

**ARDUINO SOLAR TRACKER**

PRESENTED

SWATI JHA (IT)

MENTOR:

ARPITA DAS

*ARDUINO SOLAR TRACKER*

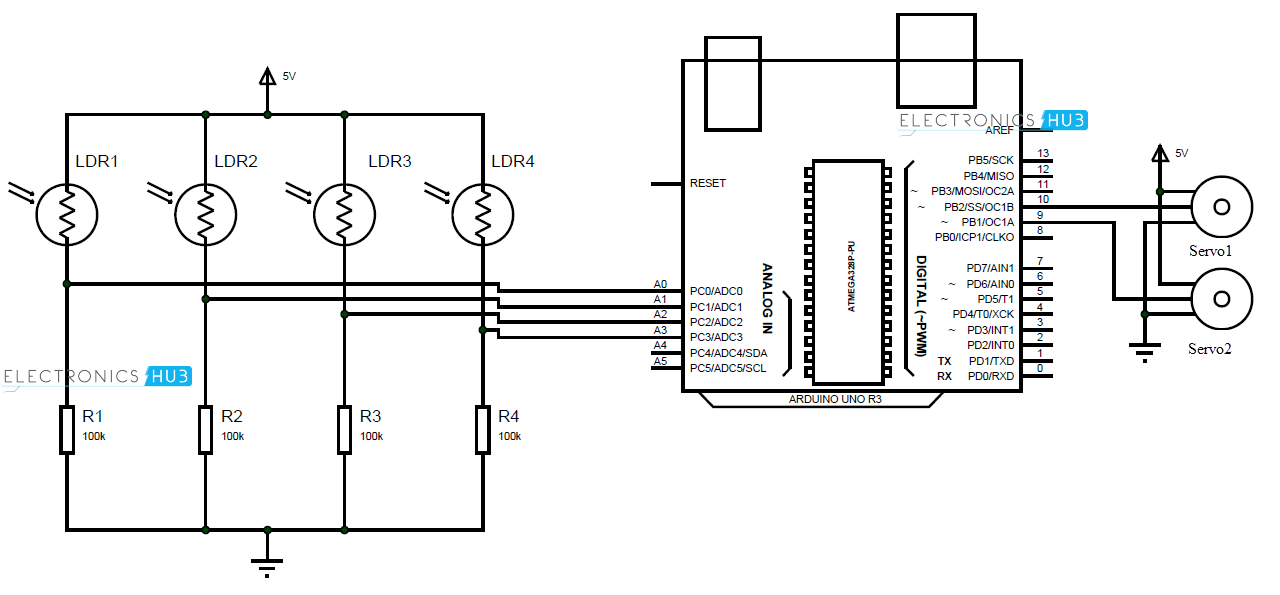
AIM OF PROJECT:

**To design an effective sensor array to provide directional information to guide a major drive system to track the movement of the sun which will position a solar panel to the best angle of exposure to sunlight where intensity is high for collection of maximum amount of solar energy.**

APPARATUS USED:

* **BREADBOARD**
* **SOLAR PANEL**
* **LIGHT DEPENDENT RESISTORS (LDRs) – 4**
* **SERVO MOTORS – 2**
* **ARDUINO BOARD**
* **RESISTORS (100k) – 4**
* **CARDBOARD**
* **CONNECTING WIRES**
* **JUMPER WIRES**
* **PERFORATED METAL STRIPS – 2**
* **BUS WIRE**
* **DOUBLE SIDED STICKY TAPE**
* **HOT GLUE**
* **SOLDERING IRON**

CIRCUIT DIAGRAM:



WORKING PRINCIPLE:

**LDRs are used as the main light sensors. Two servo motors are fixed to the structure that holds the solar panel. The program for Arduino is uploaded to the microcontroller. The working of the project is as follows.**

**LDRs sense the amount of sunlight falling on them. Four LDRs are divided into top, bottom, left and right.**

**For east – west tracking, the analog values from two top LDRs and two bottom LDRs are compared and if the top set of LDRs receive more light, the vertical servo will move in that direction.**

**If the bottom LDRs receive more light, the servo moves in that direction.**

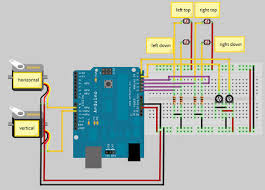
**For angular deflection of the solar panel, the analog values from two left LDRs and two right LDRs are compared. If the left set of LDRs receive more light than the right set, the horizontal servo will move in that direction.**

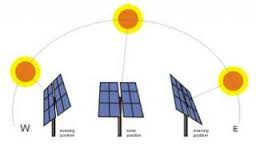
**If the right set of LDRs receive more light, the servo moves in that direction.**

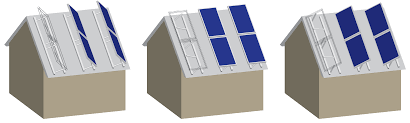
APPLICATIONS:

**Solar trackers can be mostly used in remote village areas where electricity through power plants cannot be reached. Solar trackers can be used in large scale industries where large amount of power is required, using renewable sources. When stationary solar panels are replaced by solar trackers, the efficiency is increased by 20% and can be used in housing complexes to save power. Further in a solar tracker if we connect a powerbank, it will be able to store the solar energy which can be used in charging various appliances and also provide power to generate electricity for futher use. We can also use a step up transformer to increase the power stored in the powerbank and use that power to light up a single house in a remote village area or even light up an entire village in an eco-friendly manner. As a result, in the near future we can use solar energy to generate electricity to be used in regions where electricity cannot reach, free of cost. Large scale industries can also use this solar energy to generate electricity according to their necessity as solar energy is a renewable source of energy.**

RELEVANT PICTURES:



REFERENCES:

* [**http://www.electronicshub.org/arduino-solar-tracker/#Working**](http://www.electronicshub.org/arduino-solar-tracker/#Working)
* [**https://en.wikipedia.org/wiki/Solar\_tracker**](https://en.wikipedia.org/wiki/Solar_tracker)

PROGRAM CODE:

|  |
| --- |
| #include <Servo.h> |
|  | //defining Servos |
|  | Servo servohori; |
|  | int servoh = 0; |
|  | int servohLimitHigh = 160; |
|  | int servohLimitLow = 20; |
|  |  |
|  | Servo servoverti; |
|  | int servov = 0; |
|  | int servovLimitHigh = 160; |
|  | int servovLimitLow = 20; |
|  | //Assigning LDRs |
|  | int ldrtopl = 2; //top left LDR green |
|  | int ldrtopr = 1; //top right LDR yellow |
|  | int ldrbotl = 3; // bottom left LDR blue |
|  | int ldrbotr = 0; // bottom right LDR orange |
|  |  |
|  | void setup () |
|  | { |
|  | servohori.attach(10); |
|  | servohori.write(0); |
|  | servoverti.attach(9); |
|  | servoverti.write(0); |
|  | delay(500); |
|  | } |
|  |  |
|  | void loop() |
|  | { |
|  | servoh = servohori.read(); |
|  | servov = servoverti.read(); |
|  | //capturing analog values of each LDR |
|  | int topl = analogRead(ldrtopl); |
|  | int topr = analogRead(ldrtopr); |
|  | int botl = analogRead(ldrbotl); |
|  | int botr = analogRead(ldrbotr); |
|  | // calculating average |
|  | int avgtop = (topl + topr) / 2; //average of top LDRs |
|  | int avgbot = (botl + botr) / 2; //average of bottom LDRs |
|  | int avgleft = (topl + botl) / 2; //average of left LDRs |
|  | int avgright = (topr + botr) / 2; //average of right LDRs |
|  |  |
|  | if (avgtop < avgbot) |
|  | { |
|  | servoverti.write(servov +1); |
|  | if (servov > servovLimitHigh) |
|  | { |
|  | servov = servovLimitHigh; |
|  | } |
|  | delay(10); |
|  | } |
|  | else if (avgbot < avgtop) |
|  | { |
|  | servoverti.write(servov -1); |
|  | if (servov < servovLimitLow) |
|  | { |
|  | servov = servovLimitLow; |
|  | } |
|  | delay(10); |
|  | } |
|  | else |
|  | { |
|  | servoverti.write(servov); |
|  | } |
|  |  |
|  | if (avgleft > avgright) |
|  | { |
|  | servohori.write(servoh +1); |
|  | if (servoh > servohLimitHigh) |
|  | { |
|  | servoh = servohLimitHigh; |
|  | } |
|  | delay(10); |
|  | } |
|  | else if (avgright > avgleft) |
|  | { |
|  | servohori.write(servoh -1); |
|  | if (servoh < servohLimitLow) |
|  | { |
|  | servoh = servohLimitLow; |
|  | } |
|  | delay(10); |
|  | } |
|  | else |
|  | { |
|  | servohori.write(servoh); |
|  | } |
|  | delay(50); |
|  | } |