ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 5, May 2018

ADQ Solar Tracking System (RTC/FCT) in ARM7 with Isis

[1] Srivani Athmakur, [2] Rohith Bandari
[1] Department of Electronics and Communication Engineering, INDIA
[2] Department of Mechanical Engineering, INDIA

Abstract: **-- As now-a-days renewable energy is a rapidly gaining notoriety. Specially, Solar filed is trending with new scope of life for a better change in efficiency with advanced sun tracking system. Here by using fuzzy logic with advanced quadratic RTC system in ARM7 by virtually controlling. By considering the meteorological data for a month to calculate the Altitude, Azimuthal angle by using the DC motor with LDR's to get accurate result with 50% more efficiency then normal result as applying the mirrors in by adjacent to panel. We use 3D accelerometer for better accuracy in fuzzy logic of controlling RS232 communication. Additionally, Filtering the unnecessary data by using optimization of logic. This system has advantage of controlling circuit in virtual media of schematic (ISIS PROTUES) and code controlling in keil4IDE(software).**

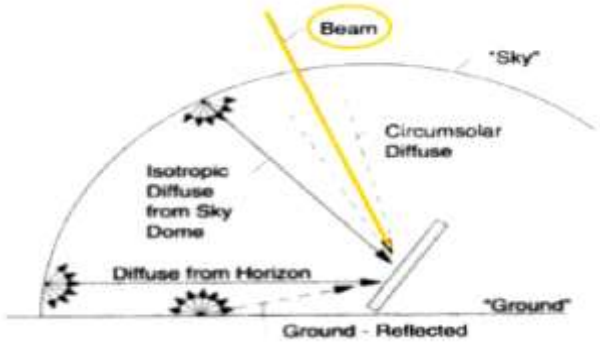
Keywords **— 3D accelerometer, δ, ω, θ, Az.ɳ.**

I. INTRODUCTION

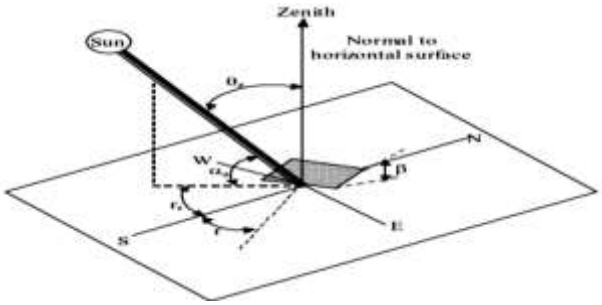
The aim of the project is to keep the solar PV panel perpendicular to the sun throughout the year/month in order to make it more efficient. The efficiency of the PV system depends on the meteorological conditions of solar radiation, ambient temperature and wind speed, matching of the system with the load and appropriate placement of the solar panels. The tracking system has the capability to always point the solar array toward the sun and can be installed in various regions with minor modifications. The fuzzy logic has been used to control the position using 3D accelerometer. A solar tracker will track the sun throughout the day and adjust the angle of the solar panel to make the sun normal to the solar panels at all times using the forecasting data. The orientation of the solar panels may increase the efficiency of the system from 30% up to 50%. The sun tracking solar power system is a electro-mechanical system that integrates electrical and mechanical systems, and computer hardware and software processes.

II. METHODOLOGY

ADQ SOLAR TRACKING SYSTEM designed using lpc2148. This microcontroller is compatible to connect all the components with accurate data of calculations to get Az, altitude angle and ɳ. This approach reduces costs, heat and power use. These are desirable for light, portable, battery-powered device with embedded systems. A simpler design facilitates more efficient multi-core CPUs and higher core counts at lower cost, providing higher processing power and improved energy efficiency for servers and supercomputers. This memory is more than enough for almost all applications. By considering the climate data of solar window here we have designed our system which is compatible for 8 direction's of simulation with the help of forecasting the data with RTC.



Earth condition w.r.t sun.



Position of sun w.r.t a point on earth's surface

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 5, May 2018

CALCULATIONS:

$$\delta = 23.45 \sin\left(360 \frac{284+n}{365}\right)$$

-
- $\theta = f(\varphi, \delta, \beta, \gamma, \omega)$
- $\cos\theta = A + B\cos\omega + C\sin\omega$

CONNECTIONS:

• ARM uses a 32-bit RISC processor in which port p01,p02,p03,p04,p05,p08,p09 are used to connect the LDR‟s fixed to the solar panel with the help of car jacker.

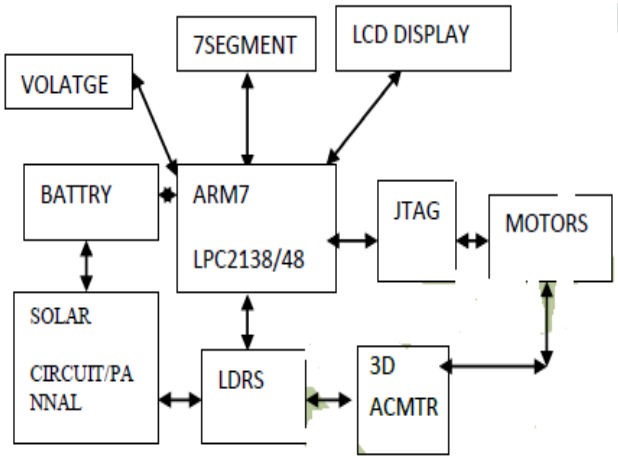
• Pins p18,p19,p20,p21 of port0 are being used to connect the dc motors which helps in moving the panel.

• The pins p20,p21,p26,p27 of port 1 are used to obtain the four quadrant system which helps in keeping the solar panel perpendicular to sun's direction always.

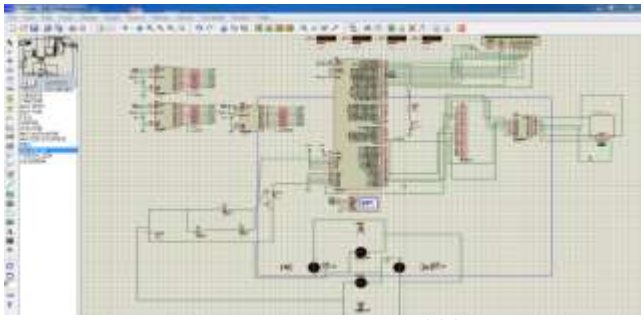
• The base motor is turned on if there is any light detected in Y-axis (i.e north or south direction).

• 3rd and 4th DC motor will be turned on, if light is detected in east or west position. These motors support wheels of vehicle to move in west or east position.

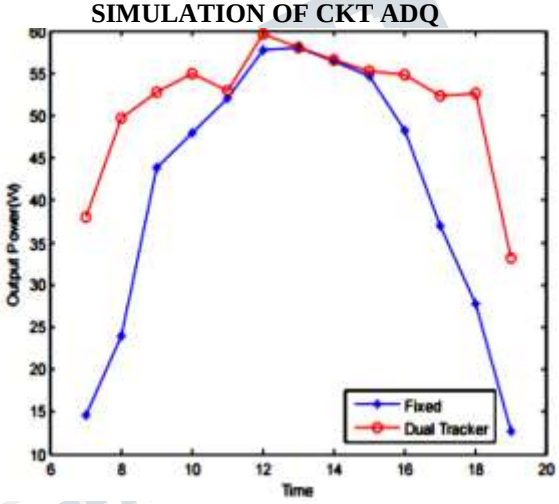
BLOCK DIAGRAM:



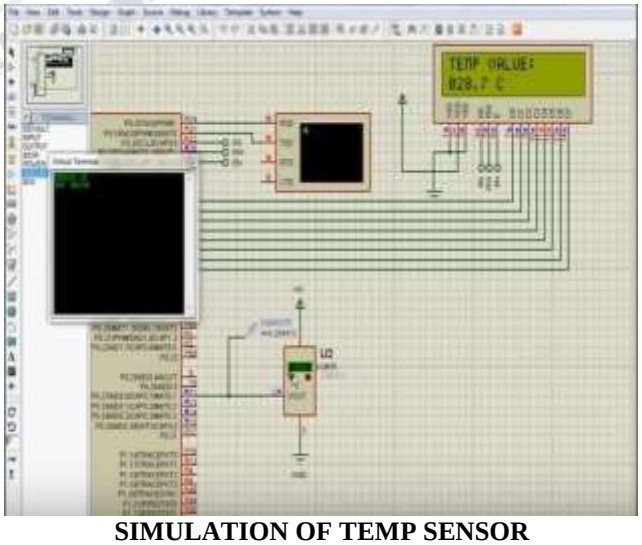
RESULTS OF ISIS:



SIMULATION OF CKT ADQ



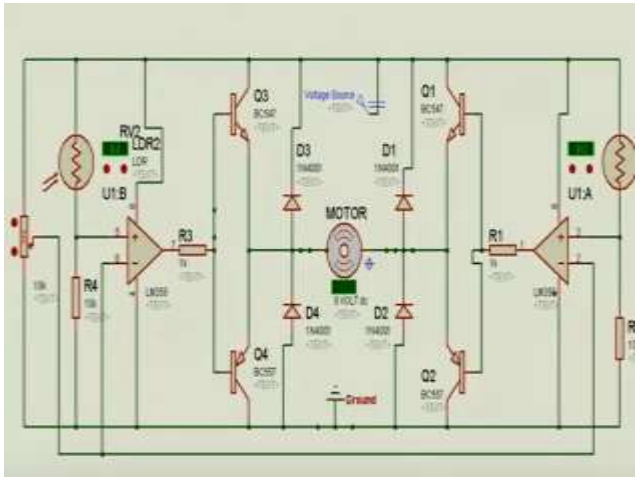
SIMULATION OF SOLAR CKT



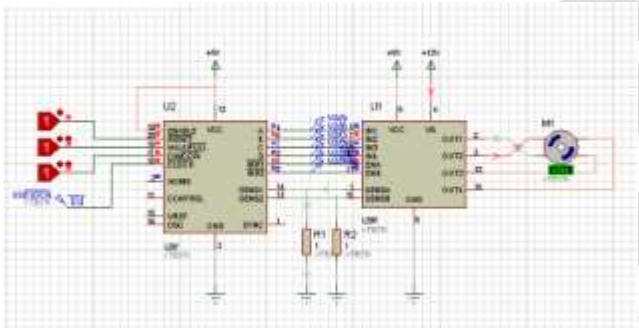
SIMULATION OF TEMP SENSOR

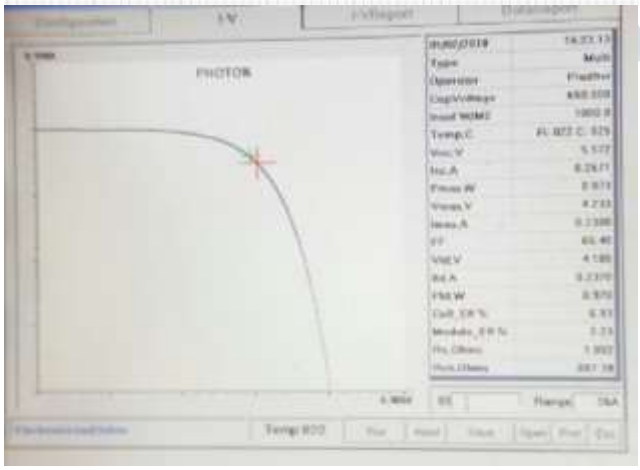
ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 5, May 2018



SIMULATION OF COMAPRISION OF FIXED AND DUAL





SIMULATION OF SOLAR CELL

#include<iostream> #include<iomanip> #include<cmath> static const double PI = 3.14159265358979323846, earthDiameterMeters = 6371.0 * 2 * 1000; double degreeToRadian (const double degree) { return (degree *

PI / 180); }; double radianToDegree (const double radian) { return (radian * 180 / PI); }; double CoordinatesToAngle (double latitude1,const double longitude1,double latitude2, const double longitude2){const auto longitudeDifference = degreeToRadian(longitude2 - longitude1); latitude1 = degreeToRadian(latitude1); latitude2 = degreeToRadian(latitude2); using namespace std;const auto x = (cos(latitude1) * sin(latitude2)) - (sin(latitude1) * cos(latitude2) * cos(longitudeDifference)); const auto y = sin(longitudeDifference) * cos(latitude2); const auto degree = radianToDegree(atan2(y, x));return (degree >= 0)? degree : (degree + 360);} double CoordinatesToMeters (double latitude1,double longitude1, double latitude2, double longitude2) { latitude1 = degreeToRadian(latitude1); longitude1 = degreeToRadian(longitude1); latitude2 = degreeToRadian(latitude2); longitude2 = degreeToRadian(longitude2); using namespace std;auto x = sin((latitude2 - latitude1) / 2), y = sin((longitude2 - longitude1) / 2);#if 1 return earthDiameterMeters * asin(sqrt((x * x) + (cos(latitude1) * cos(latitude2) * y * y))); #else auto value = (x * x) + (cos(latitude1) * cos(latitude2) * y * y); return earthDiameterMeters * atan2(sqrt(value), sqrt(1 - value)); #endif} std::pair<double,double> CoordinateToCoordinate (double latitude,double longitude, double angle,double meters){ latitude = degreeToRadian(latitude); longitude = degreeToRadian(longitude); angle = degreeToRadian(angle); meters *= 2 / earthDiameterMeters; using namespace std;pair<double,double> coordinate; coordinate.first = radianToDegree(asin((sin(latitude) * cos(meters))+ (cos(latitude) * sin(meters) * cos(angle)))); coordinate.second = radianToDegree(longitude + atan2((sin(angle) * sin(meters) * cos(latitude)), cos(meters) - (sin(latitude) * sin(coordinate.first)))); return coordinate;} int main (){using namespace std; const auto latitude1 = 17.968460, longitude1 = 77.641308,latitude2 = 17.967862, longitude2 = 77.653130; cout << std::setprecision(10); cout << "(" << latitude1 << "," << longitude1 << ") --- " "(" << latitude2 << "," << longitude2 << ")\n"; auto angle = CoordinatesToAngle(latitude1, longitude1, latitude2, longitude2); cout << "Angle = " << angle << endl; auto meters = CoordinatesToMeters(latitude1, longitude1, latitude2, longitude2); cout << "Meters = " << meters << endl; auto coordinate = CoordinateToCoordinate(latitude1,

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 5, Issue 5, May 2018

longitude1, angle, meters); cout << "Destination = (" << coordinate.first << "," << coordinate.second << ")\n";}

RESULTS: 1.Title towards NORTH direction:

Time	Solar altitude	Hour angle	Declination	Azimuth
8:00 AM	24.9	-63.29	16.8	79.7
1:00PM	83.7	5.983	16.89	251.8
5:00PM	-15.7	-61.1	16.7	65.52

2.Title towards SOUTH direction:

Time	Solar altitude	Hour angle	Declination	Azimuth	Elevation
8:00 AM	5.2	-56.17	0.273	79.61	0
1:00PM	-26.3	62.81	0.274	251.7	37.25
5:00PM	2.7	-71.63	0.275	65.51	0

3.Title towards EAST direction:

Time	Solar altitude	Hour angle	Declination	Azimuth	Elevation
8:00 AM	3.2	-72.122	0.273	72.4	0
1:00PM	87.9	-0.051	0.274	178.6	73.63
5:00PM	-0.88	71.78	0.275	283.2	0

4.Title towards WEST direction:

Time	Solar altitude	Hour angle	Declination	Azimuth	Elevation
8:00 AM	42.7	-44.46	16.8	72.4	0
1:00PM	-72.9	-0.41	16.89	178.6	73.62
5:00PM	6.9	89.7	16.7	288.2	0

5.SHOWING TIME WITH TEMPERATUURE.

S.no	Time	Solar voltage	Temperature	Intensity
1.	8:00 AM	0V	17'c	67Lux
2.	1:00 PM	0.01V	21'c	73Lux

East (longitude =78')		North (latitude = 18')		
Time	Declination (N hemisphere)	Altitude	Azimuth(E of S)	Hour angle
8AM	-22.99	48.96	-2.96	-2.11
12PM	-22.97	3.88	-64.33	-77.59
5PM	-22.95	-49.62	-75.47	-137.07

East (longitude =78')		South (latitude=18')		
Time	Declination (N hemisphere)	Altitude	Azimuth(E of S)	Hour angle
8AM	-22.99	84.64	-21.29	-2.11
12PM	-22.97	17.99	-70.98	-77.59
5PM	-22.95	-31.38	-47.28	-137.07

West (longitude= 78')		North (latitude=18')		
Time	Declination (N hemisphere)	Altitude	Azimuth(E of S)	Hour angle
8AM	-23.02	-65.02	73.96	153.84
12PM	-23.00	3.20	64.55	78.36
5PM	-22.99	45.05	24.94	18.88

West (longitude= 78')		South (latitude=18')		
Time	Declination (N hemisphere)	Altitude	Azimuth(E of S)	Hour angle
8AM	-23.02	-41.67	32.90	153.84
12PM	-23.00	17.30	70.78	78.36
5PM	-22.99	71.64	71.05	18.88

ALGORITHM:

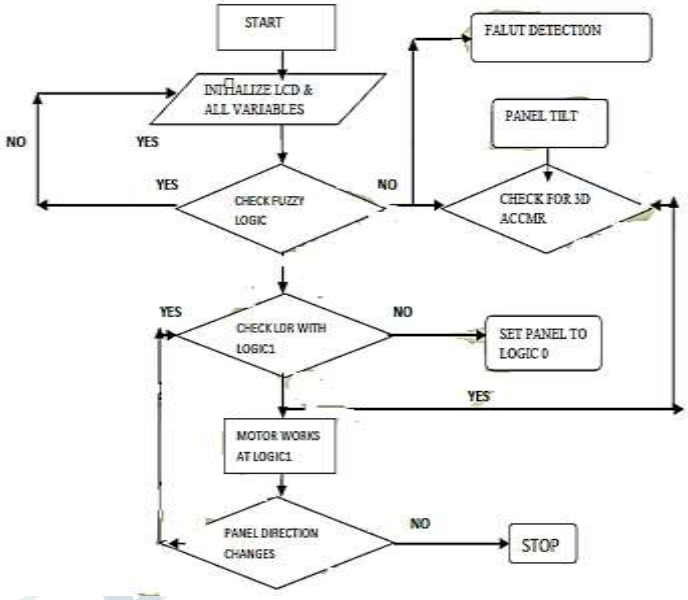


Fig. 21:showing the ALGO functioning of tracker.

III. CONCLUSION

This paper describes the simulation of a sun tracking solar power using ISIS. The simulation will be used for demonstration and experiments. The solar tracker follows the sun with 8 directions in a day followed with stepper motor. More expensive because of the size of panel and location of the project. Fixed tracking systems offer more field adjustability than single-axis tracking systems.

REFERNCE

•https://www.tigoe.com/pcomp/code/circuits/motors/stepper-motors/

• http://www.electronicshub.org