Have you ever woke up early in the morning to find out that the kids (or adults) in your home forgot to turn off the lights? I've had that happen a number of times in my kitchen, which has 5 lights built in. A lot of wasted electricity.

My project will monitor lights in any room between 10 pm and 5 am. This of course can easily be changed in software. The first time lights are sensed, a short beep will occur. Then a grace period of 30 minutes or 1 hour will take place. No alarm during this grace time. This will be selectable with a switch. The grace period is for the midnight snacks or the person who can't sleep and goes to watch a little TV. Not enough grace time? Press a switch to reset the grace period.

If the person walks away without turning off the lights, then after the grace period, an alarm will sound off to remind someone to turn the lights off. If lights are off, then the device will simply go back being a clock and continue to monitor until 5 am.

This project will teach you about working with Arduino Uno, LCD display, real time clocks (RTC), piezo buzzers, and photocell sensors.

Time Required: 3 hours depending on experience

Experience Level: Intermediate

Required tools and parts:
- Needle nose pliers
- Wire strippers
- Cutters
- Soldering iron- solder

Bill of Materials:

<table>
<thead>
<tr>
<th>Qty</th>
<th>Jameco SKU</th>
<th>Component Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2151486</td>
<td>Arduino Uno</td>
</tr>
<tr>
<td>1</td>
<td>2170839</td>
<td>AdaFruit DS1307 Real Time Clock Breakout Board Kit</td>
</tr>
<tr>
<td>1</td>
<td>2118598</td>
<td>DISPLAY,LCD,16X2</td>
</tr>
<tr>
<td>1</td>
<td>2137756</td>
<td>Pass-Through Female Headers for Arduino Shields</td>
</tr>
<tr>
<td>2</td>
<td>241146</td>
<td>POTENTIOMETER,10K OHM,3299W-103.25-TURN,.5 WATT,CERMET</td>
</tr>
<tr>
<td>1</td>
<td>691104</td>
<td>Resistor Carbon Film 10k Ohm 1/4 Watt 5% (In Bags of 10 and 100)</td>
</tr>
<tr>
<td></td>
<td>20723</td>
<td>BREADBOARD 6.5 X2.125 830PNTS 100/CS</td>
</tr>
<tr>
<td>1</td>
<td>105180</td>
<td>Prototype Builder 5&quot; X6.5&quot; PCB Epoxy Glass 0.062&quot; Thick 0.1&quot; Hole Spacing 0.08&quot; Pad Size</td>
</tr>
<tr>
<td></td>
<td>202362</td>
<td>PHOTOCELL,90mW,150VPACK,10Kohm,MAX LITE,20Mohm MIN DARK</td>
</tr>
</tbody>
</table>

Ten resisters minimum but the project only uses 2.

Bigger than needed. Will be cut to size similar to size of Arduino uno.
Other CDS photocells should work fine as there is an adjustment for light sense threshold.

<table>
<thead>
<tr>
<th>Item No.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BUZZER, PIEZO, 5Vp-p, 80dB, 4 kHz, 5mA, PIN CONTACTS</td>
</tr>
<tr>
<td>1</td>
<td>Tactile Switch Off Momentary (On) Single Pole Single Throw Round Button PC Pins 0.05 Amp 12 Volt Through Hole</td>
</tr>
<tr>
<td>1</td>
<td>Resistor Carbon Film 330 Ohm 1/4 Watt 5% (In Bags of 10 and 100)</td>
</tr>
<tr>
<td>1</td>
<td>Wire Jumper Kit 350 each 22 AWG 14 Lengths 10 Colors 25 Of Each Length</td>
</tr>
</tbody>
</table>

**Step 1 - Build protoboard shield**

- Using a hack saw, cut the protoboard to a minimum dimension of 2-1/8 inch by 2-5/8 inch.
- Arrange the four Arduino long pin headers as shown in the picture below. Use the Arduino Uno to help align the long pins into place.

![Protoboard Shield](image)

**Step 2 - Solder headers to protoboard.**

- Turn the assembly upside down and solder each pin while the proto shield is still attached to the Arduino Uno. Make sure the black headers are flush to the protoboard as you solder it in place.
- Place masking tape on the headers like I did to mark the name of the pins that correspond to the Arduino pins. The names of the Arduino pins will be hard to see when this shield is installed.

![Soldering Headers](image)
**Step 3 - Add breadboard to shield**

- On back of the breadboard, cut along the length of adhesive pad on one side of the power rails. You can choose either side.
- Remove this power rail as shown in the picture.
- Cut enough of the paper backing on the adhesive pad to cover the protoboard surface where the breadboard will sit. Install the breadboard to the protoboard as shown.

![Step 3 - Add breadboard to shield](image)

**Step 4 - Position LCD display**

- Add LCD to protoShield just built as shown in picture.
- Jump GND and 5V from the Arduino headers to the breadboard power rail as shown.
**Step 5 - Wire up LCD**

- Use the schematic shown to make all electrical connections to the LCD.

**Step 6 - Test LCD**

- Open the Arduino software environment and click on the open icon (arrow pointing up between New and Save.).
- Go near the bottom of the menu and choose LiquidCrystal > Display as shown in picture.
- The C program will show in the Arduino window.
- Find this line of code: `LiquidCrystal lcd(12, 11, 5, 4, 3, 2);`
- Change it to this: `LiquidCrystal lcd(5,6,7,8,9,10);` - Verify then upload this code into your Arduino uno.
- Install the protoShield with the LCD on to the Arduino.
- Power up the Arduino. You should see "Hello World" blinking on the display.
- If not, turn the 10k pot in both directions until you do. Got it?
- If not, check your connections against the schematic. Don't go further until this step is working.
Step 7 - Show LCD is working

- If all is good so far, then you should see what the picture shows.

Step 8 - Position RTC

Add RTC in general area as shown in picture.
Step 9 - Wire up RTC

- Wire up RTC using the schematic shown.

Step 10 - Get RTC library

- Open a new tab on your browser and copy and paste this URL:
  https://github.com/adafruit/RTClib
- Click on the "Download ZIP" button on the bottom right of the page. Open the folder where your downloads go to and extract the zipfiles.
- Change the name of the folder you just unzipped to remove the dash ("-"). The Arduino IDE doesn't like this. You can rename it to just RTClib_master for example. Underscores are OK. Dashes are bad. - Right click on the filename and copy the folder.
Step 11 - Install library

- Paste the file with the new name (I call mine RTClib_master) to the library subfolder of Arduino folder as shown in the picture. This is the Arduino folder that was created when you installed the Arduino IDE on your computer.
- If not open, then open the Arduino IDE and click on the OPEN icon (up arrow with dots on bottom). You should see the RTC library you just installed near the bottom of the list that comes up.
- If not, you probably stuck it in the wrong folder or you didn't change the name so the dash or spaces are not part of the name. Underscores are OK.

Step 12 - Set the time

- Go to the ‘open’ icon and go to the RTClib_master > ds1307. This will open up a program in the Arduino IDE.
- Verify and upload this program. Reset your Arduino Uno with the reset switch or by removing usb power.
- Open the serial monitor icon and make sure the bottom right pulldown is set for 57600 baud.
- You should see the correct time and date on the serial monitor window. (LCD will be blank for now.)
Step 13 - Verify time and date.

- If time and date is showing correct then continue to the next step. If not, repeat step 12.

Step 14 - Download software for project.

- Copy and paste this URL to your browser: https://github.com/Hacker62/LightsOnReminder
- Click on Download ZIP on bottom right of web site.
- Go to folder where the file downloaded to and right click on the zip folder to do an "Extract All".
- In the extracted folder with the same name as the zip folder, open the text file named "clockWithRtc". Right click on the white space and choose "select all".
- Right click again to choose "copy".
- Open the Arduino IDE and right click on the white space of the empty sketch and choose "paste".
- Verify and upload the program to the Arduino Uno.
- If all goes well you should see the correct time you set now on the LCD as shown in the picture.
Step 15 - Hookup the remaining components.
- Add the remaining components using the schematic shown.

Step 16 - Load application software
- When you downloaded the zip file in step 14 and extracted it to your folder of choice, there should also be a text file called "Lightson Reminder". Find this file and open it.

- Right click on any white space on the file you opened and click on "Select All". Right click on any white space again and click on "Copy".

- Open your Arduino IDE and on the blank sketch that comes up, right click on any white space and click on "Paste". Save your file and give it a name you want.

- On the Arduino IDE, click on the checkmark icon to verify the code. It should pass with no errors.

- Plug in your Arduino Uno to the USB connector and press the upload icon (arrow pointing to the right). On the last step (#17) we willfully test and adjust any parameters that need tweaking. Your complete hardware should look similar to the picture on this step.
Step 17 - Test Project

- Near the beginning of the program for this project, there is a section marked as " // Easily change adjustable variables here." Go there and make these changes. See picture for example.

- Change gracePeriod to 3 minute and startMonLgt to the hour previous to the time you’re in now. For example, if it’s 2:10 pm for you, then adjust it to 14 (time is in military format).

- If there's light in the room, then you should hear a beep and the LCD display should say "Minutes left:" with the minutes on the second row showing you 3.

- Find the serial monitor button on the top far right of the Arduino IDE. Press it and another window should show. Make sure the bottom right of the window is showing 9600 baud. If not then pick it form the list.

- The analog value of the photo detector sensor should be scrolling on the left. Divide the max value you can see in half and enter it as the threshold value in the adjustable variables section of the code.

- Cover the photo detector and the analog value should fall to a low value such as less than 50. Also note that the LCD will show you the time instead of counting down the minutes.

- When you uncover the photo detectors then a beep will sound again and the countdown of minutes start over again.

- Press the tactile switch until the minutes go back to 3 minutes. This verifies the more grace time circuit works.

- Let the minutes count down until the beeper starts which will remind someone that the lights need to be turned off.

- If all functions as described above, then adjust the gracePeriod to the minutes you desire.

- Also set the startMonLgt and endMonLgt to the times you want your project to start and end the monitoring of lights. I set mine for 21 to start and 5 to end (9 pm to 5 am).