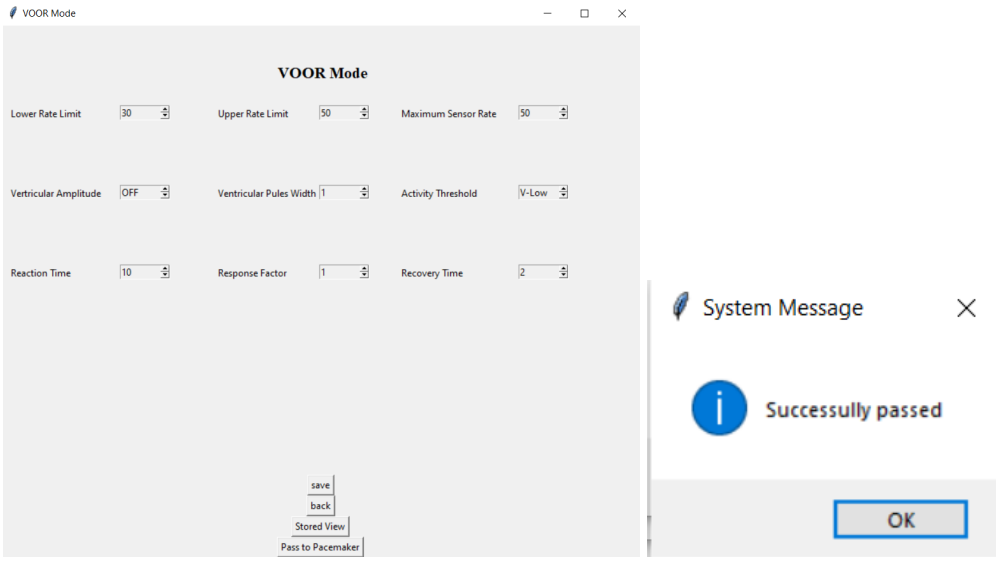
Testcase 46:

Input: VOOR Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



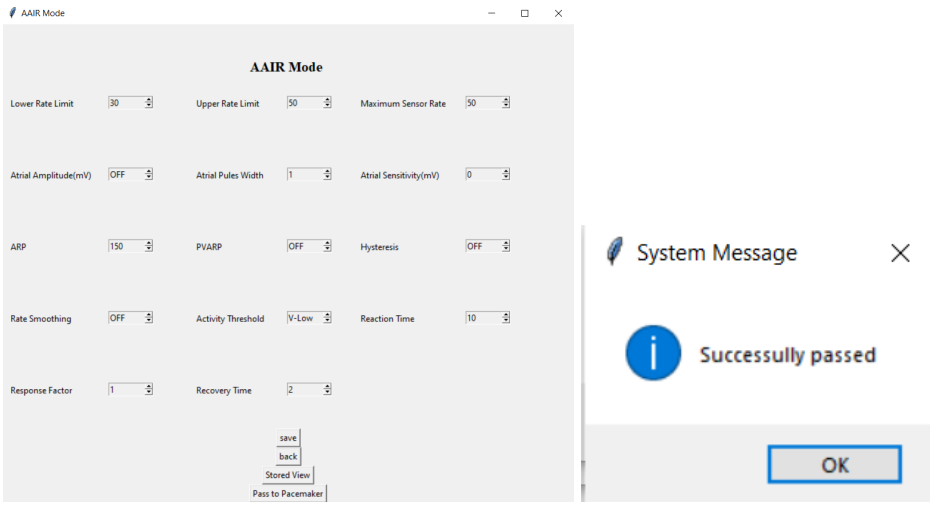
Testcase 47:

Input: AAIR Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



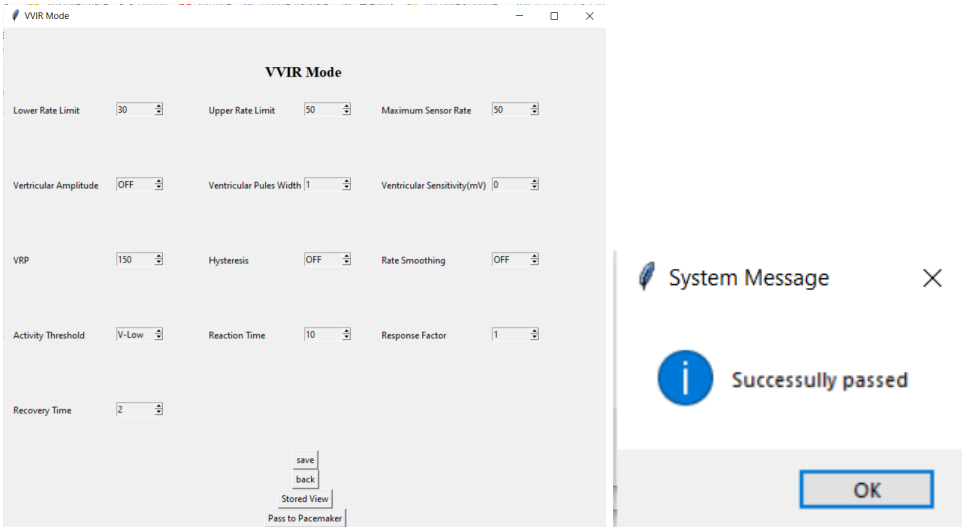
Testcase 48:

Input: VVIR Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



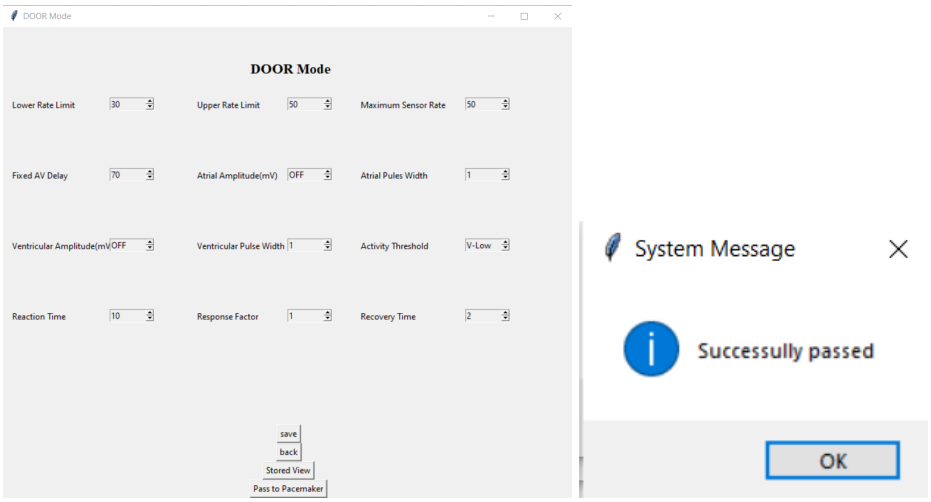
Testcase 49:

Input: DOOR Mode pass the saved parameter data by clicking “Pass to Parameter”

Expected Output: “Successfully passed” message window

Output: “Successfully passed” message window

result: pass



test case50:

egram

input :each parameter

expected output: display egram

output: does not display

result: fail

Reference

[1]Boston Scientific (2007).*PACEMAKER System Specification*. Retrieved November 30, 2021, from McMaster University.

[2]Guy Meyer (2020, November 2*).Pacemaker Microcontroller Shield*. Retrieved November 30, 2021, from McMaster University.

[3]Alan Wassyng (2011, September 27*).3K04 Requirements Specification of the Pacemaker*. Retrieved November 30, 2021, from McMaster University.