Embedded systems midterm

For the midterm we had planned to wire the RTC clock to the 7-segment displays to show the current time, wire up some buttons to control the basic functionalities of an alarm clock and playing a melody as the alarm sound using a passive buzzer. The following paragraphs will explain how these functionalities have been implemented.

RTC clock

The RTC clock, which uses I2C communication protocol on the analog pins A4 and A5, is initialized with the time of last compilation upon startup and it persists using the attached battery. The clock also has settings for the current date but it is currently not used until a proper display for it will be installed. The seconds are not displayed but are used for other internal purposes, such as turning back on the blinking when there haven't been recent button presses.

7-segments display

The 7-segment displays are able to show the current time or the currently set alarm time and it is refreshed every half second using timed interrupts. When the current time is being shown the middle semicolon is blinking. When current time or alarm time is being edited, the appropriate digits start blinking, and stop blinking for about a second when they are being modified upon button press.

The displays are managed by the TM1637 chip, which uses a custom protocol apparently similar to I2C but incompatible with it and without slave address. More information is present on the datasheet and on this webpage : http://www.microcontroller.it/english/Tutorials/Elettronica/componenti/TM1637.htm

Buttons

There are three connected buttons which have different functions according to machine state.

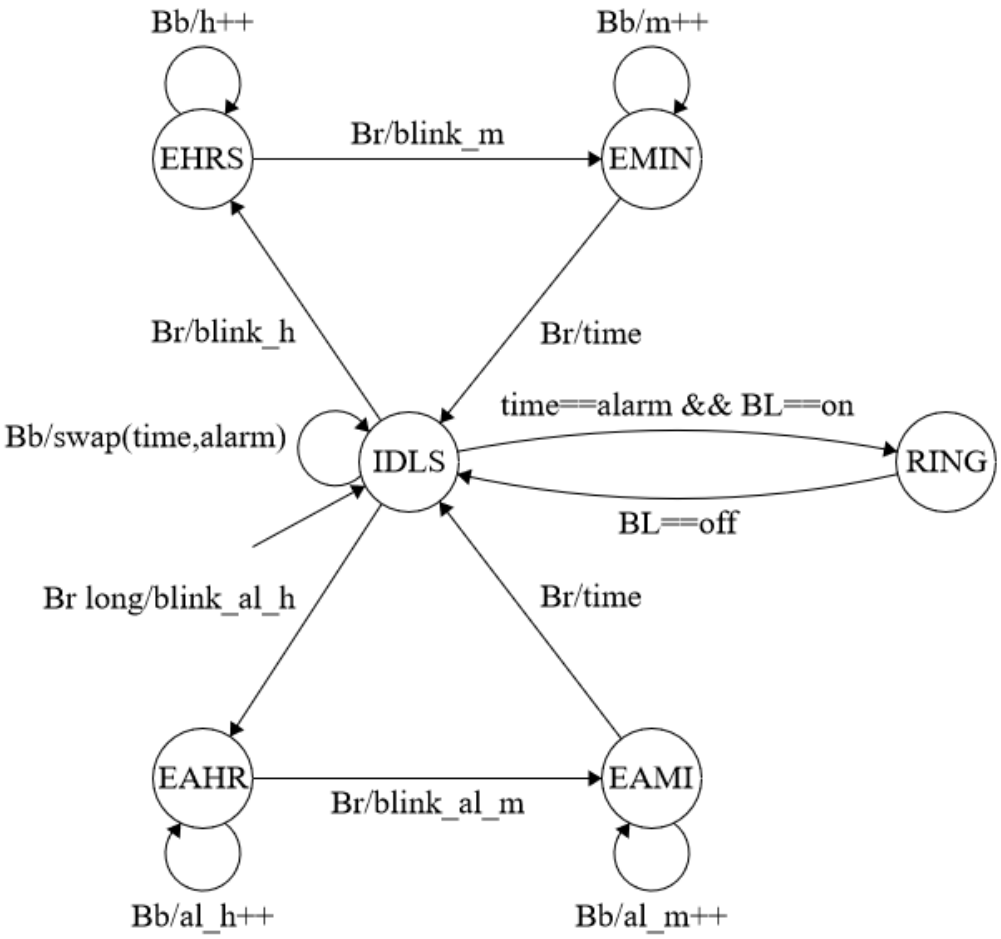
The black button (BB) is able to swap current time with alarm time on the display while the state is IDLS (IDLE State), but it is also used to change the current time or alarm time when we are editing the values. The values are increased on each press; eventually we'll introduce a second button to decrease them.

The red button (BR) is used to change the machine state. Contrary to other button, this button is active HIGH, but on the other hand, is also the only button which executes an action on button release instead of button press. This button also has a dual mode of operation: short button press or long button press, using the `millis()` function that returns the milliseconds elapsed since the Arduino was turned on or reset. A short press from IDLS state transitions the machine into the alarm editing states, while a long press enables current time editing states. This button can, from a user perspective, be seen as the "OK" button.

The DPDT ON-ON (BL) switch is used to enable the alarm on the currently set alarm time or to disable it. In the future, the alarm will be disabled in a different, more intricate, way. Contrary to previous buttons, which are

being polled, this button is read through an interrupt when there is a change on the associated input pin. Despite being an ON-ON button, it is wired as a normal ON-OFF button.

A finite state diagram can help visualize the Mealy machine state transitions:



Buzzer

The buzzer we are using is a passive one, which means it requires an AC signal to produce sound. The Arduino `tone()` function generates a square wave of the specified frequency and it is used to play a custom melody which loops until the alarm is disabled.