

# Sistema de Posicionamento para Placas Fotovoltaicas

*Aplicação de microcontroladores para o melhoramento da eficácia de painéis solares*

Jéssica de Souza Alves  
Graduanda em Engenharia Eletrônica  
Universidade de Brasília - UnB  
Brasília, Brasil  
14/0042784  
jessicaturunenn@gmail.com

Lorena Albernaz  
Graduanda em Engenharia Eletrônica  
Universidade de Brasília - UnB  
Brasília, Brasil  
14/0025715  
lorena.albernazz@gmail.com

**Resumo— O projeto visa criar um sistema de rotação de placas fotovoltaicas de acordo com o posicionamento do sol, também conhecido como solar tracker, sendo este controlado por um MSP 430.**

**Palavras-chave, MSP, Seguidor Solar, Fotovoltaico**

## 1 . Introdução

A energia solar é uma fonte limpa e sustentável de energia, e nos últimos tempos tem sofrido impacto devido ao seu crescimento. Dessa forma, métodos para o melhoramento de captação melhora consequentemente o aproveitamento e custos relacionados à energia desse tipo, sendo assim cada vez mais eficaz.

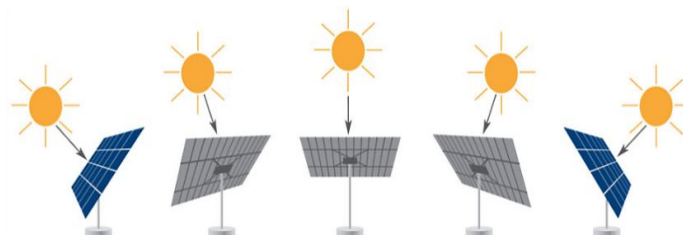
A energia solar fotovoltaica é a energia obtida através da conversão direta da luz solar em eletricidade, sendo a célula fotovoltaica, um dispositivo fabricado com material semicondutor, sendo essa célula fotovoltaica a unidade fundamental desse processo de conversão. A forma em que a célula fotovoltaica converte a luz é chamado de efeito fotovoltaico [1].

Uma célula solar ou célula fotovoltaica é um dispositivo elétrico de estado sólido capaz de converter a luz proveniente do sol (energia solar) diretamente em energia elétrica por intermédio do efeito fotovoltaico. [2]

Atualmente, as células fotovoltaicas apresentam eficiência de conversão da ordem de 16%. Até existem células fotovoltaicas com

eficiências de até 28%, fabricadas de arseneto de gálio, mas o seu alto custo limita a produção dessas células solares para o uso da indústria espacial. [3]

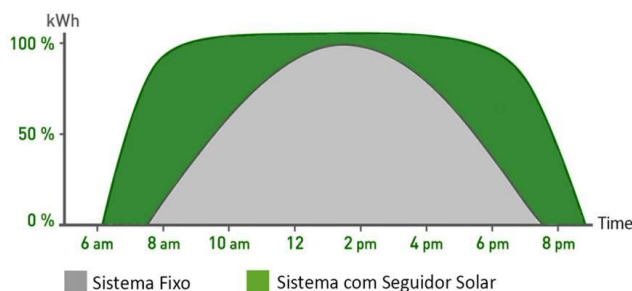
Uma boa maneira melhorar o rendimento de placas de captação de energia fotovoltaica, é criando dispositivos, que consideram o movimento do sol como um fator real, e fazem com que as placas solares sigam a direção em que a incidência solar se torna maior.



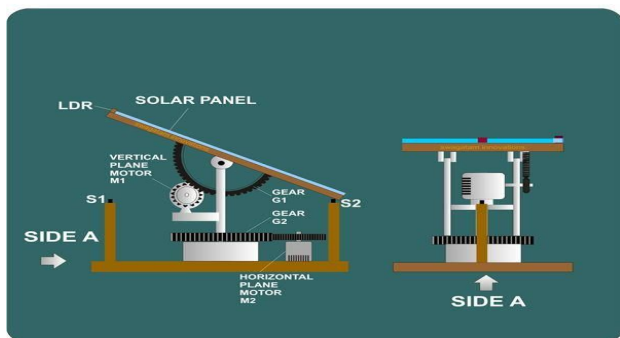
**Figura 1 - Funcionamento simplificado de um seguidor solar**

Seguidores Solares atuam na movimentação de placas fotovoltaicas adequando suas posições para o sentido de maior incidência solar de modo a otimizar a produção de energia da placa.

Sistemas com seguidores solares aumentam em torno de 30% a produção de energia quando comparados à sistemas fixos. Isso ocorre devido ao aumento da exposição direta aos raios solares. [4]



**Figura 2 - Gráfico comparativo entre os sistemas fotovoltaicos fixos e com seguidor solar**



**Figura 3 - Possível mecanismo para um sistema de movimentação de um painel solar com dois eixos [5]**

## 1.2 Justificativa

A energia solar por fotovoltaica tem custos relativamente altos para a sua implementação, dessa forma, a fim de ter um maior aproveitamento da energia disponível nas placas fotovoltaicas e colocar a energia fotovoltaica viável economicamente, coloca-se um dispositivo a fim de aumentar a eficiência das placas.

Dessa forma, a escolha do projeto foi motivada pela constante necessidade da otimização da produção de energia, deixando-a mais eficiente, e de modo a torná-la mais acessível ao consumidor. Sendo que, produtos similares ao funcionamento do projeto escolhido, são encontrados com valores elevados vistos comercialmente.

## 1.3 Objetivos

Aplicar os conhecimentos da disciplina de Microcontroladores e Microprocessadores para fazer um protótipo de um seguidor solar com uma implementação mais simples e eficaz do que as que atualmente existem.

## 1.4 Requisitos

- Projetar um circuito que controla os eixos de rotação em bases de placas fotovoltaicas;
- Monitorar com eficácia dados de sensores de luminosidade e sensores de irradiação solar de forma com que a placa gere mais energia;
- Rotacionar a placa fotovoltaica de forma otimizada de acordo com os dados obtidos pelos sensores

## 1.5 Benefícios

Um seguidor solar torna o custo da produção de energia de origem fotovoltaica mais barato pois aumenta a exposição da placa à raios solares diretamente aumentando a taxa de irradiação solar local.

Além disso, o protótipo visa ter um custo mais atrativo do que os disponíveis no mercado, beneficiando a sociedade e aumentando o acesso à energia limpa.

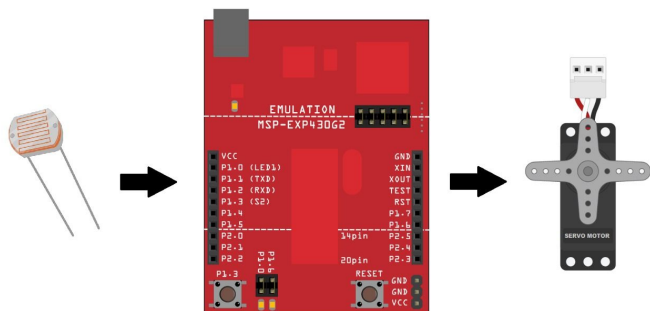
## 2. Desenvolvimento

### 2.1 Descrição de Hardware

O Hardware do projeto se dará de forma geral por uma placa microcontroladora que irá controlar dois motores (cada um em um eixo), a partir de dados de luminosidade coletado por sensores.

Serão utilizados quatro sensores LDR's (Light Dependent Resistor). Os sensores serão utilizados para o posicionamento das placas, de forma que ao fazerem as coletas de luminosidade do

ambiente, o software irá comparar os dados dos sensores de forma dois a dois. Assim, a partir de quatro sensores que irão fazer as coletas de intensidade luminosa do ambiente, dois motores serão acionados com cada um responsável por um eixo, e dessa forma posicionar a placa. Contudo, terá um motor em cada eixo (totalizando dois eixos), e cada motor é responsável por cada eixo de posicionamento.

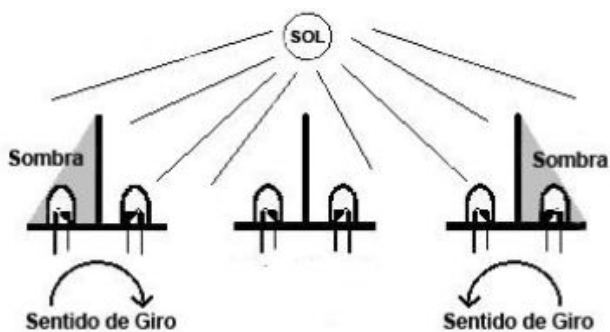


**Figura 4 - Esquemático de Hardware do seguidor solar (da esquerda para a direita: sensores LDR, MPS430, Motores Servo).**

## 2.2 Descrição de Software

Cada LDR é no circuito funciona como um divisor de tensão ao ser montado com um outro resistor. A saída deste divisor de tensão é atribuída a um pino do microcontrolador e então variáveis serão atribuídas para guardar o valor de cada um dos quatro LDR's.

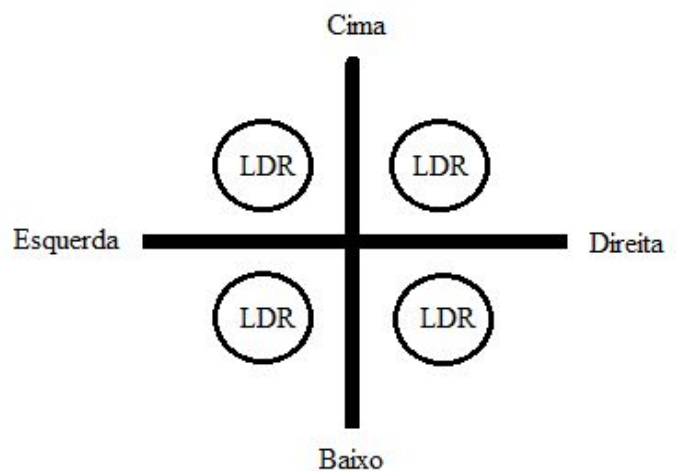
Para acionar o microcontrolador, são utilizados dois conjuntos com dois LDR's cada um. Um conjunto controla o movimento azimuthal e outro conjunto controla o ângulo de declinação.



**Figura 5 - Perfil "T" separando os LDR's de um conjunto**

Os LDR's de cada conjunto são separados um do outro por meio de um perfil "T" conforme a Figura 4. O microcontrolador é acionado quando existe uma diferença entre as impedâncias dos LDR's. O microcontrolador envia um pulso para movimentação (pwm) ao motor quando nota essa disparidade e reposiciona o sistema até que a discrepância entre as impedâncias seja mínima.

Um dos conjuntos de LDR's compara a intensidade luminosa entre o lado direito e o lado esquerdo enquanto o outro conjunto compara os valores de um referencial em cima e outro abaixo conforme os referenciais da Figura 6.



**Figura 6 - Referencial de orientação utilizado**

## 2.3 Protótipo

Para um primeiro protótipo funcional, foi utilizado o Arduino Uno pois assim é possível utilizar o Energia (equivalência do Arduino para o MSP430).

O sistema foi montado e testado conforme o esquemático da Figura 7 utilizando 4 LDR's, dois servo motores SG90, 4 resistores de 10K ohms e um Arduino Uno. Além disso, para validação do teste, o código compilou também na IDE Energia.

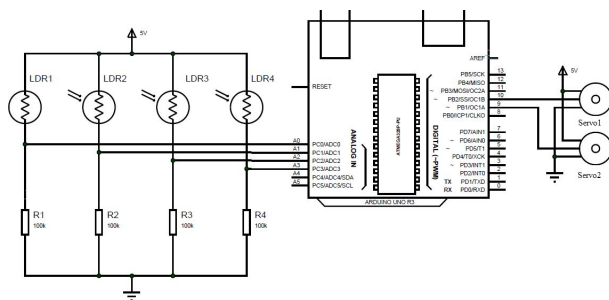


Figura 7 - Esquemático de teste do protótipo

A partir do código de teste do protótipo (Anexo 1), é possível ver que os dados dos divisores de tensão dos LDR's são entradas de 4 pinos do Arduino onde seu valor é armazenado analogicamente. Também são criadas duas variáveis para controlar os servo motores.

O loop (linha 35) utiliza das variáveis que armazenam os valores dos LDR's de acordo com a incidência para fazer comparações entre os dois LDR's de cada conjunto. No primeiro conjunto, é comparado a incidência entre os lados direito e esquerdo (observado no primeiro if na linha 45). Enquanto o segundo conjunto compara os valores de incidência entre a parte de cima e a parte de baixo (observado no segundo if na linha 59).

Ainda dentro do loop, o servo R1 é controlado a partir dos dados do primeiro conjunto e o servo R2 é controlado com os dados do segundo conjunto. Após cada análise nos valores de incidência de luminosidade de cada conjunto os motores são movimentados até que a discrepância entre os valores analisados seja mínima

## 2.4 Descrição do código utilizado no MSP

A lógica de funcionamento dos LDR's para o controle dos motores conforme explicada na descrição de software foi validada no teste do protótipo.

Após o teste e validação do protótipo, as alterações foram feitas para que houvesse o devido funcionamento no MSP430.

O esquemático de montagem feito com raciocínio análogo ao esquemático da Figura 7, está

de acordo com o a Figura 8. O diagrama foi feito no Proteus para simulação.

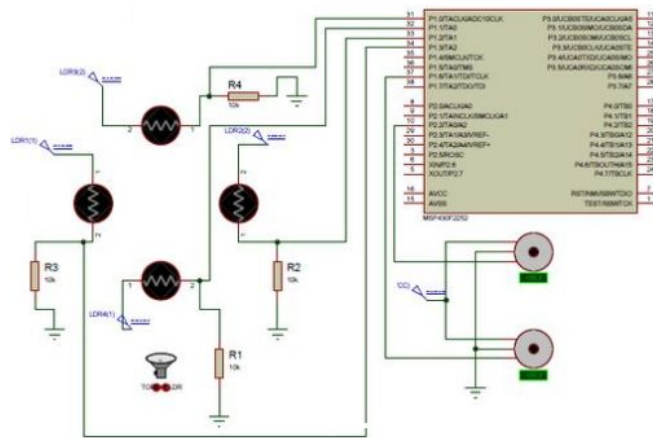


Figura 8 - Esquemático de montagem definitivo

O código para o uso do Msp foi:

```
#include "msp430.h"
#define ADC_CH 4
unsigned int samples[ADC_CH];
```

```
#define ldr_E BIT0
#define ldr_D BIT1
#define ldr_C BIT2
#define ldr_B BIT3
```

```
#define MCU_CLOCK      1000000
#define PWM_FREQUENCY  46
#define SERVO_STEPS    30
```

```
unsigned int PWM_Period =
(MCU_CLOCK / PWM_FREQUENCY);
unsigned int PWM_Duty = 0;
int i ;
int count ;
int count2 ;

void ConfigureAdc(void){

    ADC10CTL1 = INCH_3 | ADC10DIV_0 |
    CONSEQ_3 | SHS_0;

    ADC10CTL0 = SREF_0 | ADC10SHT_2 |
    MSC | ADC10ON | ADC10IE;
```

```
ADC10AE0 =ldr_E + ldr_D + ldr_C +
ldr_B ;
```

```
ADC10DTC1 = ADC_CH;
```

```
}
```

```
void main(void) {
```

```
WDTCTL = WDTPW | WDTHOLD;
```

```
TA1CCTL1 = OUTMOD_7 ;
```

```
TA0CCTL1 = OUTMOD_7 ;
```

```
TA1CTL = TASSEL_2 + MC_1 ;
```

```
TA0CTL = TASSEL_2 + MC_1 ;
```

```
TA1CCR0 = PWM_Period-1;
```

```
TA0CCR0 = PWM_Period-1 ;
```

```
TA1CCR1 = 0 ;
```

```
TA0CCR1 = 0 ;
```

```
P1DIR = 0;
```

```
P1SEL = 0;
```

```
P1OUT = 0;
```

```
P1REN |= (ldr_E | ldr_D | ldr_C | ldr_B);
```

```
P1DIR |= BIT6 ;
```

```
P2DIR |= BIT2 ;
```

```
P1SEL |= BIT6 ;
```

```
P2SEL |= BIT2 ;
```

```
ConfigureAdc();
```

```
__enable_interrupt();
```

```
while (1) {
```

```
__delay_cycles(1000);
```

```
ADC10CTL0 &= ~ENC;
```

```
while (ADC10CTL1 & BUSY);
```

```
ADC10SA = (unsigned int)samples;
```

```
ADC10CTL0 |= ENC + ADC10SC;
```

```
__bis_SR_register(CPUOFF + GIE);
```

```
for (i = 0; i < SERVO_STEPS; i++) {
```

```
TA1CCR1 = count2 ;
```

```
TA0CCR1 = count ;
```

```
__delay_cycles(20000);
```

```
TA1CCR1 = count2 ;
```

```
TA0CCR1 = count ;
```

```
__delay_cycles(20000);
```

```
}
```

```
}
```

```
#pragma vector = ADC10_VECTOR
```

```
__interrupt void ADC10_ISR (void){
```

```
if (samples[0] < samples[1]){
```

```
if(samples[1]> 550){
```

```
count = 850;
```

```
}else {
```

```
count = 1350;
```

```
}} else if ((samples[0] +
```

```
samples[1])/2 < 500) {
```

```
count = 1350;
```

```
} else {
```

```
if(samples[0]>550){
```

```
count = 1850;
```

```
}else {
```

```
count = 1350;
```

```
}}
```

```
if (samples[2] < samples[3]) {
```

```
if(samples[3]> 550){
```

```
count2 =850;
```

```
}else {
```

```
count2 = 1150;
```

```
}
```

```
} else if ((samples[2] +
```

```
samples[3])/2 < 500) {
```

```
count2 = 1150;
```

```
} else {
```

```
if(samples[2]>550){
```

```
count2 = 1850;
```

```
}else {
```

```
count2 = 1150;
```

```
}
```



```

    }

    __bic_SR_register_on_exit(CPUOFF);
}

```

### 3. Referências

- [1] Manual de engenharia para sistemas fotovoltaicos. Rio de Janeiro: CEPEL - CRESESB. Março de 2014.
- [2] Roper, L. David (24 de agosto de 2011). World Photovoltaic Energy.
- [3] Água quente solar - Caracterização do Solar Térmico em Portugal 2008. Agência para a Energia.
- [4] Disponível em <http://www.portalsolar.com.br/blog-solar/painel-solar/seguir-solar---tracker-vantagens-e-desvantagens-parte-1.html>
- [5] Disponível em <http://www.brighthub.com/environment/renewable-energy/articles/76226.aspx>.
- [6] GOETZBERGER, A.; HEBLING, C.; SCHOCK, H.-W. Photovoltaic materials, history, status and outlook. Materials Science and Engineering R, v. 40, p. 1-46, 2003.
- [7] GREEN, M. A. Photovoltaics: technology overview. Energy Policy, v. 28, p. 989-998, 2000.
- [8] OLIVEIRA, Fernanda P. Monografia: Célula Fotovoltaica de Silício. São Paulo: USP, 2010

### 4. Anexos

#### 4.1 Código de teste do protótipo

```

1  #include <Servo.h>
2  //Inclui biblioteca de servos
3
4  Servo servoR1;
5  // Cria variaveis para controlar servos
6  Servo servoR2;
7  // Cria variaveis para controlar servos
8
9  int ldr_E = A2; // Ldr da esquerda
10 int ldr_D = A3; // Ldr da direita
11 int ldr_C = A0; // Ldr de cima
12 int ldr_B = A1; // Ldr de baixo
13
14 // Posicao inicial dos servos
15 int valR1 = 90;
16 int valR2 = 135;
17
18 //Variaveis para guardar valores
19 //analógico dos pinos
20 int val_E;
21 int val_D;
22 int val_C;
23 int val_B;
24
25 // Tempo de delay dos servos
26 int delay_movimento= 100;
27
28 void setup()
29 {
30 // Atribui pino aos servos R1 e R2
31   servoR1.attach(10);
32   servoR2.attach(9);
33 }
34
35 void loop()
36 {
37 //leitura analógica do ldr de acordo
38 // com a incidência solar
39   val_E = analogRead(ldr_E);
40   val_D = analogRead(ldr_D);
41   val_C = analogRead(ldr_C);
42   val_B = analogRead(ldr_B);
43
44 //comparação dos ldr para estabilidade
45   if (val_E - val_D > 2 || val_E - val_D < -2)
46   {
47     {
48       if (val_E > val_D && valR1 > 0)
49       {
50         valR1--;
51       }
52       if (val_D > val_E && valR1 < 180)
53       {
54         valR1++;
55       }
56     }
57   }
58 }

```

```
57
58 //comparação dos ldr para estabilidade
59 if (val_C - val_B > 2 || val_C - val_B < -2)
60 {
61     if (val_C > val_B && valR2 > 90)
62     {
63         valR2--;
64     }
65     if (val_B > val_C && valR2 < 180)
66     {
67         valR2++;
68     }
69 }
70
71 //posição de acordo com o valor calculado
72 servoR1.write(valR1);
73 servoR2.write(valR2);
74
75 // Espera o servo se mover até a posicao
76 delay(delay_movimento);
77 }
```