Indoor Light Automation

Shashank R

June 21, 2020

1 Introduction

This module is built mainly for indoor PIR sensor testing. This module has household application like home automation. The module contains a PIR sensor integrated with a microcontroller which controls a relay for switching of 220V AC load (bulb for example) connected to the module.

2 Technical Details

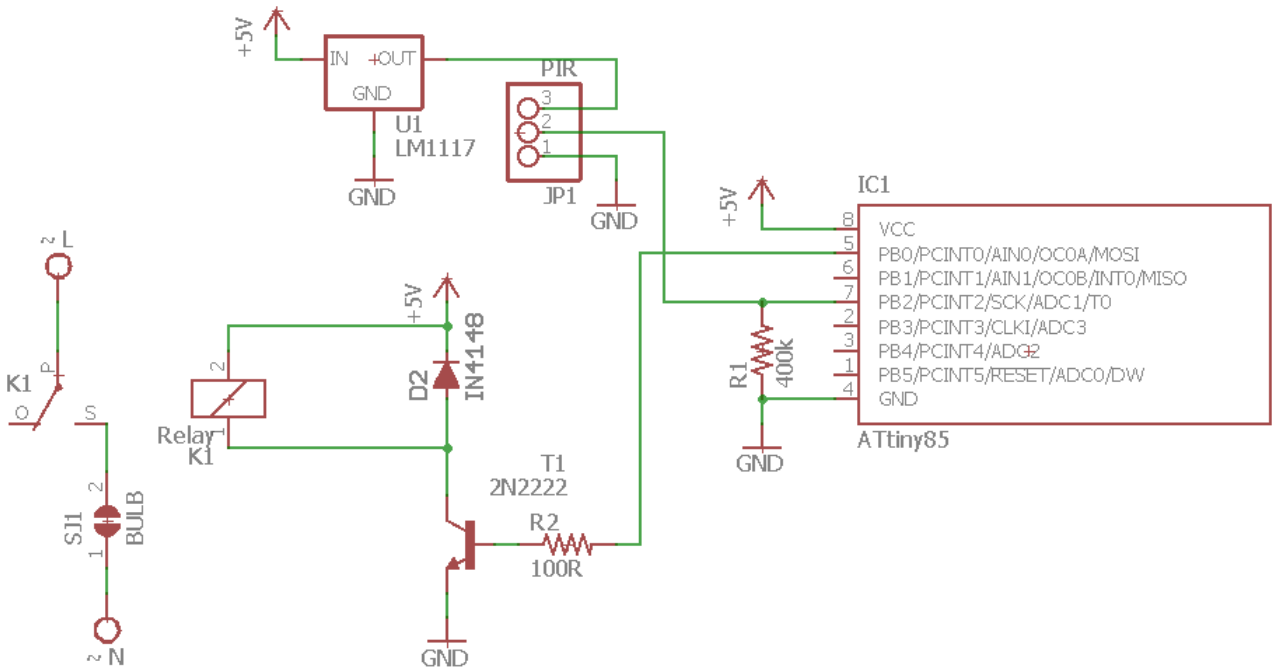
2.1 Circuit Components Used

1. PIR Sensor - Panasonic EKMC1603111, 12m range.
2. Microcontroller - ATtiny85.
3. Transistor - 2N2222A.
4. Resistor - 100Ω $1/4W$; $400k\Omega$ $1/4W$.
5. Diode - IN4148.
6. SPDT Relay - 5V DC/ 220V 7A AC.
7. Voltage Regulator - 3.3V LM1117.
8. 4.3V/350mA SMPS (isolated).
9. Bulb and Bulb holder.
10. Miscellaneous: GCB, jumpers, IC base, etc.

2.2 Circuit Design

A simple circuit with minimal components was designed to reduce space consumption and cost. The circuit uses a 8-pin cheap microcontroller called ATtiny85 which is best suited for running codes consuming very less program memory. The circuit schematic is shown below. The supply voltage for the circuit is 4.3V which is supplied by a 4.3V/350mA SMPS with isolation from mains. A 3.3V regulator ic - LM1117 is used to set the operating voltage of the PIR sensor to 3.3V. When the PIR sensor detects motion, the PB0 pin of the microcontroller goes HIGH which drives the transistor 2N2222A into saturation, actuating the relay coil which switches on the bulb (or the connected load). The microcontroller delays for a dead time of 30 seconds during which period the relay is actuated. After this delay the microcontroller reads the sensor data for every 100ms. This reading of data continous for the next 30 seconds and the relay is switched off only when no data/signal is detected from the PIR sensor. If signal is deteted then the dead time delay of 30 seconds during which the relay is actuated, prevails.

2.3 Circuit Schematic



2.4 Arduino Code

1. *void setup()* {
   pinMode(A1, INPUT);
   pinMode(0, OUTPUT);
   }
   Pins Analog Input 1 and digital output pin 0 are declared.

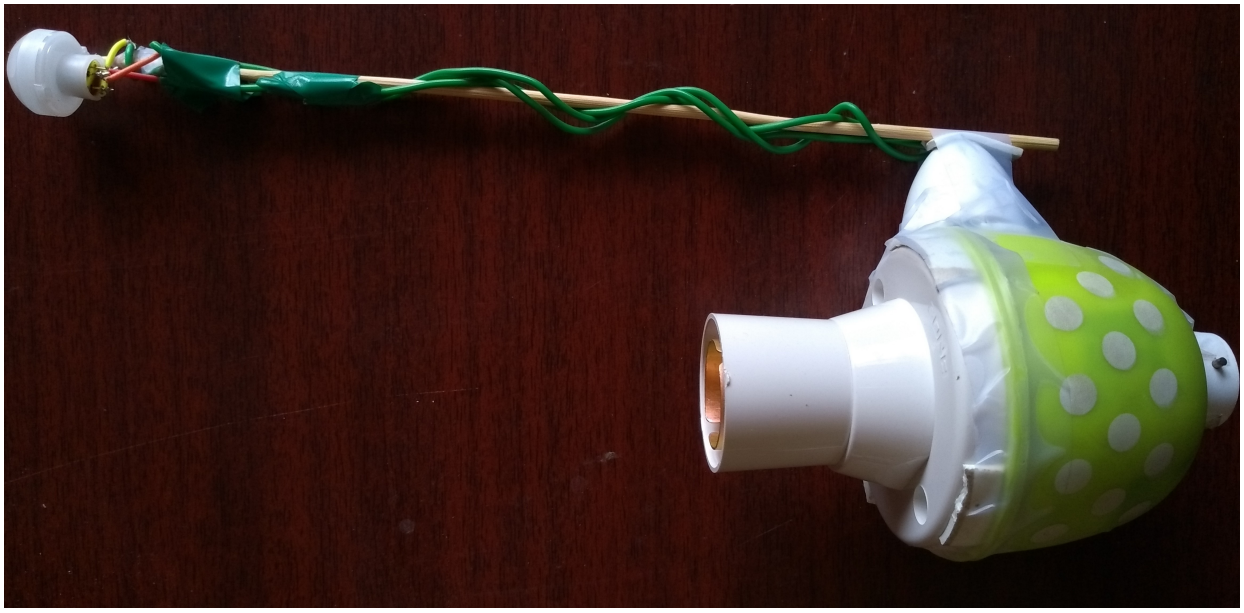
2. *void Dly(int dly)* {
   int j = 1;
   for(j = 1; j <= dly; j++)
   {
      delay(1000);
   }
   }
   Dly function is written for iterative delaying, to achieve long time delays. The delay occurs for *dly* seconds, delaying for 1 second for every interation, iterating *dly* times.

3. *void loop()* {
   int d = analogRead(A1);
   if(d >= 600)
   {
      digitalWrite(0, HIGH);
      Dly(30);
   }
   The code written in this point and next goes on looping forever. If the sensor value goes above 3V (equivalent to 600 in analog value of ATtiny85) then pin 0 (PB0) goes HIGH and delays for 30 seconds.

4. *else*
   {
      int i = 0;
      while((i <= 299)&&(d <= 600))
      {
         d = analogRead(A1);
         delay(100);
         i++;
         if(i == 299)
         {
            digitalWrite(0, LOW);
            delay(100);
         }
      }
   }
   }
   After delaying for 30 seconds, if no sensor data is detected, then the sensor data is read for every 100ms. This runs in a for loop 300 times which accounts for another 30 seconds. During this time pin0 is HIGH. It is only when no sensor data is detected after all this reading, that the pin0 goes LOW.

3 Pictures

Module:



Module as an add on to bulbs:

Module Working:

4 References

1. ATtiny85 datasheet: http://www.atmel.com/images/atmel-2586-avr-8-bit-microcontroller-
datasheet.pdf.

2. ATtiny85 programming using Arduino UNO as ISP: https://create.arduino.cc/
projecthub/arjun/programming-attiny85-with-arduino-uno-afb829

3. Panasonic PIR: https://na.industrial.panasonic.com/products/sensors/sensors-automotiv
pir-motion-sensor-papirs