Distributed Systems (Fall 2018)

Task 1: state system modelling

Practical information:
Due date: Thursday, Nov 15, midnight
Each hour late after midnight is 10% less points
– After 6 hours, value of the task is 50%
– This task can be done in a team of max 3 persons.

Submit your solution to huber.flores@helsinki.fi CC. abhishek.kumar@helsinki.fi

Task description
In this assignment, as shown in Figure 1, you will specify a state model for the system behavior of a Digital Watch (inspired by the 1981 Texas Instruments LCD Alarm Chronograph).

Figure 1: Watch overview

As for the in-class practice, you must design the Statechart (you must include this statechart in your report), test it with SVM and finally plug the SVM simulation into a Python GUI. Don't be worry about your current proficiency on Python and Tkinter, for this assignment you will be provided with the implementation of the GUI.

Behavior requirements
1. The time value should be updated every second, even when it is not displayed (as for example, when the chrono is running). However, time is not updated when it is being edited.
2. Pressing the top right button turns on the background light. The light stays on for as long as the button remains pressed. From the moment the button is released, the light stays on for 2 more seconds, after which it is turned off.
3. Pressing the top left button alternates between the chrono and the time display modes. The system starts in the time display mode. In this mode, the time (HH:MM:SS) and date (MM/DD/YY) are displayed

4. When in chrono display mode, the elapsed time is displayed MM:SS:FF (with FF hundredths of a second). Initially, the chrono starts at 00:00:00. The bottom right button is used to start the chrono. The running chrono updates in 1/100 second increments. Subsequently pressing the bottom right button will pause/resume the chrono. Pressing the bottom left button resets the chrono to 00:00:00. The chrono will keep running (when in running mode) or keep its value (when in paused mode), even when the watch is in a different display mode (for example, when the time is displayed).
    Note: interactive simulation of a model containing time increments of 1/100 second is possible, but it is difficult to manually insert other events. Hence, while you are simulating your model, it is advisable to use larger increments (such as 1/4 second) for simulation purposes
5. When in time display mode, the watch will go into time editing mode when the bottom right button is held pressed for at least 1.5 seconds.
6. When in time display mode, the alarm can be displayed and toggled between on or off by pressing the bottom left button. If the bottom left button is held for 1.5 seconds or more, the watch goes into alarm editing mode. This is not an example of good User Interface design, as going to editing mode will also toggle on/off and that may not be desired. It is however how the 1981 Texas Instruments LCD Alarm Chronograph works. The first time alarm editing mode is entered, the alarm time is set to 12:00:00. The alarm is activated when the alarm time is equal to the time in display mode. When it is activated, the screen will blink for 4 seconds, and then the alarm turns off. Blinking means switching to/from highlighted background twice per second. The alarm can be turned-off before the elapsed 4 seconds by a user interrupt (i.e.: if any button is pressed). After the alarm is turned off, activity continues exactly where it was left-off.
7. When in (either time or alarm) editing mode, briefly pressing the bottom left button will increase the current selection. Note that it is only possible to increase the current selection, there is no way to decrease or reset the current selection. If the bottom left button is held down, the current selection is incremented automatically every 0.3 seconds. Editing mode should be exited if no editing event occurs for 5 seconds. Holding the bottom right button down for 2 seconds will also exit the editing mode.

To help clarify the requirements, you can find a working solution (without the alarm activation) in the zip file (directory: demo; to run the demo use "python DigitalWatch.py"). The statechart behind is precompiled however.

Starting point
The zip file provided contains the implementation of the GUI (directory: startingpoint). This implementation is organized in a way to facilitate the interaction with the Statechart simulation. The GUI is implemented with two classes:
*   **LowLevelGUI**   Implements the rendering and the tracking of mouse events.
*   **DWatchGUI**   Provides a simplified view to the GUI and serves as the bridge between the LowLevelGUI and the Statechart simulation (event forwarding and user interface updating).

The following table summarizes the set of operations provided by DWatchGUI.

getTime()	Returns the current clock time.
getAlarm()	Returns the alarm time set.
checkTime()	Checks if the alarm time set is equal to the current clock time. If so, it will broadcast the "alarmStart" event to the statechart and return true. Otherwise,

	it returns false. Note that checkTime() does not care/check whether the alarm has been set "on".
refreshTimeDisplay()	Redraw the time with the current internal time value. The display does not need to be cleaned before calling this function. For instance, if the alarm is currently displayed, it will be deleted before drawing the time.
refreshChronoDisplay()	See refreshTimeDisplay()
refreshDateDisplay()	See refreshTimeDisplay()
refreshAlarmDisplay()	See refreshTimeDisplay()
resetChrono()	Resets the internal chrono to 00:00:00.
startSelection()	Selects the leftmost digit group currently displayed on the screen.
increaseSelection()	Increases the currently selected digit group's value by one.
selectNext()	Select the next digit group, looping back to the leftmost digit group when the rightmost digit group is currently selected. If the time is currently displayed on the screen, select also the date digits. If the alarm is displayed on the screen, don't select the date digits. (to simplify the statechart).
stopSelection()	
increaseTimeByOne()	Increase the time by one second. Note how minutes, hours, days, month and year will be modified appropriately, if needed (for example, when increaseTimeByOne() is called at time 11:59:59, the new time will be 12:00:00).
increaseChronoByOne()	Increase the chrono by 1/100 second.
setIndiglo()	Turn on the display background light.
unsetIndiglo()	Turn off the display background light.
setAlarm()	Flag the alarm to be on or off.

GUI events to be handled by the Statechart: (due to button press)
topRightPressed
topRightReleased
topLeftPressed
topLeftReleased
bottomRightPressed
bottomRightReleased
bottomLeftPressed
bottomRightReleased

(generated by checkTime() if "current time" == "alarm time") alarmStart

Deliverables
Source code, a diagram of your state logic, and a report about implemented functionality.