Scott Leonard   
Mattics Phi   
CPE 329-03   
Spring 2012   
4/30/12   
John Oliver

Project 3: Environmental Sensing: Occupancy Sensor

**Brief Statement of the Project**  
The goal of this experiment was to implement an occupancy sensor with the Arduino Uno using a PIR motion detector and a solid state relay system.  A motion sensor was used to detect motion in a confined space, like a room.  The general purpose of an occupancy sensor is to react in a useful way to human presence in a fixed area.  Our system works by activating lights in the presence of people without the action of a physical switch by the occupants.  When people exit the space, the sensor turns off the lights when presence is not detected for some time, saving power for the building.  Occupancy sensors are also used in security applications, where they can be used to trigger alarms to deter intruders.

**System Requirements**  
An enormous amount of electrical  energy is wasted in our society from active lights inside empty rooms.  Although sometimes it is desirable to light areas for safety and security reasons, many general purpose lights are left on by forgetful users.  An occupancy sensor is a simple and elegant solution to this problem.  By sensing the presence of people, an occupancy sensor system can turn on lights for a specific location, then deactivate them after the occupants have left.  

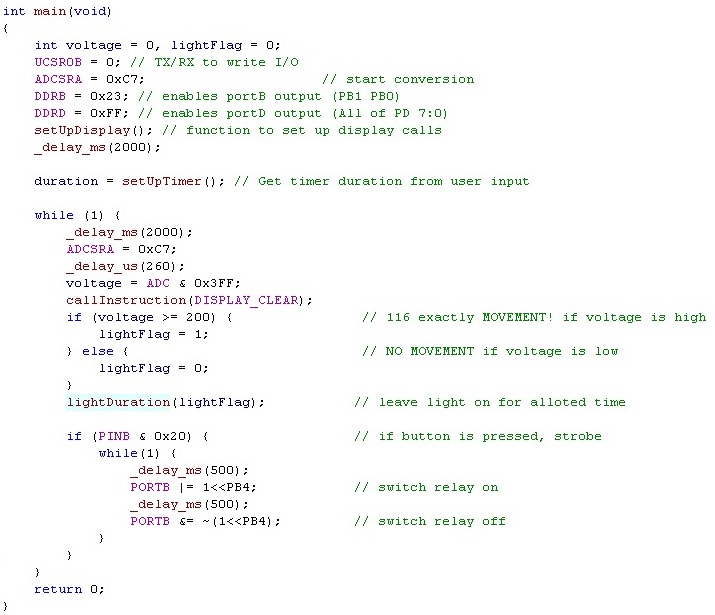
**System Specifications**  
Our sensor must be capable of detecting the presence of people over a fixed location.  This is accomplished by monitoring a set space for motion and reacting when movement is sensed.  For the system to be useful, the sensor must be able to distinguish the motion of people from other environmental changes, like sunlight or moving inanimate objects (like a moving fan).  This is accomplished by using infrared (heat) sensing to limit the sensors detection range to humans (or other mammals).  When the presence of people is detected, the sensor reports back to the microprocessor, which has control over a relay system.  This relay will switch on standard 120v outlet, bringing power to almost any device or subsystem.

**System Parameters**  
Our sensor must be capable of sensing motion in a medium sized room (about 15 feet by 15 feet).  The sensor will only trip if people are detected.  Upon detection, the sensor will bring power to a standard 120v outlet, capable of switching 15 amps through a solid state relay.  The turn off time for the relay must also be adjustable by the user upon system start-up, so users can change the amount of time the outlet is powered after they leave.   
  
**Partitioning of Software and Hardware Tasks**  
For this project, Mattics wrote the code for the occupancy sensor system, handling the software control of user input and feedback.  In addition, Mattics interfaced the PIR motion sensor with the Arduino controller.  Scott was tasked with the construction of the solid state relay system and its 120v interface, in addition to selecting system components.  Both teammates handled the final testing of the system.  The final system layout is shown below is **Figure 1**.

**Figure 1: Block Diagram of System**

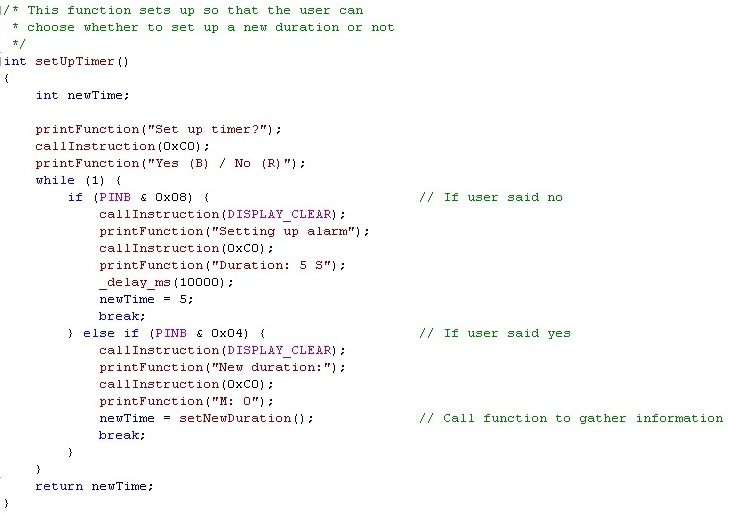
**Design and Selection of Components**  
A PIR sensor was used instead of a traditional motion sensor because of its ability to distinguish inanimate objects from living things.  Infrared sensors are also better at detecting user presence even when they are producing very little motion.  A solid state relay was used in place of a latching relay for improved efficiency, reliability, and output power.

**Code Examples**  
The software to run this project is based around a couple different tasks.  First and foremost, the program is run through a while loop in the main function.  The function of the while loop is to keep the program running while keeping track of button inputs.  The code example can be seen below in **Figure 2**.



**Figure 2: Main Loop in Main Function**

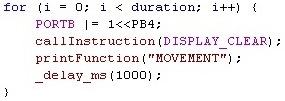
First and foremost, the code asks the user if they want to input a new duration or use the default one.  If the user wants to input a new duration, then the user will press the correct button accordingly and then follow the instructions on how to choose a new time.  If the user would rather use the default duration, then the user would press the corresponding button and then continue on with the program.  The code snippet can be seen below in **Figure 3**.  The amount of time the light is on is calculated by converting the user’s input into seconds.  By putting that in a “for loop” that has one delay instruction, the duration of the light is calculated.  The code for this can be seen below in **Figure 4a** and **Figure 4b**.



**Figure 3: Setting Up the Timer Duration**

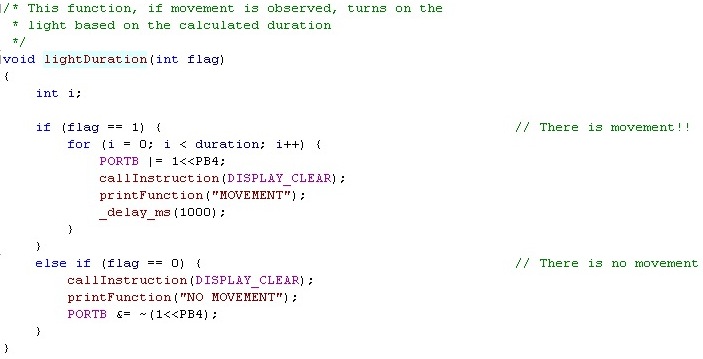
**Viet's HD:Users:Mattics:Desktop:Screen shot 2012-05-14 at 12.19.03 AM (2).jpg**

**Figure 4a: Converting the Time into Seconds**

****

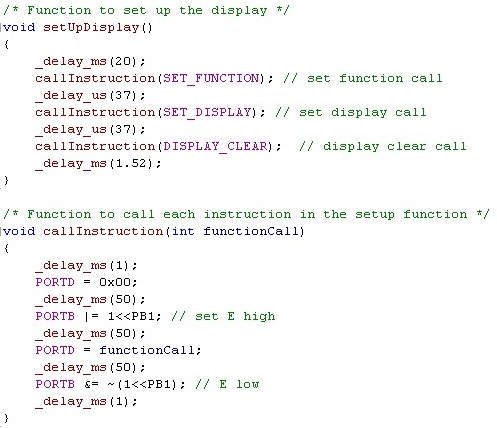
**Figure 4b: Delaying the Timer to Correct Duration**

Whenever the PIR sensor detects motion, a flag will appear in the code and that flag will toggle the relay.  The amount of time the light will be on will depend on the user’s decision earlier in the code.  The code snippet can be seen below in **Figure 5**.

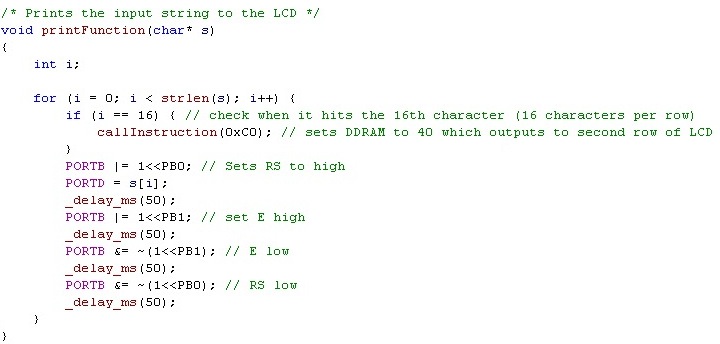
****

**Figure 5: Switching the Relay and Timer**

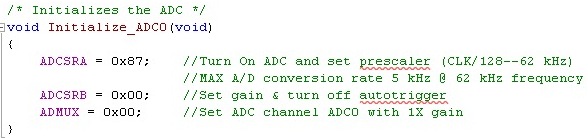
Some other code that was reused from past projects include: printing to LCD function, setting up the LCD function, and initializing the ADC function.  All three of these functions can be seen below in **Figure 6a**, **Figure 6b**, and **Figure 6c**.



**Figure 6a: Function to Set Up the LCD**



**Figure 6b: Function to Print to the LCD**

****

**Figure 6c: Function to Initialize ADC**

**Schematics of Project (Wiring Diagram)**  
I’ll do the wiring diagram in class tomorrow.

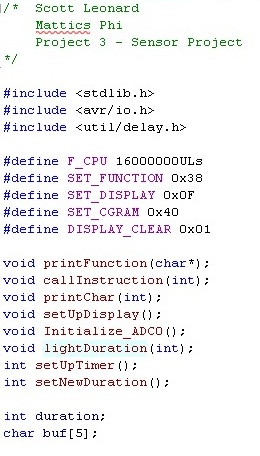
**Circuits Designed**  
Reference wiring diagram here I guess...

**System Integration**  
The first component that was implemented in our system was the PIR motion sensor.  The sensor needed two power lines from the Arduino board and a third wire to report the presence of motion.  After this component was tested, an LCD screen was added to the board to report set-up information back to the user.  Then, two buttons were added to allow the user to input a turn-off time on the sensor from a simple interface on the LCD.  Finally, the solid state relay module was connected to the board for complete system functionality.  To take advantage of our last open output port, an additional button was added to activate a strobe feature on the relay.  This demonstrated the high speed switching time allowed by the solid state relay.

**Testing Procedures**  
The system was initially tested by simply waving hands in front of the sensor module to establish basic functionality.  When the control line of the sensor was able to toggle between high and low upon the presence of motion, it was quickly integrated with the rest of the system.  The sensor’s performance measures were then evaluated by placing it in different sized rooms and lighting conditions.  In addition, various items were moved around the sensor to verify that it could distinguish between living things and inanimate objects.  Finally, the detection angle of the sensor was verified by moving 360 degrees around the sensor and monitoring its feedback to the controller.  The sensor behaved correctly in accordance with its datasheet and our required specifications.

**Observations**  
Only one notable problem was encountered during the construction of our system.  The 120v solid state relay that we intended to use in our system could not be activated by the Arduino Uno’s standard output port power.  This was unexpected because the solid state relay datasheet did not accurately specify the input power required to activate its switching functionality.  To solve this problem, the relay output line from our Arduino was instead connected to an NMOS transistor.  When the relay needed to be turned on by the system, the Arduino changed this semiconductor’s operating state from cut-off to saturation, creating a short between two of its three terminals.  This semiconductor switch was used in conjunction with our Agilent benchtop supply, which allowed the solid relay to be activated at 20 volts instead of 5.  This provided us with more than enough power to trigger the relay when motion was detected, without drawing any significant power from the Uno.

**Conclusion**  
When our occupancy sensor was finally assembled, the system prompted the user for a turn-off delay time from the LCD display.  If a turn-off delay was selected, the system then prompted the user for a desired delay time, which could be entered in minutes and seconds using two buttons.  If no delay time was selected, the system would turn off the power relay as soon as no motion was detected.  In both modes of operation, the LCD reported if motion was currently being detected by the sensor in real time.  The solid state relay system was successfully interfaced with the rest of the system using one input control line.  Because this relay handles high voltages and currents, the entire relay body and its 120v outlet were encased in grounded junction boxes in accordance with standard electrical codes.  This guarantees complete safety against electrical shocks during system operation. Aside from our minor difficulties with activating the solid state relay, no major issues were encountered with the system.  Our final product worked perfectly according to our proposal for an A grade.

**References and C Code**  
Solid state relay: <http://www.fotek.com.hk/solid/SSR-1.htm>  
PIR Motion Sensor: <http://www.sparkfun.com/datasheets/Sensors/Proximity/SE-10.pdf>  
LCD Screen: <http://www.sparkfun.com/datasheets/LCD/GDM1602K-Extended.pdf>  
ATMega328P Datasheet

