CENTRO UNIVERSITÁRIO DE BRASÍLIA

FACULDADE DE TECNOLOGIA E CIÊNCIAS SOCIAIS APLICADAS

CURSO DE ENGENHARIA DA COMPUTAÇÃO

BRUNO QUEIROZ CUNHA

PERSIANA AUTOMATIZADA UTILIZANDO SENSOR DE LUZ PARA APROVEITAMENTO DA LUZ SOLAR

Orientadora: Profª. M.C. Maria Marony Sousa Farias

BRASÍLIA

2º SEMESTRE DE 2012

BRUNO QUEIROZ CUNHA

PERSIANA AUTOMATIZA UTILIZANDO SENSOR DE LUZ PARA APROVEITAMENTO DA LUZ SOLAR

Trabalho apresentada ao UniCEUB – Centro Universitário de Brasília como pré-requisito para obtenção de Certificação de Conclusão do Curso de Engenharia de Computação.
Orientadora: Profª. M.C. Maria Marony Sousa Farias Nascimento.

BRASÍLIA

2º SEMESTRE DE 2012

BRUNO QUEIROZ CUNHA

PERSIANA AUTOMATIZADA UTILIZANDO SENSOR DE LUZ PARA APROVEITAMENTO DA LUZ SOLAR

Trabalho apresentada ao UniCEUB – Centro Universitário de Brasília como pré-requisito para obtenção de Certificação de Conclusão do Curso de Engenharia de Computação. Orientadora: Profª. M.C. Maria Marony Sousa Farias Nascimento.

Este Trabalho foi julgado adequado para a obtenção do Título de Engenheiro de Computação, e aprovado em sua forma final pela Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS.

_____

Prof. Abiezer Amarília Fernandez

Coordenador do Curso

Banca Examinadora:

_____

Profª. Maria Marony Sousa Farias, mestre em Engenharia Elétrica – UFPB – PB. Orientadora

_____

Profª Irene de Azevedo Lima Joffily, mestre em Estruturas e Construção Civil – UNB.

_____

Prof. José Julimá Bezerra Júnior, mestre em Engenharia Elétrica.

_____

Prof. Sidney Cerqueira Bispo dos Santos, Doutor.

Dedico a todos que influenciaram direta e indiretamente a concluir esse projeto e mais uma fase da minha vida.

AGRADECIMENTOS

A minha família, pelo suporte e dedicação ao longo da minha vida, me estimulando e dando confiança para que eu crescesse como pessoa e como profissional.

A minha namorada, Juliana, pelo incentivo, força e companheirismo dado ao longo do projeto.

Aos amigos com quem estudei que contribuíram para meu crescimento acadêmico e profissional ao longo do curso de graduação.

E a todos que participaram direta e indiretamente para que eu conseguisse concluir meu projeto.

“A imaginação é mais importante que o conhecimento.”

Albert Einstein.

RESUMO

Este trabalho apresenta uma persiana automatizada que faz o uso de sensores de luz para o aproveitamento de luz solar. O projeto faz integração de uma persiana comum, microcontrolador PIC 16F877A, servomotor e de sensores de luminosidade. O trabalho faz com que a persiana abra suas lâminas de acordo com a luz que é incidida nos sensores de luz. Os sensores estão posicionados estrategicamente ao longo da estrutura da persiana, e quando a luz incide neles, é feita uma comparação de tensão. Após isso, o motor movimentará o sistema de abertura da persiana movimentando suas lâminas para a posição onde se encontra o LDR com a menor tensão.

Palavras-chave: LDR, persiana, microcontrolador, luz e motor.

ABSTRACT

This work presents an automatic shutter that uses light sensors to take advantage of sunlight. The project consists in integrating a common shutter, PIC 16F877A microcontroller, servo motor and lighting sensors. The design makes the shutter open its blades according to the amount of light inciding on the lighting sensors. The sensors are strategically positioned throughout the structure of the shutter, and when light hits them, a comparison of electric tensions takes place, after that, the motor will move the opening system moving the shutter's blades to the position where the LDR with the lowest voltage is.

Keywords: LDR, shutter, microcontroller, light engine.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A/D	Analógico / Digital
ASCII	American Standard Code for Information Interchange
CA	Corrente Alternada
CC	Corrente Continua
CDS	Sulfeto de Cádmio
CEPEL	Centro de Pesquisas de Energia Elétrica
CM	Centímetros
EEPROM	Erasable Electronically Programmable Read Only Memory
FIEC	Federação das Indústrias do Estado do Ceará
LCD	Liquid Cristal Display
LDR	Light Dependent Resistor
PIC	Programmable Interface Controller
PWM	*Pulse Width Modulation*

CAPÍTULO 1 - INTRODUÇÃO

1.1 Apresentação do Problema

A maioria das persianas é composta por um conjunto de lâminas, um eixo e um sistema de abertura e fechamento. O objetivo delas é basicamente impedir que luz entre no ambiente. Porém, na maioria dos casos, a persiana fica sempre na posição fechada, pois as pessoas veem mais conforto em apenas ligar um interruptor ao invés de utilizar as cordas do sistema de abertura e fechamento e abrir a persiana para que entre luz natural no ambiente.

Além de trazer certo conforto para os usuários, que não irão precisar ficar abrindo e fechando a persiana, este projeto visa minimizar o uso de luz artificial. Utilizando uma persiana automatizada, a mesma irá se movimentar com intuito de aumentar a luz que entrará no ambiente, minimizando assim, a necessidade de luz artificial, em certo período do dia.

Para automação da persiana, é utilizado um sensor LDR (*Light Dependent Resistor*), que fará com que a persiana funcione como um girassol. O sensor enviará informações a um microcontrolador, que movimenta um motor interligado ao sistema de abertura e fechamento da persiana de acordo com a luminosidade incidente, fazendo com que o ambiente tenha um aproveitamento mais eficiente da luz natural em detrimento da iluminação artificial por lâmpadas.

1.2 Visão Geral do Projeto

Persianas automatizadas não são mais uma novidade nos dias de hoje. Existem diversos tipos e modelos. Normalmente, esses modelos são acionados por um botão localizado no interruptor da casa, e os mais modernos, por controle remoto. Esse tipo de automatização gera um conforto grande para as pessoas, pois não precisam de cordas que normalmente são frágeis, arrebentam facilmente e se enroscam umas nas outras.

Este projeto consiste na construção de um protótipo que faz a movimentação de forma automática, com a mínima necessidade de interação.

Utilizando sensores de luz e um motor servo, as frestas da persiana seguirão o sol, tal como um girassol, fazendo com que a luz entre facilmente e adentre o ambiente, diminuindo assim, a necessidade de luz artificial, economizando assim, energia.

1.3 Objetivos do Trabalho

O objetivo deste projeto é desenvolver e implementar um sistema automatizado de persianas que além de trazer conforto para os usuários, atende ao principal objetivo, que é economia de energia.

O circuito tem que avaliar em qual LDR está incidindo a maior quantidade de luz, decidindo então, em qual posição deve ficar a persiana. Um motor é então acionado para o melhor ângulo, para que a luz natural passe pelas frestas da persiana, iluminando assim, o ambiente.

Este projeto poderia ser utilizado em ambientes que necessitem de luz por longos períodos, como por exemplo, escolas, faculdades e escritórios comerciais.

1.4 Justificativa e Importância do Trabalho

O uso racional de energia traz benefícios enormes para o meio ambiente, e consequentemente para a sociedade. Segundo Brum (2009) O consumo de energia intensifica-se com o passar dos tempos. Como a energia é proveniente da utilização e transformação de forças oferecidas pela natureza, a agressão ao meio ambiente é cada vez maior. Os países que realizam esta transformação e fornecimento consomem sem preocupar-se em reparar os danos causados. A energia é produzida por: usinas hidroelétricas (água), usinas térmicas (queima de combustível), usinas nucleares (fissão e fusão de átomos). A produção descontrolada de energia acarreta em sérios prejuízos ao meio ambiente, modificando a paisagem e o clima. Afeta assim, o ecossistema, a fauna e a flora.

A economia de energia elétrica está diretamente ligada a esse tema, sendo necessária a atenção de todos, pois ao passar dos anos, degradamos cada vez mais o planeta em que vivemos.

Este projeto visa primeiramente à economia de energia. Ele é projetado para que a luz natural substitua a luz artificial dentro do ambiente sempre que possível, tendo em vista de como o consumo exagerado de energia impacta no meio ambiente.

1.5 Escopo do Trabalho

O projeto utiliza o microcontrolador PIC 16F877A que vai controlar o motor servo de acordo com uma comparação feita entre os LDRs. No total são 4 LDrs que vão ser comparados entre si. Ao comparar os valores, o microcontrolador vai movimentar o motor para a direção que se encontra o LDR com menor tensão.

Além disso, o trabalho tem um botão de ligar e desligar e também um botão de reset, caso venha a ter algum problema durante as gravações do programa no microcontrolador, ou algum tipo de travamento.

1.6 Resultados Esperados

Os LDRs (sensor) estão posicionados estrategicamente ao longo da persiana. Assim que a luz incidir nos LDRs, eles vão mandar um sinal analógico para o microcontrolador. O microcontrolador irá fazer a conversão analógico/digital de cada um dos sinais enviados pelos sensores. Ao fazer essa conversão, ele vai comparar os valores das tensões e acionará o motor servo. O motor será movimentado de acordo com valores pré-estabelecidos no microcontrolador e posicionará as lâminas da persiana de acordo com o LDR que tiver com menor tensão (maior incidência de luz).

Na figura 1.1, é mostrada a topologia do projeto:

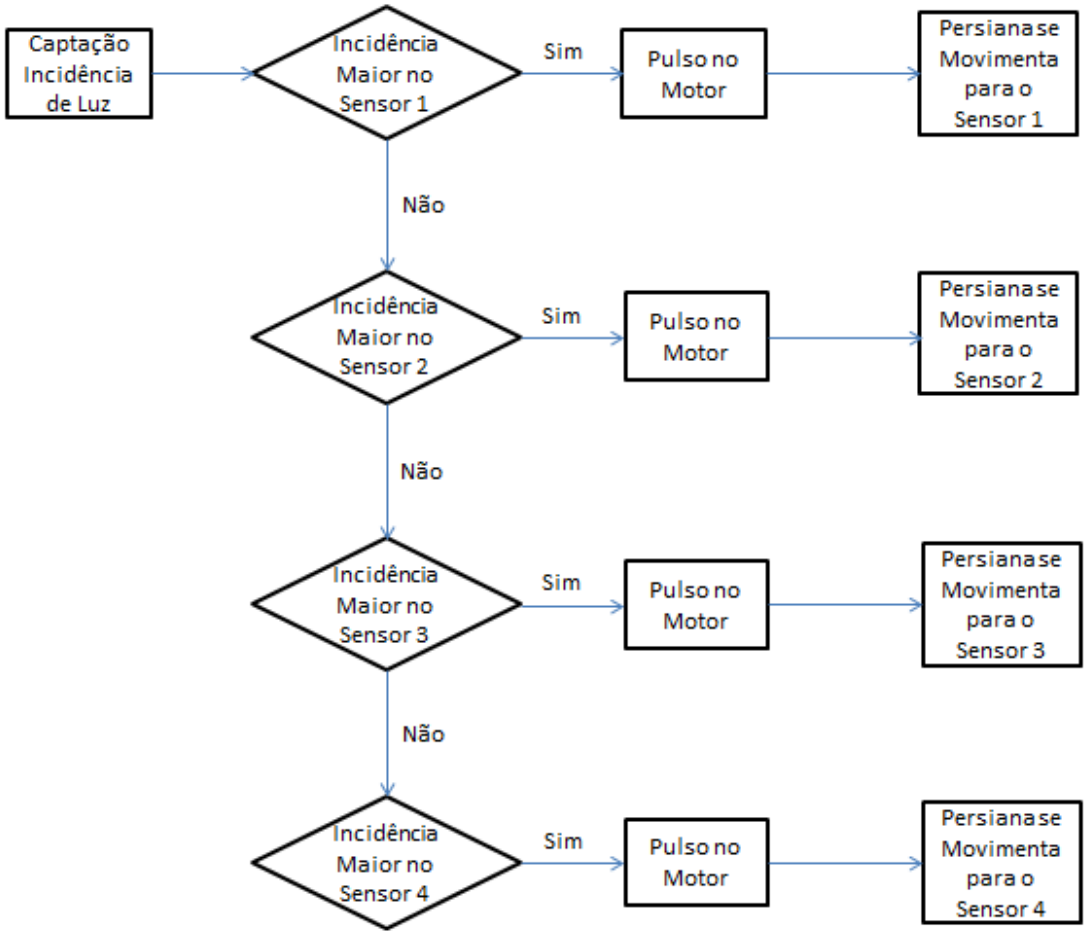

Figura 1.1 – Topologia

1.7 Estrutura da Monografia

A monografia esta estruturada da seguinte maneira:

Capítulo 2: Esse capítulo descreve de forma aprofundada a apresentação do problema que o projeto pretende resolver, assim como, soluções e tecnologias existentes, impacto que esses problemas causam na sociedade, como resolver o problema e como o projeto pode ajudar a solucionar este problema.

Capítulo 3: Nesse capítulo é descrito o referencial teórico e tecnológico, tratando sobre todos os dispositivos utilizados ao longo do projeto, necessários para a compreensão do desenvolvimento do mesmo.

Capítulo 4: É o capítulo em que é demostrado como todo o sistema foi desenvolvido. Aqui é mostrado como foi feita desde a programação até a construção do hardware utilizado, assim como sua explicação detalhada de alguns componentes principais utilizados ao longo do projeto.

Capítulo 5: Nesse capítulo são mostrados todos os resultados obtidos ao longo da monografia, tal qual, figuras ilustrativas do protótipo, e os testes realizados sobre a solução.

Capítulo 6: É o capítulo onde é descrita a conclusão acerca do assunto tratado no projeto, assim como, as dificuldades encontradas e sugestões para trabalhos futuros em temas relacionados ao da monografia.

CAPÍTULO 2 – APRESENTAÇÃO DO PROBLEMA

Neste capítulo são apresentadas as questões que motivaram o trabalho, como por exemplo, a economia de energia e a comodidade que projeto trará. Essas informações são descritas de forma resumida na seção ─ Apresentação do Problema do capítulo anterior.

É apresentado também características de soluções existentes no mercado, assim como o preço comercial das mesmas.

2.1 Visões do Projeto

> A iluminação doméstica é responsável por uma parcela importante do gasto com energia na casa. No Brasil, a fonte de energia mais usada para iluminação é a elétrica. Economizando com iluminação você gasta menos e ajuda o meio ambiente porque a geração de energia elétrica tem impacto ambiental. Mesmo no Brasil, onde predominam as usinas hidrelétricas, a redução no consumo de energia elétrica é fundamental para o ambiente. Primeiramente porque nosso sistema é integrado e também usa usinas termoelétricas e nucleares, que causam dano ambiental maior. Segundo, porque as usinas hidrelétricas também têm impacto ambiental. Finalmente, se todos economizarem, não será preciso ampliar o sistema, o que vai trazer mais custo ambiental. (Manosso, 2010)

A persiana automatizada tem como objetivo o aproveitamento da luz solar. Diminuindo assim, o consumo abusivo de energia elétrica. A persiana automatizada trará também benefícios como o aumento da vida útil da persiana e diminuição da manutenção na mesma, além do grande conforto para o usuário, pois interação entre o usuário/persiana será mínima.

Segundo uma pesquisa em conjunto da FIEC, CEPEL e ELETROBRÁS, uma lâmpada incandescente de 40 W ligada 5 horas por dia durante um mês tem um gasto médio mensal de 6kwh e uma fluorescente compacta de 60 W, utilizada no mesmo pelo mesmo tempo da anterior, tem um gasto médio mensal de 9kwh, tendo assim, o mesmo gasto que uma televisão em cores de 14'' ligada pelo mesmo período de tempo. Como o normal, é de que essas lâmpadas ficam ligadas durante quase

todo o dia, o consumo mensal é bem maior, trazendo assim, um impacto ambiental grande.

Para o maior aproveitamento da luz solar, esse projeto utiliza sensores de luminosidade (LDR) como é mostrado na figura 2.1.

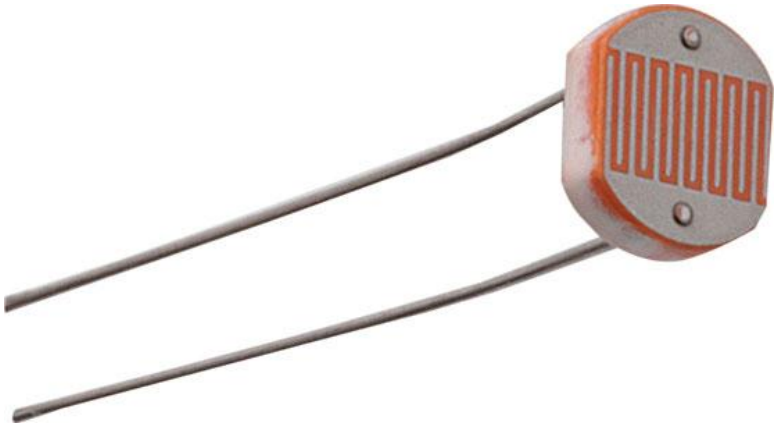

Figura 2.1 – Sensor de Luminosidade (LDR).

Fonte: (http://www.te1.com.br/2011/03/circuito-sensor-luz-ldr/)

A persiana será movimentada de acordo com a luz incidida nos LDRs estrategicamente posicionados, fazendo com que a luz que passe pelas suas frestas seja a maior possível, aumentando assim, a iluminação do ambiente.

2.2 Características das Persianas

> Uma persiana funciona com uma mola que é controlada por uma lingueta e uma pequena alavanca de retenção. Elas podem ser consideradas como um tipo de cortina. Possuem função térmica, podem bloquear a entrada indesejada de calor nas épocas de clima mais quente no ano e manter o calor em épocas de clima mais frio. Em ambos os casos, dependendo da cor e do formato, reduzem a luz em graus variados. (Alfarone, 2010)

O objetivo das persianas é o bloqueio da luz solar. Isso se faz necessário por vários motivos, entre eles esta o dano que o sol excessivo traz a alguns aparelhos eletrônicos e móveis.

Com o uso da persiana automatizada, a luz que vai entrar no ambiente será controlada, pois as frestas irão impedir a passagem de um percentual de iluminação, evitando assim, tais problemas.

2.3 Preço Comercial

Persianas automatizadas não são mais uma novidade. Existem vários tipos e preços. Comumente a automatização se dá com o controle do usuário, seja apertando um botão do controle remoto ou um interruptor na parede do ambiente.

Não foi encontrado no mercado nenhum modelo de persiana com o objetivo do aproveitamento da luz solar, mas é suposto que o preço da mesma seja parecido com o preço de uma persiana automatizada comum. Conforme a pesquisa de preço de Alfarone (2010, p.25), uma persiana com 2 metros por 1,5 metros com motor, mecanismos e o modelo mais barato, sairiam por R$ 2.208,20, porém, os modelos mais vendidos chegam a quatro mil reais.

CAPÍTULO 3 – REFERENCIAL TEÓRICO

Nesse capítulo é descrito o referencial teórico e tecnológico, tratando sobre todos os dispositivos utilizados ao longo do projeto, necessários para a compreensão do desenvolvimento do mesmo.

3.1 Motores Elétricos

O motor elétrico é uma máquina que faz a transformação de energia elétrica em energia mecânica, normalmente, essa energia é utilizada em um eixo de rotação, como é ilustrado na figura a seguir.

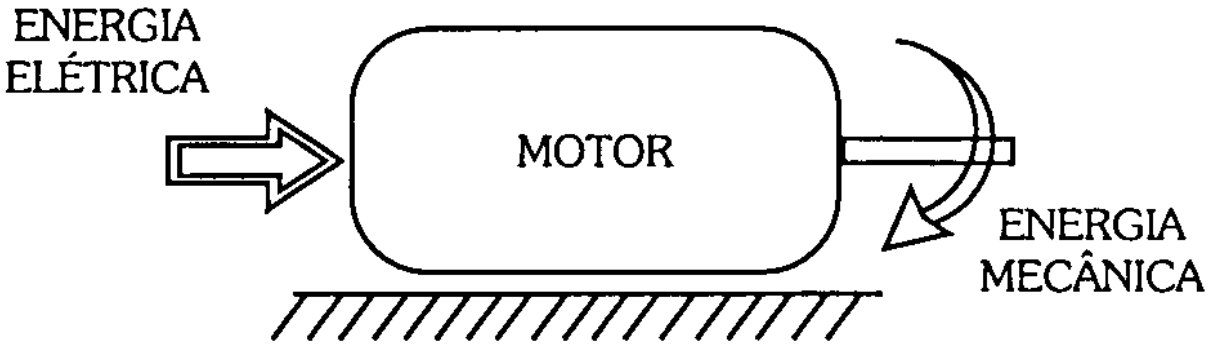

Figura 3.1 – Funcionamento de um motor elétrico

(Fonte: http://s3.amazonaws.com/magoo/ABAAAA49cAF-1.png)

Segundo Filho (2000, p.1), o motor elétrico é a máquina mais largamente utilizada na indústria, no meio rural, no comércio, nos serviços e nas residências. Isto se deve às qualidades inerentes da energia elétrica, ou seja, trata-se de uma energia limpa, de baixo custo e de fornecimento instantâneo. Pelo lado dos motores eles são fabricados desde potencias minúsculas de alguns watts, até potencias gigantescas de milhares de quilowatts. Suas características operacionais atendem a quaisquer tipos de cargas, aliadas a um alto rendimento na transformação de energia. A instalação e manutenção são razoavelmente simples. É uma maquina extremamente segura. São do ponto de vista econômico são imbatíveis frente a quaisquer outros tipos de motores.

3.1.1 Tipos Básicos

Existem diversos tipos de motores elétricos, os quais são divididos em duas grandes famílias. A primeira é a família dos motores acionados por corrente continua (motores cc). A segunda é a dos motores acionados por corrente alternada (motores ca). (FILHO, 2000) É apresentado na figura 3.2, o quadro geral dos motores elétricos.

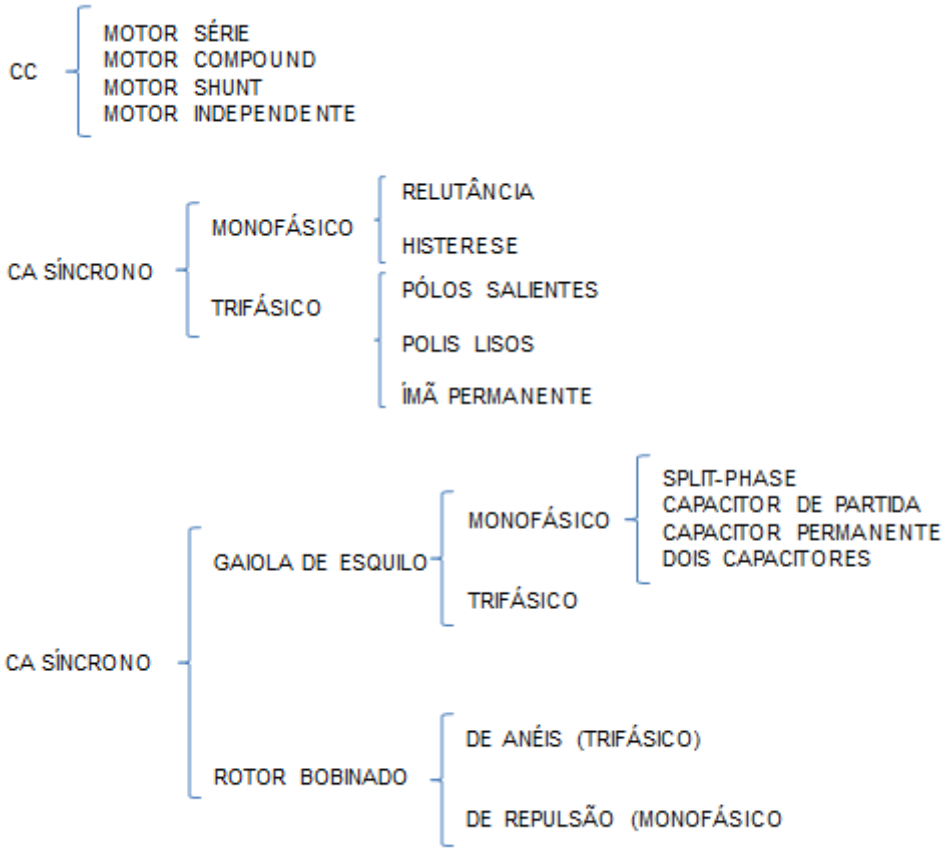

Figura 3.2 – Tipos de motores elétricos

Os motores ca podem ser dos tipos síncrono, assíncrono ou especial. O motor assíncrono é denominado como motor de indução, posto que seu princípio de funcionamento esta baseado na indução eletromagnética, à semelhança dos transformadores. Esses motores são fabricados em alta tensão quando se destinam a aplicações de alta potencia. Tipicamente são utilizados em siderúrgicas, refinarias, entre outras. (FILHO, 2000)

3.1.2 Motor Utilizado do Projeto

No projeto foi utilizado um servomotor. Ele foi escolhido devido a sua fácil utilização com aplicações de posicionamento, já que seu papel movimentar o sistema de abertura e fechamento da persiana, posicionando assim, as lâminas em ângulos específicos.

Segundo Francisco (2008, p.120) Os servomotores são constituídos basicamente de um pequeno motor, um circuito eletrônico de controle, engrenagens e três condutores para ligação. Na figura 3.3, é mostrada a constituição do servomotor.

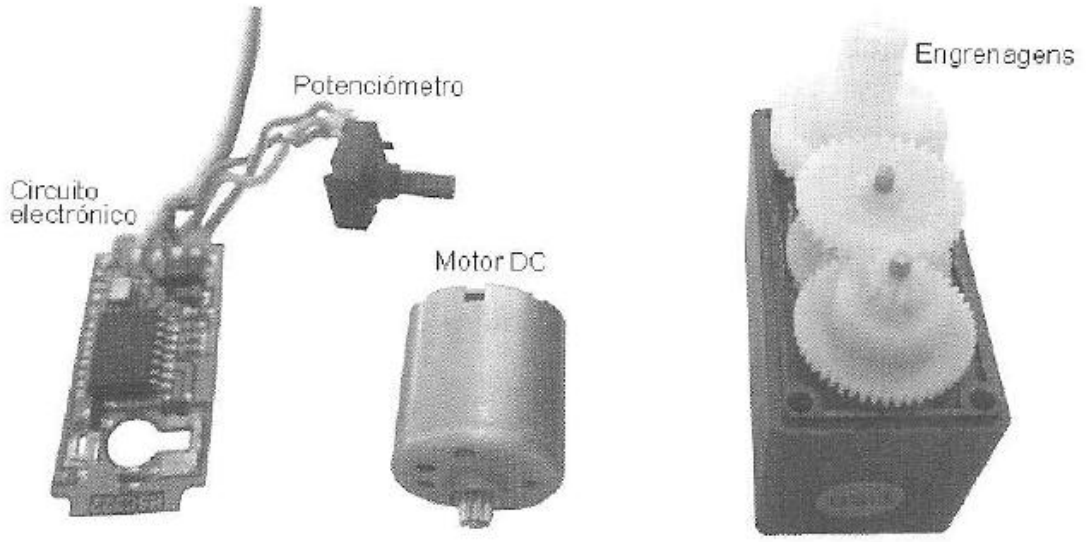

Figura 3.3 – Constituição do servomotor. (Fonte: Francisco, 2008)

O ângulo de rotação do servomotor é determinado pela duração do impulso (tempo ON) que se aplica na entrada do comando, nesta aplica-se um sinal PWM. Trata-se de uma onda quadrada em que se varia a duração do tempo t_{on}, mantendo o período da mesma. (FRANCISCO, 2008)

> O giro mínimo e máximo depende do tipo de servomotor. No geral, se o servo receber na sua entrada impulsos com a duração de:
>
> - 1 ms, o seu eixo roda no sentido anti-horário, até atingir o limite do intervalo de rotação, o que corresponde a 0º;
> - 1,5 ms, o seu eixo roda até ficar estável no centro do intervalo de rotação, a que corresponde o ângulo de 90º;

- 2 ms, o servo roda no sentido horário, até atingir o outro limito do intervalo de rotação (180º ou um pouco mais) (Francisco, 2008)

Na figura 3.4, é ilustrado o controle de ângulos de acordo com os pulsos no servo.

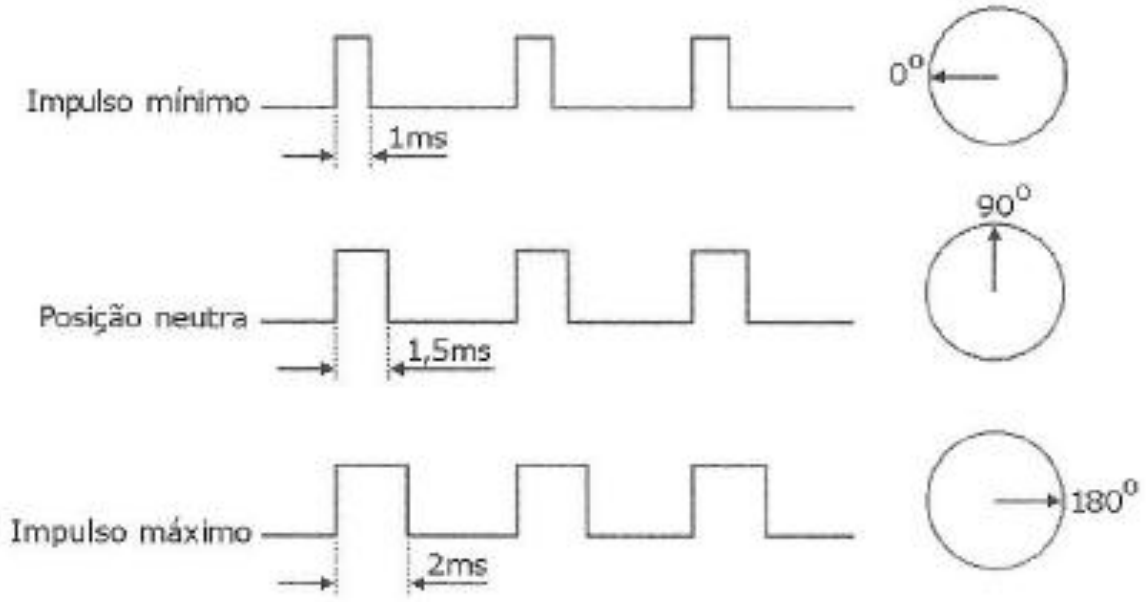

Figura 3.4 – Ângulos atingidos de acordo com os pulsos gerados. (Fonte: Francisco, 2008)

O modelo utilizado no projeto foi um Hobbico CS-60 Standard Sport, é um motor que tem torque de 42 oz-in(3,06 kg-cm) e velocidade de 0,19 rotações por segundos quando é alimentado com 4,8V, e 49 oz-in(3,57 kg-cm) e velocidade de 0,15 rotações por segundo quando alimentado com 6V. (HOBBICO)

A seguir é mostrado o modelo do motor.

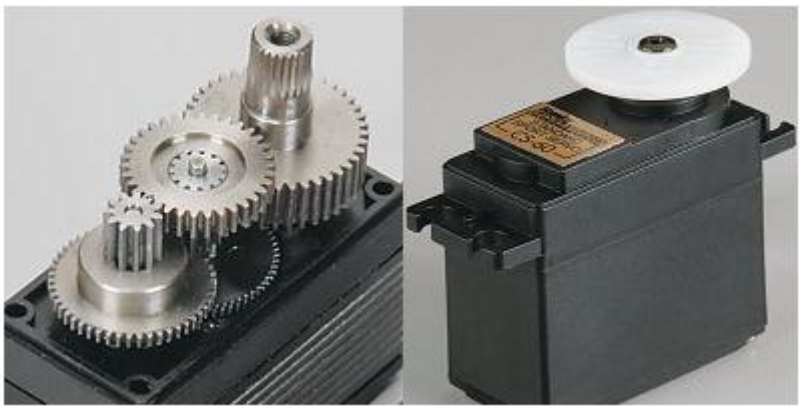

Figura 3.5 – Hobbico CS-60

(Fonte: http://www.hobbico.com/radioaccys/hcam1000.html)

3.2 Sensores

Termo empregado para designar dispositivos sensíveis a alguma forma de energia do ambiente que pode ser luminosa, térmica, cinética, relacionando informações sobre uma grandeza que pode ser medida, como: temperatura, pressão, velocidade, corrente, aceleração, posição, etc. Conforme indica a figura 3.6. (THOMAZINI, 2007)

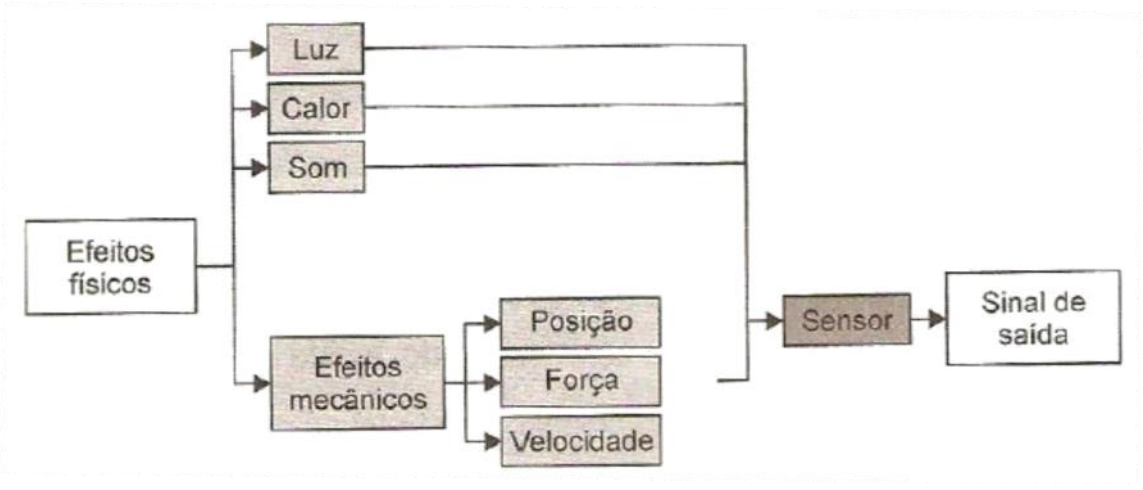

Figura 3.6 – Ilustração das formas de energia em um sensor.

(Fonte: Thomazini, 2007)

Um sensor nem sempre tem as características elétricas necessárias para ser utilizado em um sistema de controle. Normalmente o sinal de saída deve ser manipulado antes da sua leitura no sistema de controle. Isso geralmente é realizado com um circuito de interface para produção de um sinal que possa ser lido pelo controlador. (THOMAZINI, 2007)

3.2.1 Sensores Analógicos

Algumas das grandezas físicas que podem assumir qualquer valor ao longo do tempo são: pressão, temperatura, velocidade, umidade, vazão, força, ângulo, distância, torque e luminosidade. Essas variáveis são mensuráveis por elementos sensíveis com circuitos eletrônicos não digitais. A figura 3.7 ilustra a variação de uma grandeza física (temperatura) de forma analógica. (THOMAZINI, 2007)

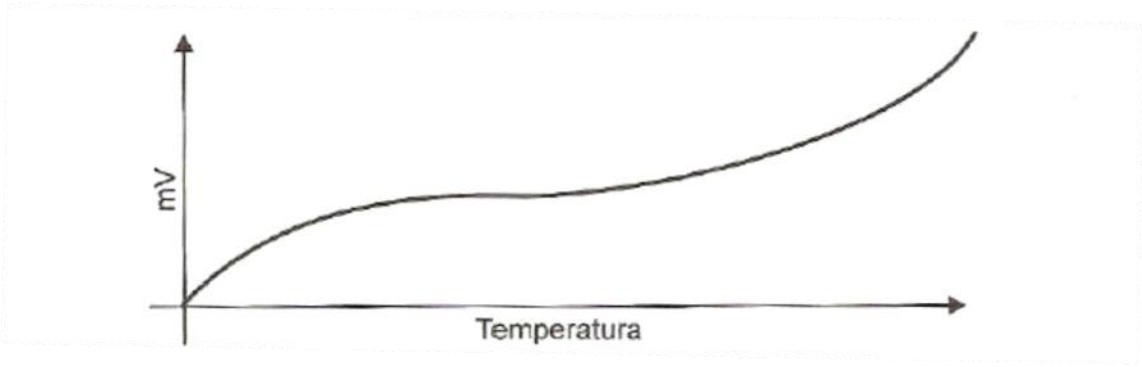

Figura 3.7 – Ilustração da variação de uma grandeza física de um sensor analógico. (Fonte: Thomazini, 2007)

Para que o microcontrolador entenda o sinal de saída de um sensor, é necessário fazer a conversão digital do mesmo. Quando fazemos esse tipo de conversão, o sinal analógico muda totalmente, e podem acontecer algumas distorções no sinal. Conforme é mostrado na figura 3.8.

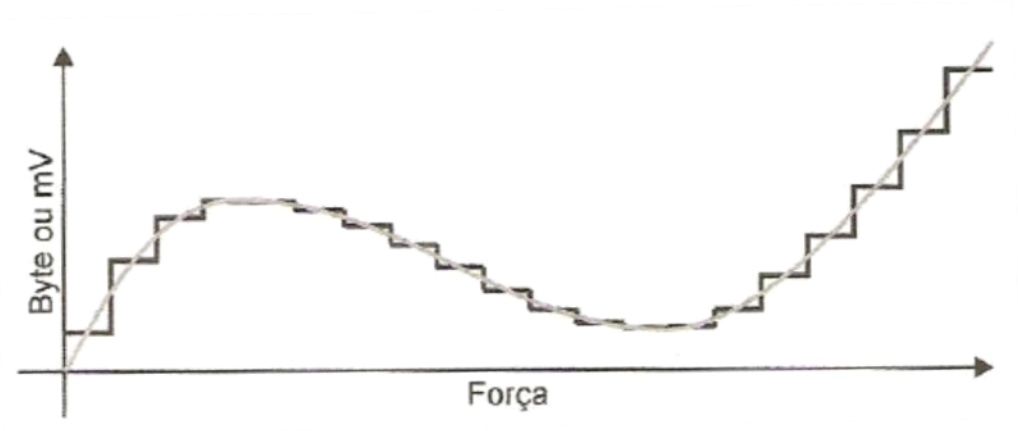

Figura 3.8 – Ilustração a conversão analógico/digital de um sinal. (Fonte: Thomazini, 2007)

3.2.2 LDR *(Light Dependent Resistor)*

Neste projeto, foram utilizados quatro sensores LDRs devido seu comportamento ao ser incidido pela luz. Quando a luz incide no sensor LDR, ele altera sua resistência. No trabalho, foram utilizados quatro sensores para fazer o controle da persiana. Ao serem incididos pela luz, é feita uma comparação entre eles para saber em qual deles a luz está com mais força, após isso, a persiana é movimentada para que a luz ilumine de melhor forma o ambiente. Detalhes referentes ao funcionamento do LDR no projeto são abordados no próximo capítulo.

A tradução de LDR é resistor dependente de luz, também chamado de fotorresistor, é usado em varias aplicações, fazendo o controle de vários equipamentos.

Quando a luz incide em determinadas substâncias cujas resistências são alteradas devido à quantidade de luz que recebem, ocorre a liberação de portadores de carga que ajudam a condução da corrente elétrica, conforme é mostrado na figura 3.9. (THOMAZINI, 2007)

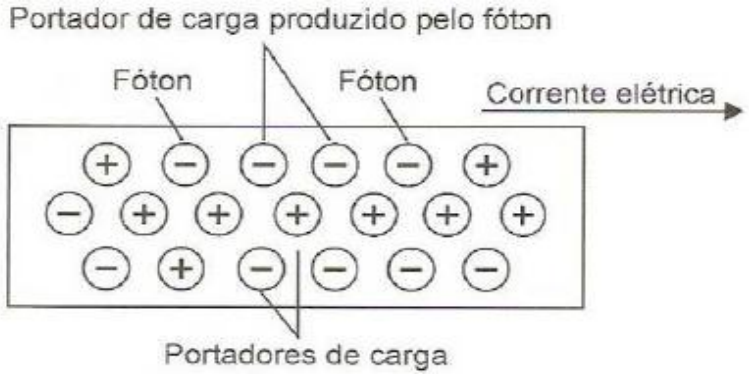

Figura 3.9 – A luz libera portadores de carga que conduzem a resistência elétrica de determinados materiais. (Fonte: Thomazini, 2007)

Segundo Thomazini (2007, p.62) O sulfeto de cádmio, cuja fórmula química é CdS, é utilizado na construção dos LDRs. São chamados de fotocélulas de sulfeto de cádmio ou simplesmente células de CdS. Apresenta uma resistência extremamente elevada no escuro, da ordem de milhões de ohms. Essa resistência é diminuída para algumas centenas de milhares de ohms quando recebe iluminação direta, a luz forte, uma lâmpada próxima ou a luz direta do sol. A figura 3.10 ilustra a variação da resistência em função da incidência luminosa sobre o fotorresistor.

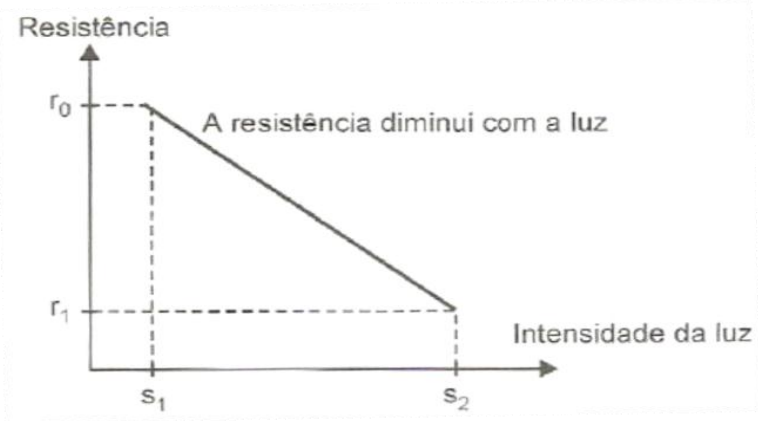

Figura 3.10 – A variação de resistência com a luz. (Fonte: Thomazini, 2007)

A superfície é composta de sulfato de cádmio. Pequenas trilhas do material condutor, eventualmente ouro, se entrelaçam com material condutor de modo a aumentar a superfície de contato e assim serem conseguidas maior capacidade de corrente e maior sensibilidade. (THOMAZINI, 2007)

É mostrado na figura 3.11 mostra um dos tipos de encapsulamento do LDR e sua simbologia.

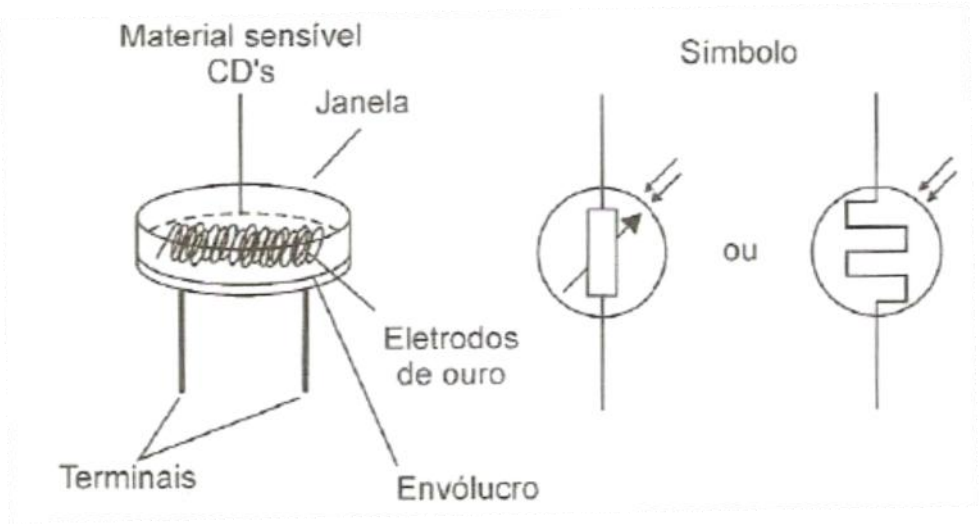

Figura 3.11 – LDR, aspecto e símbolo. (Fonte: Thomazini, 2007)

Os LDRs não são componentes polarizados, o que significa que a corrente pode circular nos dois sentidos. As variações da resistência com a luz são iguais em qualquer sentido. (THOMAZINI, 2007)

São mostrados na figura 3.12 os tipos de LDR mais comuns.

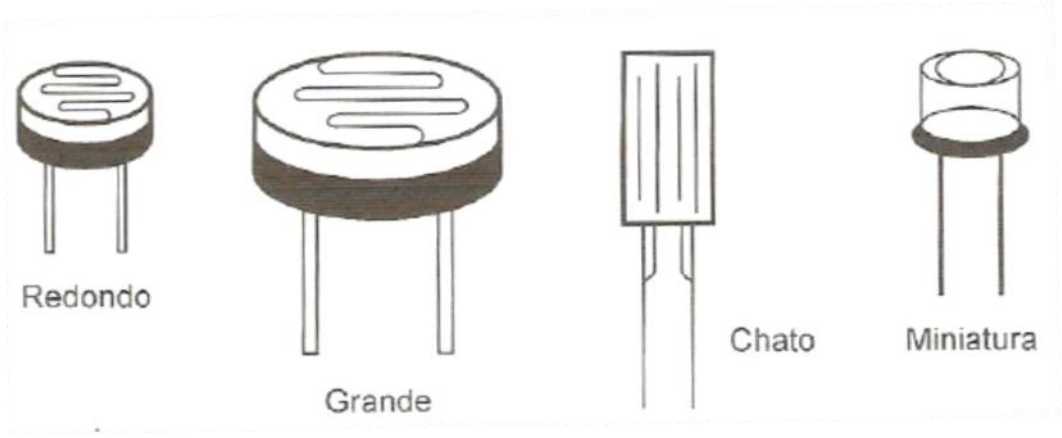

Figura 3.12 – Alguns tipos de LDRs encontrados no mercado. (Fonte: Thomazini, 2007)

Conforme Thomazini (2007, p.64) os LDRs são sensores lentos, não operando em velocidades maiores do que algumas dezenas de quilohertz. Enquanto outros dispositivos de sensores como os fotodiodos e os fototransistores podem perceber variações muito rápidas de luz, em frequências que chegam a dezenas ou mesmo centenas de megahertz, o LDR tem um “tempo de recuperação” muito longo. Estando totalmente iluminado e sendo a luz cortada, ocorre um determinado intervalo de tempo para que a resistência, inicialmente no valor mínimo, volte ao valor máximo, como indica a figura 3.13.

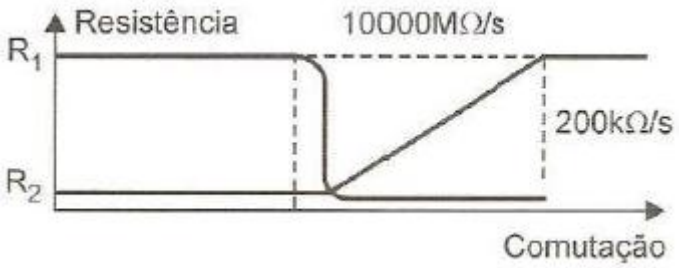

Figura 3.13 – Faixa de operação do LDR. R1 se refere à iluminação e R2 ao escurecimento sobre um LDR. (Fonte: Thomazini, 2007)

A baixa taxa de comutação do LDR impede que ele seja usado em sensores do tipo leitor de cartões perfurados, códigos de barras ou sistemas de alarmes modulados. No entanto, em aplicações mais simples, cujos tempos necessários para a atuação sejam maiores, como alarmes, brinquedos, sensores de luz ambiente, detectores de nível de iluminação, fotômetros, ele é muito útil. (THOMAZINI, 2007)

3.3 Microcontrolador

Um microcontrolador é um sistema computacional completo, no qual estão incluídos internamente uma CPU (Central Processor Unit), memórias RAM (dados), flash (programa) e E2PROM, pinos de I/O (Input/Output), além de outros periféricos internos, tais como, osciladores, canal USB, interface serial assíncrona USART, módulos de temporização e conversores A/D, entre outros, integrados em um mesmo componente (chip). (JUCÁ)

> O microcontrolador é programável, pois toda a lógica de operação é estruturada na forma de um programa e gravada dentro do componente. Depois disso, toda vez que o microcontrolador for alimentado, o programa interno

> será executado. Quanto à “inteligência” do componente, pode-se associá-la à Unidade Lógica Aritmética (ULA), pois é nessa unidade que todas as operações matemáticas e lógicas são executadas. Quanto mais poderosa a ULA do componente, maior sua capacidade de processar informações. (SOUZA, 1999, p.4)

Os microcontroladores são utilizados em várias áreas, e passam despercebidos no nosso dia a dia. Usamos por exemplo, nos portões de nossas residências, nos interfones, circuito de TV fechado, trancas e fechaduras eletrônicas, sensores de presença e de iluminação, etc.

Estima-se que sejam fabricados aproximadamente 63 milhões de veículos anualmente no mundo, sendo que cada veículo atual conta com aproximadamente 30 microcontroladores para controle de suas funções básicas, podendo chegar a mais de 70 microcontroladores nos modelos mais completos. São aplicações que agregam conforto, segurança e eficiência ao veículo, tais como: freio ABS, direção eletrônica, controle de tração, injeção eletrônica de combustível, controle de suspensão, acionamento inteligente de vidros e travas elétricas, acionamento de air-bags, redes internas, aquisição e tratamento de informações colhidas por sensores, controles de aceleração, entre outros. (CORTELETTI, 2006)

3.3.1 A Família PIC

Segundo Corteletti (2006, p. 6) A MICROCHIP TECHNOLOGY INC. é uma empresa de grande porte, com sede em Arizona, nos Estados Unidos da América. É uma empresa de expressiva participação no mercado de microcontroladores e semicondutores analógicos. Entre seus principais produtos, destaca-se o microcontrolador PIC® (Periferal Interface Controler), que possui uma boa diversidade de recursos, capacidades de processamento, custo e flexibilidade de aplicações.

Os microcontroladores PIC estão classificados em famílias, cada qual com uma característica relativa à sua performance e funcionalidade. A família PIC10, de menos recursos, é aplicada a funções de controle on-off mais simples e de menor porte, possuindo custo relativamente baixo (abaixo de US$ 1,00 por unidade) e a família dsPIC30 e dsPIC33 são adequadas para processamento e controle envolvendo aquisição, tratamento e processamento veloz de sinais analógicos, permitindo

desenvolvimento de aplicações mais complexas, ligadas à área de telecomunicações, comunicações sem fio de alta performance, controles em tempo real de alta velocidade, entre outras. (CORTELETTI, 2006)

Os microcontroladores PIC apresentam uma estrutura de máquina interna do tipo Havard, enquanto grande parte dos microcontroladores tradicionais apresenta uma arquitetura tipo Von-Neumann. A diferença está na forma como os dados e o programa são processados pelo microcontrolador. Na arquitetura tradicional, tipo Von-Neumann, existe apenas um barramento (bus) interno (geralmente de oito bits), por onde passam as instruções e os dados. Já na arquitetura tipo Havard existem dois barramentos internos, sendo um de dados e outro de instruções. No caso dos microcontroladores PIC, o barramento de dados é sempre de oito bits e o de instruções pode ser de 12, 14 ou 16 bits, dependendo do microcontrolador. Esse tipo de arquitetura permite que, enquanto uma instrução é executada, outra seja “buscada” da memória, o que torna o processamento mais rápido. Além disso, como o barramento de instruções é maior do que oito bits, o OPCODE da instrução já inclui o dado e o local onde ela vai operar (quando necessário), o que significa que apenas uma posição de memória é utilizada por instrução, economizando assim muita memória de programa. (SOUZA, 1999)

3.3.2 Microcontrolador PIC 16F788A

Neste projeto, foi utilizado o microcontrolador 16F877A da família PIC. Nele integramos periféricos como os sensores LDR e o motor. De acordo com Souza (2007 p. 18), o microcontrolador tem 40 pinos, possibilitando a montagem de um hardware complexo e capaz de interagir com diversas funções e recursos ao mesmo tempo; Via de programação de 14 bits e 35 instruções, Memória de programação EEPROM FLASH, que permite a gravação rápida do programa diversas vezes no mesmo chip. Memória RAM de 368 bytes; Três Timers (2x8 bits e 1x16 bits); Dois módulos CCP: Capture, Compare e PWM; Conversores analógicos de 10 bits (8x) e comparadores analógicos (2x); entre outros.

É mostrado na figura a seguir a pinagem do PIC 16F788A, de acordo com a figura, estão representados quais pinos são de entrada, saída ou entrada/saída.

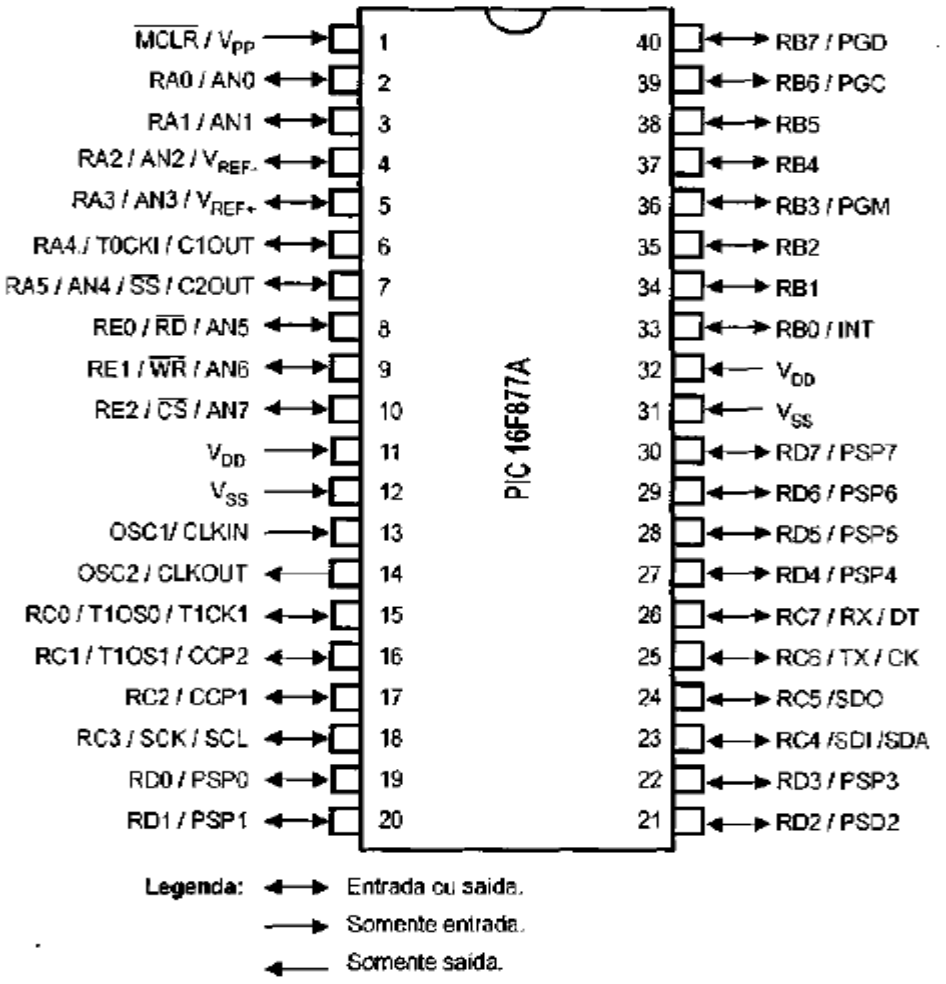

Figura 3.14 – Pinagem do PIC 16F877A. (Fonte: Souza, 2007)

São nesses pinos que conectamos os periféricos. Cada pino tem sua característica (descrição de cada um em anexo), como por exemplo, pinos de I/O, como o número 2, pinos de alimentação, como o número 11, etc.

3.4 Programação

A criação de programas para microcontroladores e microprocessadores pode ser uma tarefa desgastante à medida que aumenta a complexidade da aplicação sem desenvolvida. (PEREIRA, 2003)

De acordo com Pereira (2003 p.15) os primeiros dispositivos programáveis tinham seus programas escritos com códigos chamados códigos de maquina, que consistiam normalmente em dígitos binários. Estes eram inseridos por meio de um dispositivo de entrada de dados para então serem executados pela máquina, como por exemplo, 00110000010001100. Após isso, surgiu uma nova forma de programação, o Assembly. A partir dai, o programador poderia simplesmente utilizar o comando MOVLW 0x8C.

A linguagem Assembly é de baixo nível, ou seja, não possui nenhum comando, instrução ou função além daqueles definidos no conjunto de instruções do processador utilizado. Isto implica em um trabalho extra do programador para desenvolver rotinas e operação que não fazem parte do conjunto de instruções do processador. Produzindo, por conseguinte, programas muito extensos e complexos com um fluxo muitas vezes difícil de ser seguido. (PEREIRA, 2003)

É ai que entram as chamadas linguagens de alto nível. Elas são criadas para permitir a programação utilizando comandos de alto nível e que são posteriormente traduzidos para a linguagem de baixo nível (assembly ou diretamente para código de máquina) do processador utilizado. (PEREIRA, 2003)

3.4.1 Linguagem de Programação C

Segundo Pereira (2003, p. 18) a utilização de C para programação de microcontroladores com PICs parece uma escolha natural e realmente é. O uso de C permite a construção de programas e aplicações muito mais complexas do que seria viável utilizando apenas Assembly. Além disso, o desenvolvimento em C permite uma grande velocidade na criação de novos projetos, devido às facilidades de programação oferecidas pela linguagem e também a sua portabilidade, o que permite adaptar programas de um sistema para outro com um mínimo esforço.

O aspecto eficiência é realmente muito importante quando tratamos de microcontroladores cujos recursos são tão limitados como nos PICs, afinal, quando dispomos de apenas 512 palavras de memoria de programa 25 bytes de RAM (como no PIC12C508 e 16C54), é imprescindível que se economize memoria. (PEREIRA, 2003)

Além disso, a utilização de uma linguagem de alto nível como C permite que o programador preocupe-se mais com a programação da aplicação em si, já que o compilador assume para si tarefas como o controle e localização das variáveis, operações matemáticas e lógicas, verificação de bancos de memoria, etc.. (PEREIRA, 2003)

Após a programação é gerado um arquivo binário (hex file). De acordo com Corteletti (2006 p.14) esse arquivo contém o programa já em linguagem de máquina, ou seja, já com as instruções nativas do microcontrolador a ser programado.

Por padrão utiliza-se para representação o formato INTEL HEX, que há muito tempo é utilizado como regra de representação do programa em linguagem de máquina. Este formato facilita a representação dos bytes de instruções e dados do programa através de caracteres ASCII, permitindo que, em sistemas mais antigos, pudesse haver a inserção e modificação de instruções diretamente em linguagem de máquina. (CORTELETTI, 2006)

Na figura 3.15 é mostrado um exemplo de um programa em formato INTEL HEX 8 bits.

```
:1000000000308A00192800002130840000080319FC
:10001000182801 30F800F701F70B0C28F80B0B2813
:100020004A30F700F70B1228000000000800B092867
:10003000003484011F3083050630831 69F00FF3093
:100040008312A000201020088316870083120714 53
:100050006430A1000420201020088316870083123A
:0C00600007106430A100042022286300 77
:00000001FF
;PIC16F877
```

Figura 3.15 – Exemplo de um programa no formato INTEL HEX 8 bits. (Fonte: Corteletti, 2006)

Após a criação do hex file, o arquivo é gravado no microcontrolador. Na figura 3.16, é mostrado o esquema de gravação do gravador Microchip PIC.

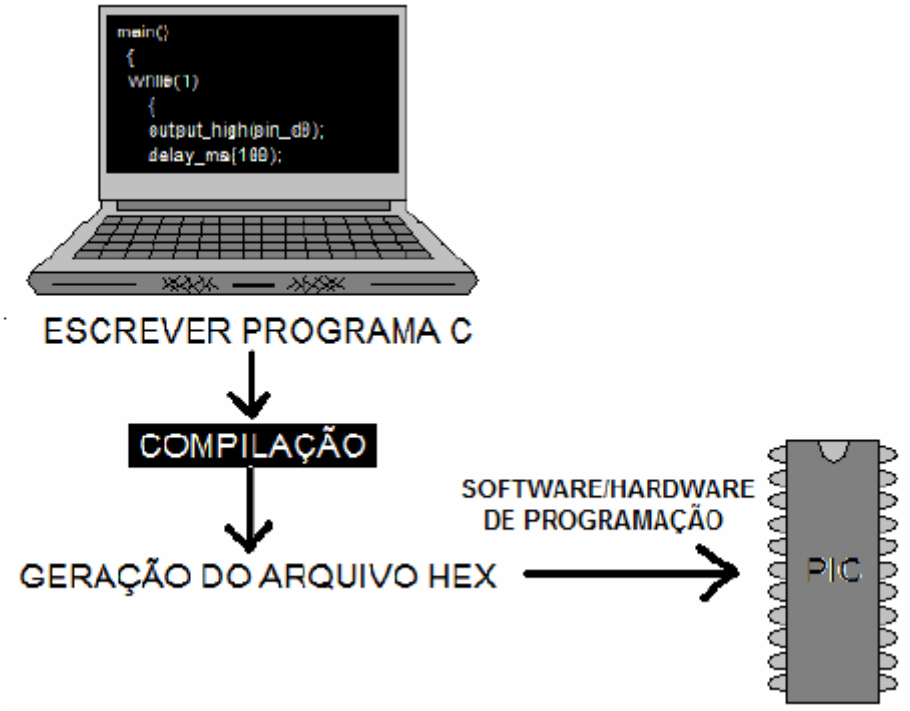

Figura 3.16 – Gravação do programa no PIC. (Fonte: Corteletti, 2006)

Segundo Corteletti (2006, p.18), na maioria dos compiladores, o processo é realizado em duas etapas, sendo isso transparente ao usuário. Na primeira etapa, existe a leitura do programa fonte, a verificação dos símbolos (nomes usados no programa), e da sintaxe dos comandos, verificando se há algum erro sintático. Em paralelo a isso, através da análise do significado de cada estrutura da linguagem, é produzido um programa em linguagem intermediária (em alguns casos o código intermediário é em assembly). Na segunda etapa, o compilador executa a tradução do código intermediário para a linguagem de destino, neste caso, em um arquivo HEX com as instruções de máquina do microcontrolador.

CAPÍTULO 4 – DESCRIÇÃO DO HARDWARE E SOFTWARE

4.1 Apresentação Geral

O objetivo deste projeto é a construção de um protótipo que visa o aproveitamento da luz. Mas, não apenas isso, pois o protótipo aqui apresentado também proporciona conforto ao usuário, dado que com a automatização da persiana reduz-se a necessidade abertura ou fechamento da persiana para o controle da luz natural incidente no ambiente.

Neste capitulo é apresentado o protótipo da persiana automatizada. O protótipo utiliza o PIC 16F877A, um motor interligado ao sistema de abertura/fechamento e que faz o movimento de acordo com o comando enviado pelo PIC, e também quatro sensores LDR que indicam para qual lado a persiana deverá se movimentar. Além disso, é apresentada toda a programação feita em C necessária pelo controle do projeto.

A figura 4.1 ilustra o fluxograma do protótipo.

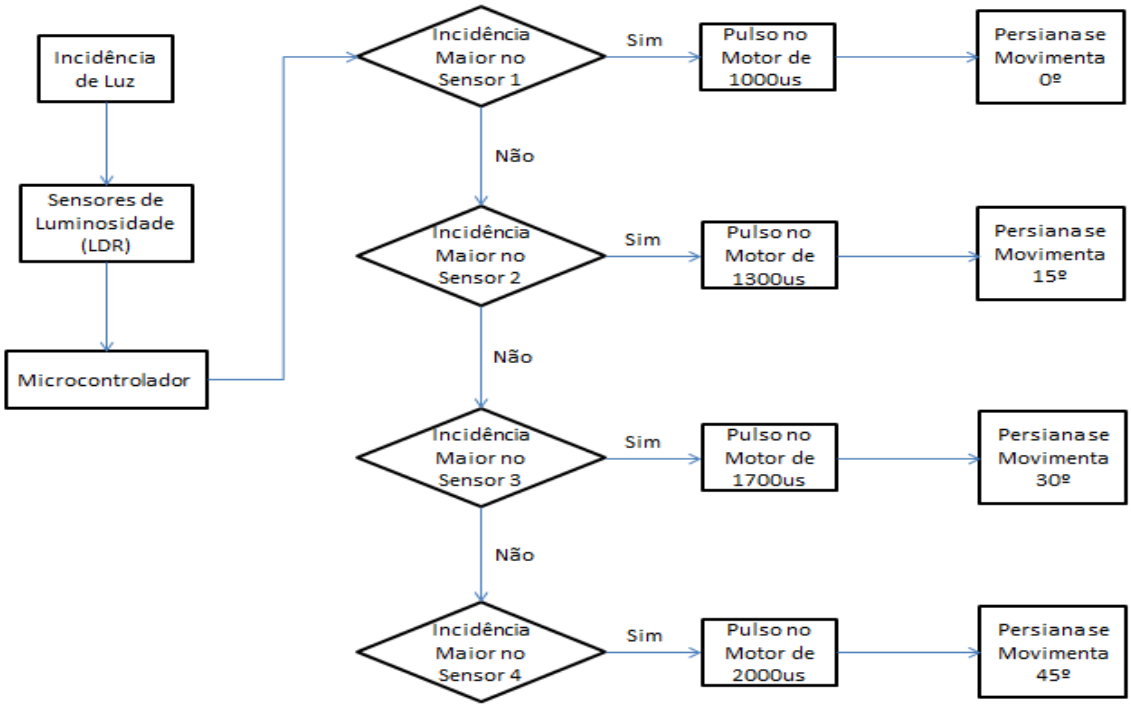

Figura 4.1 – Fluxograma do Protótipo

Quando a luminosidade incide sobre os sensores, a resistência dos LDRs varia, variando a tensão sobre eles. O microcontrolador faz a comparação de qual sensor tem o menor valor de tensão (quanto menor a tensão, maior será a incidência de luz devido à regra do divisor de tensão utilizada), e então, fará o motor movimentar a persiana para o ângulo onde está o sensor de luz com a menor tensão. Fazendo assim, que a luz passe por suas fretas, iluminando o ambiente. Para fazer o desligamento do circuito, basta desligar a fonte de energia da placa, cortando assim, a tensão que alimenta o microcontrolador.

4.2 Apresentação Final do Circuito

Antes da construção do circuito, foi feita a construção do protótipo no software de desenho de simulação de circuitos eletrônicos, Proteus 7.6. O Proteus é dividido em duas plataformas, o ISIS e ARES.

O ISIS permite a elaboração do diagrama esquemático do circuito e, além disso, permite a simulação do funcionamento do circuito. Tal funcionalidade é de extrema importância nos projetos, pois permite que várias configurações de hardware e software sejam testadas antes da montagem física do protótipo. Além disso, este programa também possibilita a utilização de instrumentos de medição e inclusão de gráficos que mostram os sinais obtidos na simulação. É um software completo, que tem desde vários tipos de microcontroladores aos menores componentes, permitindo a simulação de quase todos os tipos de circuitos.

O ARES é voltado para o desenvolvimento de placas de circuito impresso. Nele é feito todo o desenho da pinagem, das trilhas, e também, dos componentes. Assim como o ISIS, ele é bem completo, e caso não tenha o componente desejado, ele propicia que o mesmo seja construído. Ele também tem uma ótima integração com o ISIS, dado que após o projeto desenhado no ISIS, o mesmo pode ser facilmente convertido para o ARES com apenas um clicar de botão.

Na figura 4.2 são mostrados os componentes e as ligações feitas no ISIS.

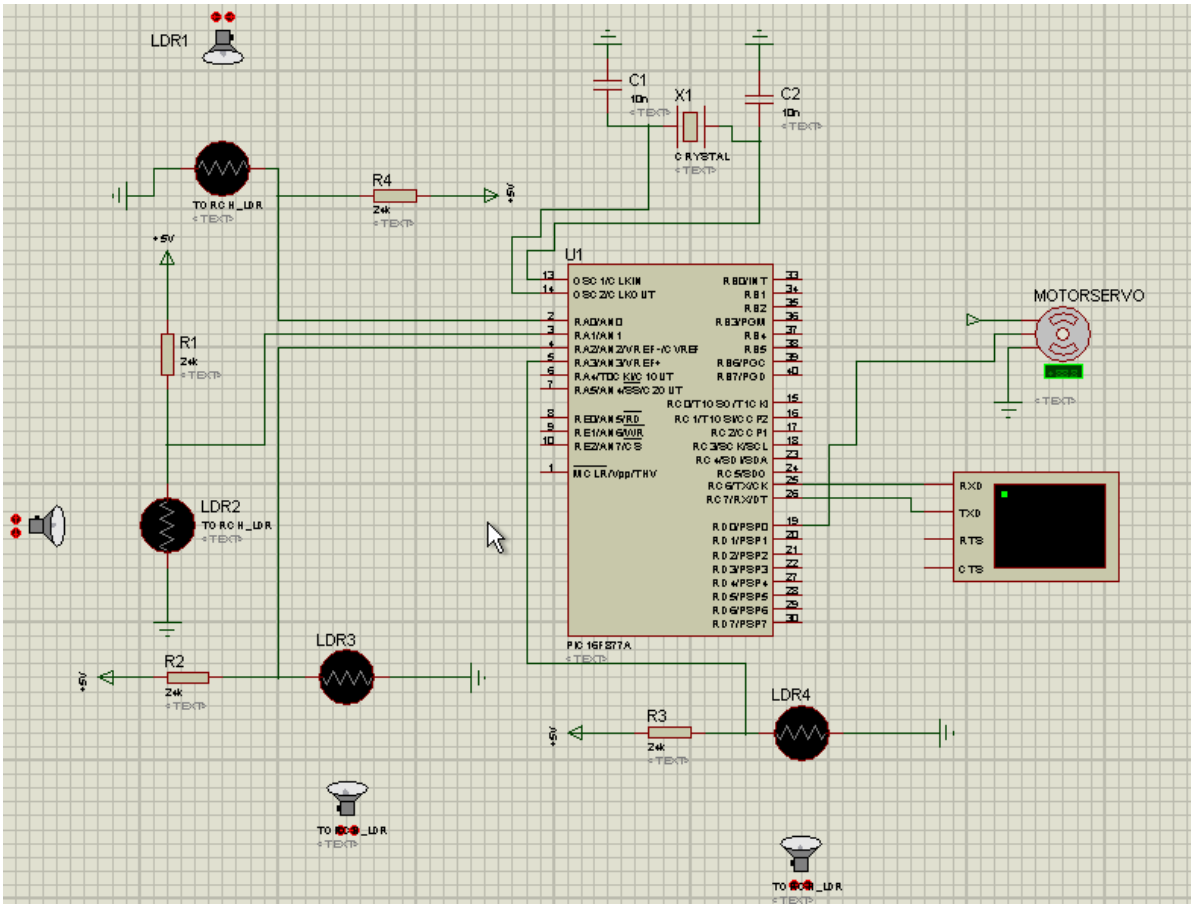

Figura 4.2 – Ligações ISIS.

Nesse projeto, temos os quatro sensores de luz (TORCH_LDR no ISIS) que são os responsáveis pela captação da luz. Os mesmos estão ligados na entrada do microcontrolador, pelas portas RA0/AN0, RA1/AN1, RA2/AN2 e RA3/AN3 respectivamente. Ao selecionar o sensor, ele já traz uma lanterna, para simular a incidência de luz no mesmo. Na figura 4.3, é mostrado o elemento de um sensor no ISIS.

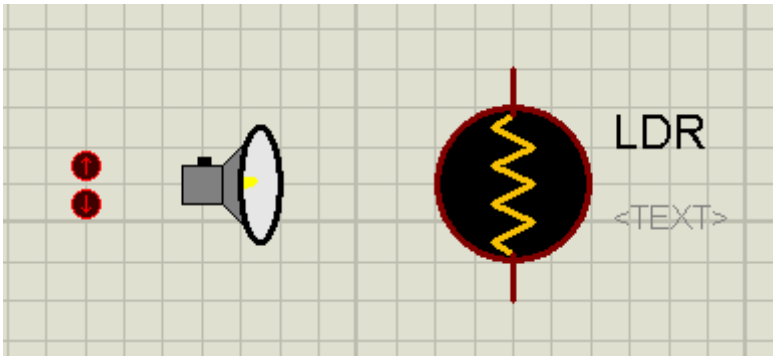

Figura 4.3 – Sensor LDR no ISIS.

Temos também, na figura 4.4, o cristal (CRYSTAL no ISIS) de 10MHz, que é responsável pelo clock externo do microcontrolador. Ele está ligado às portas OSC1/CLKIN e OSC2/CLKOUT do microcontrolador.

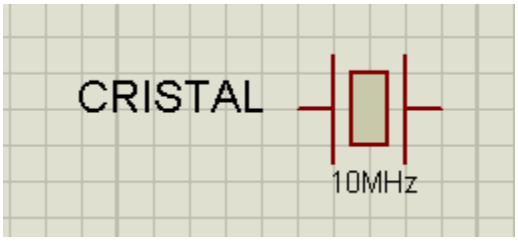

Figura 4.4 – Cristal no ISIS.

O motor servo é representado no ISIS pela sigla MOTOR-PWMSERVO, ele está ligado a porta de saída RD0/PSP0 do microcontrolador. O motor tem três conectores, a alimentação, o terra, e a entrada para o microcontrolador. É mostrado na figura a seguir, o elemento do motor no ISIS.

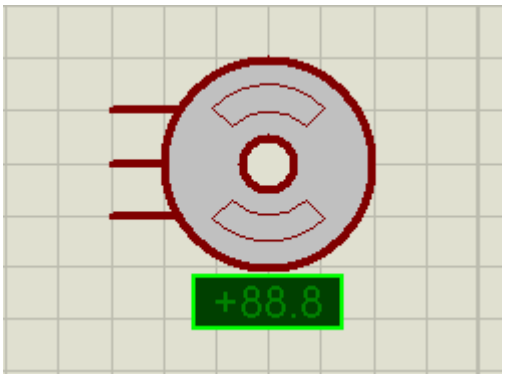

Figura 4.5 – Motor Servo no ISIS.

Temos também, o microcontrolador, é nele que carregamos o arquivo HEX que é gerado pelo compilador. Neste projeto, foi utilizado o PIC16F877A e pode-se observar na figura 4.2, todas as portas que foram utilizadas para fazer o teste neste circuito. Também foi colocado no desenho do projeto, apenas em nível de teste, o Virtual Terminal. Ele tem o mesmo objetivo que um LCD e foi colocado para mostrar na tela as tensões dos LDRs. Ele foi ligado às portas RC6/TX/CK e RC7/RX/DT do microcontrolador.

4.3 Controle do Microcontrolador

Foi utilizada nesse projeto, a linguagem de programação C. Utilizamos o PIC C *Compiler* para fazer a compilação e geração do arquivo HEX. É ilustrado na figura 4.6 um trecho do código onde foram feitas a definição do pino utilizado no motor, incluída a biblioteca utilizada, tal como definição do clock utilizado, a resolução do conversor A/D e configurações necessárias para o compilador.

```
1   /*****************************************Foto Sensor*****************************************/
2
3   #include <16f877A.h>          //Inclui a biblioteca do PIC;
4   #device adc=10                //Define o conversor A/D na resolução de 10bits;
5   #use delay (clock=10000000) //Cristal oscilador de 10Mhz;
6   #fuses HS                     //High-speed OSC > 4Mhz
7   #fuses NOWDT                  //No Watch Dog Timer;
8   #fuses PUT                    //Power Timer Up;
9   #fuses BROWNOUT               //Brown out Reset;
10  #define motor pin_D0          //Define motor na porta D0 do PIC;
```

Figura 4.6 – Trecho do código com pré-configurações.

Foram utilizadas algumas funções para não poluir tanto a função principal do programa. Na figura 4.7 é mostrado o código com elas.

```
110 /*************************************Inicialização do ADC*************************************/
111
112 void init_adc(void)                        // Configura O Canal 0 (RA0), Canal 1 (RA1), Canal 2 (RA2) e o Canal 3 (RA3);
113 {
114  setup_adc_ports(ALL_ANALOG);              //Configura todos os pinos da porta A como analogicos;
115  //setup_adc_ports(RA0_RA1_RA2_RA3_ANALOG); //Configura as portas RA0, RA1, RA2 e RA3 como analogicos;
116  setup_adc(ADC_CLOCK_INTERNAL);            //Clock A/D interno;
117 }
118
119 /*************************************Interrupção do Timer*************************************/
120
121 #INT_RTCC
122 void TIMER_INTERRUPT(void)                 //fica aqui por 32ms;
123 {
124  Timer_Flag = 1;
125 }
126
```

Figura 4.7 – Função da conversão A/D e do Timer.

4.4 Controle do Sensor de Luz

Para os sensores de luz, após a captação, é necessário fazer a conversão A/D, pois os valores que são obtidos são valores analógicos, e para que o microcontrolador possa processá-los, é necessário que sejam convertidos em valores digitais. A partir daí, os valores que foram captados pelos sensores nas portas são atribuídos às variáveis para em seguida, ser feita a comparação. Na figura 4.8, temos parte do código, onde foi feito esse procedimento.

```
32  /***********************************Leitura da Conversão ADC**********************************/
33
34  init_adc();                                  //Configura o conversor A/D;
35
36     do
37     {
38     set_adc_channel(0);                       //Configura o Canal AN0;
39     delay_ms(10);                             //Delay de 10ms;
40     ldr1 = read_adc();                        //Lê valor analógico convertido p/ digital;
41     leitura_ldr1 = (ldr1 * (5.0/1023.0));     //Converte 10 bits e o resultado em ponto flutuante;
42     //printf("\f\nVal1 = %f", leitura_ldr1);  //Mostra o valor da variável leitura_ldr1;
43     //delay_ms(500);                          //Delay de 500ms;
44
45     set_adc_channel(1);                       //Configura o Canal AN1;
46     delay_ms(10);                             //Delay de 10ms;
47     ldr2 = read_adc();                        //Lê valor analógico convertido p/ digital;
48     leitura_ldr2 = (ldr2 * (5.0/1023.0));     //Converte 10 bits e o resultado em ponto flutuante;
49     //printf("\f\nVal2 = %f", leitura_ldr2);  //Mostra o valor da variável leitura_ldr2;
50     //delay_ms(500);                          //Delay de 500ms;
51
52     set_adc_channel(2);                       //Configura o Canal AN2;
53     delay_ms(10);                             //Delay de 10ms;
54     ldr3 = read_adc();                        //Lê valor analógico convertido p/ digital;
55     leitura_ldr3 = (ldr3 * (5.0/1023.0));     //Converte 10 bits e o resultado em ponto flutuante;
56     //printf("\f\nVal3 = %f", leitura_ldr3);  //Mostra o valor da variável leitura_ldr3;
57     //delay_ms(500);                          //Delay de 500ms;
58
59     set_adc_channel(3);                       //Configura o Canal AN3;
60     delay_ms(10);                             //Delay de 10ms;
61     ldr4 = read_adc();                        //Lê valor analógico convertido p/ digital;
62     leitura_ldr4 = (ldr4 * (5.0/1023.0));     //Converte 10 bits e o resultado em ponto flutuante;
63     //printf("\f\nVal4 = %f", leitura_ldr4);  //Mostra o valor da variável leitura_ldr4;
64     //delay_ms(500);                          //Delay de 500ms;
```

Figura 4.8 – Trecho do código onde é feita a leitura e conversão A/D.

4.5 Controle do Motor Servo

Nesse projeto foi utilizado um motor servo, devido à facilidade de se trabalhar com este tipo de motor em diferentes angulações. Precisamos dele, pois a

persiana vai girar em ângulos específicos, por pulsos gerados no motor. Segue o trecho do código que com as pré-configurações do motor e a declaração das variáveis utilizadas no programa. Isso é mostrado na figura 4.9.

```
{
unsigned long ldr1, ldr2, ldr3, ldr4 = 0;                          //Criação de variáveis;
float leitura_ldr1 , leitura_ldr2, leitura_ldr3, leitura_ldr4=0;   //Criação de variáveis;
disable_interrupts(GLOBAL);                                        //Desabilita todas as interrupções;
enable_interrupts(INT_RTCC);                                       //Habilita interrupção do timer;
set_tris_D(pin_D0);                                                //Configura RD0 como saída do pulso PWM p/ o servo;
setup_counters(RTCC_INTERNAL, RTCC_DIV_128);                       //Incrementa o timer a cada 128us;
Timer_Flag = 0;                                                    //Inicializa o flag de interrupçaõ do timer;
enable_interrupts(GLOBAL);                                         //Habilita interrupções;
```

Figura 4.9 – Trecho do código que com as pré-configurações do motor e a declaração das variáveis utilizadas no programa.

O trecho a seguir, na figura 4.10, faz a rotação do motor para o lado do LDR1. Ele foi replicado quatro vezes, uma para cada LDR, sendo modificado o número do LDR e o delay_us para movimentar o motor para o lado desejado.

```
/*************************************Comparação dos LDRs*************************************/

   if((leitura_ldr1 < leitura_ldr2 && leitura_ldr1 < leitura_ldr3) && leitura_ldr1 < leitura_ldr4)
   {
   while(Timer_Flag == 0);                  //Aguarda até que o flag configure a interrupção do time;
   output_high(motor);                      //Coloca em nível alto o pino RD0;
   delay_us (2800);                         //Servo = 2.8ms
   output_low(motor);                       //Coloca em zero o pino RD0;
   //printf("\f\nVal1 = %f", leitura_ldr1); //Mostra na tela qual o valor da leitura do LDR1;
   }
```

Figura 4.10 – Parte do código que faz a comparação entre os LDRs.

Observamos na figura 4.10, que demos um pulso de 2,8ms no motor servo. Com esse pulso, o servomotor gira, e é forçado a chegar ao ângulo de rotação máximo. Foi necessário forçar o motor, pois ao rotacionar o sistema de abertura e fechamento da persiana, as lâminas não alcançaram um ângulo desejado.

4.6 Inserção do Algoritmo no Microcontrolador

Como dito anteriormente, para inserir o código que foi programado no microcontrolador do ISIS, é necessário fazer a compilação e geração do arquivo HEX. Nesse projeto, utilizamos o PIC C Compiler na versão 3.43.

Ao compilar o código, aparecem quais são os arquivos de saída gerados pela compilação, dentre eles estão arquivos do tipo ERR, SYM, LST, COF, PJT, TER, STA e o mais importante para o nosso teste, o arquivo do tipo HEX. Observa se também, quanto de memória ROM e RAM do microcontrolador está sendo utilizada para fazer o carregamento do programa no mesmo. Na figura 4.11 é mostrada a compilação do programa.

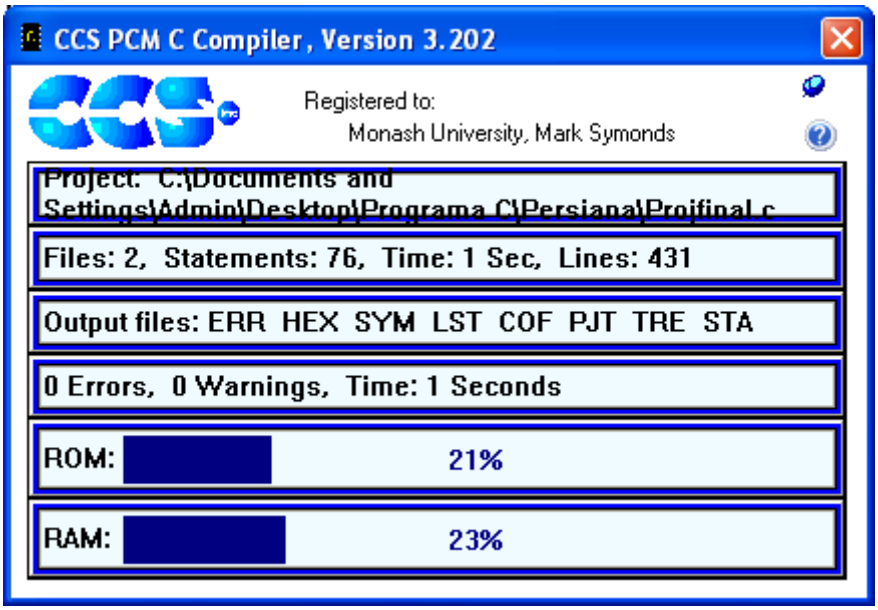

Figura 4.11 – Compilação e criação dos arquivos de saída.

Após a criação do arquivo HEX, é necessário fazer a inserção deste no ISIS. Esse procedimento é necessário para que todos os testes sejam feitos no programa antes de fazer a gravação no microcontrolador.

Para fazer a gravação do programa no microcontrolador, foi utilizado o gravador PIC KIT 2 USB. A placa de gravação contém um socket para que seja inserido o microcontrolador, no nosso caso, o PIC 16F877A. Além da placa de gravação, foi utilizado o programa *PICkit 2 programmer*. Esse programa, assim como o driver da placa, devem ser instalados para que a placa seja reconhecida pelo computador. Nas figuras 4.12 e 4.13, temos a placa de gravação e a pagina inicial do programa com explicações dos campos respectivamente.

Figura 4.12 – Foto da placa de gravação.

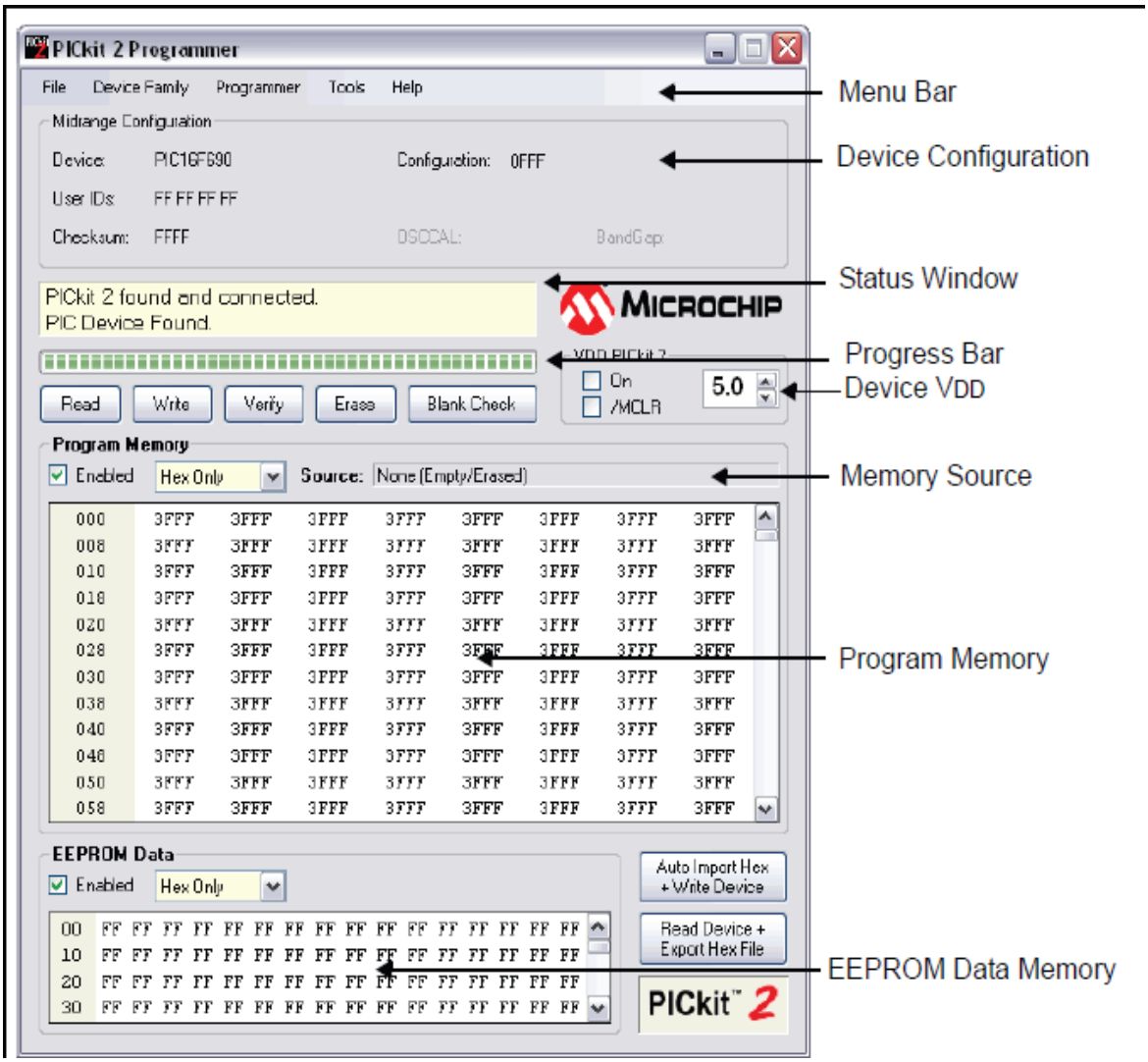

Figura 4.13 – Tela inicial do PICkit 2 (MICROCHIP)

Após fazer a instalação correta do programa, ao conectar a placa ao computador usando um cabo USB, o microcontrolador é imediatamente reconhecido.

Aparecendo assim, tudo o que está gravado na memória deste. Após a gravação, observa se que os espaços de memória são escritos por vários números hexadecimais e uma mensagem é mostrada na tela, dizendo que a gravação foi um sucesso. Nas figuras 4.14 e 4.15, são mostrados respectivamente, os endereços de memoria escritos e a mensagem dizendo que a gravação foi bem sucedida.

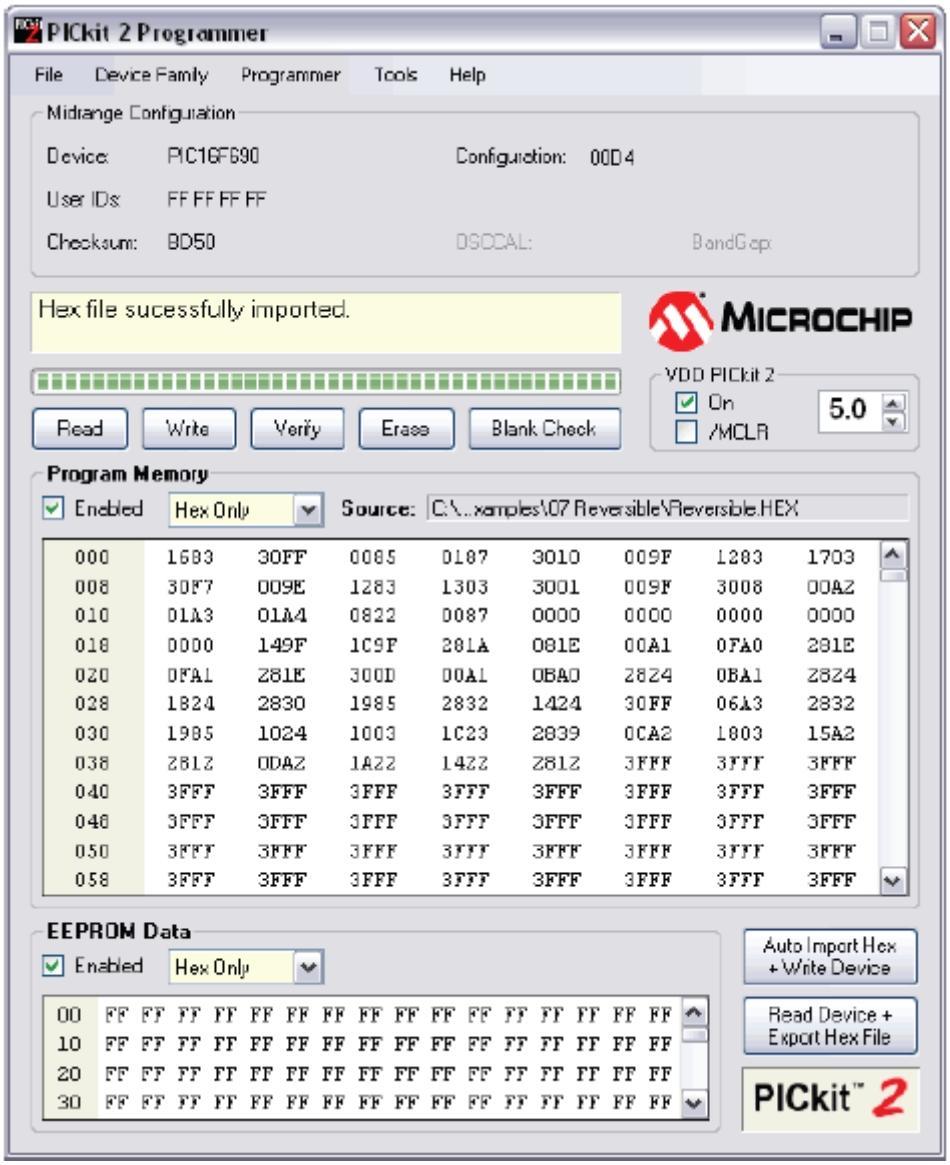

Figura 4.14 – Arquivo HEX importado e gravado na memória. (MICROCHIP)

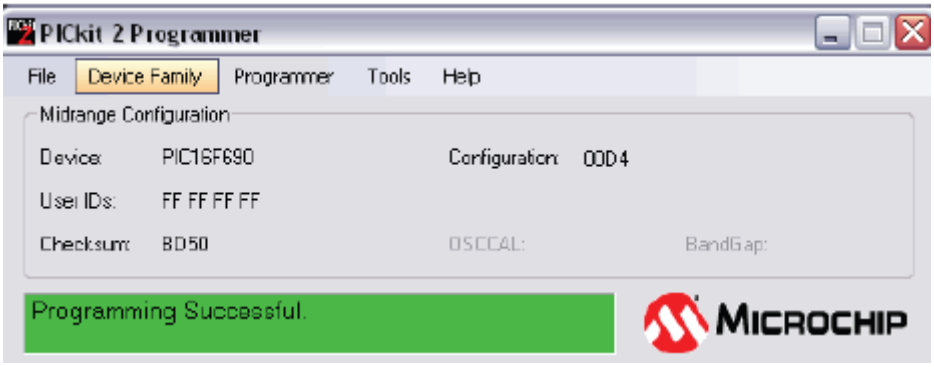

Figura 4.15 – Mensagem com status da gravação. (MICROCHIP)

4.7 Apresentação do Protótipo

Após a construção do protótipo e teste do programa no software de simulação ISIS, foi feita a arquitetura da placa no software de desenho, o ARES. Foram desenvolvidas 5 placas, uma para o circuito principal, que acopla o microcontrolador, cristal, regulador de tensão, entrada para a fonte de energia, circuitos eletrônicos em geral, pinos para conexão com o motor e com as placas do LDR, além de uma entrada para o gravador PICkit. As outras 4 placas, foram posicionadas estrategicamente no topo da estrutura com uma distância de 5cm separando-as. Estas são compostas apenas pelo sensor de luz e pelo pino que faz a conexão com a placa principal. Na figura 4.16 e 4.17 são mostradas as arquiteturas no ARES da placa principal e das placas dos LDRs respectivamente.

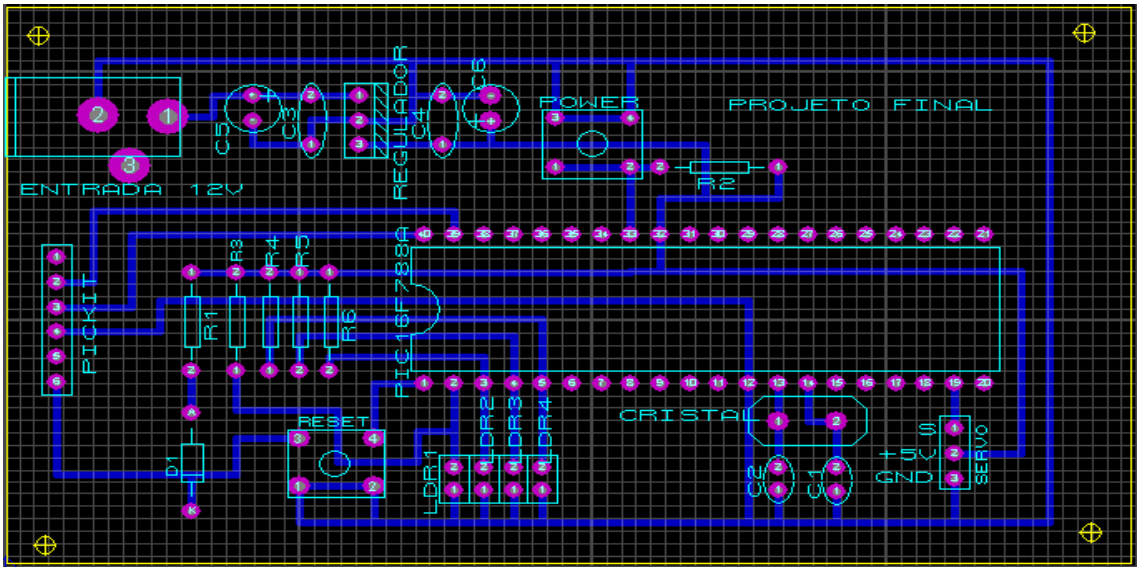

Figura 4.16 – Desenho da placa principal no ARES.

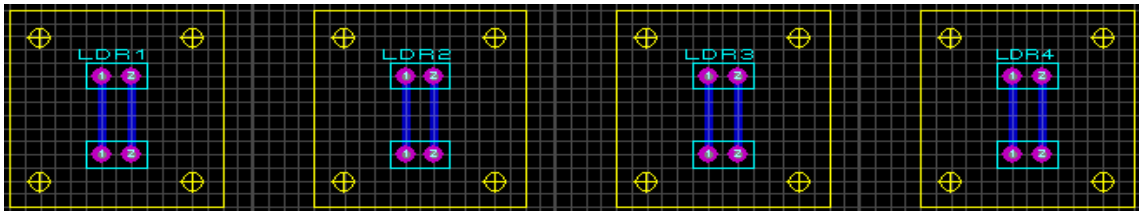

Figura 4.17 – Desenho das placas dos LDRs no ARES.

Após o desenho das placas no ARES, foi iniciado o processo de construção. Que vai desde a impressão do desenho até a solda dos componentes. Nas figuras 4.18 e 4.19 são mostradas as placas já impressas e com os componentes soldados.

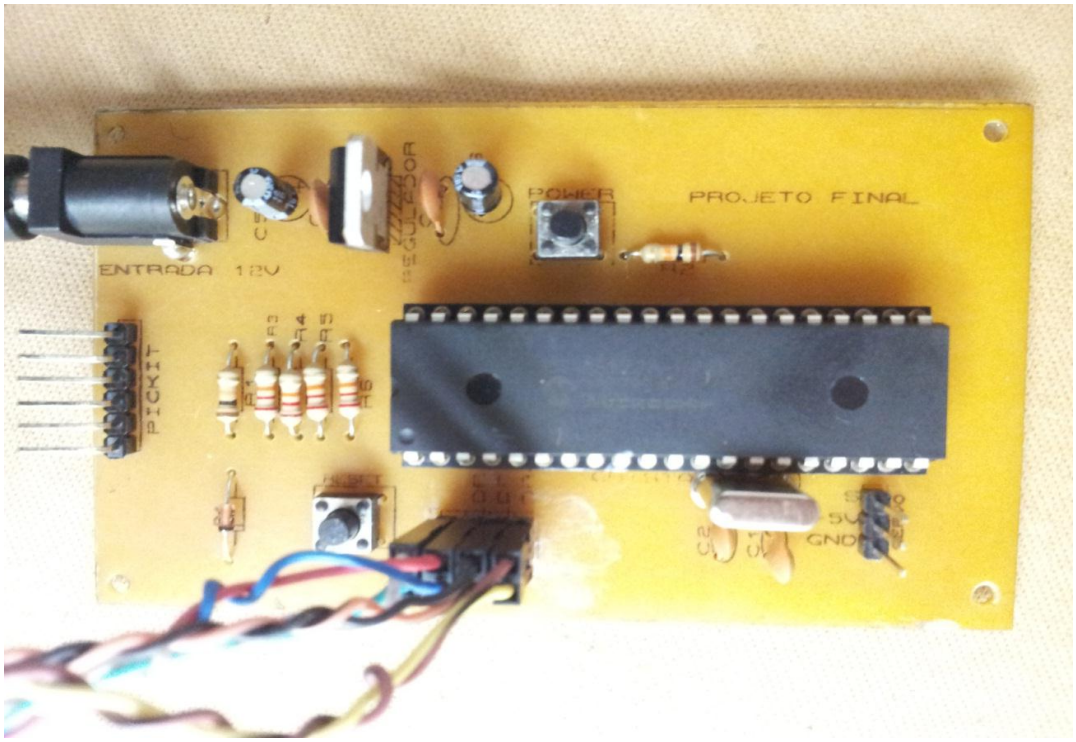

Figura 4.18 – Placa principal com os componentes.

Figura 4.19 – Placa do LDR.

Com a finalização da construção das placas, o próximo passo foi construir a estrutura que vai manter a parte que faz a movimentação persiana junto ao motor e placas que fazem o controle do mesmo. É ilustrada na figura 4.20 a estrutura já com a persiana, motor e placas já integradas.

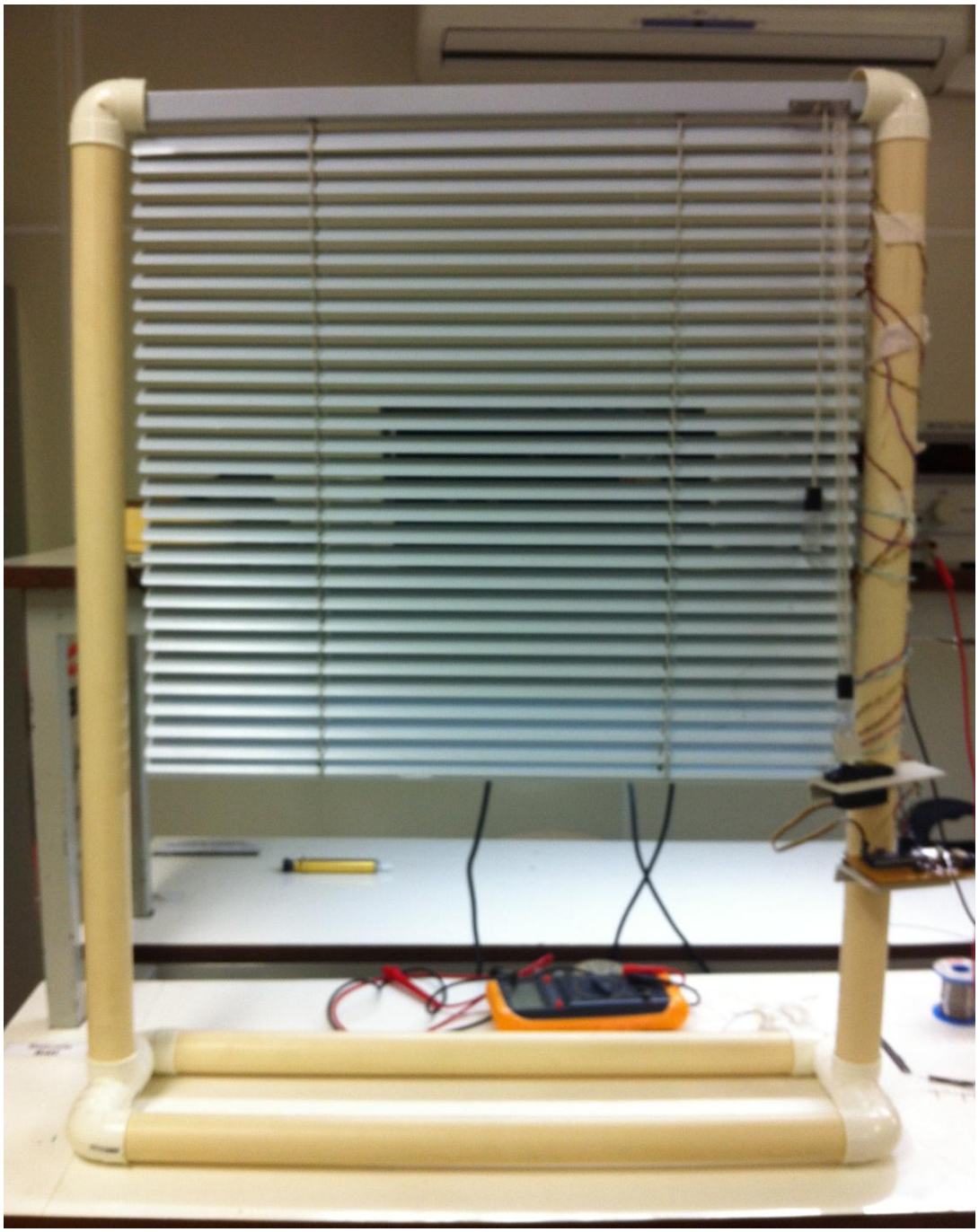

Figura 4.20 – Estrutura com persiana, motor e placas.

A ligação física entre motor e persiana é mostrada na figura 4.21.

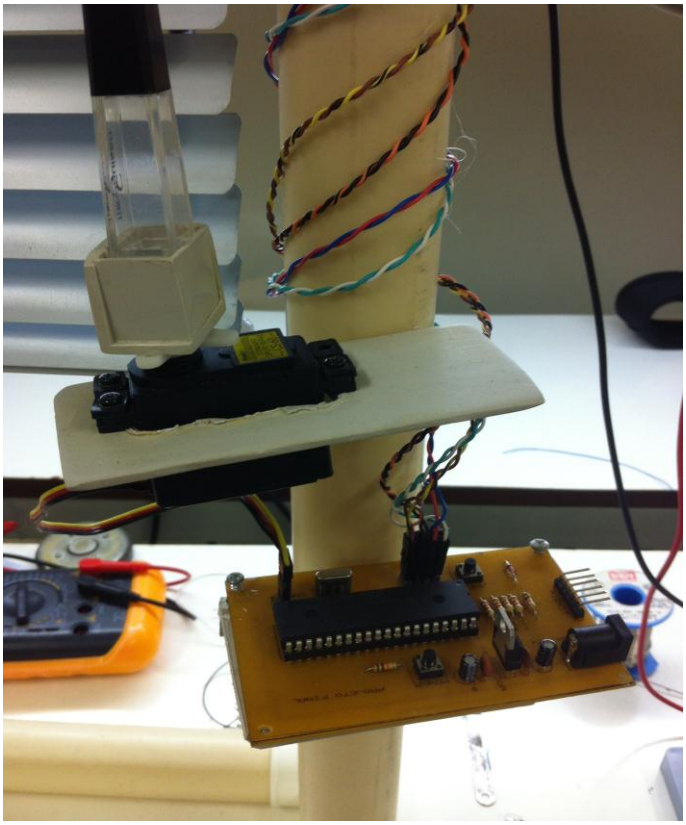

Figura 4.21 – Ligação entre motor e Persiana.

Para fazer a integração do motor com a persiana, foi feita uma estrutura de plástico presa ao motor, essa estrutura foi construída para que o sistema de abertura/fechamento da persiana não fosse danificado.

CAPÍTULO 5 – RESULTADOS OBTIDOS

Neste capítulo são mostrados todos os resultados obtidos ao longo da monografia, tal qual, figuras ilustrativas do protótipo, e os testes realizados sobre a solução.

5.1 Apresentação da Área de Aplicação do Modelo

O projeto tem uma área de aplicação vasta. Como o objetivo é aproveitar a luz solar, este projeto poderia ser utilizado em ambientes que necessitem de luz por longos períodos, como por exemplo, escolas, faculdades e escritórios comerciais. Tendo em vista maior iluminação é necessário usar uma persiana maior, para que assim, os raios do sol passem com maior facilidade entre as suas lâminas.

Devido a espaços menores dos ambientes nas residências, podemos utilizar uma persiana menor para que parte dos raios solares sejam filtrados pela própria persiana. É necessário ficar atento ao fato de que se a luz passar com muita intensidade, pode trazer certo desconforto aos moradores.

5.2 Testes

O projeto foi iniciado no programa ISIS, nele foi construída toda a estrutura do projeto para que pudéssemos dar inicio a programação e posteriormente aos testes do programa. Após todos os testes feitos no programa utilizando o ISIS, foi iniciado o desenho da placa no ARES, para depois partir para montagem da placa de circuito impresso.

O programa basicamente faz uma comparação entre os sensores de luz, Após isso, é enviado um pulso para o motor se movimentar para a posição desejada. É mostrado na figura 5.1, os testes feitos no programa pelo software ISIS. No primeiro caso, a lâmpada está mais próxima do sensor 1, indicando assim, que ele está recebendo mais luz do que os outros três sensores. Quando isso acontece, a resis-

tência dele fica menor e é gerado um pulso no motor que faz com que ele gire em +90º em relação a sua posição inicial.

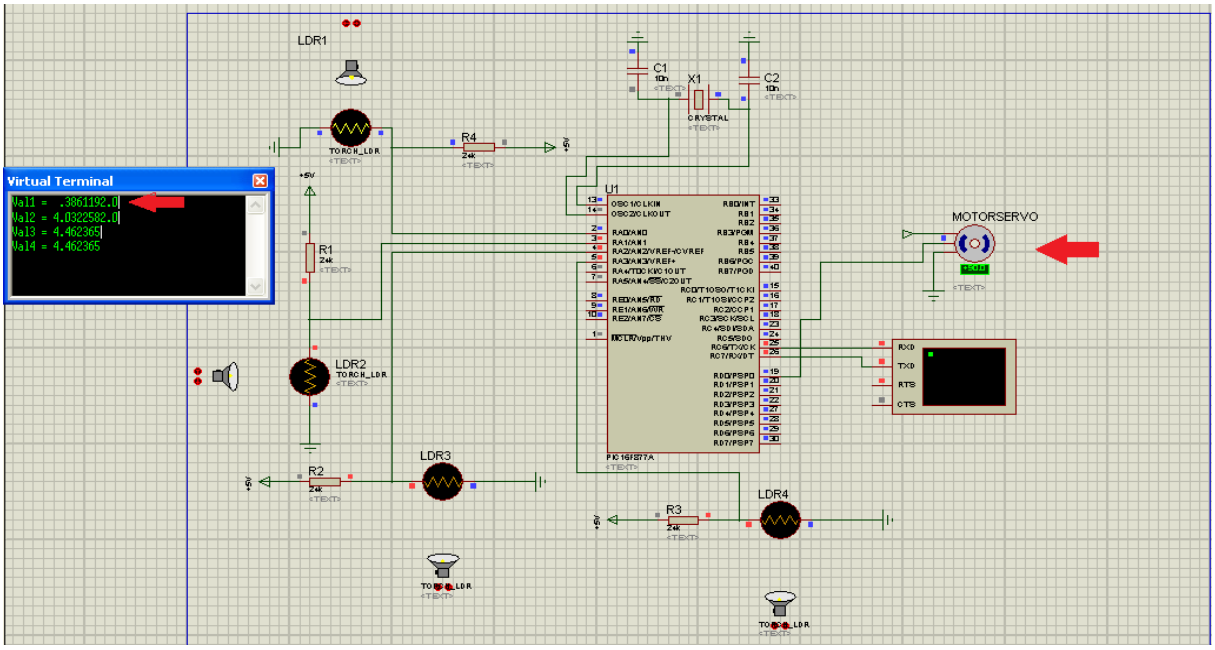

Figura 5.1 – Sensor 1 recebendo mais luz.

Na figura 5.2 é ilustrado o sensor 2 recebendo mais luz. Com isso sua resistência passa a ser a menor e é gerado um pulso no motor que faz com que ele gire em +33,2º em relação sua posição inicial.

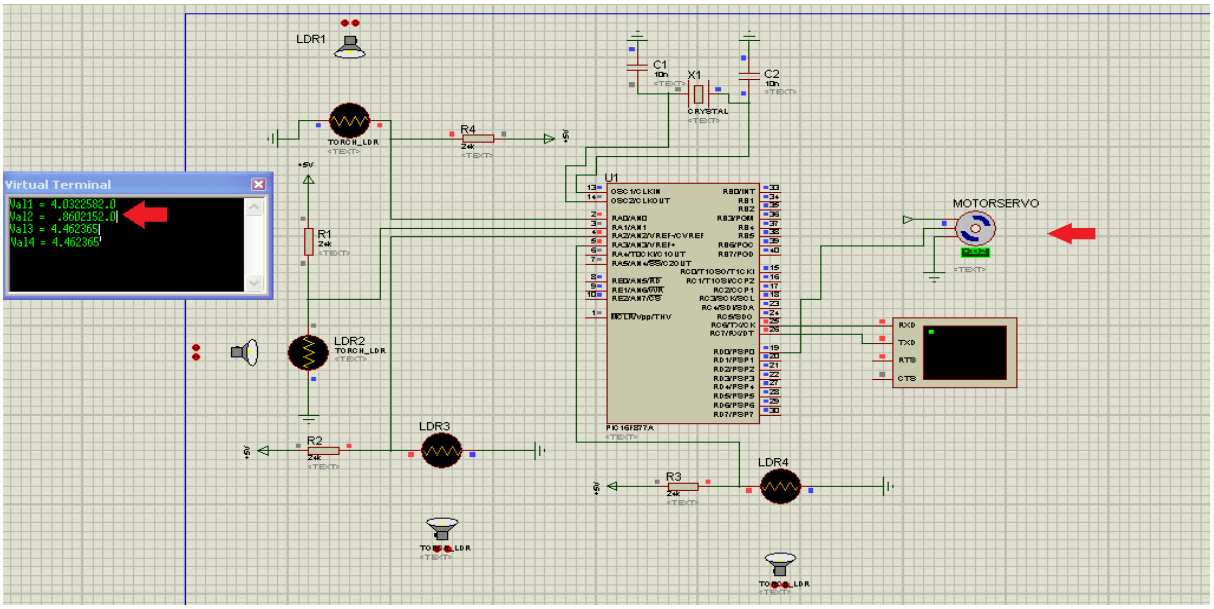

Figura 5.2 – Sensor 2 recebendo mais luz.

Quando a luz está com maior incidência no sensor 3, sua resistência diminui como aconteceu nos casos anteriores. O pulso gerado no motor faz com que

ele seja movimentado em -61,9º em relação a sua posição inicial, como é mostrado na figura 5.3, a seguir.

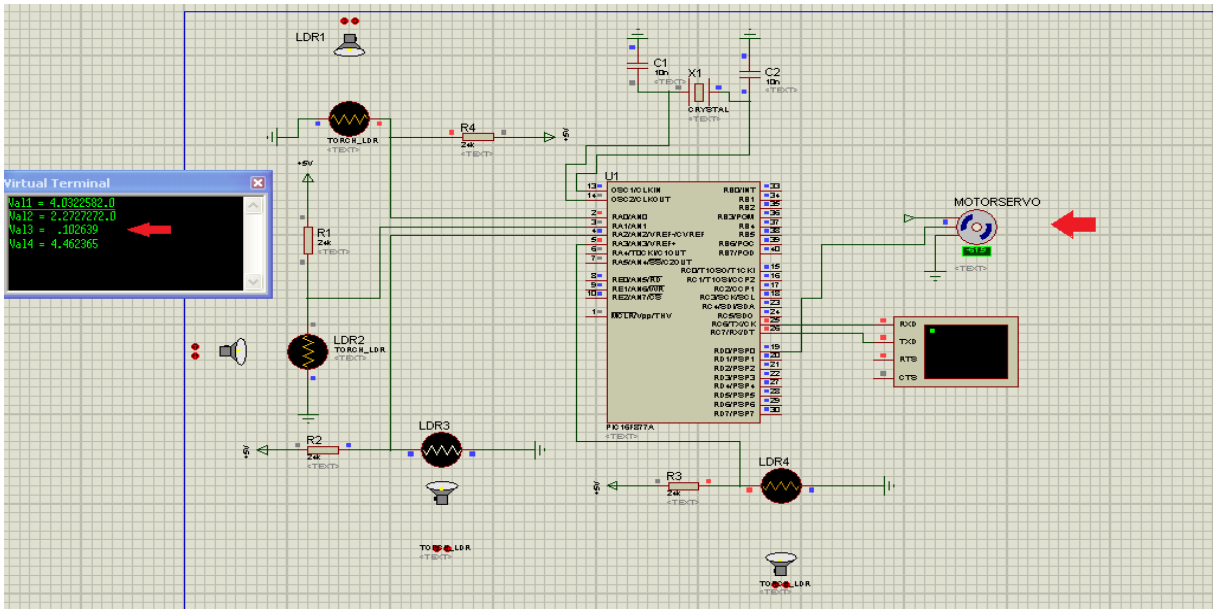

Figura 5.3 – Sensor 3 recebendo mais luz.

Na figura 5.4 é ilustrado quando o sensor 4 está recebendo mais luz. Podemos notar que sua resistência diminuiu, sendo a menor dentre os outros sensores. Quando isso acontece o motor vai se movimenta e para em -89º em relação a sua posição inicial.

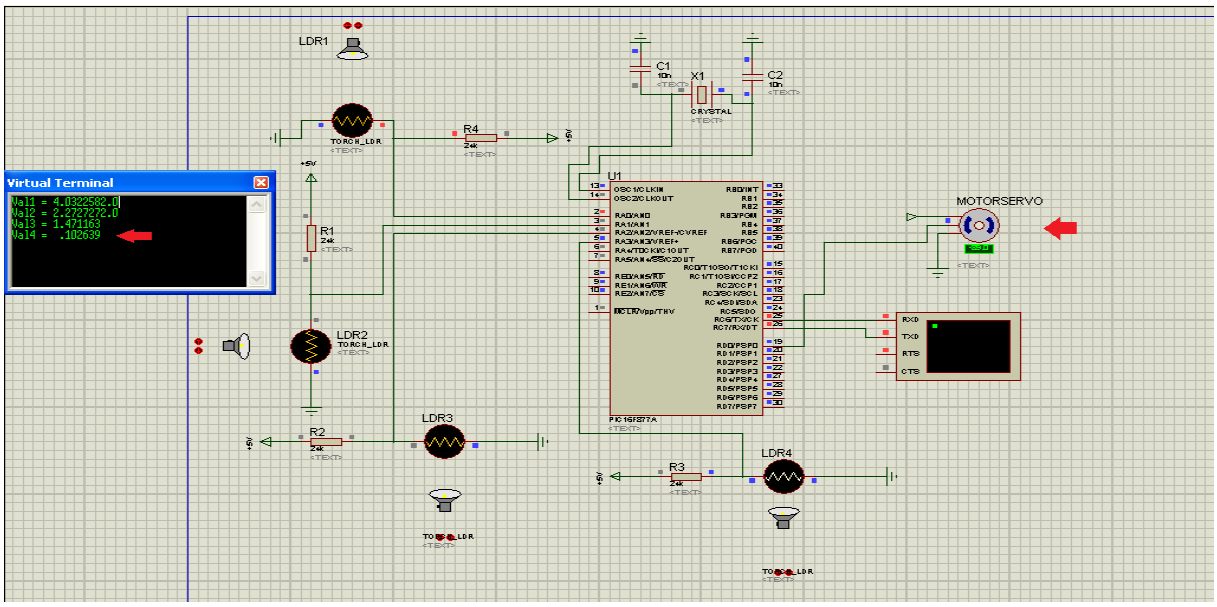

Figura 5.4 – Sensor 4 recebendo mais luz.

5.3 Problemas Encontrados

Foi visto nos testes realizados no IRIS, que o projeto estava funcionando de acordo com o esperado. Ou seja, o motor estava se movimentando de acordo com os valores dos sensores. Porém, para a implementação física desta etapa várias dificuldades tiveram de ser superadas. Uma delas foi à aquisição dos equipamentos. O servo motor não foi encontrado no comércio local e nem na internet, tendo sido adquirido por empréstimo.

A persiana foi outro problema, pois para o projeto, era necessário uma persiana bem específica. O sistema de abertura/fechamento da persiana tem que girar suficientemente as lâminas com o menor movimento possível. Tendo que ser instalado com engrenagens para que isso fosse possível. Ele foi adquirido por empréstimo em uma fábrica de persianas no comércio local.

Além das aquisições de equipamentos, ao longo do desenvolvimento foram encontradas outras dificuldades. Um dos problemas foi quanto ao giro do servomotor. O modelo adquirido tem certa de giro, sendo necessário forçar um pouco o mesmo para que as lâminas da persiana sejam movimentadas da forma desejada.

Uma das dificuldades encontradas na programação foi o controle do motor, pois além de ter que chamar uma série de interrupções, foi necessário ajustar os pulsos que façam com que a persiana seja movimentada de maneira adequada. Esse controle pode ser conferido no apêndice A.

5.4 Avaliação do Modelo

O projeto se mostrou útil e pode ser instalado em vários ambientes. Além de economizar energia elétrica, ele traz conforto e reduz a necessidade de reparos na persiana.

Os custos do projeto podem diminuir consideravelmente caso seja feita uma produção em escala, fazendo assim, um projeto economicamente viável para a grande parte das pessoas, tendo em vista que traria uma economia de energia que em longo prazo seria um investimento para a família ou empresa.

5.5 Custos de Modelo

Para realizar o projeto, foi necessário à compra de alguns componentes. A seguir é mostrada a tabela de preços.

Tabela 5.1 – Tabela de Preços

COMPONENTES	QUANTIDADE	PREÇO POR UNIDADE (R$)	PREÇO TOTAL (R$)
MICROCONTROLADOR PIC 16F877A	1	14,30	14,30
REGULADOR DE TENSÃO LM7805	1	1,50	1,50
DIODO 1N4148	1	0,10	0,10
CHAVE MINI PUSH BUTTON 4P	2	2,00	4,00
TERMINAL MODU FÊMEA	50	0,06	3,00
ALOJAMENTO MODU FÊMEA - 01 X 06	10	0,10	0,10
CABO MANGA SEM BLINDAGEM 16X26*	1	3,30	3,30
CAPACITOR ELETROLÍTICO 10uF-16V	2	0,5	1,00
CAPACITOR CERÂMICO 22pF	2	0,10	0,20
CAPACITOR CERÂMICO 100nF	2	0,10	0,20
RESISTOR 22k	4	0,15	0,60
RESISTOR 10k	2	0,10	0,20
BARRA DE PINOS 180º	1	1,00	1,00
BARRA DE PINOS 90º	1	0,50	0,50
MOTOR SERVO (GRANDE)	1	31,00	31,00
SENSORES LDR	4	0,60	2,40
CRISTAL 10MHZ PERFIL BAIXA	1	2,50	2,50
GRAVADOR USB PIC PICKIT2 MPLAB	1	129,00	129,00
FONTE	1	22,00	22,00
ENTRADA PARA A FONTE	1	0,50	0,50
KIT PARA CIRCUITO IMPRESSO	1	58,00	58,00
PERSIANA	1	90,00	90,00
TUBOS DE PVC*	4	6,25	25,00
TOTAL			**R$ 390,40**

*Componentes medidos em metros.

Uma maneira de economizar com esse projeto é criando a placa de gravação do microcontrolador, a mesma foi o componente mais caro do projeto, totalizando R$129,00. Também é possível encomendar a construção de uma persiana que faz com que o preço reduza, além de ter uma persiana feita sob encomenda para as necessidades do comprador.

CAPÍTULO 6 – CONSIDERAÇÕES FINAIS

6.1 Conclusões

Neste trabalho foi desenvolvido um protótipo de uma persiana automatizada. Com as devidas adaptações, este projeto pode tanto atender a área de automação residencial como para escritórios comerciais e até fábricas.

Para a realização do projeto, foi feita a integração do microcontrolador PIC16F877A que faz o controle de todos os componentes, os LDRs que tem suas tensões comparadas, uma persiana, que é movimentada para que a luz passe por suas frestas, e um servomotor, que faz toda a movimentação da mesma.

O trabalho teve como foco principal a economia de energia, tendo em vista que com o aproveitamento de luz solar, o uso de luz artificial se faz menos necessária, pois ao invés de acendermos três lâmpadas para iluminar o ambiente, será necessário acender duas ou somente uma. Com essa economia, além de ajudar ao meio ambiente, pois menos energia será gasta, o usuário terá um investimento, pois sua conta de energia diminuirá ao longo do tempo.

Mesmo com os problemas encontrados ao longo do desenvolvimento do projeto, o objetivo do trabalho foi alcançado. A persiana foi automatizada, e ela é capaz de se movimentar de acordo com a luz incidente nos sensores, trazendo assim, economia de energia, além de certo conforto para o usuário.

É possível inferir que o projeto tem potencial e pode vir a ser utilizado em diversos ambientes.

6.2 Propostas para Trabalhos Futuros

Como proposta para trabalhos futuros, a troca do servomotor por um outro motor que faça com que a persiana tenha um giro maior, e possa ser fazer movimento em ângulos maiores.

Também a inserção de mais LDRs, com isso, a persiana teria mais posições de aproveitamento com a luz do sol. Atualmente ela possui apenas quatro posições.

Outra sugestão seria a integração da persiana com o sistema de iluminação do ambiente. Fazendo assim, um controle automático da luz do ambiente, apagando e acendendo luzes automaticamente de acordo com a luz que se encontra no local.

Além dessa integração, pode se fazer uma integração com automação da janela, abrindo e fechando a janela com sensores de vento e de chuva, protegendo assim a persiana. Com isso, gastos com a manutenção da persiana iriam diminuir.

REFERÊNCIAS BIBLIOGRÁFICAS

ALFARONE, Rafael de Mello. **Projeto Final – Persiana Residencial Automatizada Utilizando Sensor de Luminosidade.** Brasília/DF (2º semestre de 2010)

BRUM, Lilian. **Consumo de Energia X Meio Ambiente: Uma luta constante.** Publicado em: 24 ago. 2009. Disponível em: <http://www.geniodalampada.com/index.php?option=com_content&view=article&id=84:consumo-de-energia-x-meio-ambiente-uma-luta-constante&catid=87::gestao-ambiental&ltemid=131> Acesso em: 08 set. 2012.

CORTELETTI, Daniel. **Introdução à programação de microcontroladores Microchip PIC.** Publicado em Out. 2006. <http://sbrt.ibict.br/dossie-tecnico/downloadsDT/NTE=> Acesso em 02 out. 2012.

FILHO, Guilherme Filippo**. Motor de Indução.** 1ª Edição. Editora Erica Ltda, 2000.

FRANCISCO, António. **Motores Eléctricos.** 1ª Edição. Editora ETEP, 2008.

HOBBICO; ***Servo Specifications and Applications.*** Disponível em <http://www.hobbico.com/radioaccys/hcam1000.html> Acesso em 05 nov. 2012.

MANOSSO, Radamés. **Iluminação e lâmpadas.** Disponível em <http://radames.manosso.nom.br/ambiental/energia/iluminacao-e-lampadas/> Acesso em 10 out. 2012.

MICROCHIP. ***PICkit 2 Programmer/Debugger, User's Guide*****.** 2008.

JUCÁ, Sandro. **Apostila de Microcontroladores PIC e Periféricos**. Disponível em <http://www.ebah.com.br/content/ABAAAAmX0AH/apostila-microcontroladores-pic-perifericos> Acesso em 08 set. 2012.

PEREIRA, Fábio. **Microcontroladores PIC, Programação em C.** 2ª Edição. Editora Érica Ltda., 2003.

PROCEL. CEPEL. ELETROBRÁS. **Consumo médio de aparelhos.** Disponível em <http://www.fiec.org.br/acoes/energia/informacoes/consumo_medio.htm> Acesso em 10 out. 2012.

SOUZA, David José de. LAVINIA, Nicolás César. **Conectando o PIC 16F877A**. 3ª Edição. Editora Érica Ltda., 2007.

SOUZA, David José de. **Desbravando o PIC.** 4ª Edição. Editora Érica Ltda., 1999.

THOMAZINI, Daniel. ALBUQUERQUE, Pedro Urbano Braga; **Sensores Industriais - Fundamentos e Aplicações.** 4ª Ed. São Paulo: Editora Érica Ltda., 2007.

APÊNDICE A - CÓDIGO FONTE DO DISPOSITIVO

```
/*=====================================================================
                PROJETO FINAL - Engenharia da Computação - UniCEUB
                2o. Semestre de 2012

                BRUNO QUEIROZ CUNHA
                RA: 2082917/9

                CONTROLE DA PERSIANA AUTOMATIZADA
=====================================================================*/

/*=====================================================================
                FOTO SENSOR
=====================================================================*/

#include <16f877A.h> //Inclui a biblioteca do PIC;
#device adc=10 //Define o conversor A/D na resolução de 10bits;
#use delay (clock=10000000) //Cristal oscilador de 10Mhz;
#fuses HS,NOWDT,PUT,BROWNOUT
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)

#define motor pin_D0 //Define motor na porta D0 do PIC;

void init_adc(); //Protótipo de função de inicialização do ADC;

static char Timer_Flag; //Variável global do tipo char;

void main()
{
unsigned long ldr1, ldr2, ldr3, ldr4 = 0;
float leitura_ldr1 , leitura_ldr2, leitura_ldr3, leitura_ldr4=0;
disable_interrupts(GLOBAL); //Desabilita todas as interrupções;
enable_interrupts(INT_RTCC); //Habilita interrupção do timer;
set_tris_D(pin_D0); //Configura RD0 como saída do pulso PWM p/ o servo;
setup_counters(RTCC_INTERNAL, RTCC_DIV_128); //Incrementa o timer a cada 128us.
Timer_Flag = 0; //Inicializa o flag de interrupção do timer;
enable_interrupts(GLOBAL);

output_b(0x00); //Limpa a porta B;
output_c(0x00); //Limpa a porta C;
output_d(0x00); //Limpa a porta D;
output_e(0x00); //Limpa a porta E;

/*=====================================================================
                LEITURA DA CONVERSÃO ADC
=====================================================================*/

init_adc(); //Configura o conversor A/D;

  do
```

```
  {
  set_adc_channel(0); //Configura o Canal AN0;
  delay_ms(10); //Delay de 10ms;
  ldr1 = read_adc(); //Lê valor analógico convertido p/ digital;
  leitura_ldr1 = (ldr1 * (5.0/1023.0)); //Converte 10 bits e o resultado em ponto flutuante;
  //printf("\f\nVal1 = %f2.0", leitura_ldr1);
  //delay_ms(500); //Delay de 500ms;

  set_adc_channel(1);
  delay_ms(10);
  ldr2 = read_adc();
  leitura_ldr2 = (ldr2 * (5.0/1023.0));
  //printf("\f\nVal2 = %f2.0", leitura_ldr2);
  //delay_ms(500); //Delay de 500ms;

  set_adc_channel(2);
  delay_ms(10);
  ldr3 = read_adc();
  leitura_ldr3 = (ldr3 * (5.0/1023.0));
  //printf("\f\nVal3 = %f", leitura_ldr3);
  //delay_ms(500); //Delay de 500ms;

  set_adc_channel(3);
  delay_ms(10);
  ldr4 = read_adc();
  leitura_ldr4 = (ldr4 * (5.0/1023.0));
  //printf("\f\nVal4 = %f", leitura_ldr4);
  //delay_ms(500); //Delay de 500ms;

/*=============================================================================
                                        LEITURA DOS LDRs
=============================================================================*/
  if((leitura_ldr1 < leitura_ldr2 && leitura_ldr1 < leitura_ldr3) && leitura_ldr1 < leitura_ldr4) //Compara o
LDR1 com os outros;
  {
  while(Timer_Flag == 0); //Aguarda até que o flag configure a interrupção do time;
  output_high(motor); //Coloca em nível alto o pino RC2;
  delay_us (2800); //Servo = 2.0ms
  output_low(motor); //Coloca em zero o pino RC2;
  //printf("\f\nValTESTE1 = %f", leitura_ldr1); //Mostra na tela qual o valor da leitura do LDR1;
  }
    else if((leitura_ldr2 < leitura_ldr1 && leitura_ldr2 < leitura_ldr3) && leitura_ldr2 < leitura_ldr4)
//Compara o LDR2 com os outros;
    {
    while(Timer_Flag == 0); //Aguarda até que o flag configure a interrupção do time;
    output_high(motor); //Coloca em nível alto o pino RC2;
    delay_us (2000); //Servo = 1.8ms
    output_low(motor); //Coloca em zero o pino RC2;
   // printf("\f\nValTESTE2 = %f", leitura_ldr2); //Mostra na tela qual o valor da leitura do LDR2;

    }
       else if((leitura_ldr3 < leitura_ldr1 && leitura_ldr3 < leitura_ldr2) && leitura_ldr3 < leitura_ldr4)
//Compara o LDR3 com os outros;
```

```
        {
        while(Timer_Flag == 0); //Aguarda até que o flag configure a interrupção do time;
        output_high(motor); //Coloca em nível alto o pino RC2;
        delay_us (1200); //Servo = 1.6ms
        output_low(motor); //Coloca em zero o pino RC2;
        //printf("\f\nValTESTE3 = %f", leitura_ldr3); //Mostra na tela qual o valor da leitura do LDR3;
        }
            else if((leitura_ldr4 < leitura_ldr1 && leitura_ldr4 < leitura_ldr3) && leitura_ldr4 < leitura_ldr2) //Compara o LDR4 com os outros;
            {
            while(Timer_Flag == 0); //Aguarda até que o flag configure a interrupção do time;
            output_high(motor); //Coloca em nível alto o pino RC2;
            delay_us (500); //Servo = 1.4ms;
            output_low(motor); //Coloca em zero o pino RC2;
            //printf("\f\nValTESTE4 = %f", leitura_ldr4); //Mostra na tela qual o valor da leitura do LDR4;
            }
  else
  {
  output_low(motor);
  }
  Timer_Flag = 0; //Reseta o flag de modo que o loop irá aguardar 32ms p/ a próxima interrupção;
  delay_ms(100);
  }while(TRUE);
} //Fim da função void main.

/*=====================================================================
                                    INICIALIZAÇÃO DO ADC
=====================================================================*/

void init_adc(void) // Configura O Canal 0 (RA0), Canal 1 (RA1), Canal 2 (RA2) e o Canal 3 (RA3);
{
setup_adc_ports(ALL_ANALOG); //Configura todos os pinos da porta A como analógicos;
//setup_adc_ports(RA0_RA1_RA2_RA3_ANALOG); //Configura as portas RA0, RA1, RA2 e RA3 como analógicos;
setup_adc(ADC_CLOCK_INTERNAL); //Clock A/D interno;
}

/*=====================================================================
                                    INTERRUPÇÃO DO TIMER
=====================================================================*/

#INT_RTCC
void TIMER_INTERRUPT(void) //fica aqui por 32ms;
{
Timer_Flag = 1;
}
```

ANEXO

A – Pinagem Microcontrolador

Nome do Pino	Núm. Pino	I/O/P	Tipo	Descrição
MCLR/Vpp	1	I/p	ST	Master Clear (reset) externo. 0 microcontrolador só funciona quando este pino encontra-se em nível alto. Entrada para tensão de programação (13V).
Vss	12/3	p	-	GND.
vdd	11/3	p	-	Alimentação positiva.
				PORTA (I/Os digitais bidirecionais e sistema analógico):
RA0/AN0	2	I/O	TTL	RA0: I/O digital ou entrada analógica ANO.
RA1/AN1	3	I/O	TTL	RA1 : I/O digital ou entrada analógica AN1 .
RA2/AN2/ V_{ref-}/CV_{ref}	4	I/O	TTL	RA2: I/O digital ou entrada analógica AN2 ou tensão negativa de referência analógica.
RA3/AN3/V_{REF+}	5	I/O	TTL	RA3: I/O digital ou entrada analógica AN3 ou tensão positiva de referência analógica.
RA4 / T0CKI / C1OUT	6	I/O	ST	RA4: I/O digital (quando saída é open drayn, isto é, não consegue impor nível alto) ou entrada externa do contador TMR0 ou saída do comparador 1 .
RA5/SS/AN4/ C2OUT	7	I/O	TTL	RA5: I/O digital ou entrada analógica AN4 ou habilitação externa (slave select) para comunicação SPI ou saída do comparador 2.
				PORTB (I/Os digitais bidirecionais). Todos os pinos deste PORT possuem pull-up interno que podem ser ligados/ desligados pelo software:
RB0/INT	33	I/O	TTL/ST(1)	RB0: I/O digital com interrupção externa.
RB1	34	I/O	TTL	RB1: I/O digital.
RB2	35	I/O	TTL	RB2: I/O digital.
RB3/PGM	36		TTL	RB3: I/O digital ou entrada para programação em baixa tensão (5V).
RB4	37		TTL	RB4: I/O digital com interrupção por mudança de estado.
RB5	38		TTL	RB5: I/O digital com interrupção por mudança de estado.
RB6/PGC	39		TTL/ST	RB6: I/O digital com interrupção por mudança de estado ou clock da programação serial ou pino de in-circuit debugger.
RB7/PGD	40	I/O	TT17ST(2)	RB7: I/O digital com interrupção por mudança de estado ou data da programação seriai ou pino de in-circuit debugger.
				PORTC (I/Os digitais bidirecionais):
RCO/T10SO/ T1CKI	15	I/O	ST	RC0:I/O digital ou saída do oscilador externo para TMR1 ou entrada de incremento para TMR1 .
RC1/T10SI/CCP2	16	I/O	ST	RC1: I/O digital ou entrada do oscilador externo para TMR1 ou entrada do Capture2 ou saídas para Compare2/PWM2.
RC2/CCP1	17	I/O	ST	RC2: I/O digital ou entrada do Capturei ou saídas para Compare1/PWM1.
RC3/SCK/SCL	18	I/O	ST	RC3: I/O digital ou entrada/saída de clock para comunicação serial SPI / I^2C.
RC4/SDI/SDA	23	I/O	ST	RC4: I/O digital ou entrada de dados para SPI ou via de dados (entrada/saída) para I^2C.

Nome do Pino	Núm. Pino	I/O/P	Tipo	Descrição
RC5/SDO	24	I/O	ST	RC5: I/O digital e saída de dados para SPI.
RC6/TX/CK	25	I/O	ST	RC6: I/O digital ou TX (transmissão) para comunicação USART assíncrona ou clock para comunicação síncrona.
RC7/RX/DT	26	I/O	ST	RC7: I/O digital ou RX (recepção) para comunicação USART assíncrona ou data para comunicação síncrona.
				PORTD (I/Os digitais bidirecionais) ou porta de comunicação paralela.
RD0/PSP0	19	I/O	TTL/ST(3)	RD0: I/O digital ou dado 0 (comunicação paralela).
RD1/PSP1	20	I/O	TTL/ST(3)	RD1: I/O digital ou dado 1 (comunicação paralela).
RD2/PSP2	21	I/O	TTL/ST(3)	RD2: I/O digital ou dado 2 (comunicação paralela).
RD3/PSP3	22	I/O	TTL/ST(3)	RD3: I/O digital ou dado 3 (comunicação paralela).
RD4/PSP4	27	I/O	TTL/ST(3)	RD4: I/O digital ou dado 4 (comunicação paralela).
RD5/PSP5	28	I/O	TTL/ST(3)	RD5: I/O digital ou dado 5 (comunicação paralela).
RD6/PSP6	29	I/O	TTL/ST(3)	RD6: I/O digital ou dado 6 (comunicação paralela).
RD7/PSP7	30	I/O	TTL/ST(3)	RD7: I/O digital ou dado 7 (comunicação paralela).
				PORTE (I/OS digitais bidirecionais e sistema analógico):
RE0/RD/AN5	8	I/O	TTL/ST(3)	RE0: I/O digital ou controle de leitura da porta paralela ou entrada analógica AN5.
RE1/WR/AN6	9	I/O	TTL/ST(3)	RE1: I/O digital ou controle de escrita da porta paralela ou entrada analógica AN6.
RE2/CS/AN7	10	I/O	TTL/ST(3)	RE2: I/O digital ou habilitação externa da porta paralela ou entrada analógica AN7.

Legenda:

I	=	Input (entrada)
O	=	Output(saída)
I/O	=	Input/Output (entrada ou saída)
P	=	Power (alimentação)
-	=	Não-utilizado
TTL	=	Entrada tipo TTL
ST	=	Estrada tipo *Schmitt Trigger*

Notas:

(1) Esta entrada é do tipo ST, somente quando configurado como interrupção externa.
(2) Esta entrada é do tipo ST, somente durante o modo de programação serial.
(3) Esta entrada é do tipo ST, quando configurado como I/O de uso geral e TTL quando.usado em modo de porta paralela.
(4) Esta entrada é ST quando em modo RC e CMOS nos demais casos.