DANIEL AKIRA NUMAJIRI

DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE DISPOSITIVOS VIA ACESSO REMOTO

Projeto de um protótipo apresentado ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do Curso de Ciência da Computação, para obtenção do título de bacharel

Orientador

Prof. Rêmulo Maia Alves

Lavras
Minas Gerais - Brasil
2003

DANIEL AKIRA NUMAJIRI

DESENVOLVIMENTO DE UM SISTEMA DE CONTROLE DE DISPOSITIVOS VIA ACESSO REMOTO

Projeto de um protótipo apresentado ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do Curso de Ciência da Computação, para obtenção do título de bacharel

Aprovada em 11 de Dezembro de 2003.

_____
Prof. Anderson Bernardo do Santos

_____
Prof. Luciano Mendes do Santos

_____
Prof. Rêmulo Maia Alves
(Orientador)

Lavras
Minas Gerais – Brasil

Agradecimentos

Agradeço aos meus pais, pela base pessoal ensinada e todo o incentivo que eles me proporcionaram, a minha irmã pela paciência e companheirismo e aos meus amigos pelo apoio, dicas e risadas trocadas nos momentos de dificuldade.

Aos professores do Curso de Bacharelado em Ciências da Computação pelos ensinamentos no decorrer do curso. E ao Luciano e Fabrício, muito obrigado.

Resumo

Desenvolvimento de um Sistema de Controle de Dispositivos via acesso remoto

Atualmente, com os avanços da tecnologia, coisas que nem imaginávamos há alguns anos atrás se tornam possíveis e viáveis de implementar, tal como o fato de podermos controlar dispositivos na nossa casa em qualquer parte do mundo através da internet. Essa monografia tem como objetivo mostrar o desenvolvimento de um sistema de monitoramento e controle de uma lâmpada e o estudo das formas de conexão e transmissão de dados na rede, os dispositivos de controle e comando gerados por controladores e atuadores. E a partir disso estabelecer um modelo base para o desenvolvimento futuro de outros sistemas de controle de dispositivos, como o monitoramento de estufas, granjas, controle de irrigação e dispositivos eletrônicos.

Abstract

Development of a System of Control of Devices saw access remote

Currently, with the advances of the technology, things that nor we imagined has some years behind become possible and viable to implement, such as the fact to be able to control devices in our house in any part of the world through the InterNet. This monograph has as objective to show to the development of a monitoramento system and control of a light bulb and the study of the connection forms and transmission of data in the net, the devices of control and command generated by controllers and actuators. E to leave of this to establish a model base for the future development of other systems of control of devices, as the monitoramento of greenhouses, farms, control of irrigation and electronic devices.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1

Introdução

Com o crescimento das novas tecnologias de comunicação e o aumento progressivo do número de eletro-eletrônicos utilizando essas tecnologias, torna-se interessante o incentivo à pesquisas nesta direção.

Um dos exemplos mais conhecidos do momento é a utilização de aparelhos celulares permitindo o acesso à Internet. Juntamente com outros dispositivos portáteis como Palms, Cassiopeias, etc, estes equipamentos visam permitir que usuários acessem serviços já existentes e, em sua maioria, a um custo maior e com uma menor qualidade.

Outros exemplos de sistemas que atuam nesta área, porém com outro enfoque, são os desenvolvedores de tecnologias para a interligação e comunicação entre eletrodomésticos, nas quais equipamentos como geladeiras são capazes de fazer o controle de qualidade e de estoque dos mantimentos nelas armazenados. Sistemas como este, acabam por aproveitar os recursos que já estão disponíveis como, por exemplo, compras on-line.

Outro fator importante que podemos destacar é a automação predial que já está presente em muitos lugares. Na atualidade não só os grandes edifícios corporativos de escritório e comércio requerem a introdução de sistemas de automação e controle em suas instalações, o objetivo de gerenciar a utilização de energia, o conforto dos ocupantes dentro dos ambientes, o acesso de pessoas, a comunicação, os sistemas prediais e de informática e a prevenção de acidentes e falhas de equipamentos, também são requeridos pelas residências. A implantação de um simples controle de acesso e detecção de intrusão ou o

monitoramento de acesso e o controle remoto de eletrodomésticos estão cada vez mais presentes no sistema de controle e gerenciamento das residências.

Partindo deste ponto de vista, as redes de computadores, que até algum tempo atrás, se ligavam apenas com outras redes de computadores (inclusive a Internet), agora estão conhecendo novos vizinhos e parceiros. Cada vez mais novas aplicações são idealizadas e projetos são colocados em desenvolvimento no sentido de atingir tais idéias, de modo que podemos prever desde agora como serão algumas das novidades futuras.

É procurando permanecer atualizado em relação a estas novas tecnologias, e buscando possuir o conhecimento e a experiência de trabalhar com elas, que foi proposto o desenvolvimento deste projeto.

O objetivo deste trabalho é desenvolver um sistema de controle de dispositivos simples acessível pela internet ou pela rede local a fim de torná-lo um modelo base para que, sempre que possível, abstrair as idéias básicas do processo possibilitando sua utilização em quaisquer outros tipos de sistemas, com pequenas adaptações. O objetivo secundário será tentar adaptar o projeto no sistema operacional Linux, pelo fato de ser um ambiente de código livre e possuir programas semelhantes ao do sistema operacional Windows, porém com custo zero.

A motivação principal para o desenvolvimento deste trabalho foi suprir a falta de um projeto que envolvesse os conhecimentos adquiridos ao longo do curso de graduação em Ciência da Computação da Universidade Federal de Lavras. Tomando por base a grade curricular do curso, procurou-se por um projeto que englobasse os conhecimentos de hardware e software que são ministrados.

Partindo-se disto, o tema escolhido inicialmente foi o desenvolvimento de uma interface para o computador de uma rede local. Sendo assim, esperamos

que o objetivo de se desenvolver um projeto envolvendo os conhecimentos de hardware possam ser alcançados.

Para alcançar os objetivos de software, o tema foi expandido para a criação de uma arquitetura de rede que desse suporte para as aplicações distribuídas poderem acessar a interface de controle.

Desta forma, para atender ambos os objetivos (os de software e de hardware), o trabalho deverá ser o desenvolvimento de uma arquitetura para possibilitar aplicações controlarem dispositivos eletrônicos não convencionais conectados a computadores. Além disso, estaria em estudo também, todo o processo de implantação e configuração de um servidor, aquisição de conhecimento sobre a linguagem de script CGI (Common Gateway Interface), linguagem de programação C e páginas web, para que o processo fosse executado pela internet utilizando um browser qualquer.

Como objetivos secundários deste trabalho destacamos o ganho de experiência no desenvolvimento e gerenciamento de projetos, o aprendizado e utilização de novas tecnologias e a possibilidade de desenvolvimento de possíveis produtos comerciais futuramente.

No capítulo 1, será apresentada uma breve introdução sobre o tema proposto, os objetivos e a organização do trabalho.

No capítulo 2, serão mostrados os fundamentos e os conceitos sobre a Internet, o protocolo *Transmission Control Protocol* (TCP) e o *Internet Protocol* (IP), a arquitetura de rede Ethernet, sistemas de controles, interfaces de comunicação, microcontroladores e relés para uma melhor compreensão da área e que serve de base para o desenvolvimento do trabalho.

No capítulo 3, será mostrado o tipo de metodologia que o projeto seguirá, informando o material utilizado e os passos de desenvolvimento do protótipo.

No capítulo 4 serão mostradas as dificuldades encontradas durante o desenvolvimento do protótipo, uma breve conclusão e trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Neste capítulo são apresentados os fundamentos e os conceitos das tecnologias que servem de base para o desenvolvimento do trabalho. Estas tecnologias são importantes para que sejam alcançados os objetivos do trabalho e o conseqüente entendimento destes objetivos.

2.1 A Internet – Um pouco da história

Grande parte da Internet resultou da "evolução" de um sistema criado em 1969. Nesta época, uma divisão do Departamento de Defesa dos EUA - a Agência de Projetos de Pesquisa Avançada em Defesa (DARPA) – concluiu que o país precisava criar um modo fácil de trocar informações militares entre cientistas e pesquisadores localizados em diferentes regiões geográficas.

O objetivo, segundo [Stang (1994)], era desenvolver um conjunto de protocolos de comunicação, os quais permitiriam computadores ligados em rede comunicarem-se de forma transparente entre várias redes diferentes. O sistema de protocolos que foi criado chamou-se TCP/IP.

Uma rede simples de quatro computadores, conhecida como DARPANET, foi desenvolvida. Algum tempo depois foi rebatizada de

ARPANET e, em 1972, cresceu a ponto de incluir 37 computadores e, ao mesmo tempo, o modo de utilização da rede começou a mudar. Além de ser empregado para trocar informações importantes, sobre atividades militares, os usuários da ARPANET, começaram a enviar mensagens eletrônicas por meio de caixas de correio pessoais.

Em 1990 a Internet comercial começou a funcionar. Desde então, o crescimento da internet tem sido simplesmente um fenômeno. O número de usuários saltou de 5.000 pessoas para cerca de 30 milhões em apenas dez anos, resultando em crescimento vertiginoso de 6.000%.

2.1.1 Organização

A estrutura da Internet é formada basicamente por três figuras principais: os *backbones*, os provedores de acesso e os usuários finais.

Backbones são grandes redes de computadores de alta velocidade, projetadas para a transmissão de um grande volume de dados. Pode-se dizer que os *backbones* são a espinha dorsal das redes de computadores. A Internet é formada por vários *backbones* espalhados pelo mundo, sendo que seu *backbone* principal encontra-se nos EUA.

Um provedor de acesso conecta-se diretamente aos *backbones* da Internet, formando conexões dedicadas e permanentes. O objetivo do provedor de acesso é fornecer aos usuários finais acesso à Internet e a outros serviços correlatos.

Os usuários finais podem ser particulares ou empresas que desejam obter acesso à Internet. Os usuários finais conectam-se aos provedores de acesso através de uma rede local ou da rede telefônica, via modem, e podem utilizar todos os recursos disponíveis na Internet.

A estrutura dos *backbones*, provedores de acesso ou usuários finais na Internet é formada por *hosts,* que são computadores que estão conectados a uma única rede física e têm o objetivo de executar programas aplicativos, e roteadores, que são computadores conectados a duas ou mais redes físicas e são responsáveis pela transmissão entre as redes.

2.1.2 Os Recursos da Internet

Mesmo possuindo recursos ilimitados, a cada dia surgem novas possibilidades e tecnologias de utilização da rede. Cita-se os principais:

a) **correio eletrônico:** baseado no protocolo de aplicação *Simple Mail Transfer Protocol* (SMTP), é um recurso que permite a troca de mensagens entre usuários. A troca das mensagens é baseada em endereços no formato usuário@domínio, onde "usuário" corresponde a um único usuário ou a um conjunto de usuários, e "domínio" corresponde ao nome do domínio de uma rede onde está localizada a caixa postal do usuário;

b) **telnet (*Telecomunications Network*):** trata-se de um emulador de sistemas que, através de um programa "cliente telnet", permite ao usuário conectar-se a outro *host* e, dependendo das suas permissões, executar aplicativos deste *host*, como se estivesse trabalhando nele localmente;

c) ***network news*:** baseado no protocolo de aplicação *Network News Transfer Protoco* (NNTP), fornece artigos referentes a vários assuntos, que são agrupadas em categorias denominadas *newsgroups*. Através de um programa de leitura de *news* o usuário pode conectar-se a um servidor de *network news* e ler os artigos disponíveis;

d) **transferência de arquivos:** baseado no protocolo de aplicação *File Transfer Protocol* (FTP) é um recurso especializado na transmissão de arquivos pela Internet. Através de um cliente de FTP o usuário pode conectar-se a um *host*, analisar o sistema de arquivos em que está conectado e escolher quais arquivos deseja pegar ou enviar;

e) **www (*World Wide Web*):** baseado no protocolo de aplicação *Hipertext Transfer Protocol* (HTTP) é um sistema de busca de informações em que a passagem de um documento para outro está embutida no próprio documento. A este mecanismo é dado o nome de navegação por hipertexto. A www permite que sejam incorporados aos documentos recursos de multimídia e imagens. O usuário utiliza um programa denominado *browser* para ler os documentos da www.

É no protocolo HTTP que focaremos parte do nosso processo de desenvolvimento do sistema de controle de dispositivos, criando uma interface de computador capaz de comunicar com um microcontrolador que será responsável pelo envio de dados de leitura e escrita do dispositivo(liga/desliga) que será criado, no caso, uma lâmpada.

Nesta interface será feito um estudo na parte de programação de páginas web, linguagem de script CGI e linguagem de programação C, que fará a comunicação entre o usuário e o dispositivo.

2.2 Ethernet e IEEE 802.3

A *Ethernet* é a tecnologia de rede local (LAN) mais amplamente usada. A *Ethernet* foi projetada para ocupar o espaço entre as redes de longa distância, com baixa velocidade e as redes especializadas de sala de computação que transportam dados em alta velocidade por distâncias muito limitadas. A *Ethernet*

é bem adequada a aplicativos em que um meio de comunicação local deva transportar tráfego esporádico, ocasionalmente intenso, a altas taxas de dados.

A arquitetura de rede *Ethernet,* segundo [Tanenbaum (1997)], tem suas origens nos anos 60, na Universidade do Havaí, onde o método de acesso que é usado pela *Ethernet*, *carrier sense multiple access/collision detection* (CSMA/CD), foi desenvolvido. O *Palo Alto Research Center* (PARC), da *Xerox Corporation*, desenvolveu o primeiro sistema *Ethernet* experimental no início dos anos 70. Isso foi usado como base para a especificação 802.3 do *Institute of Electrical and Electronic Engineers* (IEEE), lançada em 1980.

Logo após a especificação 802.3 de 1980 da IEEE, a *Digital Equipment Corporation*, a *Intel Corporation* e a *Xerox Corporation* desenvolveram conjuntamente e lançaram uma especificação *Ethernet*, versão 2.0, que foi substancialmente compatível com a IEEE 802.3. Juntas, a *Ethernet* e a IEEE 802.3 detêm atualmente a maior fatia de mercado de todos os protocolos LAN. Hoje, o termo *Ethernet* é freqüentemente usado para se referir a todas as LANs baseadas em CSMA/CD que normalmente estão em conformidade com as especificações *Ethernet*, incluindo a especificação IEEE 802.3.

A *Ethernet* e a IEEE 802.3 especificam tecnologias similares: ambas são LANs baseadas em CSMA/CD. Estações em uma LAN CSMA/CD podem acessar a rede a qualquer momento. Antes de enviar dados, as estações CSMA/CD "escutam" a rede para determinar se ela já está em uso. Se estiver, então elas aguardam. Se a rede não estiver em uso, as estações transmitem. Uma colisão ocorre quando duas ou mais estações "escutam" o tráfego da rede, não ouvem nada e transmitem simultaneamente. Neste caso, as transmissões são prejudicadas e as estações devem retransmitir mais tarde. Algoritmos de recuo determinam quando as estações que colidiram podem retransmitir. As estações CSMA/CD devem detectar colisões, assim, elas sabem quando devem parar a transmissão para retransmitir mais tarde.

[Tanenbaum (1997)] diz que ambas as LANs *Ethernet* e IEEE 802.3 são redes de *broadcast*. Isso significa que todas as estações de uma LAN podem ver todos os quadros, independentemente de serem ou não o destino daqueles dados. Cada estação deve examinar os quadros recebidos para determinar se ela é o destino. Se for, o quadro é passado a um protocolo de camada mais alto dentro da estação para processamento apropriado.

As diferenças entre as LANs *Ethernet* e IEEE 802.3 são sutis. A *Ethernet* fornece serviços correspondentes às camadas 1 e 2 do modelo de referência OSI, enquanto a IEEE 802.3 especifica a camada física, a camada 1, e a parte de acesso a canais da camada de enlace, a camada 2, mas não define um protocolo de controle de enlace lógico, o que é feito pelo IEEE 802.2 (*Logical Link Control* - LLC). Ambas as LANs *Ethernet* e IEEE 802.3 são implementadas através de hardware. Normalmente, a parte física desses protocolos é uma placa de interface em um *host* ou um conjunto de circuitos em uma placa de circuitos principal no *host*.

2.2.1 Árvore da família Ethernet

Segundo [Tanenbaum (1997)], existem pelo menos 18 variedades de *Ethernet* especificadas ou em processo de especificação. A Tabela 1 realça algumas das mais comuns e mais importantes tecnologias *Ethernet*.

Tabela 1: Tecnologias do padrão *Ethernet*

Tipo	Meio	Taxa Máxima	Comprimento máximo do segmento	Topologia física	Topologia lógica
10Base-T	UTP Cat 5	10Mbps	100m	Estrela	Barramento
10Base-FL	Fibra óptica	10Mbps	2000m	Estrela	Barramento
100Base-TX	UTP Cat 5	100Mbps	100m	Estrela	Barramento
100Base-FX	Fibra óptica	100Mbps	2000m	Estrela	Barramento

2.3 Introdução ao TCP/IP

O termo TCP/IP refere-se aos dois principais protocolos que o constituem: *Transmission Control Protocol* (TCP) e o *Internet Protocol* (IP). Existem inúmeros outros protocolos que permitem a comunicação entre computadores, sendo o TCP/IP o mais difundido atualmente.

O início se deu quando o Departamento Americano de Defesa, no final da década de 60, resolveu criar um *software* de comunicação entre computadores de forma que fosse possível a comunicação remota ou local, entre sistemas operacionais iguais ou diferentes utilizando ou não o mesmo tipo de hardware. O protocolo permitiu que, no começo da década de 70, os computadores pudessem comunicar-se remotamente, sendo de suma importância em navios de guerra ou entre porta-aviões.

Segundo [Starlin (1999)], para que isto fosse possível, foi necessário criar um sistema de comunicação onde a rede como um todo fosse dividida em "pequenas" redes independentes, passando a usar como padrão de interconexão o TCP/IP.

Como o Departamento Americano de Defesa sempre foi um dos maiores compradores de sistemas de informática, comunicou que somente compraria as tecnologias que tivessem incorporado o TCP/IP. Desde então, todos os sistemas operacionais e produtos trazem implementado o conjunto TCP/IP.

O TCP/IP possui como algumas de suas características básicas a independência tecnológica, interconexão universal, comunicação fim-a-fim, padrões de protocolos para as diversas aplicações. Uma das características mais importantes é a flexibilidade de adaptação às tecnologias de redes existentes e futuras, pois o TCP/IP foi concebido de forma independente das tecnologias de redes.

A comutação por pacote *packet-switching*, segundo [Redes (2001)], é a base de sua tecnologia de comunicação e utiliza o pacote *datagram* como unidade de transmissão de dados. Cada *host* conectado à rede possui um endereço único, possibilitando que a máquina emitente reconheça a máquina receptora.

2.3.1 Arquitetura TCP/IP

Segundo [Comer (1998)], a arquitetura TCP/IP, apresentada na Figura 1, trata de um conjunto de protocolos divididos em quatro camadas, sendo que cada uma destas camadas possui funções bem definidas.

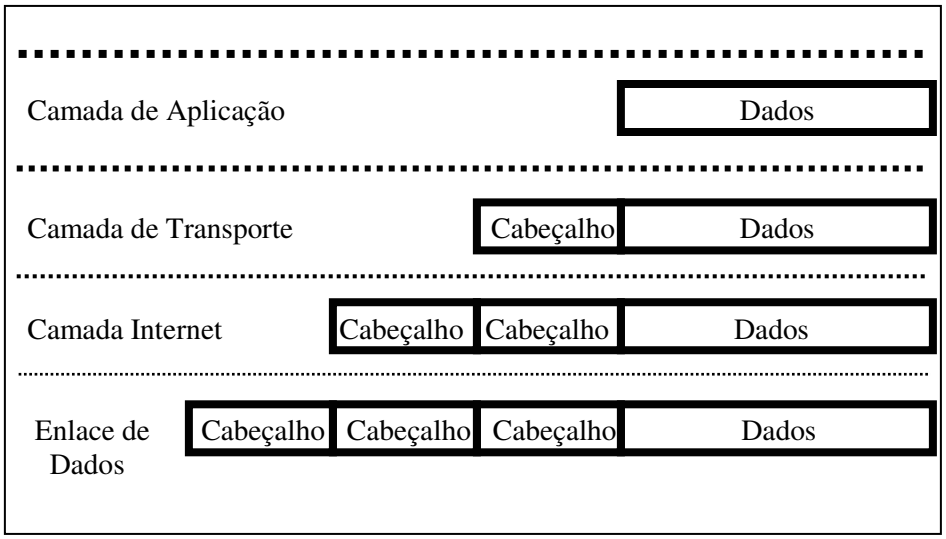


Figura 1: Camadas da arquitetura TCP/IP

Conforme a Figura 1, as camadas são:

a) camada de aplicação: segundo [Stevens (1994)] é nesta camada que estão os protocolos de alto nível como terminal virtual (TELNET), protocolo de transferência de arquivos (FTP), protocolo de envio de correio eletrônico (SMTP) entre outros. É nesta camada que estão os programas dos usuários.

b) camada de transporte: responsável por uma comunicação entre dois *hosts* fim a fim, podendo oferecer comunicações orientadas ou não orientadas à conexão. Fazem parte desta camada os protocolos *Transmission Control* Protocol (TCP) e *User Datagram Protocol* (UDP).

c) camada inter-redes ou internet: é onde está implementado o protocolo *Internet Protocol* (IP). Equivalente à camada de Rede no modelo OSI, é onde é feito o roteamento e a entrega dos pacotes IP.

d) camada de enlace: é responsável por encapsular os pacotes da camada inter-redes no formato específico da rede associada e extrair os pacotes

dos quadros vindos da rede e encaminhá-los à camada Inter-redes. O protocolo ARP encontra-se nessa camada.

2.3.2 O Protocolo IP

O *Internet Protocol* (IP) teve origem em 1970 no desenvolvimento da ARPANET, a qual foi depois interligada a outras redes formando em 1980 um vasto conjunto que passou a ser conhecido por Internet. Com a inclusão do protocolo IP no UNIX, em 1982, um grande número de universidades passou a formar as suas redes que por sua vez também foram ligadas à Internet.

O *Internet Protocol* (IP) é designado para o uso em sistemas de computadores interconectados (redes), utilizando troca de pacotes. O IP fornece transmissão de blocos de dados chamados datagramas entre origem e destino, sendo que origem e destino são *hosts* identificados por um endereço de tamanho fixo.

Outra função do IP, definido na [RFC 791 (2001)], é a fragmentação e desfragmentação de datagramas longos, se necessário, para transmissão através de redes que apresentam um desempenho melhor com pacotes menores.

O protocolo IP é especificamente limitado em escopo para fornecer as funções necessárias para entregar um pacote de bits (um datagrama internet) de uma origem para um destino em uma rede de computadores. Não há mecanismos para aumentar a confiança fim-afim dos dados, assegurar a seqüência dos datagramas, ou outros serviços comumente encontrados em outros protocolos *host-to-host*.

Segundo [Comer (1998)], o pacote pode ser perdido, reproduzido, atrasar-se ou ser entregue com problemas, mas o serviço não detectará tais condições, nem informará isso ao transmissor nem ao receptor. Também é considerado sem conexão porque cada pacote, segundo [Comer (1998)], é independente dos outros. Uma seqüência de pacotes enviados de um computador

a outro pode trafegar por caminhos diferentes, ou alguns podem ser perdidos enquanto outros são entregues.

O formato do datagrama IP está ilustrado na Figura 2:

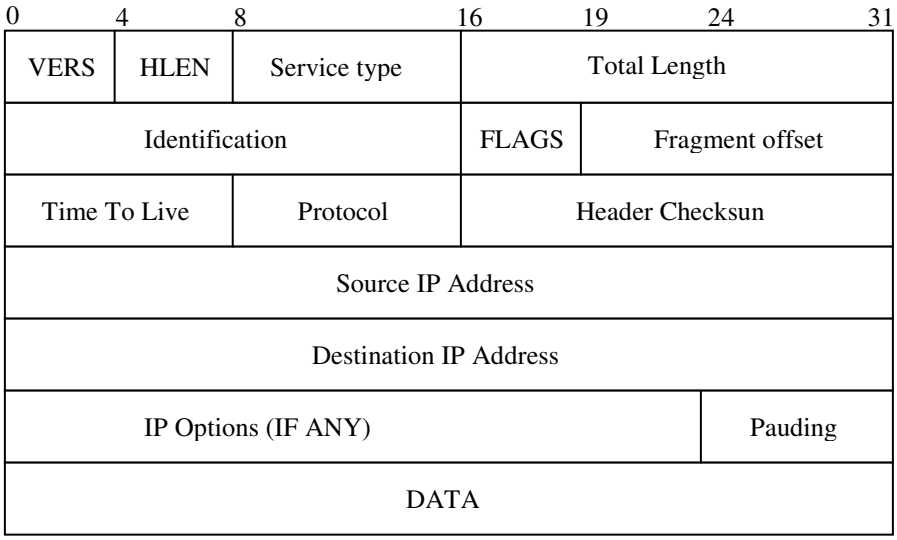


Figura 2: Formato do datagrama IPv4

2.3.3 O Protocolo TCP

O *Transfer Control Protocol* (TCP) é um protocolo de transporte orientado à conexão. Ele garante a confiabilidade da comunicação entre pares de processos localizados em máquinas ligadas ou não a uma mesma rede.

Possui como características principais o estabelecimento e a liberação de conexões em três fases (*handshake* triplo); a manipulação da entrega confiável dos pacotes; a entrega seqüencial dos pacotes (na ordem em que foram enviados); controle do fluxo de dados, impedindo que pacotes sejam processados em uma velocidade superior à capacidade do aplicativo; a recuperação de erros no caso de perda, alteração ou duplicação de pacotes; a demultiplexação do tráfego de entrada entre as múltiplas aplicações usuárias.

O protocolo TCP especifica o formato dos dados e das confirmações que dois computadores trocam para oferecer uma transferência confiável e, também, os procedimentos de que se valem os computadores para assegurar que os dados cheguem corretamente. Especifica ainda como o software TCP confirma os múltiplos destinos em determinada máquina e como as máquinas recuperam-se de erros como pacotes duplicados ou perdidos.

O TCP trata o fluxo de dados recebidos da aplicação como uma cadeia (*stream*) de bytes ou octetos, que vão sendo agrupados dentro do *buffer* de transmissão. Periodicamente o TCP separa um determinado conjunto de dados a adiciona um cabeçalho, formando o que é chamado de segmento, sendo este a unidade básica de transferência de dados.

O TCP permite que programas aplicativos múltiplos, de determinada máquina, comuniquem-se simultaneamente. Como o UDP, o TCP utiliza números de porta de protocolo para identificar o destino final em uma máquina. Os projetistas preferiram utilizar o mesmo número de portas para qualquer serviço que seja acessível pelo TCP e UDP, tornando-os assim independentes.
Os segmentos são trocados para estabelecer conexões, transferir dados, enviar confirmações, informar tamanho de janelas e encerrar conexões. O formato do segmento TCP pode ser visualizado na Figura 3.

Os protocolos TCP de ambas as pontas precisam concordar sobre o segmento máximo que irão transmitir, pois nem todas as redes têm o mesmo tamanho de segmento. Essa negociação é feita utilizando o campo opções do segmento, especificado pelo MSS (*maximum segment size*). O tamanho ideal é o que permite que os datagramas IP que transportam os segmentos sejam os maiores possíveis, sem exigir a fragmentação em qualquer ponto ao longo do caminho.

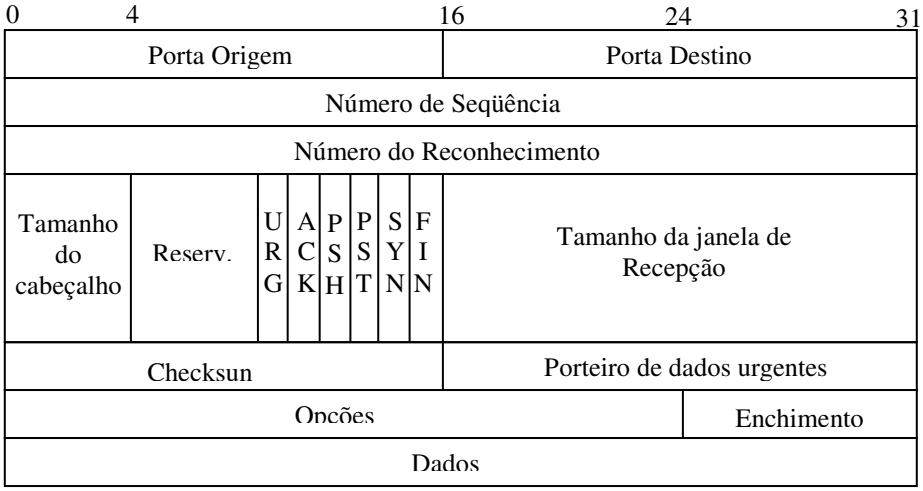


Figura 3: Formato do segmento TCP

Para que seja possível que as diversas aplicações que utilizam o protocolo TCP comuniquem-se entre si, é necessário que ocorra um estabelecimento de conexão. Neste momento às máquinas passam a se conhecer trocando diversas informações de controle e realizando uma verificação de autenticidade entre elas.

O TCP é um protocolo complexo que promove a comunicação através de uma grande variedade de tecnologias de rede. Segundo [Redes (2001)], a generalidade do TCP não parece interferir no seu desempenho. Informações mais detalhadas podem ser encontradas em [Comer (1998)].

2.4 Sistema de Controles

Segundo [Kuo (1995)], um sistema de controle são coisas que acontece em nossa vida diária, que são numerosos objetivos que precisam ser realizados. Por exemplo, no domínio doméstico, nós necessitamos regular a temperatura e umidade de residências e edifícios para uma vida confortável. Para o transporte, nós necessitamos controlar o automóvel e avião para ir exatamente e com

segurança de um ponto a outro. Industrialmente, os processos de produção contêm objetivos numerosos para os produtos que vão satisfazer às exigências da eficácia da precisão e de custo. Um ser humano é capaz de executar uma escala larga das tarefas, incluindo a tomada de decisão. Algumas destas tarefas tais como pegar um objeto e andar de um ponto a outro, são realizadas geralmente em uma forma rotineira.

Sob determinadas circunstâncias, algumas destas tarefas são executadas da melhor maneira possível. Por exemplo, um atleta que percorre um trecho de 100 jardas tem o objetivo de percorrer essa distância no menor tempo possível. Um corredor de maratona, em outro caso, não somente deve percorrer a distância no menor tempo possível, como também, ele ou ela deve controlar o consumo da energia e planejar a melhor estratégia para a corrida. Os meios de conseguir estes "objetivos" envolvem geralmente o uso de sistemas de controle que implementam certas estratégias de controle.

Baseado nos acontecimentos da nossa vida diária procurarei desenvolver um sistema de controle capaz de fornecer maneiras mais eficazes, rápidas e confortáveis de suprir nossas tarefas do dia-a-dia.

2.5 Interfaces de Comunicação

No computador existem duas interfaces de comunicação clássicas: a interface paralela e a interface serial. A alguns anos atrás, toda impressora era conectada à interface paralela e todo mouse era um dispositivo serial. Hoje, muitas inpressoras usam a interface USB e nos PCs mais novos, a porta padrão para o mouse é a PS/2. Contudo, as interfaces paralela e serial ainda existem em todos os computadores e podem ser muito úteis.

2.5.1 Interface Serial

O termo oficial promulgado pela IBM para a porta serial é porta de comunicações de dados assíncrona. Embora normalmente seja abreviado para porta assíncrona ou porta de comunicação. Além disso, como a grande maioria das ligações seriais aceita pela industria do PC opera sob um padrão chamado RS-232 (um nome dado por uma associação industrial, a Electronics Industry Association, ou EIA), a porta serial comum normalmente é descrita por sua especificação como porta RS-232.

Devido a simplicidade do hardware, a interface serial é muito utilizada dentro da indústria eletro-eletrônica. Hoje, o padrão de comunicação serial mais utilizado é o EIA/TIA – 232 ou, simplesmente, RS-232 (Recommended Standard). Com esse padrão, pode-se estabelecer uma comunicação bidirecional entre dois dispositivos usando apenas três fios. Daí vem a sua simplicidade e praticidade.

A comunicação RS-232 é muito utilizada em microcontroladores. A maioria deles tem um periférico exclusivo para comunicação RS-232. Por isso, a interface serial é uma maneira simples e prática de fazer o computador comunicar-se com o seu projeto. Por exemplo, você pode construir um pequeno robô e fazer com que a sua programação seja feita conectando-se o microcontrolador (que controla o robô) ao seu computador através de uma interface serial.

O problema de se utilizar comunicação serial entre o PC e um microcontrolador são os níveis de tensão exigidos pela interface serial. Geralmente, os microcontroladores trabalham com níveis de tensão entre 0 e 5V. Na interface serial do PC, os níveis envolvidos são -12 e 12V. Por isso, você deve utilizar drivers de tensão para não "torrar" o seu microcontrolador. O driver de interface serial que utilizei neste projeto foi o MAX232.

2.5.2 Porta Serial

A porta serial é o denominador comum das comunicações por computador. Até mesmo os PC's e periféricos mais primitivos aceitam uma porta serial.

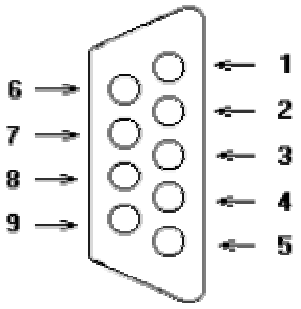
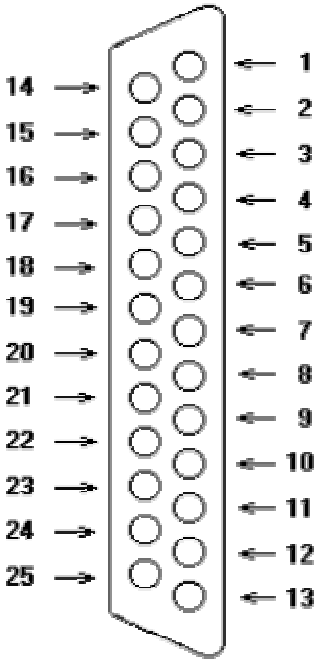
Não importa o nome usado, todas as portas seriais da IBM são iguais, pelo menos funcionalmente. Todas usam 8, 16 ou 32 bits paralelos que um computador apanha do seu bus (barramento) de dados e vira de lado – transformando uma tira de dados digitais em uma cadeia de pulsos que caminha bit a bit. Essa forma de comunicação recebe o nome serial porque os bits individuais de informação são transferidos em longas séries.

Muitos infortúnios podem cair sobre o vulnerável bit de dado serial à medida que percorre a conexão. Um dos bits de um byte inteiro de dados pode se desviar deixando um pedaço de dado com um valor menor na chegada do que na partida. Com a vaga no fluxo de dados, todos os outros bits pularão um lugar e assumirão novos valores. Ou então pode ocorrer o caso oposto, de erros no fluxo da comunicação empurrar todos os bits para trás. De qualquer forma, o prognóstico não é bom. Com essa forma elementar de comunicações seriais, um bit a mais ou a menos causará um erro em todos os bytes seguintes.

Estabelecer comunicações seriais confiáveis significa contornar esses problemas de erro de bit e muitos outros mais. Graças a alguma engenhosidade digital, no entanto, as comunicações seriais funcionam e funcionam bem – o bastante para que um PC dependa delas.

A tabela 3 apresenta a pinagem de conectores de 9 (DB-9) e 25 (DB-25) pinos de acordo com o padrão RS-232. A maioria dos computadores atuais utilizam o conector de 9 pinos.

Tabela 2: COM1/COM2 RS-232 PORTA SERIAL (DB-9 - DB25)

DB9 MACHO	DB 9 PINO	SINAL	DB 25 PINO	DB25 MACHO
	1	CD - Detecção da portadora	8	
	2	RXD - Recepção de dados	3	
	3	TXD - Transmissão de dados	2	
	4	DTR - Terminal de dados pronto	20	
	5	GND -Terra	7	
	6	DSR - Dado pronto	6	
	7	RTS - Permissão para envio	4	
	8	CTS - Permissão fornecida	5	
	9	RI - Indicador de chamadas	22	

No projeto da interface para conexão a um dispositivo a ser controlado via porta serial, bastam 3 pinos (3 fios): pino 5 – Terra, pino 3 – transmissão de dados, pino 2 – recepção de dados. A figura 5 mostra como ficaria o cabo serial com os devidos conectores para conectar um PC a um dispositivo externo.

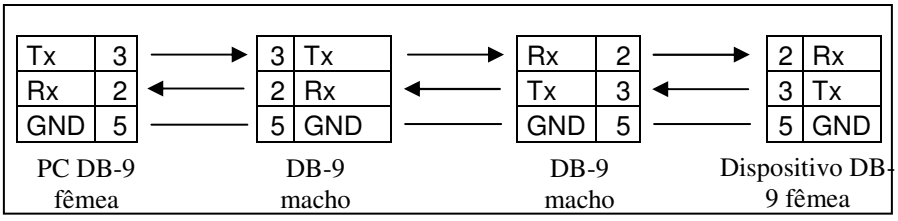


Figura 5: Interface serial entre um computador e um dispositivo externo

2.5.3 Comunicações Síncronas e Assíncronas

Dois dos principais métodos de comunicação serial são usados para evitar o desastre dos erros em bits seriais. Em um deles, os sistemas de envio e recepção são sincronizados por meio de algum tipo de sinal auxiliar, de modo que os dois lados da conexão estejam sempre alerta. Um clock, sincronizado entre a unidade de transmissão e recepção, temporiza com precisão o período que separa cada bit de dados. Um bit a mais ou a menos pode ser inesperada no fluxo de bits. Apenas olhando o relógio, você pode distinguir as comunicações por computador, um bit de dados real do ruído de interferência. Essa forma de transferência serial sincronizada pelo tempo chama-se comunicação síncrona, sendo uma técnica usada principalmente em sistemas de grande porte.

O sistema sincronizado falha sempre que os sistemas de envio e recepção perdem sua tranca de sinal mútuo. O fluxo de dados torna-se pouco mais do que um ruído.

A alternativa é incluir marcadores de lugar no fluxo de bits para ajudar acompanhar cada bit. Um marcador poderia, por exemplo, indicar a posição atribuída a um bit. Um bit ocorrendo sem o seu marcador poderia ser considerado como erro. Naturalmente, esse esquema simples teria muito desperdício, exigindo dois sinais digitais (o marcador e o bit de dados) para cada bit de informação transferido.

Um sistema intermediário funciona melhor. Em vez de indicar cada bit, o marcador poderia indicar o inicio de um pequeno fluxo de bits. A posição de cada bit no fluxo poderia ser definida sincronizando-os a intervalos regulares. Embora esse método seja semelhante a transferência síncrona, os sistemas transmissor e receptor não precisam ficar presos, exceto por pequenos intervalos entre os marcadores. A chegada de um marcador indica ao sistema receptor para começar a procurar os bits e rodar um temporizador curto. O problema de enviar e receber timers fora de sincronismo são eliminados reiniciando-se o clock a cada marcador, reduzindo o período entre os marcadores, não há tempo suficiente para o timer se perder.

Esse sistema de curto prazo temporizado normalmente é chamado comunicação assíncrona, pois os sistemas de envio e recepção não precisam estar sincronizados exatamente um com o outro. Os bits marcadores oferecem a tranca temporária necessária para distinguir um fluxo curto de bits de dados em seguida. A maioria das comunicações seriais no PC usa esse esquema. Na maior parte dos sistemas assíncronos, os dados são divididos em pequenos pedaços, cada um correspondendo aproximadamente a um byte. Cada um deles chama-se palavra, e pode conter de cinco a oito bits de dados. Os tamanhos de palavra mais usados são sete e oito bits, o primeiro porque inclui todos os caracteres de texto maiúsculos e minúsculos em código ASCII; o segundo porque cada palavra corresponde exatamente a um byte de dados.

Como dados seriais, os bits de uma palavra são enviados um de cada vez por um canal de comunicação. Por convenção, o bit menos significativo da palavra é enviado primeiro. O restante dos bits segue em ordem de grandeza crescente.

Junto a esses bits de dados há um pulso muito especial de tamanho duplo chamado bit de início; ele indica o início de uma palavra de dados. Um ou mais bits de fim indicam o final da palavra. Entre o último bit da palavra e o

primeiro bit de fim, um bit de paridade costuma ser incluído para a verificação da integridade dos dados. Junto, os bits de dados, o bit de início, o bit de paridade e os bits de fim compõem um quadro de dados.

2.5.4 Bits de Paridade

Cinco tipos de bits de paridade podem ser usados na comunicação serial, dois deles oferecendo um meio de detectar erros de transmissão ao nível de bit. Essa detecção de erros funciona contando o número de bits na palavra de dados e determinando se o resultado é par ou ímpar. Na paridade ímpar, o bit de paridade é ativado (passado para o nível lógico um) quando o número de bits da palavra é impar. A paridade par ativa o bit de paridade quando o total de bits de uma palavra é par.

Na paridade marca, o bit de paridade é sempre ativado não importando o total de bits da palavra. A paridade espaço sempre deixa o bit de paridade desativado. Nenhuma paridade significa que o quadro não tem espaço para um bit de paridade. Embora emitindo um bit de integridade de dados (que pode ser fornecido por outros meios), isso permite comunicações mais eficientes, comprimindo mais informações em um determinado número de bits transmitidos.

2.5.5 Polaridade de Sinal

Todos os bits de um sinal serial no padrão RS-232 são enviados pela linha de comunicação como pulsos passando para negativo superpostos a uma voltagem positiva normal mantida na linha de dados. Ou seja, a presença de um bit na palavra serial interromperá uma voltagem positiva contínua com um breve pulso negativo.

Comparado com os sistemas lógicos normais, os dados no padrão RS-232 parecem estar de cabeça para baixo. Não há um motivo particularmente bom para essa inversão, exceto por ser o modo como as coisas têm sido feitas e, no caso das comunicações, as coisas funcionam melhor quando todos usam o mesmo padrão.

2.5.6 Velocidade de Envio dos Dados

Uma outra característica importante de todo sinal serial é a velocidade em que os bits no trem de dados serial são nominalmente enviados. O formato padrão dessa medida é muito simples: o número de bits enviados por segundo, com a unidade padrão sendo um bit por segundo, ou bps.

Por questões um tanto arbitrário, as velocidades de bit são enumeradas em incrementos pouco usuais. A velocidade mínima comum é 300 bps, embora esteja disponível submúltiplo mais lento: 50 100 e 150 bps. As velocidades mais rápidas simplesmente duplicam as velocidades anteriores, passando para 600, 1200, 4800, 9600 e 19200, a velocidade mais rápida aceita oficialmente pela IBM com portas seriais comuns controladas por microprocessador, encontradas nos modelos 50 a 80 do PS/2.

Essas velocidades oficiais lentas são importantes porque o controle da porta serial por software impõe uma carga tão grande sobre o microprocessador do sistema que os chips mais lentos não podem trabalhar com velocidades maiores. Como as maiores velocidades não podem ser aceitas por todo o software, a IBM escolheu não sancionar tais velocidades em seus computadores mais antigos.

2.6 Micro-controladores – Família PIC

O micro-controlador é definido como sendo um sistema computacional integrado, pois ele possui unidade de processamento, memória e entradas e saídas integrados em um único chip.

A maioria das pessoas entende o sistema computacional como sendo o nosso computador pessoal. No entanto todo sistema que a partir de dados de entrada, executa algum processamento mediante um programa armazenado em uma memória gerando uma saída é chamado de sistema computacional.

Neste projeto, utilizei um microcontrolador da família PIC, portanto darei uma visão geral das características e funções deste microcontrolador. Um dos aspectos mais interessantes do PIC é que, mesmo sendo a maior família de micro-controladores do mercado em termos de variedade de modelo, todos os modelos foram desenvolvidos a partir da mesma filosofia de produto.

Esta característica da família PIC permite a compatibilidade de códigos (linguagem assembler) e a estruturação das aplicações, pois um código escrito para um modelo de PIC poderá ser migrado para outro modelo equivalente em termos de recursos necessários sem que sejam necessárias grandes mudanças no código fonte. Isto facilita o trabalho de quem desenvolve e preserva o investimento de quem produz.

Um sistema computacional é composto por uma unidade de processamento, memória e portas de entrada/saída (I/O).

A maioria das pessoas limita os conceitos de sistema computacional ao computador que temos em casa, vulgo PC (Personal Computer). No entanto todo sistema que a partir de dados de entrada, executa algum processamento mediante um programa armazenado em uma memória gerando uma saída é chamado de sistema computacional. Podemos definir micro-controlador como sendo um sistema computacional integrado, pois ele possui unidade de

processamento, memória e entradas e saídas integrados em um único chip. Alguns ainda possuem periféricos como conversores A/D e D/A, comparadores.

Os microcontroladores chegam a custar muitas vezes mais barato do que um transistor. Existe uma quantidade grande de μC no mercado, veja alguns nomes a baixo:

• Família 8051 – fabricante Intel
• PIC – fabricante Microchip
• AVR – fabricante Atmel
• BASIC Stamp – fabricante Parallax
• BASIC Step – fabricante Tato Equipamentos

2.6.1 Arquitetura de Microcontroladores

A arquitetura de um sistema digital define quem são e como as partes que compõe o sistema estão interligadas. As duas arquiteturas mais comuns para sistemas computacionais digitais são as seguintes:

- Arquitetura de Von Neuman: A Unidade Central de Processamento é interligada à memória por um único barramento. O sistema é composto por uma única memória onde são armazenados dados e instruções.
- Arquitetura de Harvard: A Unidade Central de Processamento é interligada a memória de dados e a memória de programa por barramento específico.

2.6.1.1 Memória de Programação

A memória de programação é onde as instruções do programa são armazenadas. No caso do 16F628 esta memória é de 2048 palavras (words) de

14 bits cada uma. Partes destes 14 bits informam o OPCODE (código da instrução) e o restante traz consigo o argumento da instrução correspondente. Na familia PIC existem três tipos de memória de programa: EPROM (O.T.P. – One Time Programable), EEPROM (janelado) e FLASH. Existem duas posições da memória do programa que recebem nomes especiais: vetor de reset e vetor de interrupção.

O vetor de reset é para onde o programa vai quando ele é inicializado, enquanto que o vetor de interrupção é a posição da memória de programa para onde o processamento é desviado quando ocorre uma interrupção.

2.6.1.2 Memória de Dados

A memória de dados é uma memória volátil do tipo R.A.M. (random access memory). O mapa de memória é divido em duas partes: registradores especiais (special function register - S.F.R.) e registradores de uso geral (general purpose register - G.P.R.). Como o ponteiro da memória de programa tem capacidade de endereçar somente 128 posições de memória de cada vez (7 bits), a memória de programa é dividida em bancos (banco 0 e banco 1 no 16F84).

Esta divisão implica em termos posições de memória que somente poderão ser acessadas caso o banco a que ela pertença seja previamente selecionado através de um bit específico do S.F.R. STATUS.

2.6.1.3 Unidade Lógica Aritmética

A unidade lógica aritmética é onde todas as operações lógicas (funções lógicas booleanas): e ou, (exclusivo e complemento) e aritméticas (soma e subtração) são efetuadas. O registrador W sempre estará envolvido de alguma forma em toda operação lógica ou aritmética. Existem dois destinos possíveis

para estas operações: o W (work) ou um registrador (posição da memória de dados) definido no argumento da instrução.

2.6.1.4 Ciclo de Máquina

Um micro-controlador pode ser entendido como sendo uma máquina que executa operações em ciclos. Todos os sinais necessários para a busca ou execução de uma determinada instrução devem ser gerados dentro de um período de tempo denominado Ciclo de Máquina. Nos PIC com memória de programa de 12 e 14 bits um Ciclo de Máquina corresponde a quatro períodos de clock (1:4) denominados Q1,Q2,Q3 e Q4, conforme pode ser verificado na figura 5.

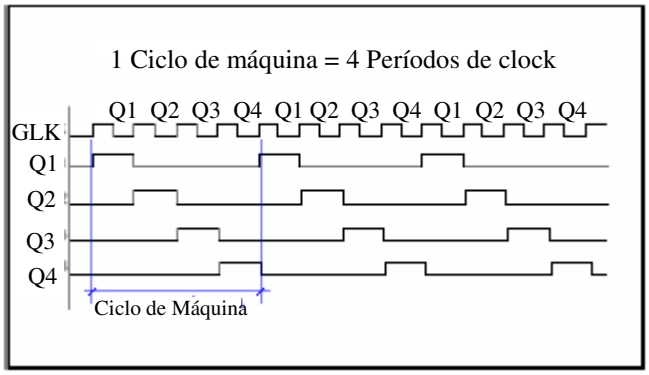


Figura 5: Ciclo de Máquina

2.6.1.5 Pipeline

Se para efeito de análise dividirmos o processamento interno do PIC em ciclos de busca e execução, podemos afirmar que para cada instrução executada foi necessária a execução prévia de um ciclo de busca. Imagine um sistema que implemente um ciclo de busca e ao mesmo tempo processe um ciclo de

execução. Desta forma, no início de cada Ciclo de Máquina haverá uma instrução pronta para ser executada.

No entanto algumas instruções fazem com que este sistema seja desarticulado: são as chamadas instruções de desvio. As instruções de desvio são aquela que alteram o valor do Program Counter (contador/ponteiro de programa).

Quando ocorre um desvio a instrução que já foi previamente buscada pelo sistema de Pipeline não é válida, pois estava na posição de memória de programa apontada pelo PC antes dele ter seu valor alterado para o destino especificado. Conseqüentemente torna-se necessário a execução de um novo ciclo de busca, que obviamente demandará mais um ciclo de máquina, resultando em um tempo de total de processamento igual a dois ciclos de máquinas. A figura 6 mostra a atuação do pipeline.

Label	Instrução	Pipeline
BT 1	MOVLW .100	ADDWF COUNT,F
	ADDWF COUNT,W	MOVWF COUNT,W
	MOVWF RESULT	GOTO MOSTRA
	GOTO MOSTRA	MOVLW .50 (DESCARTADO) MOVF MOSTRA,W (NOVA BUSCA)
BT 2	MOVLW .50	
	(...)	
	(...)	
	(...)	
MOSTRA	MOVF MOSTRA,W	MOVWF PORTB
	MOVWF PORTB	(...)

Figura 6: Atuação do Pipeline

Toda instrução do PIC demanda um Ciclo de Máquina para ser executada, exceto aqueles que provocam desvio no programa que demanda dois Ciclos de Máquina.

2.6.1.6 Timer

O PIC possui internamente um recurso de hardware denominado Timer0. Trata-se de um contador de 8bits incrementado internamente pelo ciclo de máquina ou por um sinal externo (borda de subida ou descida), sendo esta opção feita por software durante a programação (SFR). Como o contador possui 8 bits ele pode assumir 256 valores distintos (0 até 255). Caso o ciclo de máquina seja de 1us, cada incremento do Timer corresponderá a um intervalo de 1us. Caso seja necessário intervalos de tempos maiores para o mesmo Ciclo de Máquina, utilizaremos o recurso de PRE-SCALE.

2.6.1.7 Pre-scale

O Pre Scale é um divisor de freqüência programável do sinal que incrementa o Timer0. Quando temos um pre scale de 1:1, cada ciclo de máquina corresponde a um incremento do Timer0 (unidade de Timer0). Ao alterarmos o pre scale para, por exemplo, 1:4 (os valores possíveis são as potencias de dois até 256), o Timer0 será incrementado uma vez a cada quatro ciclos de máquina.

2.6.1.8 Watchdog

O watchdog é um recurso disponível no PIC que parte do princípio que todo sistema é passível de falha. Se todo sistema pode falhar, cabe ao mesmo ter recursos para que, em ocorrendo uma falha, algo seja feito de modo a tornar o sistema novamente operacional.

Dentro do PIC existe um contador incrementado por um sinal de relógio (clock) independente. Toda vez que este contador extrapola o seu valor máximo retornando a zero, é provocado a reinicialização do sistema (reset).

- Clear Watchdog: Se o sistema estiver funcionado da maneira correta, de tempos em tempos uma instrução denominada clear watchdog timer (CLRWDT) zera o valor deste contador, impedindo que o mesmo chegue ao valor máximo. Desta maneira o Watchdog somente irá "estourar" quando algo de errado ocorrer.
- Pre Scale: O período normal de estouro do Watchdog Timer é de aproximadamente 18 ms. No entanto, algumas vezes este tempo é insuficiente para que o programa seja normalmente executado. A saída neste caso é alocar o recurso de Pre Scale de modo a aumentar este período. Se sem o pre scale o período é de 18ms, quando atribuímos ao Watchdog Timer um PRE SCALE de 1:2 (um para dois) nos dobramos este período de modo que o processamento possa ser executado sem que seja feita uma reinicialização.

2.6.1.9 Interrupção

As Interrupções são causadas através de eventos assíncronos (podem ocorrer a qualquer momento) que causam um desvio no processamento. Este desvio tem como destino o vetor de interrupção.

Uma boa analogia para melhor entendermos o conceito de interrupção é a seguinte: você está trabalhando digitando uma carta no computador quando o seu ramal toca. Neste momento você, interrompe o que está fazendo, para atender ao telefone e verificar o que a pessoa do outro lado da linha está precisando. Terminada a conversa, você coloca o telefone no gancho novamente e retoma o seu trabalho do ponto onde havia parado. Observe que não precisamos verificar a todo instante, se existe ou não alguém na linha, pois quando o ramal é chamado, o telefone toca avisando que existe alguém querendo falar com você.

A habilitação das interrupções nos PIC segue a seguinte filosofia. Existe uma chave geral (general interrupt enable) e chaves específicas para cada uma das interrupções. Deste modo, se eu quiser habilitar a interrupção de tempo (TMR0) eu devo "setar" o bit da chave geral e também o bit da chave específica (T0IE), ambos presentes no registrador especial (S.F.R.) INTCON.

- POWER ON RESET: é um sistema faz com que durante a energização o pino de Master Clear (/MCLR) permaneça durante algum tempo em zero, garantindo a inicialização.
- POWER UP TIMER: é um temporizador que faz com que o PIC, durante a energização (power up), aguarde alguns ciclos de máquina para garantir que todo o sistema periférico (display, teclado, memórias, etc) estejam operantes quando o processamento estiver sendo executado.
- BROWN OUT: O Brown Out monitora a diferença de tensão entre VDD e VSS, provocando a reinicialização do PIC (reset) quando esta cai para um valor inferior ao mínimo definido em manual.
- SLEEP: O modo de operação Sleep foi incluído na família PIC para atender um mercado cada vez maior de produtos que devem funcionar com pilhas ou baterias. Estes equipamentos devem ter um consumo mínimo para que a autonomia seja a máxima.

Quando o PIC é colocado em modo Sleep (dormir), através da instrução SLEEP, o consumo passa da ordem de grandeza de mA (mili àmperes) para uA (micro àmperes). Existem três maneiras de "acordar o PIC": por interrupção externa/estado, estouro de Watchdog ou reinicialização (/MCRL).

2.6.2 Microcontrolador PIC

Os microcontroladores da família PIC são dispositivos RISC (Reduced Instruction Set Computer – Computador com conjunto reduzido de instruções) com arquitetura HARVARD (Barramentos de memória de programa e de memória de dados diferentes) e fluxo de instruções PIPELINE (Enquanto se executa uma determinada instrução, a próxima já está sendo lida) de última geração, seu uso é praticamente ilimitado e seu preço é baixo, levando-se em conta as suas excelentes características. Os PICs são muito versáteis, podem possuir de 6 até 66 pinos de I/O, trabalhar em frequências de até 40MHz, eles podem ter ainda:

- ADCs = Conversores analógico para digital;
- DACs = Conversores digital para analógico;
- Endereçadores de memória externa;
- I/Os seriais;
- Memória EEPROM de dados, entre outros.

2.6.3 PIC16F628

O PIC16F628 possui arquitetura Harvard. A memória de dados é do tipo RAM (volátil) e a memória de programa são do tipo Flash (letra F no código).

Este PIC possui 18 pinos e pode ser classificado na faixa superior em termos de recursos disponíveis, pois ele possui interrupções e conversores AD ou portas de comunicação serial. O PIC16F628 melhorou as características do núcleo, a pilha de oito níveis, e fontes internas e externas múltiplas da interrupção. As barras de comunicação separadas da instrução e de dados da arquitetura de Harvard permitem que uma palavra de 14-bit de larga instrução use dados de 8-bit separados. A faixa de dois estágios da instrução permite que

toda a instrução seja executada em um único-ciclo, à exceção das filiais do programa (que requerem dois ciclos).

O microcontrolador consegue tipicamente uma compressão de código de 2:1 e uma melhoria da velocidade de 4:1 sobre outros microcontroladores 8-bit em sua classe. Os dispositivos do PIC16F628 têm as características especiais para reduzir componentes externos, assim reduzindo o custo do sistema, realçando a confiabilidade do sistema e reduzindo o consumo de potência.

Há oito configurações do oscilador, apenas o oscilador do pino ER fornece uma solução de baixo custo. O oscilador LP minimiza o consumo de potência, XT é um cristal padrão, INTRC é um oscilador interno e o HS é para cristais de alta velocidade.

A modalidade do SLEEP (baixo consumo) oferece economias de energia. O usuário pode ativar a micro plaqueta do SLEEP de diversas interrupções e restauração externas e internas.

A tecnologia FLASH faz a customização dos programas de aplicação (níveis da detecção, geração do pulso, temporizadores, etc.) extremamente rápido e conveniente. Os pacotes pequenos da entrada fazem esta série do microcontrolador ideal para todas as aplicações com limitações do espaço. Baixo custo, baixo consumo, alta performance, facilidade de utilização e flexibilidade de I/O faz do PIC16F628 muito versátil. A figura 7 mostra a pinagem do PIC16F628.

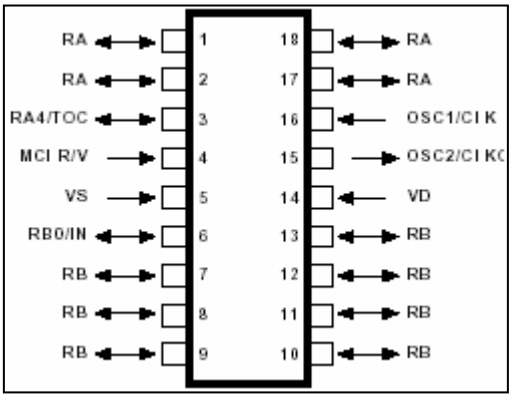


Figura 7: Pinagem do PIC16F628

Características técnicas:

- Freqüência de operação DC a 20 MHz
- Memória do programa 3.5K
- Numero de instruções 2048
- RAM 224 bytes
- EEProm 128 bytes
- PORT A: RA0 ... RA4 (5 PINOS)
- PORT B: RB0 ... RB7 (8 PINOS)
- 1 Módulo CCP
- 2 Comparadores
- Presença de Power On Reset
- Presença de Brown-out Detect
- Temporizador 1x16bit 2x8bit WDT W/RC
- Presença da Tensão de Referência Vref
- Capacidade de corrente 25mA
- Presença do Programa de Circuito Serial

2.7 Relés

Os relés são dispositivos comutadores eletromecânicos. A estrutura simplificada de um relé é mostrada na figura 8 e a partir dela explicaremos o seu princípio de funcionamento.

Nas proximidades de um eletro-imã é instalado uma armadura móvel que tem por finalidade abrir ou fechar um jogo de contatos. Quando a bobina é percorrida por uma corrente elétrica é criado um campo magnético que atua sobre a armadura, atraindo-a. Nesta atração ocorre um movimento que ativa os contatos, os quais podem ser abertos, fechados ou comutados, dependendo de sua posição, conforme mostra a figura 9.

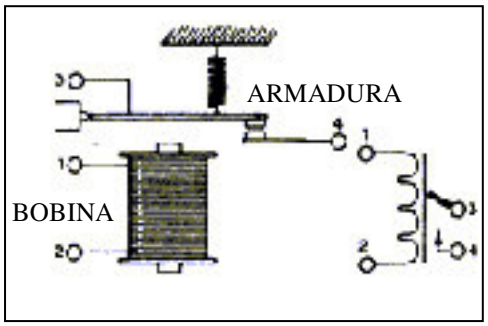


Figura 8: Estrutura elétrica do Relé

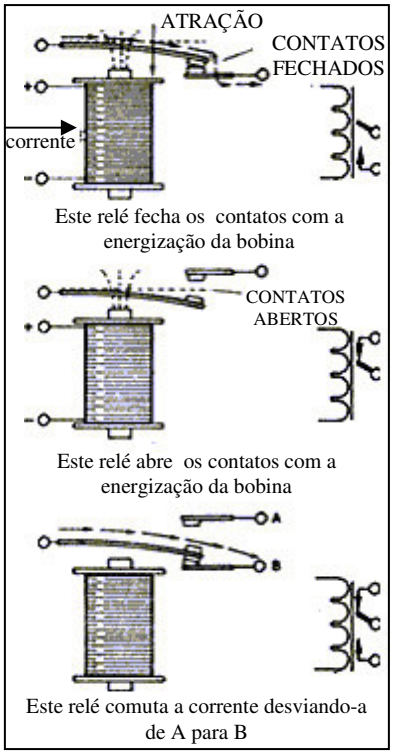


Figura 9: Abertura e fechamento dos contatos

Isso significa que, através de uma corrente de controle aplicada à bobina de um relé, podemos abrir, fechar ou comutar os contatos de uma determinada forma, controlando assim as correntes que circulam por circuitos externos. Quando a corrente deixa de circular pela bobina do relé o campo magnético criado desaparece, e com isso a armadura volta a sua posição inicial pela ação da mola.

Os relés se dizem “energizados” quando estão sendo percorridos por uma corrente em sua bobina capaz de ativar seus contatos, e se dizem “desenergizados” quando não há corrente circulando por sua bobina.

A aplicação mais imediata de um relé com contato simples é no controle de um circuito externo ligando ou desligando-o, conforme mostra a figura 10. Observe o símbolo usado para representar este componente.

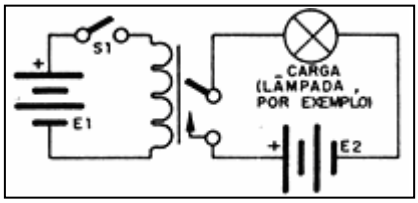


Figura 10: Relé de contato simples

Quando a chave S1 for ligada, a corrente do gerador E1 pode circular pela bobina do relé, energizando-o. Com isso, os contatos do relé fecham, permitindo que a corrente do gerador E2 circule pela carga, ou seja, o circuito controlado que pode ser uma lâmpada.

Para desligar a carga basta interromper a corrente que circula pela bobina do relé, abrindo para isso S1.

Uma das características do relé é que ele pode ser energizado com correntes muito pequenas em relação à corrente que o circuito controlado exige para funcionar. Isso significa a possibilidade de controlarmos circuitos de altas correntes como motores, lâmpadas e máquinas industriais, diretamente a partir de dispositivos eletrônicos fracos como transistores, circuitos integrados, fotoresistores etc.

A corrente fornecida diretamente por um transistor de pequena potência da ordem de 0,1A não conseguiria controlar uma máquina industrial, um motor ou uma lâmpada, mas pode ativar um relé e através dele controlar a carga de alta potência. (figura 11).

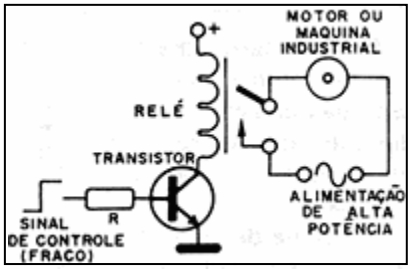


Figura 11: Ativar uma carga de alta potência

Outra característica importante dos relés é a segurança dada pelo isolamento do circuito de controle em relação ao circuito que está sendo controlado. Não existe contato elétrico entre o circuito da bobina e os circuitos dos contatos do relé, o que significa que não há passagem de qualquer corrente do circuito que ativa o relé para o circuito que ele controla.

Se o circuito controlado for de alta tensão, por exemplo, este isolamento pode ser importante em termos de segurança.

Do mesmo modo, podemos controlar circuitos de características completamente diferentes usando relés: um relé, cuja bobina seja energizada com apenas 6 ou 12V, pode perfeitamente controlar circuitos de tensões mais altas como 110V ou 220V.

O relé que tomamos como exemplo para analisar o funcionamento possui uma bobina e um único contato que abre ou fecha.

Na prática, entretanto, os relés podem ter diversos tipos de construção, muitos contatos e apresentar características próprias sendo indicados para aplicações bem determinadas.

2.7.1 Os Relés na Prática

O que determina a utilização de um relé numa aplicação prática são suas características. O entendimento dessas características é fundamental para a escolha do tipo ideal.

A bobina de um relé é enrolada com um fio esmaltado cuja espessura e números de voltas são determinados pelas condições em que se deseja fazer sua energização.

A intensidade do campo magnético produzido e, portanto, a força com que a armadura é atraída depende tanto da intensidade da corrente que circula pela bobina como do número de voltas que ela contém.

Por outro lado, a espessura do fio e a quantidade de voltas determinam o comprimento do enrolamento, o qual é função tanto da corrente como da tensão que deve ser aplicada ao relé para sua energização, o que no fundo é a resistência do componente. Todos estes fatores entrelaçados determinam o modo como a bobina de cada tipo de relé é enrolada.

De um modo geral podemos dizer que nos tipos sensíveis, que operam com baixas correntes, são enrolados milhares ou mesmo dezenas de milhares de voltas de fios esmaltados extremamente finos, alguns até mesmo mais finos que um fio de cabelo! (figura 12).

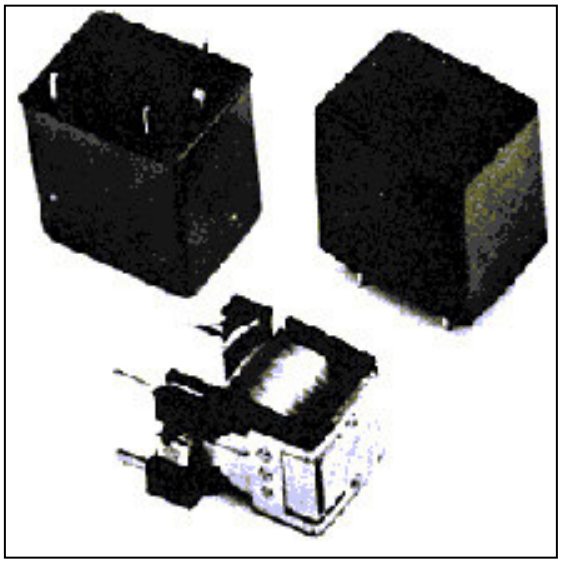


Figura 12: A bobina de um Relé

As armaduras dos relés devem ser construídas com materiais que possam ser atraídos pelos campos magnéticos gerados, ou seja, devem ser de materiais ferromagnéticos e montadas sobre um sistema de articulação que permita sua movimentação fácil, e retorno à posição inicial quando o campo desaparece.

Peças flexíveis de metal, molas ou articulações são alguns dos recursos que são usados na montagem das armaduras.

A corrente máxima que os relés podem controlar depende da maneira como são construídos os contatos. Além disso, existe o problema do faiscamento que ocorre durante a abertura e fechamento dos contatos de relé, principalmente no controle de determinado tipo de carga (indutivas).

O material usado deve então ser resistente, apresentar boa capacidade de condução de corrente e, além disso, ter um formato próprio, dependendo da aplicação a que se destina o relé.

Dentre os materiais usados para a fabricação dos contatos podemos citar o cobre, a prata e o tungstênio. A prata evita a ação de queima provocada pelas faíscas, enquanto os contatos de tungstênio evitam a oxidação.

O número de contatos e sua disposição vão depender das aplicações a que se destinam os relés.

2.7.2 Tipos de Relés

2.7.2.1 Contatos NA ou Normalmente Abertos

Os relés são dotados de contatos do tipo normalmente abertos, quando estes permanecem desligados até o momento em que o relé seja energizado. Quando o relé é energizado, os contatos fecham, e com isso pode circular corrente pelo circuito externo. Podemos ter relés com um ou mais contatos do tipo NA, conforme mostra a figura 13.

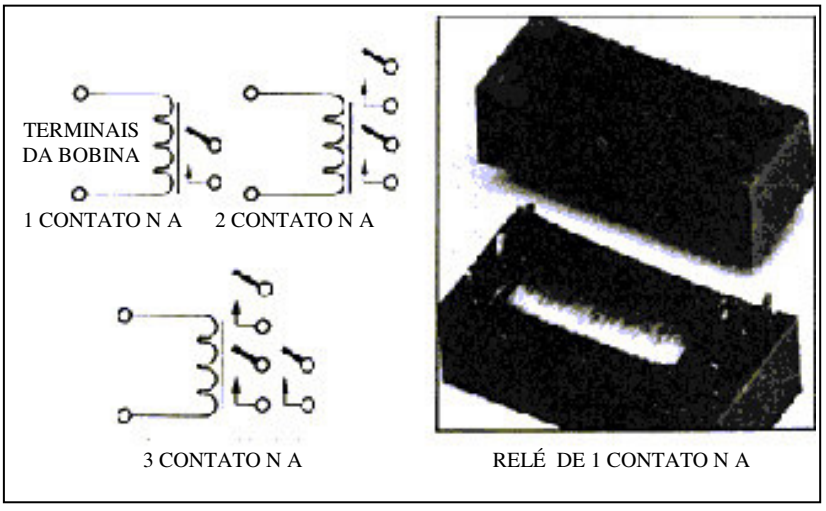


Figura 13: Relés tipo NA

Usamos relés com contatos do tipo NA quando queremos ligar uma carga externa ao fazer uma corrente percorrer a bobina do relé, ou seja, quando o energizarmos.

2.7.2.2 Contatos NF ou Normalmente Fechados

Estes relés apresentam um ou mais contatos que estão fechados, permitindo a circulação pela carga externa, quando a bobina estiver desenergizada. Quando a bobina é percorrida por uma corrente, o relé abre seus contatos, interrompendo a circulação de corrente pela carga externa. (figura 14).

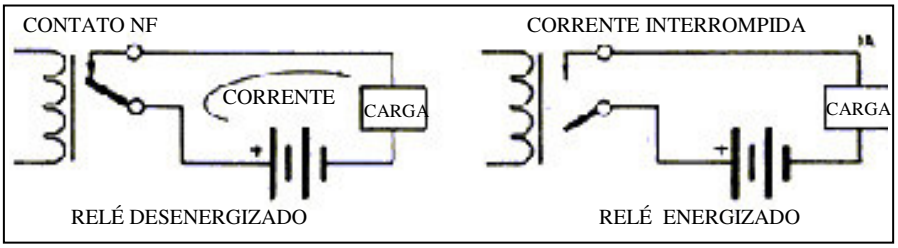


Figura 14: Relés tipo NF

Usamos este tipo de relé para desligar uma carga externa ao fazer uma corrente percorrer a bobina do relé.

2.7.2.3 Contatos NA e NF ou Reversíveis

Os relés podem também ter contatos que permitem a utilização simultânea dos contatos NA e NF ou de modo reversível, conforme mostra a figura 15.

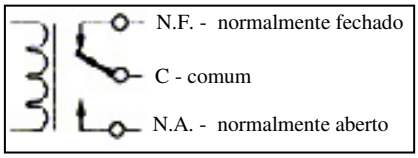


Figura 15: Relé Reversível

Quando o relé está com a bobina desenergizada, o contato móvel C faz conexão com o contato fixo NF, mantendo fechado este circuito. Energizando a bobina do relé o contato C (comum) passa a encostar-se ao contato NA, fechando então o circuito. Podemos usar este tipo de relé para comutar duas cargas, conforme sugere a figura 16.

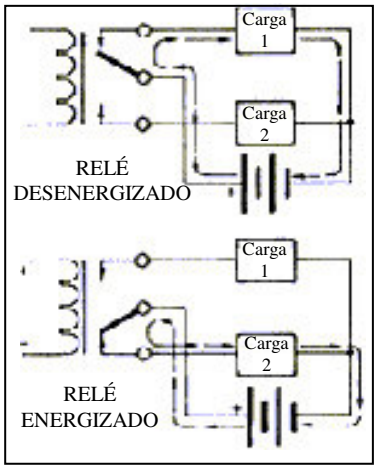


Figura 16: Comutação de duas cargas através do relé reversível

A energia da fonte E passa então do circuito de carga 1 para o circuito de carga 2.

O número de contatos NA e NF de um relé pode variar bastante, o que garante uma enorme versatilidade para este componente. Assim, jogando com os dois contatos reversíveis, podemos fazer inversões do sentido de circulação da corrente.

Os relés podem ainda ter bobinas para operar tanto com corrente contínua como com corrente alternada.

No caso de corrente contínua, a constância do campo garante um fechamento firme, sem problemas. No entanto, no caso do acionamento por corrente alternada, a inversão do sentido da corrente numa determinada freqüência faz com que o campo magnético apareça e desapareça dezena de vezes por segundo, o que leva a armadura e os contatos a uma tendência de vibração.

Para evitar este problema técnicas especiais de construção são usadas, sendo que a mais eficiente consiste na colocação numa das metades do núcleo da bobina de um anel de cobre. Neste anel é então induzida uma forte corrente que cria um segundo campo magnético, o qual divide o campo principal em dois fluxos defasados. Assim, não existe um instante em que o campo seja nulo, quando a armadura pode "descolar", e com isso causar as vibrações.

Por este motivo, os relés usados em corrente contínua não são os mesmos empregados em circuitos de corrente alternada.

2.7.2.4 Relés abertos, fechados e selados

Dependendo das aplicações, temos ainda para os relés montagens diferentes do conjunto de peças que o formam. Os relés podem ser abertos, ou seja, sem proteção, se forem usados em equipamentos fechados, que não estejam sujeitos a poeira, umidade ou outros elementos que prejudiquem o componente.

Temos também relés fechados, mas sem vedação alguma que são utilizados na maioria das aplicações comuns. Estes relés possuem coberturas de materiais diversos, como por exemplo, o plástico que pode ser opaco ou transparente.

Existem ainda os relés herméticos que são encerrados em invólucros que impedem a penetração de ar do meio ambiente.

Em especial estes relés são empregados em aplicações que ficam em atmosferas combustíveis, já que o acionamento dos contatos pode ser acompanhado de faíscas que causariam a ignição do combustível e com isso o perigo de explosão. Na figura 17 há três tipos da linha de produtos relés com as mais diversas especificações adicionais.

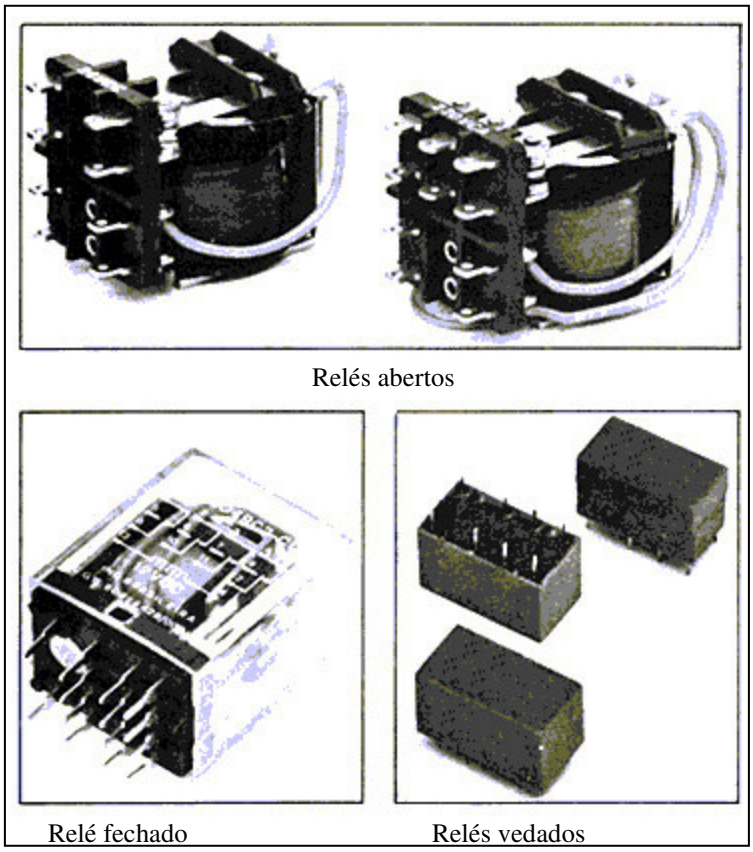


Figura 17: Relés abertos, fechados e selados

Esta proteção evita que a poeira se acumule principalmente nos contatos, vindo a prejudicar o funcionamento do relé.

2.7.3 Ligação dos relés ao circuito externo

Outro fato importante na construção de um relé é a maneira como ele vai ser ligado ao circuito externo. Para esta finalidade, os relés são dotados de terminais. O tipo mais simples possui, então, 4 terminais sendo 2 para a conexão à bobina e 2 para os próprios contatos. (figura 18).

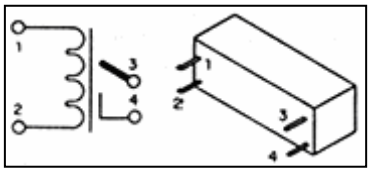


Figura 18: Relés de 4 terminais

O número de terminais aumentará na proporção em que aumenta o número de contatos e estes podem ter as mais diversas aparências. Em aplicações profissionais, onde a eventual substituição rápida de um relé deve ser feita com presteza, são usados encaixes em bases fixas. São os relés de encaixe ou plug-in. Temos ainda relés que comutam sinais de altas freqüências, e que utilizam conectores para os contatos do tipo coaxial. Este tipo de configuração é necessário para que não ocorram perdas na transferência das correntes que o relé deve comutar em seus contatos.

2.7.4 Características elétricas dos Relés

Como acionar um relé? Quais circuitos externos podem ser controlados por um relé? Na utilização de qualquer tipo de relé num projeto é fundamental ter respostas para as duas perguntas acima, e em alguns casos para outras. Nos manuais de fabricantes de relés encontramos informações que permitem a avaliação do que um relé pode fazer e como deve ser usado. No entanto, é

preciso saber interpretar estas informações, para que não aconteçam surpresas desagradáveis num projeto. Iniciaremos então nossas explicações pelas características elétricas dos relés.

2.7.4.1 Características da bobina

Para que o relé seja energizado corretamente e os contatos atuem, é preciso que uma corrente de intensidade mínima determinada circule pela sua bobina. Devemos então aplicar uma tensão de determinado valor, que em função da resistência do enrolamento vai permitir que a corrente mínima determinada seja estabelecida. Na prática os relés são especificados em termos da corrente que deve passar pelo enrolamento para uma determinada tensão que é a tensão de funcionamento. Na verdade é preciso levar em conta que, para fechar o relé, precisamos de uma certa intensidade de campo magnético que puxe a armadura para perto da bobina com certa força, mas uma vez que a armadura se aproxima, o campo já não precisa ser tão forte para mantê-la junto à bobina, e com isso o relé fechado. Devemos então distinguir a tensão que aciona o relé da tensão que o mantém fechado que é muito menor.

A corrente que aciona o relé é denominada corrente de acionamento, enquanto que a corrente que o mantém fechado (muito menor) é a corrente de manutenção. Fixando a tensão que deve disparar um relé de corrente contínua, a corrente que vai circular por sua bobina é função da resistência do enrolamento, o que pode ser calculado facilmente pela lei de Ohm. Assim, se um relé for especificado para uma tensão nominal de 24 volts, quando então circula uma corrente de 20 mA (0,02 A), podemos calcular a resistência com uma simples divisão:

$R = V/I = 24/0,02 = 1200$ ohms

As características da bobina do relé de corrente contínua (resistência, corrente e tensão) ficam então perfeitamente definidas quando temos duas das três grandezas acima citadas: Se tivermos a tensão (V) e a corrente (I), calculamos a resistência (R) pela fórmula:

$R = V/I$

Se tivermos a tensão (V) e a resistência (R), calculamos a corrente pela fórmula:

$I = V/R$

Finalmente, se tivermos a corrente (I) e a resistência (R), calculamos a tensão (V) pela fórmula:

$V = R \times I$

Veja que estas tensões são "valores nominais", ou seja, aqueles que são recomendados numa operação normal. Na prática o relé pode fechar seus contatos com tensões menores, mas este fator deve, ser levado em conta quando se desejar máxima confiabilidade do componente. Os valores superiores também são admitidos, apenas até certo limite. Se a aplicação de uma tensão num circuito que tenha uma certa resistência, como a bobina de um relé, significa a produção de calor, temos aí um motivo claro da limitação. As bobinas podem dissipar apenas uma quantidade definida de calor, que não deve ser superada. Os fabricantes de relés indicam então qual é a porcentagem acima da tensão nominal que pode ser aplicada no máximo na bobina de um relé sem o perigo de haver aquecimento. Valores típicos estão entre 10 e 15% acima da tensão nominal. Resumindo: as características elétricas da bobina de um relé, que devem ser levadas em conta num projeto, são:

- Tensão nominal, tensão de operação e tensão máxima de trabalho
- Corrente nominal
- Resistência ôhmica
- Potência nominal dissipada

2.7.4.2 Características dos contatos

Além do número de contatos e o tipo, devemos também conhecer características elétricas desses contatos, para utilizá-los sem problemas em qualquer projeto. A primeira característica que nos interessa é a corrente máxima que podem controlar. A abertura e fechamento dos contatos de um relé exigem um certo tempo, o que significa que nos pontos de aproximação máxima podem ocorrer arcos, ou seja, pequenas faíscas que tendem a queimá-los com o tempo. Estas faíscas são mais intensas quando se comuta um circuito indutivo como, por exemplo, um transformador, um motor, um solenóide etc.

A superfície dos contatos determina, por outro lado, a intensidade máxima da corrente que pode ser controlada. Estes dois fatores devem ser levados em conta na utilização de um relé. Assim, temos a especificação da corrente máxima que cada contato pode controlar tanto em circuitos resistivos como indutivos. Evidentemente, a corrente máxima num circuito resistivo é sempre maior que a permitida para um circuito indutivo. Alguns recursos permitem a proteção dos contatos com o prolongamento de sua vida útil, na comutação e controle de cargas indutivas "amortecendo" as faíscas, mas isso será visto posteriormente.

A vida útil de um relé está basicamente determinada pela durabilidade dos contatos, e como o desgaste ocorre nos momentos em que ocorrem as comutações, esta característica é dada em termos de abertura e fechamento do relé em milhares ou mesmo milhões de vezes. Temos ainda como especificação importante a tensão máxima que os circuitos do contato podem admitir. Esta característica é importante levando-se em conta a possibilidade de ocorrer faiscamentos ou mesmo fugas entre os contatos dado o seu afastamento na posição em aberto, se a tensão máxima for superada.Valores típicos estão na

faixa dos 150 aos 250V.Como a potência controlada no circuito de carga é dada pelo produto da corrente pela tensão, em alguns casos especifica-se a potência máxima também.

Existem casos em que não se recomenda que a corrente máxima especificada para os contatos seja aplicada também com a tensão máxima. Limita-se assim a potência.

Uma outra especificação importante em certas aplicações é o tempo que o relé demora para fechar seus contatos. Existe então um intervalo de tempo mínimo indicado pelo fabricante que decorre entre a aplicação da tensão na bobina e o pleno fechamento dos contatos. Este valor varia de tipo para tipo e é dado tipicamente em milisegundos (ms). Veja então que os dois tempos devem ser levados em conta quando se deseja que o relé opere em ciclos rápidos.

Do mesmo modo, existe um tempo determinado para o desaparecimento do campo magnético na bobina a partir do instante em que a corrente é interrompida. As linhas de forças do campo magnético se contraem em velocidade limitada pela indutância da bobina, e isso influi diretamente no tempo em que os contatos demoram para abrir (figura 19). Os fabricantes especificam também o tempo de abertura do relé em milisegundos.

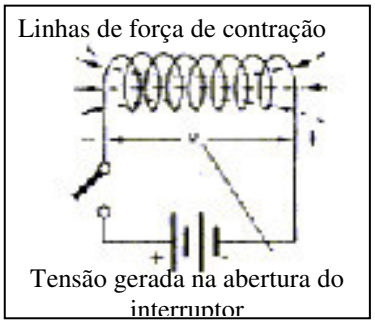


Figura 19: Condução de corrente na Bobina

Estes tempos determinam a máxima freqüência que o relé pode responder. É claro que não se recomenda a utilização deste tipo de componente em aplicações que exijam a repetição de muitos ciclos de operação rapidamente, pois existe uma limitação para a vida útil dos contatos. Esta vida útil é indicada em termos de quantidade de operações, ficando tipicamente entre 250 mil e 30 milhões, conforme a corrente controlada. Finalmente devemos levar em conta a resistência dos contatos que pode ser expressa de diversas formas.Uma das maneiras consiste em se indicar a resistência de contato inicial, que é a resistência de um contato que ainda não comutou carga e, portanto, ainda não sofreu desgaste pelo faiscamento. Esta resistência é expressa em milésimos de ohm (mohms) situando-se tipicamente entre 10 e 100. Além destas especificações todas existem outras que eventualmente podem ser necessárias nas aplicações mais críticas. Dentre elas podemos citar o isolamento entre a bobina e os contatos, a capacitância entre os contatos quando eles estão abertos, já que nestas condições podemos considerá-los como as placas de um capacitor. Temos ainda o peso do componente, a vibração, a rigidez dielétrica entre bobina e contatos e entre os contatos etc.

2.7.5 Como usar um Relé

Alguns pequenos cuidados no projeto de circuitos com relês podem ser importantes, tanto no sentido de se obter maior durabilidade para o componente, como de proteger os próprios componentes do circuito de acionamento. Analisemos os principais casos:

2.7.5.1 Proteção do circuito de acionamento

No momento em que um relé é desenergizado, as linhas de força do campo magnético da bobina, que se encontram em seu estado de expansão máxima, começam a se contrair. Nesta contração, as espiras da bobina do próprio relé são cortadas, havendo então a indução de uma tensão. Esta tensão tem polaridade oposta àquela que criou o campo e pode atingir valores muito altos.

O valor desta tensão depende da velocidade de contração do campo (di/dt) e da indutância da bobina (L). Se o componente que faz o acionamento do relé não estiver dimensionado para suportar esta tensão, se não houver uma proteção adequada, sua queima será inevitável (figura 20).

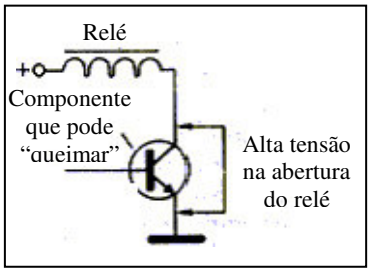


Figura 20: Proteção do Relé

Do mesmo modo, existe um tempo determinado para o desaparecimento do campo magnético na bobina a partir do instante em que a corrente é interrompida. As linhas de forças do campo magnético se contraem em velocidade limitada pela indutância da bobina, e isso influi diretamente no tempo em que os contatos demoram para abrir (figura 19). Os fabricantes especificam também o tempo de abertura do relé em milisegundos.

Diversas são as técnicas empregadas para eliminar este problema, sendo a mais conhecida a que faz uso de um diodo, conforme mostra a figura 21.

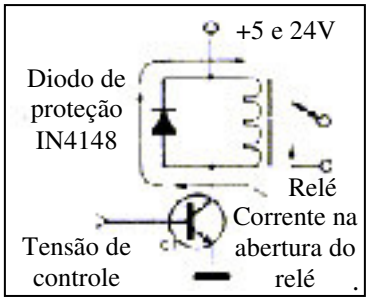


Figura 21: Uso de um diodo para proteger o relé

O que ocorre neste caso é que o diodo está polarizado inversamente em relação a tensão que dispara o relé. Assim, quando ocorre a indução de uma alta tensão nos extremos da bobina no momento da interrupção da corrente, o diodo polarizado no sentido direto passa a ter uma baixa resistência absorvendo assim a energia que, de outra forma, poderia afetar o componente de disparo.

Outra técnica, menos comum dado o custo do componente, é a que faz uso de um varistor ligado em paralelo com a bobina do relé, conforme mostra a figura 18.

O varistor ou VDR é um componente, normalmente de óxido de zinco que apresenta uma característica não linear de corrente versus tensão, conforme mostra a curva da mesma figura. Quando a tensão supera certo valor a resistência do componente cai abruptamente.

Esta propriedade pode ser usada para absorver a corrente no instante em que o relé é desenergizado e que poderia causar problemas aos componentes de disparo.

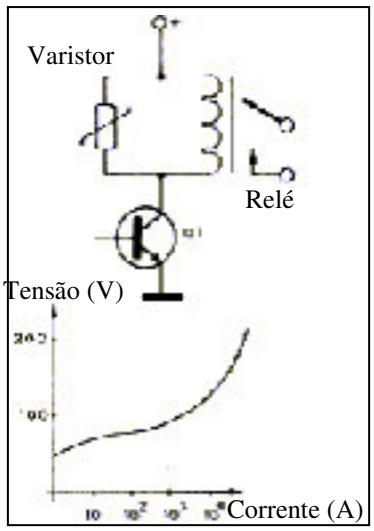


Figura 22: Uso de um varistor para proteger o relé

A tensão do VDR ou Varistor deve ser escolhida de tal modo a ser maior que a tensão de disparo do relé, porém menor que a tensão máxima suportada pelo elemento usado no disparo.

A utilização de um capacitor + resistor em paralelo com a bobina é também um meio de proteção, mas que nem sempre é recomendado, dada a velocidade com que ocorre a comutação.

2.7.5.2 Proteção dos Contatos

Além da observação das limitações de corrente e tensão que devem aparecer nos contatos de um relé, existem alguns cuidados adicionais que podem prolongar sua vida e, com isso, a vida do próprio relé.

Na comutação de cargas indutivas é conveniente agregar-se ao circuito elementos de proteção contra faiscamento.

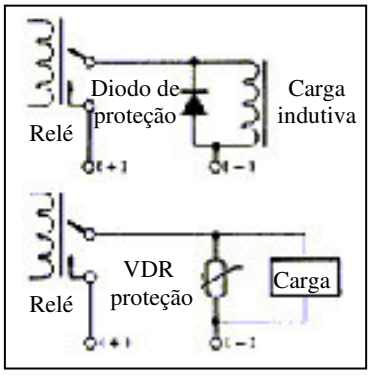


Figura 23: Uso do diodo em paralelo para evitar altas tensões

Na figura 23 temos um diodo usado em paralelo com a carga indutiva de modo que seja evitado o aparecimento de altas tensões nos contatos na sua abertura. Estas elevadas tensões poderiam causar faiscamento excessivo e com isso a queima dos contatos.

Outro recurso consiste no emprego do varistor e até mesmo de capacitores e resistores. Os capacitores e resistores são indicados para os circuitos de corrente alternada, onde o diodo não pode ser empregado.

Capítulo 3

Metodologia

3.1 Tipo de Pesquisa

Nesta monografia serão mostrados um estudo e implementação de um protótipo, desde os conceitos até a implantação de todos os recursos utilizados durante o processo de desenvolvimento de software e hardware do sistema de controle de dispositivos via web.

Este projeto foi desenvolvido nos Laboratórios da Computação, do Departamento de Ciência da Computação e na maioria das vezes em casa com auxílio do computador pessoal. O projeto iniciou efetivamente no mês de Março de 2003 e foi concluído aproximadamente no mês de Dezembro de 2003.

O projeto foi dividido nas seguintes fases:

- Escolha do dispositivo a ser utilizado
- Estudo do funcionamento do dispositivo
- Criação de uma interface de comunicação entre o dispositivo e o computador
- Desenvolvimento de softwares de controle da interface
- Desenvolvimento de um servidor de comunicação remota
- Modelagem da arquitetura final do sistema
- Desenvolvimento de sistemas web
- Desenvolvimento do protótipo
- Desenvolvimento de aplicações distribuídas sobre a arquitetura criada

Durante as fases citadas, os conceitos básicos necessários envolvem **Circuitos Eletrônicos**, onde os conceitos de componentes eletrônicos são amplamente

utilizados nas fases de desenvolvimento de microcontroladores e da interface de comunicação com o computador; **Lógica Digital**, que dá todo o suporte para a interface a partir do momento que começamos a manipular dados digitais; **Arquitetura de Computadores**, assim como a lógica digital, provê o suporte de manipulação de endereços dos dispositivos na arquitetura x86; **Programação em Linguagem C**, que permite a comunicação de softwares com o hardware sem limitações através da porta paralela, possibilitando uma maior flexibilidade na manipulação dos dispositivos; **Programação de Computadores**, que é a base para as fases de software, onde os conceitos de orientação a objeto serão amplamente explorados durante a implementação dos softwares da arquitetura proposta; **Sistemas Distribuídos**, que provê a base para a computação distribuída, os conceitos dos sistemas Cliente/Servidor e Comunicação em Grupo; **Linguagens Comerciais**, que dará suporte aos sistemas baseados na web, bem como o ambiente de programação em C para PIC, da CCS, que é um ambiente muito eficiente e proporciona uma rapidez na execução de projetos e o software Delphi da Borland; **Sistemas Operacionais**, visto que em alguma parte do projeto haverá a necessidade de aplicações utilizarem recursos de hardware, além da possibilidade de implementação deste projeto não somente em ambientes Microsoft, mas também em ambientes Unix; **Redes de Computadores**, provendo o conhecimento necessário sobre as pilhas de protocolos, arquiteturas e topologias de redes; e **Engenharia de Software**, proporcionando o conhecimento necessário para o gerenciamento deste projeto, a distribuição e evolução das fases, modelagens para os sistemas, formas de implementação e de testes.

3.2 Material Utilizado

Durante o desenvolvimento deste projeto, vários equipamentos de hardware, sistemas operacionais, softwares, editores e compiladores serão utilizados. Abaixo estão relacionados os softwares e hardwares utilizados.

Equipamentos de hardware:

- multímetro
- osciloscópio
- fonte
- Soldador e estanho
- transistor
- resistor
- Diodos
- Capacitores
- Circuito integrado (MAX232)
- Relê
- Cabo LPT (impressora)
- Conectores DB9 fêmea/macho e DB25 macho
- Leds
- Reed-switch, lâmpada, plug e tomada
- Fios para interligação dos componentes
- Microcontrolador 16F628
- Matriz de contato

Sistemas operacionais:

- Microsoft Windows 98 / 2000
- Linux

Compiladores:

- C para PIC da CCS (http://www.ccsinfo.com)
- Borland Delphi
- kylix
- gcc (http://www.gnu.org/)

Softwares adicionais:

- Apache Web Server 2.0.45 (http://www.apache.org)
- Script CGI (http://cgiclube.cidadeinternet.com.br/)
- Biblioteca parapin_0.90 para gcc
- Browser

Hosts de desenvolvimento:

- AMD AthlonXP 1700+ 256MB RAM
- Suporte à rede ethernet 100 Mbps com acesso à Internet

3.3 Planejamento

3.3.1 A escolha do dispositivo a ser utilizado

A escolha do dispositivo a ser utilizado tomou por base três fatores. Em primeiro lugar, o fator financeiro, que levou-nos a procurar um dispositivo que pudesse ser completamente acessível de modo que o prejuízo fosse mínimo o caso de danos permanentes. Em segundo, o fator dificuldade, procuramos um dispositivo que possuísse funcionalidades de fácil compreensão. Por fim, o fator praticidade, onde visamos trabalhar com algo que pudesse ser facilmente transportado e que não exigisse cuidados especiais.

Levando em conta tais fatores, o dispositivo escolhido foi uma lâmpada. Pois a partir deste dispositivo que será testado, poderemos utilizar nosso sistema para qualquer outro dispositivo semelhante, basta adaptarmos nosso sistema.

3.3.2 O estudo do funcionamento do dispositivo

Especificado o dispositivo a ser utilizado, será dado início à fase de estudos sobre seu funcionamento. Nesta fase, serão aplicados os conceitos de Circuitos Eletrônicos para a compreensão do funcionamento dessa lâmpada com relação ao sistema de controle. Serão realizados testes operacionais sobre o circuito eletrônico até que toda sua lógica seja conhecida.

3.3.3 A criação da interface para a comunicação entre o dispositivo e o computador

Após a compreensão do funcionamento do circuito eletrônico do dispositivo, iniciará uma fase de projeto de uma interface de comunicação para a

interação entre o circuito e o computador. A forma adotada para prover a comunicação entre o computador e o circuito foi a utilização da porta paralela do computador. Nesta fase serão aplicados os conhecimentos de Arquitetura de Computadores e Circuitos Eletrônicos. Os "programas teste" serão desenvolvidos sobre o Windows de forma a verificar a sua funcionalidade.

3.3.4 O desenvolvimento de softwares de controle da interface

Depois de criar e validar a interface de comunicação entre o computador e o dispositivo, iniciaremos a fase de desenvolvimento de softwares para o controle e operação deste dispositivo. Nesta fase serão utilizados códigos em linguagens orientadas a objeto (C e Delphi) juntamente com a linguagem CGI para que as aplicações possam acessar a porta paralela. Da mesma forma como nas fases anteriores, os testes de validação sobre o que foi gerado também serão realizados.

3.3.5 O desenvolvimento de um servidor de comunicação remota

Neste ponto do projeto, passaremos para a fase de desenvolvimento de servidores para suporte à computação distribuída. Utilizando os conceitos de sistemas Cliente/Servidor e o conhecimento dos protocolos correntes das redes de computadores, os servidores serão modelados e desenvolvidos utilizando o protocolo TCP (Transmition Control Protocol). Novamente, realizaremos uma fase de teste para validar nosso dispositivo. A partir daí, já teremos uma boa parte da arquitetura idealizada concluída.

3.3.6 A modelagem da arquitetura final do sistema

Em paralelo com o desenvolvimento das fases restantes do projeto, daremos início à modelagem da arquitetura do sistema, partindo-se do que iremos implementar e chegando no que seria a arquitetura final do escopo deste projeto. Baseando-se nas fases que serão concluídas, chegaremos a uma estrutura final como segue abaixo:

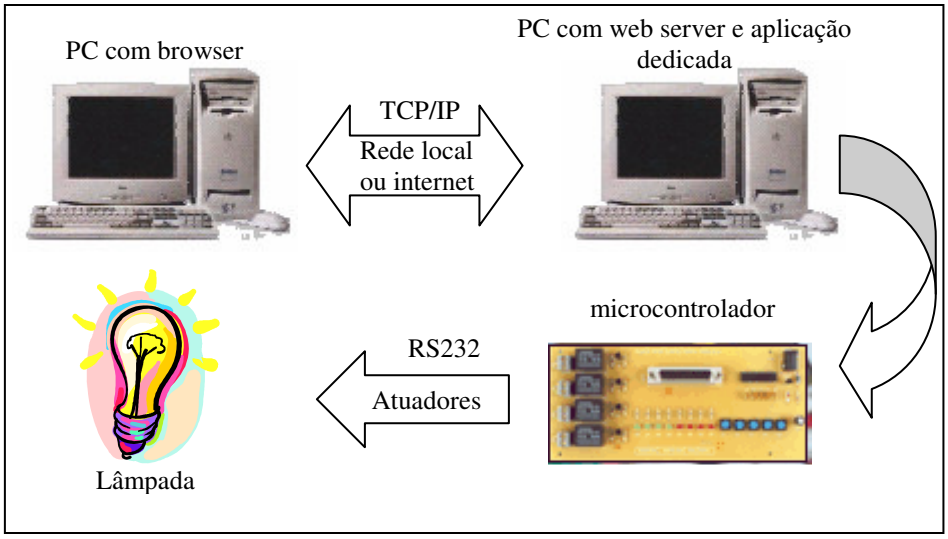


Figura 24: Arquitetura final do sistema

3.3.7 O desenvolvimento de sistemas web

Com base nos resultados das implementações das fases anteriores, a continuidade do projeto visou a disponibilização de um sistema para a manipulação do dispositivo através de browsers para a Internet. Nesta fase, os conceitos de programação para a Internet foram amplamente utilizados. Partindo da configuração de servidores apache e especificação e implementação de aplicações CGI, foi disponibilizado um sistema web para a manipulação do

dispositivo a partir de HTTP (Hiper Text Transfer Protocol). Após a implementação do sistema, o browser será utilizado para a realização de testes e validação.

3.3.8 O desenvolvimento do protótipo

Com a disponibilização de um sistema para a manipulação do dispositivo através de browsers para a Internet, será desenvolvido um protótipo para o envio de comandos através da porta serial ou paralela. Este protótipo será responsável pela interação do usuário com o dispositivo. Nesta fase, os conceitos de circuito integrado, microcontroladores e componentes eletrônicos foram amplamente utilizados. Finalizada a instalação e configuração dos sistemas web e de cabeamento, o protótipo será implantado no sistema para a realização de testes e validação.

3.3.9 O desenvolvimento de aplicações distribuídas sobre a arquitetura criada

Com o término das implementações que compõe a arquitetura final, descrita anteriormente, a fase que segue é a de desenvolvimento de aplicações sobre esta arquitetura criada. Esta fase é uma fase que não possui fim, visto que inúmeras aplicações poderão ser criadas baseando-se no que já existe e muitas outras podem vir a serem criadas partindo-se de novas tecnologias. O projeto restringiu-se apenas a criação de um web site que utiliza o servidor html criado (para a utilização do dispositivo através da internet em qualquer parte do mundo), uma aplicação remota que faz uso do servidor TCP e uma aplicação que manipula o dispositivo localmente.

3.4 Desenvolvimento do Dispositivo de Controle

O objetivo do protótipo é ser uma ferramenta para a monitoração e controle de dispositivos eletrônicos. Este protótipo terá como interface, o browser da Internet e os comandos de gerenciamento serão feitos utilizando o programa CGI, Delphi e linguagem C. O monitor estará instalado em um host conectado a uma rede Ethernet utilizando o protocolo TCP/IP. Neste capítulo serão abordados os detalhes relevantes sobre o desenvolvimento do protótipo.

3.4.1 Descrição do Sistema

Antes de entrar na parte técnica, mostrarei uma breve explicação do funcionamento do sistema e sua estrutura.

Primeiramente, observe a figura 25, note que é a planta de um cômodo de uma casa. No projeto será feita uma maquete para simular este cômodo. Este setor será nosso laboratório de testes. Nas partes onde estão localizadas a porta e a janela, serão instalados sensores do tipo reed-switch (figura 26) para saber se a porta ou a janela está fechada ou aberta. No laboratório encontra-se também uma lâmpada e uma bancada, onde serão instalados um relé e um contador para poder ligá-las remotamente.

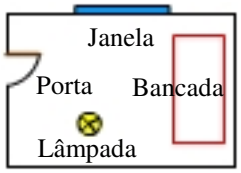


Figura 25: Maquete do laboratório

Figura 26: Sensor reed-switch

Agora observe o diagrama representado na figura 27, nele encontra-se um PC (computador) com um browser e outro PC que vai estar no laboratório que, por sua vez, está conectado em um microcontrolador através de um canal serial RS-232 que vai monitorar os sensores (porta e janela) e controlar os atuadores (luz e bancada).

Os dois computadores se comunicam entre si utilizando protocolos TCP/IP dentro de uma Rede Local ou Internet. A idéia do funcionamento é bem simples, no browser do PC o usuário digita o endereço do WEB Server que, por sua vez, se comunica com o microcontrolador executando comandos de escrita para comandar os atuadores (luz e bancada) e comandos de leitura para saber os estados dos sensores (porta e janela).

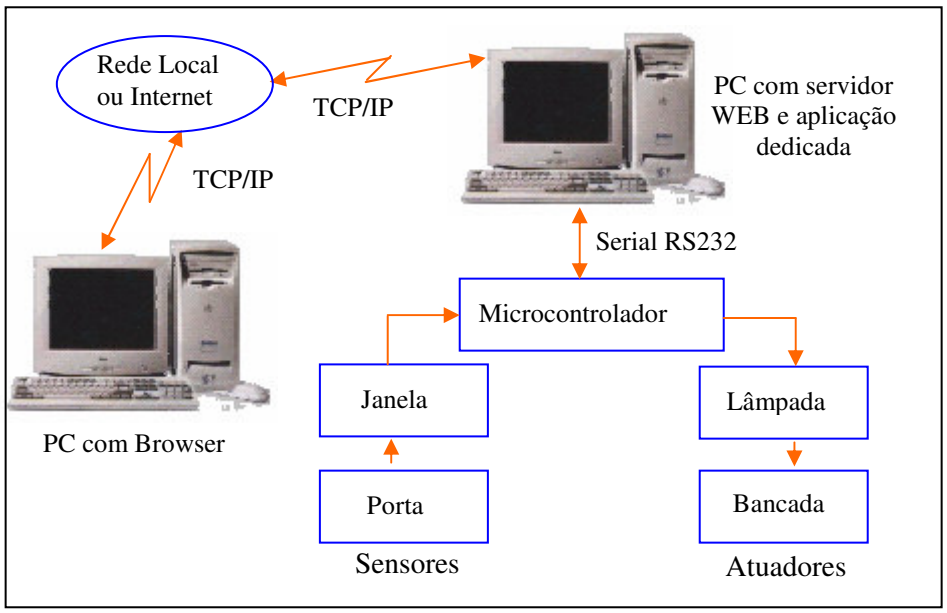


Figura 27: Diagrama do funcionamento do dispositivo de controle

O componente mais importante do protótipo responsável pela execução dos atuadores e sensores é o microcontrolador, onde será definido o protocolo de comunicação dentro do canal serial RS-232. Esse protocolo será gerenciado

pela aplicação de controle do laboratório. Teoricamente, o computador onde está instalado o WEB Server só responde se tiver uma requisição através de um browser. Isso facilitará o algoritmo que será aplicado no microcontrolador, que ficará monitorando o canal serial até receber os comandos de escrita ou leitura.

A figura 28 representa o fluxograma do programa do microcontrolador.

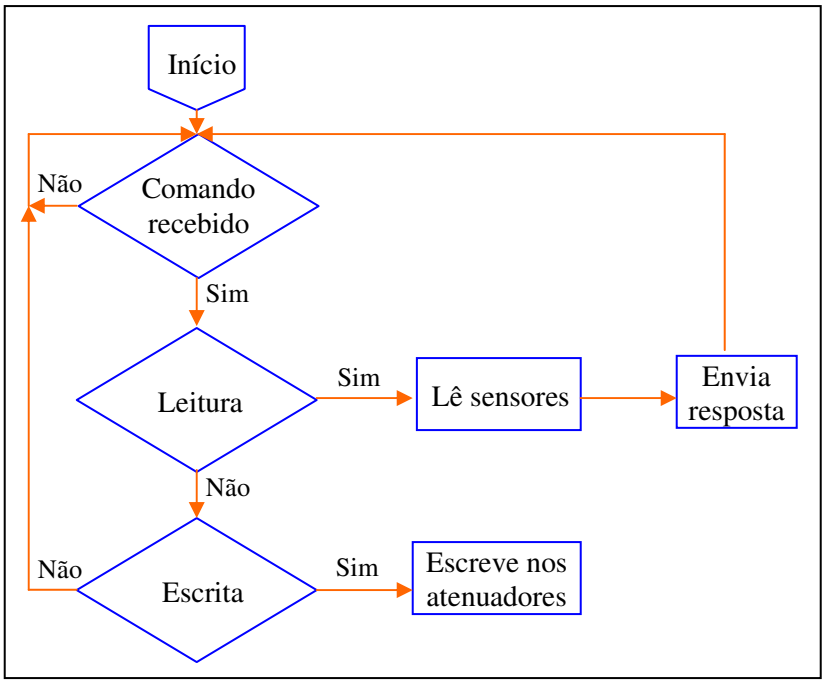


Figura 28: Fluxograma do programa do microcontrolador

Para este projeto, o microcontrolador escolhido foi o PIC16F628 que é pino compatível com o PIC16F84, a diferença é que tem um custo menor e tem mais periféricos. Foi adotado também o ambiente de programação em C para PIC, da CCS (Custom Computer Services), que é um ambiente muito eficiente e proporciona uma rapidez na execução de projetos.

Para fazer a aplicação de controle da serial e enviar resposta via HTML (Hypertext Markup Language), será utilizado o ambiente Delphi. Basicamente, a

aplicação será um CGI (Common Gateway Interface) standalone. Essa aplicação receberá requisições HTML (vindas de um browser), fará o processamento da requisição (comandos de leitura ou escrita na serial) e responderá via HTML para o browser. A figura 29 mostra o fluxo das requisições e atuações.

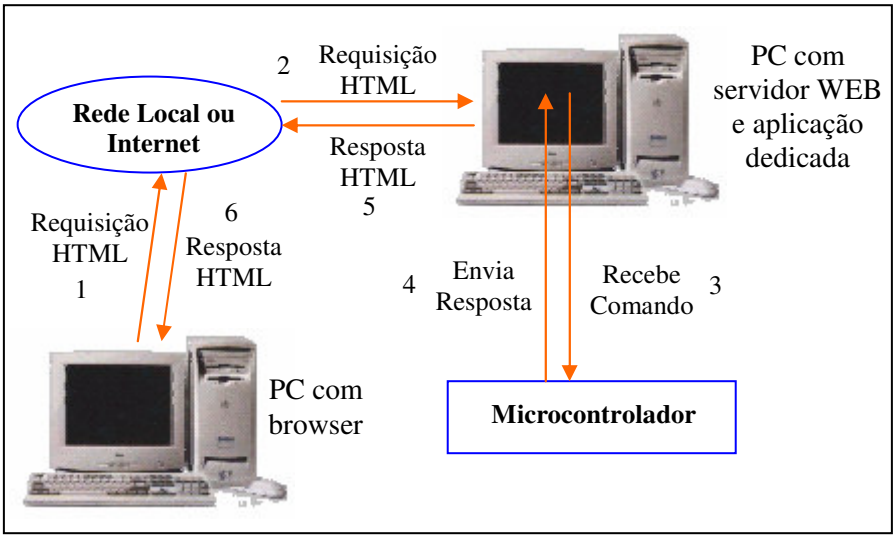


Figura 29: Fluxo das requisições e atuações

3.4.2 Conexões via Cabo Serial e Paralelo

O interessante de uma conexão via cabo paralelo ou serial é a facilidade de se executar, e o custo do material, pois utiliza somente dois conectores e alguns metros de cabo, e nenhum equipamento ou placa adicional. Utiliza-se somente a porta paralela ou serial, que já é padrão em qualquer PC.

Normalmente pode-se utilizar este tipo de conexão para jogos, transferência de arquivos entre PCs, dispositivos externos e em ambientes industriais na qual é utilizada para a comunicação dos mais diversos tipos de

equipamentos microprocessado devido exatamente ao seu baixo custo de implantação.

Antigamente era utilizado somente para a transferência de arquivos em ambiente DOS e no WINDOWS 3.1 havia o hiper terminal que ainda existe nos WIN95,98, NT ,2000. No windows 95 em diante foi criada a conexão direta via cabo, que aumentou muito as possibilidades de aplicação para este tipo de conexão. Pode-se, por exemplo, conectar-se a outro PC remoto como em um ambiente de rede completo, compartilhando arquivos, pastas, cd-rom, impressoras. Neste tipo de conexão (Internet Conection Sharing) não é possível utilizar o compartilhamento de acesso a Internet devido à inexistência de um adaptador de rede real (o adaptador Dial-Up é virtual).

Para a conexão serial utilizasse a saída RS 232 de 9 ou 25 pinos presente em qualquer PC, já para a conexão paralela, a saída paralela (LPT1 , LPT2 ...) de 25 pinos.

Há várias maneiras de ligar um dispositivo externo a um computador, como, conexão via cabo paralelo (hardware e software), rádio frequência, a interface infravermelho (IrDA), a porta USB (Universal Serial Bus) e conexão via cabo serial (hardware).

O uso do cabo paralelo foi uma das opções para realizar a conexão entre o protótipo e o computador, pelo fato de todos os computadores possuírem uma porta paralela (porta DB25 da impressora), esta porta nem sempre está sendo utilizada a todo o tempo (a serial é utilizada pelo mouse), a velocidade de transmissão é bem mais alta que a do cabo serial, pois a serial possui somente uma via de dados e a paralela possui oito, ideal para transmitir grande volume de dados entre os computadores. Apesar da vantagem do cabo paralelo, utilizei também como opção o cabo serial, pois atualmente o mouse utiliza a porta PS/2 e com isso a porta do mouse fica disponível. Portanto, para que o protótipo se

comunicasse com o computador através das duas opções, utilizei neste projeto a serial RS-232 (adaptável ao conector de 9 e 25 pinos).

3.4.3 Servidor Internet

Como foi implementado no projeto a comunicação TCP/IP, foi desenvolvido também um servidor para receber e enviar os comandos de leitura e escrita entre o browser e o microcontrolador através da interface serial RS232. O servidor utilizado para aplicação do script CGI foi o Apache, pelo fato de sua instalação e configuração ser fácil e simples de executar.

A versão utilizada para este projeto foi o Apache 2.0.45. Este servidor pode ser encontrado no seguinte site (http://www.apache.org). A instalação do Apache é feita como qualquer outro programa do Windows. Depois de instalado, é necessário fazer a configuração do servidor, através do arquivo httpd.conf. O mínimo que deve ser configurado é o diretório onde os documentos estarão, através da opção DocumentRoot.

Basta procurar a opção e escrever o nome do diretório, como no exemplo: DocumentRoot "C:\akira\" Uma outra configuração básica é a DirectoryIndex, que informa ao servidor quais os arquivos serão exibidos automaticamente como índice do diretório. É isso que faz com que ao digitar, por exemplo, "www.projeto.com.br", o servidor saiba qual dos arquivos do diretório deve ser exibido. Abaixo temos um exemplo da utilização do DirectoryIndex:

DirectoryIndex index.html index.htm.

Para definir essa configuração, cria-se um arquivo com um dos nomes definidos como índice e este é alocado no diretório definido como root. Para

saber se o servidor foi instalado corretamente, basta executar o Apache e acessar o endereço "http://localhost" pelo browser. Se a página for exibida, é porque o servidor foi instalado corretamente.

Em seguida deve-se configurar o Apache para reconhecer o script cgi. Atualmente o apache já vem configurado para executar cgi, caso isso não ocorra, deve-se configurar a diretiva ScriptAlias e LoadModule da seguinte maneira:

ScriptAlias /cgi-bin/ "C:/Apache Group/Apache2/cgi-bin/"
LoadModule cgi_module modules/mod_cgi.so (deixar ativo)

Com o servidor pronto para testar o executável do laboratório, será desenvolvido o programa que enviará e receberá os comandos vindos do microcontrolador e as respostas de envio e recebimento do status do dispositivo através da Internet.

3.4.4 Aplicação Desenvolvida

Primeiramente foi desenvolvida uma aplicação para carregar as variáveis de comando e leitura no microcontrolador e para que o software aplicativo possa executar e verificar o estado dos componentes do laboratório.

Esse aplicativo foi desenvolvido na linguagem de programação C e foi utilizado o compilador CCS que possui algumas diretivas que facilitam o acesso a porta serial e o envio e recebimento de dados. Na figura 30 é descrito o código de leitura do status dos dispositivos que estão presentes no laboratório.

```
#include <16f628.h>
#byte port_b=6
#use delay(clock=4000000)
#fuses XT,NOWDT,PUT,BROWNOUT,NOMCLR,NOLVP
#use fixed_io(b_outputs=PIN_B5 ,PIN_B4, PIN_B2)
#use  RS232(BAUD=2400, XMIT=PIN_B2, RCV=PIN_B1)

int Buffer[20];     // Buffer de recebimento
int1 A1;       // Luz
int1 A2;       // Bancada
int1 S1;       // Porta
int1 S2;       // Janela

void leitura()
 {
 if(bit_test(port_b, 6))
   S1 = 1;
  else
   S1 = 0;
 if(bit_test(port_b, 7))
   S2 = 1;
  else
   S1 = 0;
 }
```

Figura 30: Código de Leitura

Foi incluído o header 16F628 que é a classe do microcontrolador 16F628 da família PIC. As demais diretivas se referem aos pinos utilizados do microcontrolador e ao tipo de acesso a porta de comunicação. Em seguida foram definidas as variáveis referentes aos dispositivos. Esta função apenas verifica o estado de cada sensor (porta e janela).

Na figura 31 é descrito o código de escrita para os atuadores (lâmpada e bancada). E na figura 33 temos a implementação básica da main()

```
void escrita()
{
  if(buffer[4] == '1')
    {
      A1 = 1;  // Liga Luz
      output_high(PIN_B4);
    }
  else
    {
      A1 = 0; // Desliga Luz
      output_low(PIN_B4);
    }
  if(buffer[7] == '1')
    {
      A2 = 1; // Liga Bancada
      output_high(PIN_B5);
    }
  else
    {
      A2 = 0; // Desliga Bancada
      output_low(PIN_B5);
    }
}
```

Figura 31: Código de Escrita

```
void resposta()
 {
   printf("*RA1%uA2%uS1%uS2%u", A1, A2, S1, S2);
   putc(0x0D);     // carriage return
   putc(0x0A);     // line Feed

 }


void executa_cmd()
 {
   switch (buffer[1])
   {
     case 'L': leitura();
               resposta();
               break;
     case 'E': escrita ();
               leitura();
               resposta();
               break;
   }

 }
```

Figura 32: Parte do código main

```
void main()    // Programa Principal
 {
  port_b = 4;           // Limpa atuadores e Seta Stop Bit
  port_b_pullups(true); // liga pull ups
  A1 = 0;
  A2 = 0;
  while (true)
   {
     gets(buffer);          // Le comando
     if(buffer[0] == '*')  // Confirma Reader
      executa_cmd();
   }

 }
```

Figura 33: Código main()

A função resposta() é utilizada apenas para mostrar os resultados de cada requisito. A função executa_cmd() é para definir o tipo de requisição. E por fim temos o programa principal responsável pela leitura e escrita dos dispositivos.

3.4.5 Software Aplicativo

A interface que o usuário utiliza para acessar os dispositivos é o browser da Internet. Esta interface é manipulada no computador que possui o servidor web e o aplicativo de comunicação com a serial, como visto na figura B. Para ter acessa a esta interface, é necessário que o usuário esteja conectado a Internet. Caso o acesso a esta interface seja numa rede local, basta digitar o ip da máquina.

Para desenvolver esta interface, foi utilizado o Delphi para acessar a serial e o CGI para atualizar e alterar os comandos de leitura e escrita dos dispositivos. O uso do CGI facilita o monitoramento dos dispositivos, pois a cada alteração feita no laboratório, a interface é atualizada automaticamente.

O software aplicativo utiliza as seguintes bibliotecas: SysUtils, Classes, HTTPApp, HTTPProd, CPort, uCiaComport. Essas bibliotecas já vêm prontas no Delphi, isso possibilita um enfoque mais detalhado no programa principal.

Para gerar a página HTML foi criado um módulo chamado TPageProducer. A partir dele, será enviado aos atuadores os comandos de escrita e verificado nos sensores os comandos de leitura. Na figura 34 vemos um trecho do código TPageProducer.

```
  '              <div align="center"><font face="Verdana, Arial, He' +
  'lvetica, sans-serif"><#LUZ></font></div>'
'              </td>'
'              <td width="43%">'
'                <div align="center">'
'                  <#BOTAOLUZ>'
'                </div>'
'              </td>'
'            </tr>'
'            <tr>'
'              <td width="26%">'

  '              <div align="center"><font face="Verdana, Arial, He' +
  'lvetica, sans-serif">Bancada</font></div>'
'              </td>'
'              <td width="31%">'

  '              <div align="center"><font face="Verdana, Arial, He' +
  'lvetica, sans-serif"><#BANCADA></font></div>'
'              </td>'
'              <td width="43%">'
'                <div align="center">'
'                  <#BOTAOBANCADA>'
'                </div>'
'              </td>'
```

Figura 34: Trecho do código TPageProducer

No programa principal há uma rotina que manipula as tags colocados dentro do html. Tags são marcas <#BOTAOLUZ> que, quando o pageproducer processa o html e encontra a tag, chama esta rotina para a substituição da tag por algum valor texto. As tags são inseridas dentro do html que está na propriedade HTMLDoc do componente main (que é um PageProducer). Na figura 35 vemos parte do código que manipula as tags colocados dentro do html.

```
// Atuador da Luz
if TagString = 'LUZ' then
   if HardwareStatus <> 'ERRO' then
      begin
         if Pos( 'A11', HardwareStatus ) > 0 then
            ReplaceText := 'Acesa'
         else
            ReplaceText := 'Apagada';
      end
   else
      ReplaceText := '?';

// Botao da LUZ
if TagString = 'BOTAOLUZ' then
   if HardwareStatus <> 'ERRO' then
      begin
         if Pos( 'A11', HardwareStatus ) > 0 then
            ReplaceText := '<input type="submit" name="Luz" value="Apagar">'
         else
            ReplaceText := '<input type="submit" name="Luz" value="Acender">';
      end
   else
      ReplaceText := '?';

// Atuador da Bancada
if TagString = 'BANCADA' then
   if HardwareStatus <> 'ERRO' then
      begin
         if Pos( 'A21', HardwareStatus ) > 0 then
            ReplaceText := 'Ligada'
         else
            ReplaceText := 'Desligada';
      end
   else
      ReplaceText := '?';

// Botao da Bancada
if TagString = 'BOTAOBANCADA' then
   if HardwareStatus <> 'ERRO' then
      begin
         if Pos( 'A21', HardwareStatus ) > 0 then
            ReplaceText := '<input type="submit" name="Bancada" value="Desligar">'
         else
            ReplaceText := '<input type="submit" name="Bancada" value="Ligar">';
      end
   else
      ReplaceText := '?';
```

Figura 35: Parte do código que manipula as tags colocados dentro do html

Neste código vemos que há várias instruções que verificam se as tags do pageproducer foram chamadas para serem aplicadas no microcontrolador. No exemplo do atuador da luz na figura Y, se o HardwareStatus for 1, indica que a luz está acesa, caso contrário, está apagada.

Se houver algum tipo de requisição dos atuadores, estes são aplicados na variável CMD para montar as ações via serial, como visto na figura 36.

```
// Atuador da LUZ
if Pos('Luz=Acender', Request.Content) > 0 then
   cmd := '*EA11A2?'+#13;
if Pos('Luz=Apagar', Request.Content) > 0 then
   cmd := '*EA10A2?'+#13;

// Atuador da Bancada
if Pos('Bancada=Ligar', Request.Content) > 0 then
   cmd := '*EA1?A21'+#13;
if Pos('Bancada=Desligar', Request.Content) > 0 then
   cmd := '*EA1?A20'+#13;

if cmd = '' then
   cmd := '*L'+#13;
```

Figura 36: Código que monta as ações via serial

Na figura 37 vemos o código para ter acesso a porta serial e enviar e receber comandos de leitura e escrita. Neste código a variável COMM foi definida como COM1 para ter acesso a serial COM1. Para ter acesso a COM2 basta alterar a variável COMM. Para enviar o comando de leitura via serial utiliza-se a função WriteStr e para receber o status dos dispositivos utiliza-se a função ReadStr.

Ao compilar o programa, será criado um executável chamado lab1.exe para ter acesso a COM1. O lab2.exe será utilizado para ter acesso a COM2. O uso deste programa na Internet é feito da seguinte maneira. Como o servidor está pronto para executar um programa CGI, basta enviar o arquivo lab1.exe e lab2.exe para o subdiretório cgi-bin do Apache. Para testar o laboratório, utilizamos um browser qualquer e digitamos no endereço o seguinte caminho:

http://localhost/cgi-bin/lab1.exe

```
// Pega status do Hardware
HardwareStatus := '';
try
   // Abre a porta COMM
   COMM.Open;

   // Envia o comando de Leitura
   COMM.WriteStr( 'L'+#13 );
   // Aguarda 1 segundo para o Hardware responder
   Sleep(1000);
   // Le o resultado na variavel global HardwareStatus
   COMM.ReadStr(HardwareStatus, 15 );


   // Se existir comando para ser enviado via serial
   if cmd <> '' then
      begin
         // Monta o cmd correto
         if cmd[5] = '?' then
            cmd[5] := HardwareStatus[5];
         if cmd[8] = '?' then
            cmd[8] := HardwareStatus[8];

         // Envia comando para o atuador
         Sleep(1000);
         COMM.WriteStr(cmd);
      end;
   // Aguarda 1 segundo para o Hardware responder
   Sleep(1000);
   // Le o resultado na variavel global HardwareStatus
   COMM.ReadStr(HardwareStatus, 15 );
   // Fecha porta COMM
   COMM.Close;
except
   // Se houve algum erro no trato da porta COMM (Nao conseguiu abri-la e etc )
   // retorna erro que sera tratado posteriormente
   HardwareStatus := 'ERRO';
end;
// Monta a pagina de retorno para o Browser
Response.Content := Main.Content;
```

Figura 37: Trecho de código para ter acesso a porta serial

3.4.6 Funcionamento do Hardware

Os principais componentes utilizados para desenvolver o protótipo são o microcontrolador, o Relé e o circuito integrado MAX232. Capacitores, resistores, diodos e transistores foram utilizados para evitar curto circuito ou inversão da polaridade. A figura 38 mostra o esquema elétrico do protótipo.

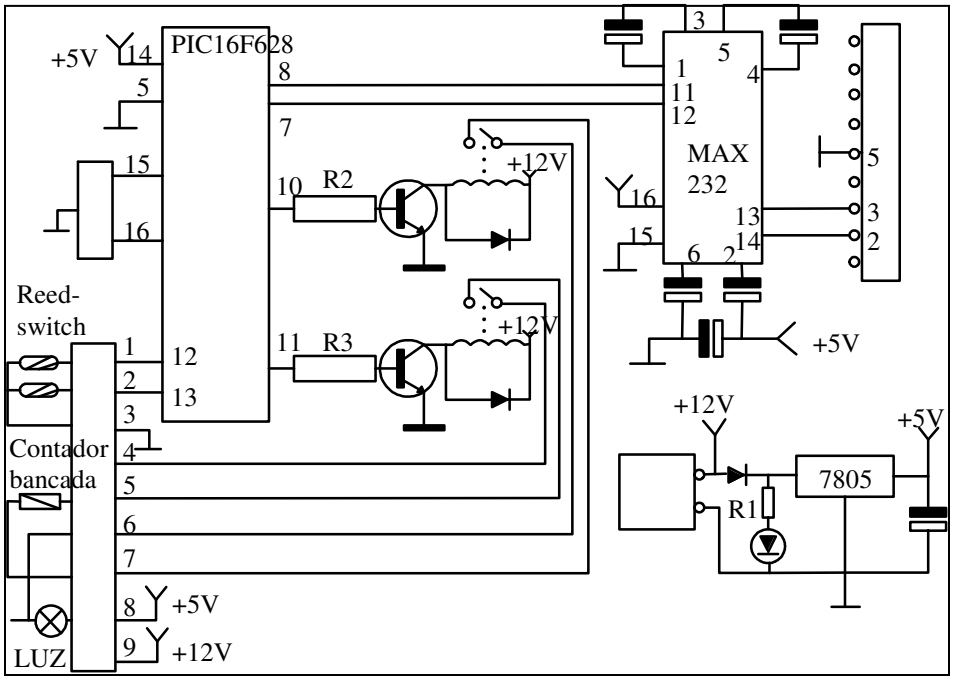


Figura 38: Esquema elétrico do protótipo

Como foi dito no referencial teórico, o microcontrolador 16F628 foi utilizado devido a sua eficiência e baixo custo. O uso de relés neste protótipo também é muito importante, pois eles são responsáveis pelo controle dos dispositivos ligando ou desligando-os. E para enviar os dados por uma saída serial, utilizou-se o chip MAX 232, que converte os níveis de tensão TTL para RS-232.

O conector DB9 (fêmea) foi utilizado para conectar na porta serial do computador, no caso na porta COM1 ou COM2, nota-se que são utilizados apenas os pinos 2, 3 e 5. O pino 2 é ligado ao pino 14 do MAX232 e o pino 3 do conector é ligado ao pino 13 do circuito integrado. Sendo o pino 2 receptor e o pino 3 transmissor de dados. O pino 5 é o terra da serial. A figura 39 mostra a estrutura do protótipo desenvolvida.

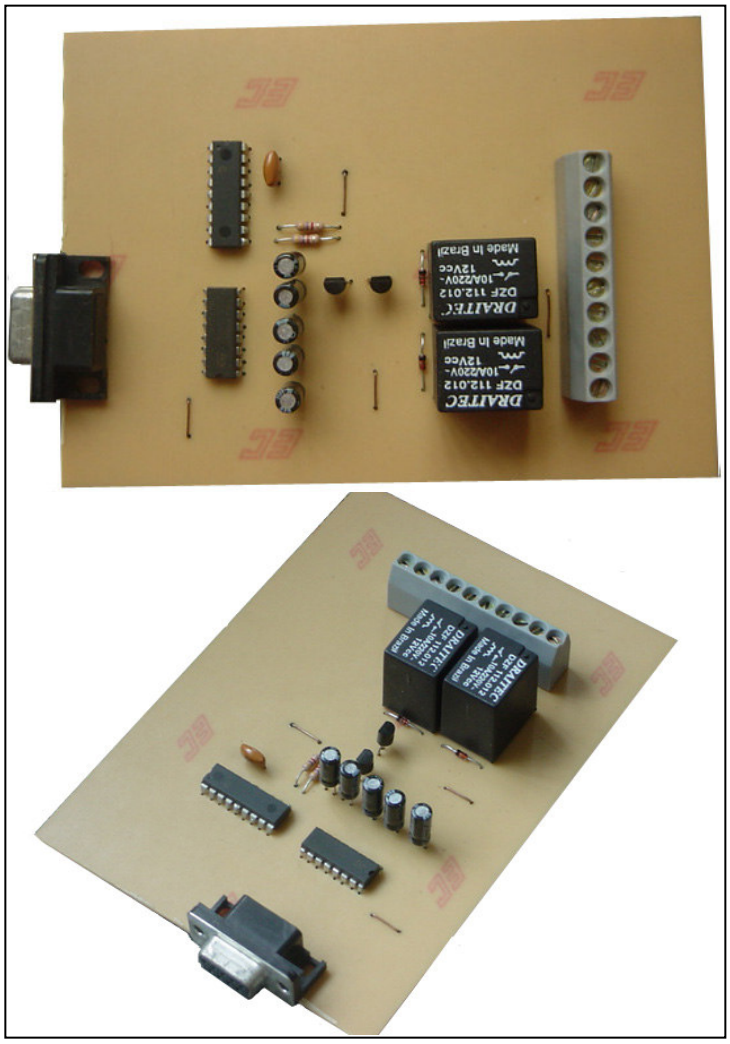


Figura 39: Protótipo

O MAX232 é utilizado para compatibilizar os níveis de tensão apresentados pela porta serial (-12V/+12V) com os níveis aceitos pelo transmissor e pelo receptor (+5 volts), a alimentação do MAX232 deve ser de +5V e esta tensão é adquirida pelo pino 16. O pino 15 é o terra do circuito. Capacitores são utilizados para permitirem o funcionamento correto do MAX232 conforme é apresentado em seu manual.

Os dados do circuito integrado são transmitidos através dos pinos 11(receptor) e 12(transmissor) e ligados aos pinos 7 e 8 do microcontrolador. O PIC converte esses sinais digitais em pequenas tensões que são transmitidas para os relés, através dos pinos 10 e 11. Como foi visto no referencial teórico, quando um relé recebe uma certa tensão, o contato é ativado e assim a corrente passa pelo circuito (Este circuito pode ser uma lâmpada).

Foi criado um regulador de tensão para alimentar o MAX232 e o microcontrolador. Este regulador utiliza uma tensão de +12V/100mA vindos da serial. Como nenhum pino da serial não tem essa tensão, criou-se um software para que um dos pinos que não está sendo utilizado na comunicação ficar com nível 0, sendo que nível 0 no RS232 é 12V e nível 1 é -12V. Esta técnica é bastante utilizada e pode alimentar circuitos que consomem pouca corrente. A figura 40 mostra o regulador de tensão.

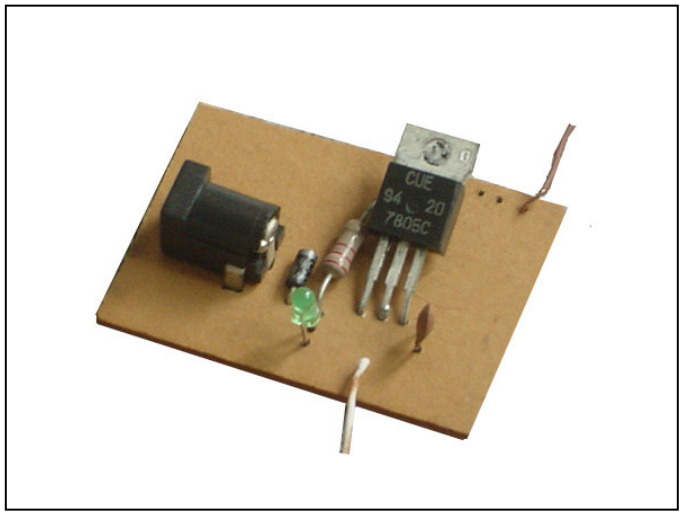


Figura 40: Regulador de Tensão

Capítulo 4

Considerações Finais

4.1 Dificuldades Encontradas

As etapas que realmente foram implementadas do protótipo foram a aquisição de dados através da Internet utilizando o protocolo TCP/IP e o acesso aos comandos de leitura e escrita através da serial RS232. Este procedimento é atualizado constantemente, de acordo com o tempo configurado. Quando o usuário quiser ligar ou desligar algum dispositivo, basta pressionar os botões que estão disponíveis na interface de comunicação e os dados serão enviados pela porta serial, para um microcontrolador e deste para o computador que retornará com o status do dispositivo.

As funcionalidades do projeto que envolvia a serial RS232 não foram implementadas. Após ter sido estudada a melhor solução para realizar esta tarefa, foi escolhido o componente MAX232, da Maxim, que possui periféricos disponíveis como a serial RS232. Foi feito então todo o estudo para integração do mesmo com o microcontrolador PIC (hardware), bem como o Relé para poder ativar ou desativar os dispositivos que foram instalados para testes. Todas estas funcionalidades já foram descritas no referencial teórico.

A primeira dificuldade encontrada foi em relação à alimentação do MAX232 e do PIC16F628. Isto foi resolvido usando um regulador de tensão, que converte a tensão de 12V para 5V utilizando um regulador 7805. Outro problema encontrado foi carregar um software no microcontrolador para ter acesso a serial e aos dispositvios. Para isto, foi desenvolvido um programa em C

para acessar seus componentes. A grande dificuldade em desenvolver esse programa foi a falta de um compilador compatível com o programa. O desenvolvimento do programa em linguagem Assembly foi uma das opções, porém, não havia material suficiente que mostrasse todos os recursos que seriam utilizados pelo protótipo como, por exemplo, o acesso a serial RS-232 e ao microcontrolador da família PIC. Havia um compilador da CCS compatível com as diretivas do programa. Porém, depois de compilado o programa, surge mais um empecilho, um aparelho que desse suporte ao microcontrolador. Quer dizer, o programa precisaria ser armazenado na memória do programa do PIC16F628, para que o software criado em html tivesse acesso ao programa do microcontrolador. Havia um gravador da ALLMAX na UFLA, porém esse aparelho era muito antigo e não era compatível com o PIC16F628. Logo não foi possível fazer o download do programa no microcontrolador.

O que foi então realmente implementado foi o microcontrolador com comunicação serial. Esta comunicação serial é feita com o MAX232 e software de configuração, responsável em receber e enviar os valores de comando e mostrá-los em um browser na Internet. No circuito foram instalados vários capacitores, diodos, transistores, resistores para evitar perda de componentes importantes tal como o circuito integrado MAX232 e o PIC16F628. O único dispositivo a ser testado foi a interface de comunicação entre o usuário e o protótipo.

Foi utilizada uma fonte de tensão externa, colocada no regulador para verificar seu correto funcionamento. O sinal de tensão é enviado para o conector P4 e deste passa para o regulador 7805 convertendo em um sinal de tensão de 5V. Os teste seriam realizados da seguinte maneira, seriam utilizados uma lâmpada, uma bancada e dois sensores, sendo um instalado na porta e outro na janela do laboratório. Seriam 2 tipos de testes, um através da Internet e outro através da rede local.

4.2 Conclusão

Nesta monografia foram descritos todo o funcionamento do microcontrolador, protocolo TCP/IP, Relés, servidor e interface serial. Foram dadas as descrições de todo o circuito e também da parte de software. O protótipo foi desenvolvido de forma correta e a configuração de software funcionou normalmente atingindo o objetivo proposto. O que não foi atingido no projeto, apresentado como proposta inicial, foi o desenvolvimento deste projeto no linux. Um dos empecilhos foi a falta de material completo para a implementação do software no kylix e o acesso a porta serial ser muito complexa. O que foi desenvolvido foi um controlador com comunicação serial RS-232. As principais dificuldades encontradas no projeto foram relacionadas ao compilador do programa, o acesso ao microcontrolador através do software desenvolvido em linguagem C, à montagem do protótipo e a implantação do projeto no sistema operacional Linux. O grande empecilho foi o gravador de PIC que estava indisponível no momento em que se realizariam os testes. Mesmo que não tenham sido implementadas toda as funcionalidades propostas, aprendi a lidar com o protocolo TCP/IP, rede Ethernet, relés, ambiente WEB e sua aplicação utilizando microcontroladores PIC. Aprendi também que todo o processo de desenvolvimento necessita de um bom gerenciamento e organização para não se perder durante o processo.

4.2 Trabalhos Futuros

A minha recomendação, caso alguém se interesse em dar continuidade ao projeto, sugiro que continue utilizando o compilador da CCS, pois o seu

planejamento de projetos é muito eficaz. Há vários outros tipos de trabalhos que podem ser feitos sobre esse projeto, como por exemplo, o meio de comunicação.

Ao invés de utilizar cabo serial, pode-se adaptar a estrutura de comunicação pela tecnologia wireless (sem fio). Embora a instalação do sistema seja um pouco mais cara, no futuro, pode economizar tempo e dinheiro, já que elimina a necessidade de instalação de um novo cabeamento. O Sistema wireless expandiria a rede de dados já existente ou aumentaria o número de dispositivos em locais onde seria difícil a passagem de cabos.

Outro trabalho bastante interessante seria adaptar o projeto sobre o sistema operacional Linux, pois assim garantiria o código livre de todas as ferramentas utilizadas durante o processo.

Outra dica seria implantar um sistema de segurança na interface de comunicação entre o usuário e o dispositivo. Um sistema de acesso com senha ou de segurança tipo firewall, antivírus e outros sistemas que impediria o acesso de qualquer pessoa a não ser o usuário ao sistema.

Como o projeto desenvolvido é um protótipo base, há uma infinidade de adaptações que podem ser feitas sobre o projeto.

Referências Bibliográficas

[Bernardes & Chostakovis (2002)] Bernardes, L. H. C. & Chostakovis, R. Controle de Dispositivos pelo Internet Explorer. Eletrônica Total nº 88, 2002.

[Comer (1998)], Douglas E; Interligação em rede com TCP/IP. Rio de Janeiro Campus, 1998.

[Kuo (1995)], Benjamin C.; Automatic Control Systems, seventh edition, 1995.

[Redes (2001)], Curso de. Trabalho sobre as camadas de rede**.** Proença, mar [2001?]. Disponível em: <http://proenca.uel.br/curso-redes-graduacao/1998/trab-08>. Acesso em: 25 mar. 2003.

[RFC 791(2001)]. Internet Protocol. Califórnia, mar [2000?]. Disponível em <http://www.netsys.com/rfc/rfc791.txt>. Acesso em: 12 mar. 2003.

[Stang (1994)], David J; MONN, Sylvia. Segredos de segurança em rede. Rio de Janeiro: Berkeley, 1994.

[Starlin (1999)], Gorki. Manual Completo do *Hacker*: como ser e evita-los. Rio de Janeiro: Book Express, 1999.

[Stevens (1994)], Richard. *TCP/IP Illustrated, Volume 1:The Protocols*,Addison -Wesley, 1994

[Tanembaum (1997)], Andrew S. Redes de computadores. Rio de Janeiro: Campus, 1997.

[MAXIM(2001)]. Datasheets de componentes. Disponível em: <http://www.maxim-ic.com>. Acesso em: 15 de julho de 2003.

[MICROCHIP(2000)]. Datasheets de componentes e notas de aplicação.

Disponível em: <http://www.microchip.com>. Acesso em: 05 de Agosto de 2003.

[SEIKO]. Datasheets de componentes. Disponível em: <http://www.iready.com>. Acesso em: 05 de Agosto de 2003.

[Souza], David José de. Desbravando o PIC. São Paulo: Ed. Érica, 2000. 2ª Edição.

[Malvino], A. P. & LEACH, D. P. Eletrônica Digital - Princípios e Aplicações. São Paulo: Editora McGraw-Hill, vol 1 e 2. 1987