Smart Home Project

Description:

- The aim of this project is to monitor and control the home climate.
- The parameters to measure and control parameters like Temperature, Light intensity.
- The temperature must be maintained between 20 - 25 celsius

System components:

- Atmel SAM3X8E – ARM embedded computer platform (Arduino due)
- Keypad (Port I/O)
- Photosensor (A/D)
- Temperature sensor (time)
- RC servo motor (PWM)
- Switch (HW-int)
- Display (port I/O)
- LEDs

Grade 3 requirements:

Req. 1: The system should have a calendar that allows showing date and time. The user should be able to configure it whenever it is needed. The date should be represented as ***DD/MM/YYYY***, while the clock as ***hh:mm:ss***. (*Use the **24-hour clock system***). The system can use **SysTick** to measure time.
Purpose: For knowing date and time and timestamp the recorded data.

Req. 2: Periodic temperature recording every minute for a duration of 7 days. The recorded temperature should be time-stamped with the time moment that it was measured at. During recording temperature, if the system memory buffer is full, the recording should start over from the beginning.
Purpose: Recording of temperature.

Req. 3: Presentation of recorded data on the LCD by text. Each day is presented by minimum, average, maximum and variance values for temperature. Maximum and minimum values should be presented along with their timestamp.
Purpose: Presentation of logged data.

Req. 4: The home has a large glass window with a motorised shading system that protects from direct sunlight. The system tracks the sun and moves the shades (controlled by the servo motor)

to face the sunlight. The sun can be simulated by any bright light source. (Problem, to achieve this req. Should operate in periodically)

Req. 5: The home temperature should be maintained between a defined upper and lower limits. An alarm should be raised in case these limits were crossed.
Purpose: Alarm at under or over temperature.

Req. 6: Create a fast mode that does exactly the same as **Req. 2** where each minute is simulated by a second.
Purpose: To simplify testing.

Req. 7: Draw a functional block diagram, which illustrates graphically how the system is composed.
Purpose: Illustrated by a figure how things are connected.

Req. 8: Documentation of the project. A report which specifies the modules of the system, with a clear connection to the project requirements, so that it is possible to track which requirements are achieved.
Purpose: Document what you have done.

Grade 4 requirements:

Req. 9: Periodic temperature recording every minute. Each minute, temperature is recorded N times and averaged to give one number; data is stored for as long as there is memory available. When memory is full an indication is made in the user interface and the oldest values are overwritten, in a circular fashion. N is selectable, range 1-10, in a settings screen by the keypad. The N temperature values, as well as the average value of them, should be stored with their timestamp. (This requirement supersedes **Req. 2**)
Purpose: Recording of temperature.

Req. 10: Use two photosensors to achieve the function of **Req. 4**. (This requirement supersedes **Req. 4**)
Purpose: Tracking of sun position and controlling window shades.

Req. 11: Presentation of recorded data on the LCD display using graphs. Each day *(specify the date)* is presented by minimum, average, maximum and variance values for temperature.
Purpose: Presentation of logged data.

Req .12: Create a test mode for **Req. 11** using pre-recorded data to test the graph mode.

Grade 5 requirements:

Req. 13: The smart home system should include a Username/Password feature to allow users to access its features. It is not secure to store the password as plaintext. A general approach is to store a hash of the password which is the output of a secure one way function. When the user inputs the password, the hash of the password is computed and compared to the stored hash. Using a regular secure hashing function directly with passwords is problematic for multiple reasons:

1) As passwords tend to be short, using one hash function is fast and makes it easy for an attacker to perform a dictionary attack and guess the password.
2) Two users with the same password will have the same hash.

To resolve those issues techniques like iterative application of hashing and extending the password using random bits (salting) can be used. Read the following information about password hashing: https://botan.randombit.net/handbook/api_ref/passhash.html

To simplify this assignment, you are asked to implement MD5 hashing in order to hash the password. MD5 is easy to implement, however you should remember that in a real setting you should not directly use it and that it is better to use a stronger function that is already implemented in a library. However, since we are not using an operating system in this project, you should implement MD5 yourself. You can find explanation of the algorithm and a method of testing the evaluation here: http://practicalcryptography.com/hashes/md5-hash/

Req. 14: The system should have one administrator with username admin. You should initially have a default password for which you store the hash. However, the user should be able to change this password. To change the password, the user should first login, then select a new password. Only the hash of the new password should be stored.

Req. 15: Extended test mode. Create a component testing mode so that the operation of each component can be validated (unit test). Purpose: To simplify testing.