DESN2000 (Computer Engineering) 2024 T2

More Project Details: Custom Clock with Laboratory Timer Functions

Laboratory timer (~30%)

1. Configuration
   a. During the configuration, the user should be able to give a number that indicates how many timers they need (1 to 4, check for nonvalid inputs).
   b. The user should then be able to set the timeout for each of those tasks while giving a short label for each task (maximum time for a task can be assumed to be 60 minutes and the minimum 30 seconds).
   c. The user should be able to change a saved value later if they wish.
   d. The saved configuration must be stored in memory as long as the power to the device is provided (assume the power supply will be replaced using a battery at the production stage – you don't have to worry about this).
2. Running
   a. Each task should have a designated button to start the relevant timer.
   b. All the timers should be displayed on the LCD screen appropriately in a countdown style.
   c. When each timer reaches its end, notify using a sound.
   d. Ensure each timer has a unique notification (e.g., different sounds) to distinguish it from others.
   e. User should be able to start tasks in parallel (ignore any little delays by the user to press buttons) or start them sequentially in any order.
   f. Once a certain task has finished, the user should be able to start the same task again later

Standard clock (~25%)

1. Time of the day clock
   a. today's date and time are displayed on the LCD in a format like 19/06/2024 Wed 12:01:00 am.
   b. a method to set this date and the time.
2. One alarm
   a. User should be able to set an alarm for a given time.
   b. Should be able to distinguish between a.m. and p.m. No need to worry about the dates for the alarms, let them ring every day.
   c. There should be a way to enable/disable the alarm.
   d. There should be a way to snooze and stop the alarm.
3. One countdown timer
   a. User should be able to set the countdown time in a format like hh:mm:ss.

    b. The countdown should be displayed on the LCD.
    c. Once the countdown is reached, a notification should be played (both auditory and visual).

4. One stopwatch
    a. User should be able to reset, start, pause and stop the stopwatch.
    b. The stopwatch should run in a format like hh:mm:ss.

<u>Switching between the two modes (~15%)</u>

1. Make sure there is a way to switch between the two modes.
2. Switching the modes should not clear any running timers/alarms/countdowns/stopwatches and instead should run in the background.
3. The user should be able to clearly distinguish in which mode the clock is currently in.
4. When in a given mode, notifications from the other mode (e.g., alarm, timeout etc.) can be ignored.

<u>Distinguishable Features (~30%)</u>

At least three distinguishable features are required to make your implementation user-friendly and attractive. Following are some examples only. Don't simply limit yourselves to these. Be creative and go beyond these.

1. A 3D-printed clock hand (you may use the maker space?) fixed onto the motor, that acts as a second indicator.
2. Being able to connect to a computer to backup/restore the configuration/settings/
3. Determine less than optimal lighting using the LDR and illuminate some LEDs to light up the board (brightness of LED inversely proportional to the darkness).
4. Make the user interface (buttons, menu etc) so good and intuitive that anyone can use it smoothly.
5. Play some nice tunes/songs for notification sounds and provide the user with a way to select between tunes.