STEMSEL Intermediate Project 6: Light Sensor Alarm

Problem

What items are important for survival in an emergency situation? How can we secure our goods? We want to create an alarm that can work even in a dark room by detecting small changes in the light level.

Background

Securing your belongings is very important, especially in an emergency situation. We'd like our food and money stored safely. If we are busy, hungry and angry birds or animals come to take away our food, we want something to works like a guard to alert us. In this project we will design an automatic guard.

Ideas

What can we use to detect the change in the light level? If we are only looking for a small change we can't use a threshold like in projects, so how can we see if there has been a change? How can we tell if the alarm is on and working? Will the alarm detect us as soon as we set it, or should we have a delay before it starts working? How should the alarm tell us it has detected an intruder?

Plan

In our kits we have two light dependent resistors (LDRs). These sensors act like an electronic eye as it can detect light and tell us about the light level in the environment. If we connect this electronic eye to our STEMSEL controller board the microchip can use it to see the light level. However, we do not want this alarm to detect us leaving the room, so we will need a delay of about 20 seconds at the start to give us enough time to set one or more of these alarms and then get out of the room.

Once it is active, the guard should read the light level in the room to get an initial reading. It should then wait for a short period of time, then read the light level again to compare it with the previous level. If they are the same, then there is no change in the room. However, if they are different then the alarm should activate a buzzer and LED to let us know there is something there.

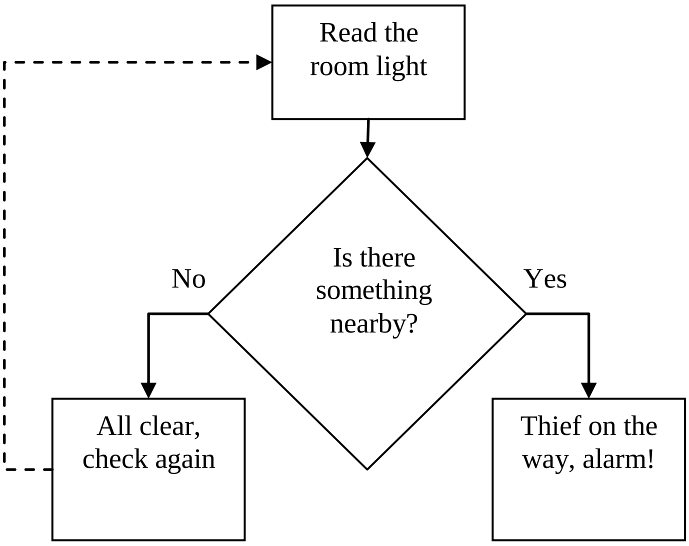
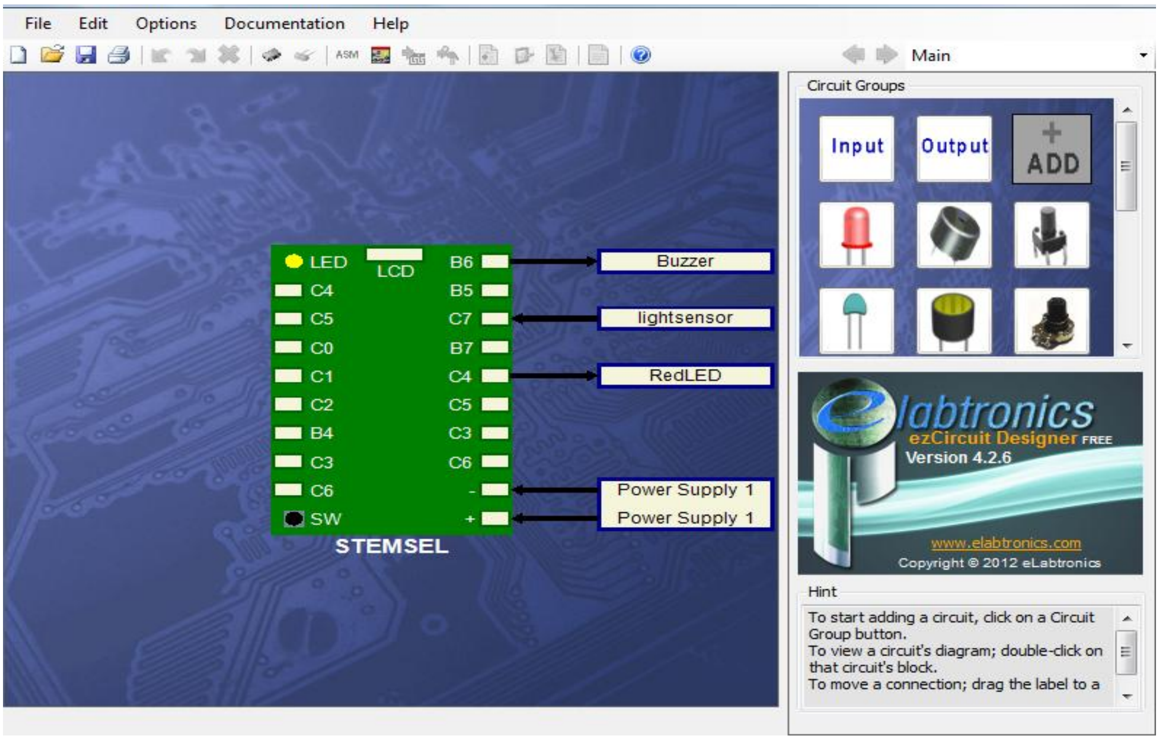


Figure 1: Action of the alarm

Design

Open the ezCircuit Designer. Create a new project using STEMSEL controller board as the chip. Add a LDR to Pin C7. Rename it to something you like.

Our guard can see the light, but it cannot speak to us. As we said in the plan, we can connect a buzzer and a red LED to the system so that it can inform us if any situation happens. Add a buzzer to pin B6 and an LED to pin C4. Now we have eyes and mouth for the guard.



To start adding a circuit, click on a Circuit Group button.
To view a circuit's diagram; double-click on that circuit's block.
To move a connection; drag the label to a

Figure 2: circuit design

Build the circuit

Now we can build the circuit. Remember that black wires connect to the negative pin, red wires connect to the positive pin, and white wires connect to the pin we designated in our design.

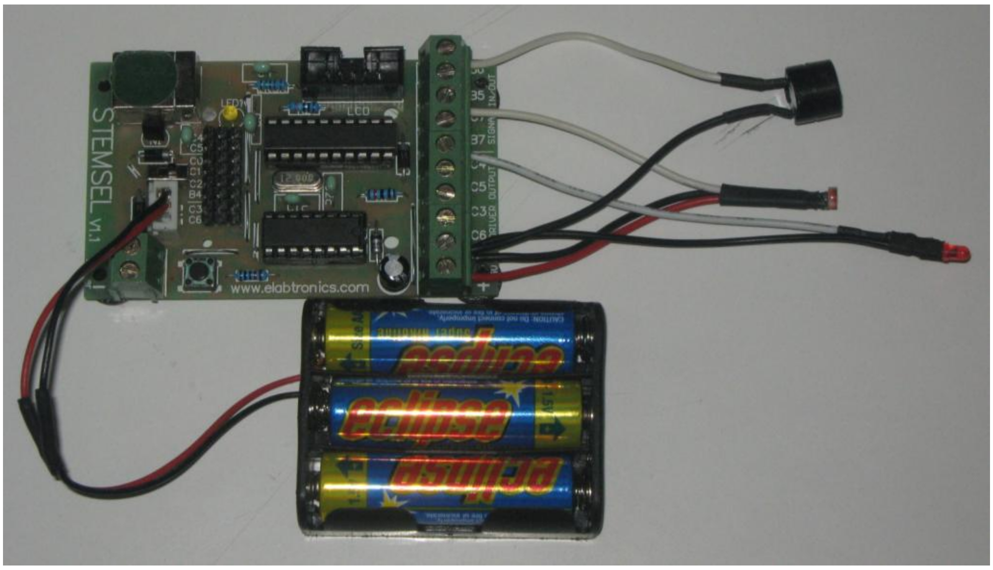


Figure 3: circuit

Programming

We need to assign work to our guard. Suppose you are storing some food in a room. We want our guard to stay next to the food and alert us if something comes close to try and steal the food.

Send the project to CoreChart and program the microchip with the test routines. Once you have verified the buzzer and LED are working correctly delete the test routines so we can write our program.

1. Just as we said in the plan, we want to have enough time so that we can leave the room without setting off the guard. Add a time delay of 20 seconds at the start of the program.

2. After the preparation is done, we would like the guard to inform us that it is active and then start to work. Add an OnOffPin icon to turn on the onboard C0_yellowLED after the time delay.

3. The guard should now start checking to see if there is anything moving in the room by collecting light data from the LDR. To get information from the guard’s eye, add an analog input to the system (Input -> Analog_In). Double click the new icon to take the value from LightSensor and save it as lightvolts. This would be the starting ambient light for our storage room since there’s nothing else in the room.

4. We’d like to compare the current light with the previous light so that we can know if there is anything entering the room. To do this, first we have to wait for a short period of time. Add a time delay of 0.25 second in the program (type 25 into HUNDREDTHS).

5. After this 0.25 second, the ‘lightvolts’ we measured in step 3 is actually a ‘previous light’. We need to save it somewhere else with a different name. Because our guard will possibly become very noisy (to alert us) if something happens, we have to adjust its sensitivity. Place an Add icon and use it to add 5 to lightvolts and save the result as previous_volts (If you find your guard is too sensitive or not sensitive enough you can change this number later). This will be the acceptable range mentioned in the plan.

6. Now that we have saved the previous light information, we need to know the current light level. Add another AnalogIn icon after the Add icon to save the lightsensor value as ‘lightvolts’

7. We now would like to see if there has been any change in the light level during this 0.25s of time. We can do this by comparing the current light level with the previous light level. Add a compare icon to the program and use it to compare lightvolts with previous_volts, and tick the ‘Above’ checkbox. Now, if the current light is suddenly lighter than the previous light it means something blocking the light has moved away, or maybe that someone is shining a torch on our guard. The guard will know that there is something there so we want it to warn us.

8. Add two OnOffPin icons to turn on the red LED and the buzzer underneath the Above decision icon. Group these OnOffPin icons together so that they can both work under the alarm situation (select them by holding shift on your keyboard, select group and rename it into alarm_on or any name you like)

9. We don’t expect the guard to work once and then stop; we want it to keep checking continually. Add an address named ‘check’ before the second time delay and a GoTo before the END of the program to create a loop (Address -> check, GoTo -> check). Ensure the GoTo sends the program back to check and not to START, otherwise the program will go back to the setup delay and any potential thieves will have 20 seconds to steal your food!

10. Download the program to the chip to see how it works. You can try passing in front of the sensor to make it activate.

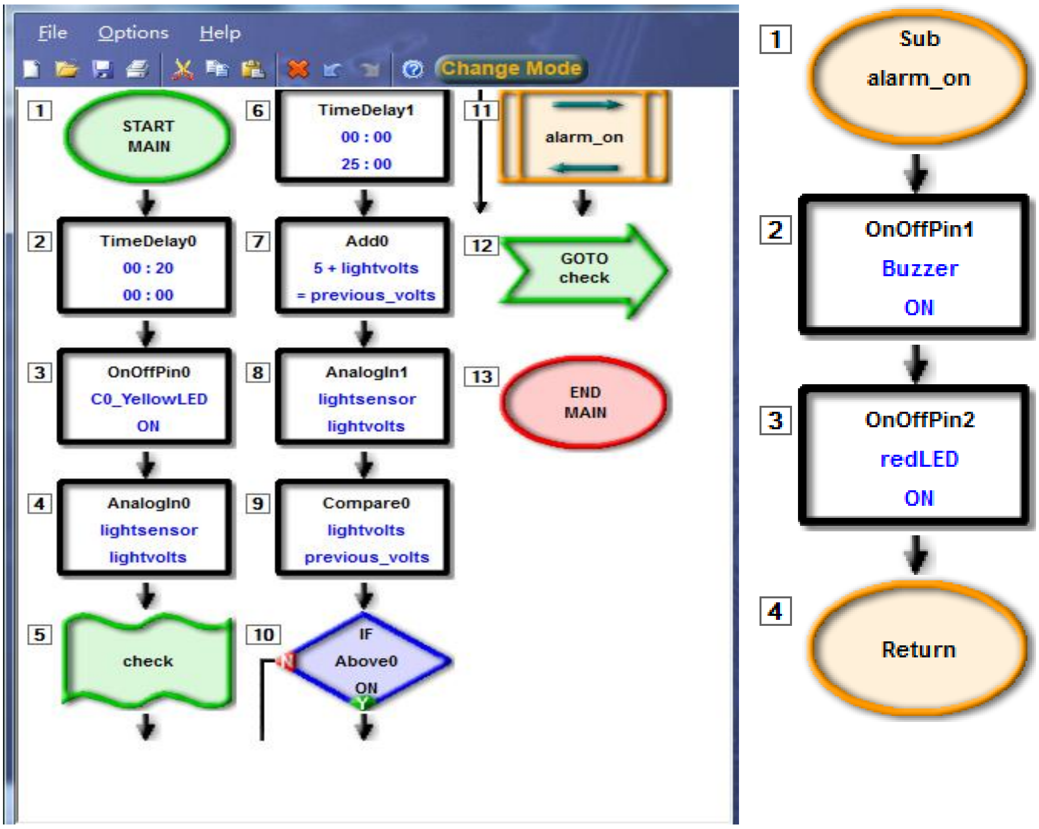


Figure 4: completed program

Explanation

The security guard will wait for 20 seconds then turn the onboard yellow LED on to tell us it's working. It then records the surrounding light level every 0.25 seconds and compares it with the previous light level. When there is a sudden increase in the light level (maybe someone passes by), it will beep loudly and turn on the red LED. The whole logic of the system works as follows:

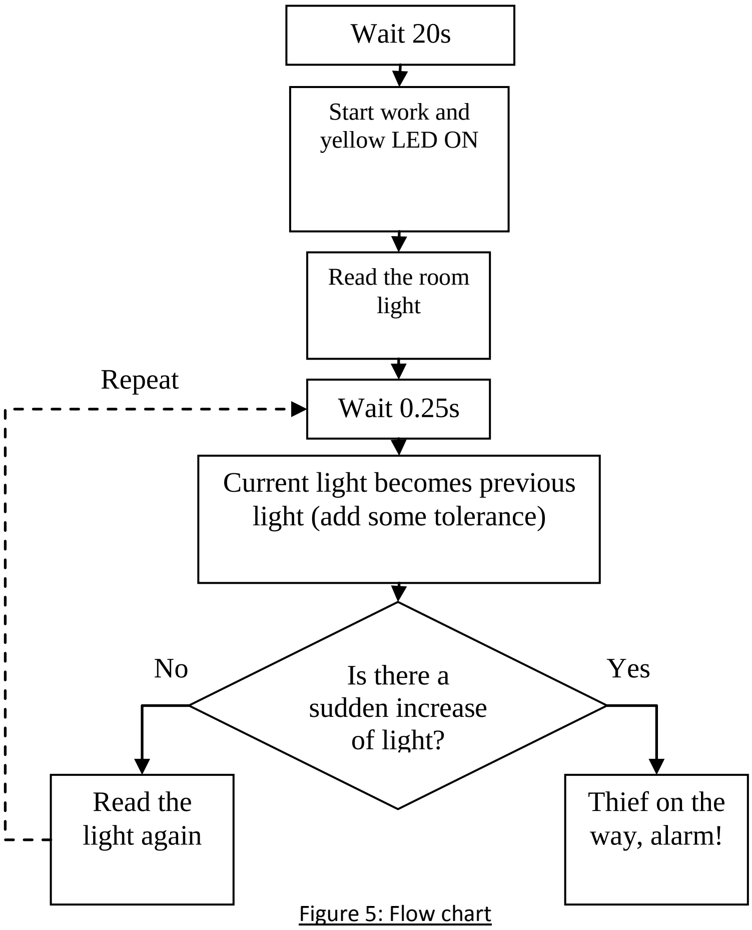

Wait 20s
Start work and yellow LED ON
Read the room light
Repeat
Wait 0.25s
Current light becomes previous light (add some tolerance)
No
Is there a sudden increase of light?
Yes
Read the light again
Thief on the way, alarm!

Figure 5: Flow chart

Question: What if the guard only compares the current light with the initial ambient light? Will the alarm still work the way we want it to?
(Hint: during the day the light will change)

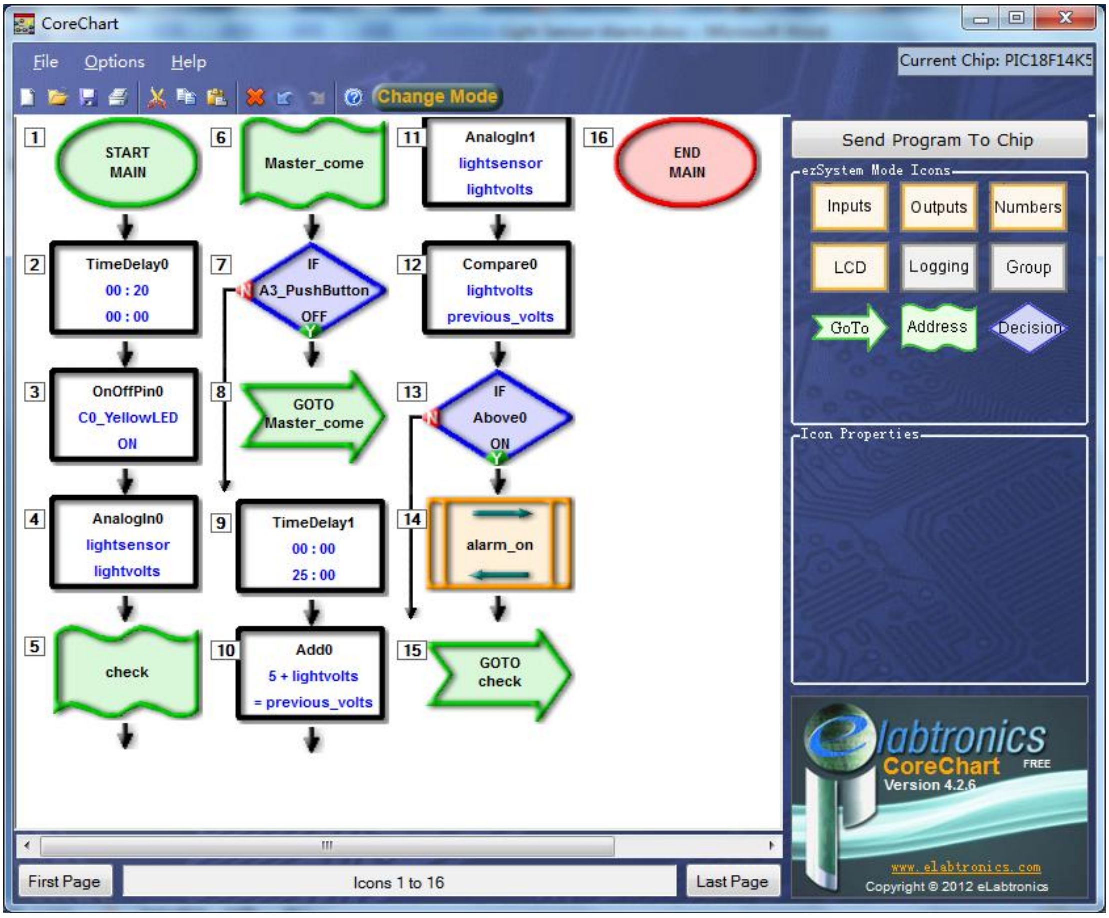
Activity

Place an object on a table then put all the guards around it. One by one everyone should try to see if they can take that object without setting off the guard. Were you caught by the guard? Did your guard catch the thief?

Discussion

1. We are currently storing our food in a bright room, but usually we store our food in a dark storage room. Also, the electricity could possibly go off during an emergency meaning the room will be even darker at that time. You would have to light the room before you can see anything. If somebody enters your storage room searching for food without your permission you can use this system to alert you. If we are using this system on this situation, is there any modification we have to make onto the system? What would our guard's logic be like in this situation?

2. Our guard is loyal. If it detects a change in the light level, it will shout. But wait! What if you are coming to get the food for yourself or to share it with your visitors? The guard is not smart enough at the moment, but we can upgrade it. Is there any method we can use to show the guard our identities? (Hint: STEMSEL has a push button mounted on the board. We may skip the guard's work by pressing the button. You may find the following diagram useful). Try to modify the system so that it will be smart enough to avoid this situation.



Summary

Even during an emergency it is important to protect certain items, not just valuables but also food and clean water. Today we learned to make a security guard that would detect light changes and alert us by beeping and turning on a LED. The guard could protect our food from thieves, but remember to help others if an emergency comes. Make sure you change your guard so that it will be smart enough to help us all.