UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
UNIDADE ACADÊMICA DE ENGENHARIA ELÉTRICA

RELATÓRIO DE ESTÁGIO

PROPOSTA DE LABORATÓRIO DE REDES DE COMPUTADORES DE BAIXO CUSTO

ALUNO: PAULO RIBEIRO LINS JÚNIOR
ORIENTADOR: PROF. DR. EDMAR CANDEIA GURJÃO

CAMPINA GRANDE, 11 DE SETEMBRO DE 2006

RELATÓRIO DE ESTÁGIO

PROPOSTA DE LABORATÓRIO DE REDES DE COMPUTADORES DE BAIXO CUSTO

Estágio realizado no Laboratório de Comunicações (LabCom) da Unidade Acadêmida de Engenharia Elétrica da UFCG, no período de 10 de julho à 8 de setembro de 2006.

Relatório apresentado como requisito necessário para a disciplina de Estágio Supervisionado, obrigatória para conclusão do curso de Engenharia Elétrica da Universidade Federal de Campina Grande.

Aluno: Paulo Ribeiro Lins Júnior

Orientador: Prof. Dr. Edmar Candeia Gurjão

Convidado: Prof. Dr. Bruno Albert

Biblioteca Setorial do CDSA. Fevereiro de 2021.

Sumé - PB

Agradecimentos

★ À energia que rege o universo e as minúsculas partes que o compõe (seja ela chamada de Deus, Alá, Rah, Tupã, Força ou qualquer outro apelido), por permitir que eu faça parte desse todo tão magnífico;

★ À minha estimada família, que da forma deles sempre me ajudaram e respeitaram a mim e ao meu trabalho. Em especial ao exemplo de força, fé e capacidade de vencer adversidades ao qual carinhosamente chamo de mãe e a grande figura da qual tirei grandes lições e que chamo de pai;

★ Aos meus grandes irmãos e amigos de longa data Jonas Agápito, Jerônimo Silva, Adrian Lívio, Erik Silva, Olímpio Cipriano, Diego Bezerra, José Luís e Danilo dentre outros sem os quais definitivamente não teria chegado até aqui, e aos mais recentes porém não menos importantes amigos Carlos Danilo, Gilney Barros e Jean, que tornam o dia-dia rotineiro do laboratório mais agradável;

★ À todos os companheiros de luta que durante um ou mais semestres sofreram as provações de ser um aluno de Engenharia Elétrica juntamente comigo;

★ Ao Departamento de Engenharia Elétrica, berço de minha formação, pela oportunidade de crescimento técnico e pessoal. Em especial as figuras de Adail e Rosilda, anjos que tornam nossa passagem pelo curso menos dolorosa;

★ Ao professor Marcelo Sampaio, por ter me dado a oportunidade de trabalhar com um grande profissional e poder crescer com essa experiência;

★ Ao meu orientador Edmar Candeia Gurjão, grande mestre e amigo, por ter sido e ainda ser uma referência de profissional e ser humano;

★ Aos professores do Departamento de Engenharia Elétrica que cumpriram sua missão como tutores, e me ajudaram a crescer tecnico e profissionalmente; também agradeço aos que não cumpriram sua missão como tutores, por terem me dado um ótimo exemplo do que não quero jamais ser.

Sumário

Lista de Figuras

Lista de Siglas

DHCP - Protocolo de Configuração Dinâmica de Hospedeiros

HTTP - Protocolo de Transferência de Hiper Texto

IP - Protocolo de Internet

LAN - Rede Local

MAC - Camada de Acesso ao Meio

TCP/IP - Protocolo de Controle de Transmissão/Protocolo de Internet

UDP - Protocolo de Usuário de Datagrama

WWW - World Wide Web

Capítulo 1

INTRODUÇÃO

1.1 Considerações Gerais

O mundo atual passa por uma mudança provocada pela difusão dos meios de comunicação, e as redes que os interligam. Um grande exemplo disso é o fato de que grande parte das pessoas trabalham, estudam e se comunicam utilizando exclusivamente a Internet.

Diariamente as pessoas se deparam com algum tipo de rede. Caixas eletrônicos de bancos, por exemplo, nada mais são do que computadores interligados através de uma rede à um computador central que armazena dados sobre a conta bancária de um cliente.

A Internet revolucionou a forma de usar o computador e o mundo das comunicações, pois uma disseminação de informações a nível mundial, representando, dessa forma, um meio de cooperação e interação entre indivíduos e seus computadores sem consideração da localização geográfica.

A Internet introduziu uma gama considerável de novas tecnologias e protocolos, tais como o Protocolo de Internet (IP), o protocolo de transferência de hipertexto (HTTP), dentre outros.

Devido a presença das redes em grande parte dos sistemas atuais, se fez necessário que os cursos de Computação e Engenharia Elétrica ofereçam disciplinas sobre o tema de redes de computadores.

Como na maioria das disciplinas ministradas em áreas ligadas a tecnologia, surge a necessidade de aliar a teoria ministrada em sala de aula com aulas práticas. Porém, montar um laboratório básico de redes, contendo alguns computadores interligados, requer um investimento nem sempre disponível.

Este trabalho teve como objetivo propor um conjunto de experimentos que podem ser usados como laboratório de baixo custo para as aulas práticas de redes de computadores. Foram utilizados microcontroladores Microchip com servidores Web embarcados que podem ser usados em substituição aos computadores para formar uma rede, diminuindo consideravelmente o custo na implementação desse laboratório.

Este relatório descreve a tecnologia utilizada e os experimentos propostos, e está dividido da seguinte forma:

Capítulo 1 - apresenta uma motivação, juntamente com os objetivos e justificativas para a realização do trabalho.

Capítulo 2 - apresenta uma revisão de redes de computadores, com enfoque no modelo de camadas TCP/IP, em especial nas camadas de aplicação, de rede e enlace.

Capítulo 3 - descreve os experimentos com base na revisão teórica feita no capítulo anterior.

Capítulo 4 - traz as conclusões do trabalho e propostas para trabalhos futuros.

Apêndice - apresenta uma descrição da placa com microntrolador utilizada nos experimentos.

Capítulo 2

REDES TCP/IP - CONCEITOS FUNDAMENTAIS

Esse capítulo objetiva criar um embasamento teórico para os experimentos propostos no capítulo seguinte.

2.1 Conceitos Gerais de Internet e TCP/IP

A Internet é uma rede pública de comunicação de dados, com controle descentralizado e que utiliza o conjunto de protocolos TCP/IP como base para a estrutura de comunicação e seus serviços de rede. A arquitetura TCP/IP fornece não somente os protocolos que habilitam a comunicação de dados entre redes, mas também define uma série de aplicações que contribuem para a eficiência e sucesso da arquitetura. Entre os serviços mais conhecidos da Internet estão o correio-eletrônico (protocolos SMTP, POP3), a transferência de arquivos (FTP), o compartilhamento de arquivos (NFS), a emulação remota de terminal (Telnet), o acesso à informação hipermídia (HTTP), conhecido como WWW (*World Wide Web*).

A Internet é dita ser um sistema aberto uma vez que todos os seus serviços básicos assim como grande parte das aplicações são definidas publicamente em documentos conhecidos como RFCs (do inglês, *Request for Comments*), podendo ser implementadas e utilizadas sem pagamento de *royalties* ou licenças para outras instituições.

O conjunto de protocolos TCP/IP foi projetado especialmente para ser o protocolo utilizado na Internet. Sua característica principal é o suporte direto a comunicação entre redes de diversos tipos. Neste caso, a arquitetura TCP/IP é independente da infra-estrutura de rede física ou lógica empregada. De fato, qualquer tecnologia de rede pode ser empregada como meio de transporte dos protocolos TCP/IP, como será visto adiante.

2.2 Protocolos TCP/IP

TCP/IP é um acrônimo para o termo *Transmission Control Protocol/Internet Protocol Suite*, ou seja é um conjunto de protocolos, onde dois dos mais importantes (o IP e o TCP) deram seus nomes à arquitetura. O protocolo IP, base da estrutura de comunicação da Internet é um protocolo baseado no paradigma de comutação de pacotes (*packet-switching*).

Os protocolos TCP/IP podem ser utilizados sobre qualquer estrutura de rede, seja ela simples como uma ligação ponto-a-ponto ou uma rede de pacotes complexa. Como exemplo, pode-se empregar estruturas de rede como Ethernet, Token-Ring, FDDI, PPP, ATM, X.25, Frame-Relay, barramentos SCSI, enlaces de satélite, ligações telefônicas discadas e várias outras como meio de comunicação do protocolo TCP/IP.

A arquitetura TCP/IP está estruturada em camadas, conforme mostrado na Figura 2.1.

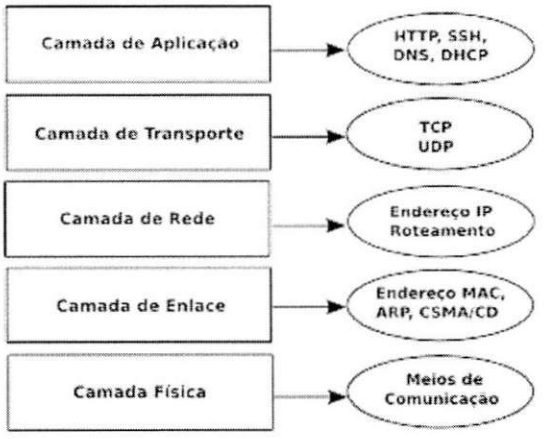

Figura 2.1: Modelo TCP/IP

2.2.1 Camada de Aplicação

A camada de aplicação reúne os protocolos que fornecem serviços de comunicação ao usuário.

Pode-se separar os protocolos de aplicação em protocolos de serviços básicos ou protocolos de serviços para o usuário:

- Protocolos de serviços básicos, que fornecem serviços para atender as próprias necessidades do sistema de comunicação TCP/IP: DNS, BOOTP, DHCP
- Protocolos de serviços para o usuário: FTP, HTTP, Telnet, SMTP, POP3, IMAP, TFTP, NFS, NIS, LPR, LPD, ICQ, RealAudio, Gopher, Archie, Finger, SNMP e outros.

2.2.2 Camada de Transporte

Esta camada reúne os protocolos que realizam as funções de transporte de dados fim-a-fim, ou seja, considerando apenas a origem e o destino da comunicação, sem se preocupar com os elementos intermediários. A camada de transporte possui dois protocolos que são o UDP (*User Datagram Protocol*) e TCP (*Transmission Control Protocol*).

O protocolo UDP realiza apenas a multiplexação para que várias aplicações possam acessar o sistema de comunicação de forma coerente.

O protocolo TCP realiza, além da multiplexação, uma série de funções para tornar a comunicação entre origem e destino mais confiável. São responsabilidades do protocolo TCP: o controle de fluxo, o controle de erro, a sequenciação e a multiplexação de mensagens.

2.2.3 Camada de Rede

Esta camada realiza a comunicação entre máquinas através do protocolo IP. Para identificar cada máquina e a própria rede onde estas estão situadas, é definido um identificador, chamado endereço IP, que é independente de outras formas de endereçamento que possam existir nos níveis inferiores. No caso de existir endereçamento nos níveis inferiores é realizado um mapeamento para possibilitar a conversão de um endereço IP em um endereço deste nível.

Os protocolos existentes nesta camada são:

Protocolo de transporte de dados: IP - Internet Protocol

Protocolo de controle e erro: ICMP - Internet Control Message Protocol

Protocolo de controle de grupo de endereços: IGMP - Internet Group Management Protocol

Protocolos de controle de informações de roteamento

Na camada de rede é realizado o roteamento, que consiste no transporte de mensagens entre redes e na decisão de qual rota uma mensagem deve seguir através da estrutura de rede para chegar ao destino.

Quando a máquina destino está na mesma rede que a máquina origem, o protocolo IP utiliza as camadas inferiores do protocolo TCP/IP para entregar a mensagem diretamente. Por outro lado, para enviar mensagem para máquinas situadas em redes distintas, ele utiliza a função de roteamento IP. Isto ocorre através do envio da mensagem para uma máquina que executa a função de roteador. Esta, por sua vez, repassa a mensagem para o destino ou a repassa para outros roteadores até chegar no destino.

Mais a frente esse protocolo e suas funcionalidades serão melhor explorados.

2.2.4 Camada de Enlace

A camada de rede é responsável pelo envio de datagramas construídos pela camada de enlace. Esta camada realiza também o mapeamento entre um endereço de identificação de nível de enlace para um endereço físico ou lógico do nível de Rede. A camada de enlace é independente do nível de Rede.

Alguns protocolos existentes nesta camada são:

Protocolos com estrutura de rede própria (X.25, Frame-Relay, ATM)

Protocolos de Enlace OSI (PPP, Ethernet, Token-Ring, FDDI, HDLC, SLIP, ...)

Protocolos de Nível Físico (V.24, X.21)

Protocolos de barramento de alta-velocidade (SCSI, HIPPI,...)

Protocolos de mapeamento de endereços (ARP - Address Resolution Protocol) - Este protocolo pode ser considerado também como parte da camada Inter-Rede.

Os protocolos deste nível possuem um esquema de identificação das máquinas interligadas. Por exemplo, cada máquina situada em uma rede Ethernet, Token-Ring ou FDDI possui um identificador único chamado endereço MAC ou endereço físico que permite distinguir uma máquina de outra, possibilitando o envio de mensagens específicas para cada uma delas. Tais rede são chamadas redes locais de computadores.

As redes ponto-a-ponto, formadas pela interligação entre duas máquinas não possuem, geralmente, um endereçamento de nível de rede (modelo TCP/IP), uma vez que não há necessidade de identificar várias estações.

2.3 Endereçamento IP e máscaras de sub-rede

Um endereço IP é composto de uma seqüência de 32 bits, divididos em 4 grupos de 8 bits cada. Numa rede TCP/IP, cada equipamento recebe um endereço IP único que o identifica na rede.

O endereço IP é dividido em duas partes. A primeira identifica a rede à qual o computador está conectado (necessário, pois numa rede TCP/IP pode-se ter várias redes conectadas entre si, como é o caso da Internet) e a segunda identifica o computador (chamado de host ou hospedeiro) dentro da rede.

Obrigatoriamente, os primeiros octetos servirão para identificar a rede e os últimos servirão para identificar o hospedeiro.

Uma característica do endereçamento IP é que nem todas as combinações de valores são permitidas. Alguns números são reservados e não podem ser usados numa rede. Os endereços IP inválidos são mostrados na tabela 2.1.

Endereço Inválido	Motivo
0.xxx.xxx.xxx	Nenhum endereço IP pode começar com zero, pois o identificador de rede 0 é utilizado para indicar que se está na mesma rede, a chamada rota padrão.
127.xxx.xxx.xxx	Nenhum endereço IP pode começar com o número 127, pois este número é reservado para testes internos, ou seja, são destinados à própria máquina que enviou o pacote. Se por exemplo a rede possuir um servidor de SMTP e um usuário configurar seu programa de e-mail para usar o servidor 127.0.0.1 ele acabará usando o próprio servidor instalado máquina
255.xxx.xxx.xxx xxx.255.255.255 xxx.xxx.255.255	Nenhum identificador de rede pode ser 255 e nenhum identificador de host pode ser composto apenas de endereços 255, seja qual for a classe do endereço. Outras combinações são permitidas, como em 65.34.255.197 (num endereço de classe A) ou em 165.32.255.78 (num endereço de classe B).
xxx.xxx.xxx.255 xxx.xxx.xxx.0	Nenhum endereço de classe C pode terminar com 0 ou com 255, pois visto anteriormente, um host não pode ser representado apenas por valores 0 ou 255. Os endereços xxx.255.255.255, xxx.xxx.255.255 e xxx.xxx.xxx.255 são sinais de broadcast que são destinados simultâneamente à todos os computadores da rede. Estes endereços são usados por exemplo numa rede onde existe um servidor DHCP, para que as estações possam receber seus endereços IP cada vez que se conectam à rede.

Tabela 2.1: Endereços IP inválidos

Caso não se deseje uma conexão da rede com a Internet, pode-se utilizar qualquer faixa de endereços IP válidos e tudo irá funcionar sem problemas. Mas, a partir do momento em que se resolve conectá-los à Web os endereços da rede poderão entrar em conflito com endereços já usados na Web.

Para resolver este problema, basta utilizar uma das faixas de endereços reservados. Estas faixas são reservadas justamente ao uso em redes internas, por isso não são roteadas na Internet.

Deve-se observar que usar uma destas faixas de endereços reservados não impede que as máquinas da rede possam acessar a Internet. Esse acesso pode ser feito, por exemplo, através de um servidor proxy.

O servidor proxy serve como um intermediário entre os terminais de acesso de uma rede e a Internet. Um servidor proxy pode ser usado com basicamente três objetivos:

1. Compartilhar a conexão com a Internet quando existe apenas um IP disponível (o proxy é o único realmente conectado à Web, os outros terminais acessam através dele);

2. Melhorar o desempenho do acesso através de um cache de páginas; o proxy armazena as páginas e arquivos mais acessados, quando alguém solicitar uma das páginas já armazenadas do cache, esta será automaticamente transmitida, sem necessidade de baixá-la novamente;

3. Bloquear acesso a determinadas páginas; como estas tem de passar pelo proxy antes de seremn acessadas nos terminais é fácil implantar uma lista de endereços ou palavras que devem ser bloqueadas.

Ao configurar o protocolo TPC/IP, seja qual for o sistema operacional usado, além do endereço IP é preciso informar também o parâmetro da máscara de sub-rede ou máscara de rede.

Ao contrário do endereço IP, que é formado por valores entre 0 e 255, a máscara de sub-rede tradicional é formada por apenas dois valores: 0 e 255, como em 255.255.0.0 ou 255.0.0.0. onde um valor 255 indica a parte endereço IP referente à rede, e um valor 0 indica a parte endereço IP referente ao host.

O número de redes interligando-se à Internet a partir de 1988 aumentou, causando o agravamento do problema de disponibilidade de endereços na Internet. Desta forma, buscou-se alternativas para aumentar o número de endereços de rede disponíveis sem afetar o funcionamento dos sistemas existentes. A melhor alternativa encontrada foi flexibilizar o conceito tradicional onde a divisão entre rede e host ocorre somente a cada 8 bits.

A máscara com novo formato é formada por 4 bytes (4 octetos) com uma sequência contínua de 1's, seguida de uma seqüência de 0's. A porção de bits em 1 identifica quais bits são utilizados para identificar a rede no endereço e a porção de bits em 0, identifica que bits do endereço identificam a estação.

Ou seja, a máscara pode ser compreendida também como um número inteiro que diz a quantidade de bits '1' utilizados. Uma máscara com valor 255.255.255.192, poderia ser representada como /26 (255.255.255.192/26). Este tipo de notação é empregada em protocolos de roteamento mais recentes. Ele indica que dos 32 bits que formam o endereço IP, 26 são inalterados, ou seja, pertencem ao endereço de rede. Por conseguinte, os primeiros 26 bits da máscara de rede possuem valor '1'.

2.4 DHCP

O DHCP, *Dynamic Host Configuration Protocol*, é um protocolo de serviço TCP/IP que oferece configuração dinâmica de terminais, com concessão de endereços IP de host e outros parâmetros de configuração para clientes de rede. O DHCP surgiu como padrão em Outubro de 1993. O RFC 2131 contém as especificações mais atuais (Março de 1997).

Resumidamente, o DHCP opera da seguinte forma:

- Um cliente envia um pacote *broadcast* (destinado a todas as máquinas) com um pedido DHCP, solicitando assim um endereço IP;

- Os servidores DHCP que capturarem este pacote irão responder (se o cliente se enquadrar numa série de critérios) com um pacote com configurações onde constará, pelo menos, um endereço IP, uma máscara de rede e outros dados opcionais, como o gateway, servidores de DNS, etc. O endereço IP é oferecido a um cliente é válido durante um certo período de tempo. Esse período é chamado de tempo de concessão e é um valor configuravél.

O DHCP usa um modelo cliente-servidor, no qual o servidor DHCP mantém o gerenciamento centralizado dos endereços IP usados na rede.

Três tipos de alocação de endereços IP são utilizados por esse protocolo:

- **Alocação manual** - onde existe uma tabela de associação entre o Endereço MAC do cliente (que será comparado através do pacote *broadcast* recebido) e o endereço IP a fornecer. Esta associação é feita manualmente pelo administrador de rede; por conseguinte, apenas os clientes cujos endereços MAC constam nesta lista poderão receber configurações desse servidor;

- **Alocação automática** - onde o cliente obtém um endereço de um espaço de endereços possíveis, especificado pelo administrador;

- **Alocação dinâmica** - o único método que dispõe a reutilização dinâmica dos endereços. O administrador disponibiliza um espaço de endereços possíveis, e cada cliente terá o software TCP/IP da sua interface de rede configurados para requisitar um endereço por DHCP assim que a máquina arranque. A alocação utiliza um mecanismo de aluguel do endereço, caracterizado por um tempo de vida. Após a máquina se desligar, o tempo de vida naturalmente irá expirar, e da próxima vez que o cliente se ligue, o endereço provavelmente será outro.

Algumas implementações do software servidor de DHCP permitem ainda a atualização dinâmica dos servidores de DNS para que cada cliente disponha também de um DNS.

2.5 Endereço MAC

O endereço MAC (*Media Access Control*) é o endereço físico do terminal, ou melhor, da interface de rede (ou adaptador) de cada terminal. É um endereço de 48 bits, representado em hexadecimal. O protocolo é responsável pelo controle de acesso de cada estação à rede Ethernet. Desses 48 bits, os primeiros seis bits são administrados pelo consórcio IEEE e

identificam o fabricante ou fornecedor da placa de rede; os seis últimos são uma identificação da placa.

Não existem duas placas com o mesmo endereço MAC, ou seja, este endereço é único para cada placa de rede em cada computador. Os endereços MAC geralmente são gravados na memória ROM e copiados para a memória de RAM quando a placa de rede é iniciada.

Antes de sair da fábrica, o fabricante do hardware atribui um endereço físico a cada placa de rede. Esse endereço é programado na placa de rede. Como o endereço MAC está localizado na placa de rede, se a placa de rede fosse trocada em um computador, o endereço físico da estação mudaria para o novo endereço MAC.

Em uma rede Ethernet, quando um dispositivo quer enviar dados para outro dispositivo, ele pode abrir um caminho de comunicação com o outro dispositivo usando o seu endereço MAC. Quando uma origem envia dados em uma rede, os dados carregam o endereço MAC do destino pretendido. Como esses dados trafegam pelos meios da rede, a placa de rede em cada dispositivo na rede verifica se o seu endereço MAC corresponde ao endereço de destino físico carregado pelo pacote de dados. Se não corresponder, a placa de rede descarta o pacote de dados. Se não houver correspondência, a placa de rede ignora o pacote de dados e permite que ele continue sua viajem pela rede até o terminal seguinte seguinte.

O protocolo ARP (*Address Resolution Protocol*) é um protocolo usado para encontrar um endereço MAC a partir do endereço IP. O host origem ou emissor difunde em *broadcast* um pacote ARP contendo o endereço IP de outro host e espera uma resposta com um endereço MAC respectivo. Cada máquina mantém uma tabela de resolução em cache para reduzir a latência e carga na rede. O ARP permite que o endereço IP seja independente do endereço MAC, mas funciona somente se todos os hosts o suportarem.

Capítulo 3

EXPERIMENTOS PROPOSTOS

Nesse trabalho são propostos dois experimentos: um envolvendo um servidor de HTTP, com uma ligeira introdução à HTML e outro envolvendo endereçamento IP e MAC.

Em ambos os experimentos, porém, o procedimento de manuseio da placa PME-10 (ver anexo 1 para uma descrição da placa) bem como a alteração da pilha e sua gravação no microcontrolador são os mesmos; por esse motivo serão descritos primeiro.

3.1 Conectando a placa PME-10

A placa PME-10 fornecida com o kit utilizado foi projetada para demonstrar as possibilidades de uso de rede Ethernet e/ou Internet com o microcontrolador PIC 18F8720[1] da Microchip®.

Sempre que um novo hardware ou software é adicionado a um sistema em rede, é aconselhável criar uma rede de testes separada da rede local (LAN - *Local Area Network*) de uso cotidiano. Isto permite testar o novo sistema em um ambiente controlado e minimizar as possibilidades de interferência na rede causadas por esse novo sistema. As maiores fontes de interferência poderiam ser:

- **Endereçamento IP-** Cada dispositivo na rede deve ter um único endereço IP. Se o protocolo DHCP (Dynamic Host Configuration Protocol) for usado, o novo sistema inserido na rede automaticamente é configurado com um endereço IP. Caso contrário, um endereço fixo deve ser fornecido, tomando-se o cuidado de não inserir um endereço já ativo na rede, o que ocasionaria um conflito de IP's.

- **Tráfego -** Enquanto o controlador Ethernet filtra mensagens não endereçadas ao novo sistema, uma rede Ethernet com muitas mensagens *broadcast* pode sobrecarregá-lo, aumentando o número de colisões.

[1]Na verdade outros microncontroladores da família PIC 18X, como o PIC 18F8621 podem ser utilizados

► **Segurança dos Dados -** Apesar de não ser comum o comprometimento da integridade dos dados ou privacidade de informações da rede pela simples adição de um dispositivo simples como é a placa PME-10, é sempre prudente executar testes intensivamente com novos sistemas antes de adicioná-los a uma rede.

A placa PME-10 esta habilitada a trabalhar tanto com DHCP quanto com endereço IP estático. Porém muitas razões podem levar o usuário a usar IP fixo, dentre as quais a falta de um servidor de DHCP se apresenta com uma das principais. Além do mais, para os propósitos do laboratório, o uso de IP fixo se torna mais interessante, tendo em vista a facilidade de se manipular o mesmo.

Sendo assim, a placa pode se comunicar diretamente com um computador qualquer que esteja conectado e devidamente configurado na mesme rede que ela. Para isso, basta que esse computador possua uma interface *Ethernet* com conector RJ-45, possua um sistema operacional com pilha TCP/IP[2] e possua um navegador WEB que suporte o protocolo HTTP 1.0 ou superior.

Nesta configuração deve-se usar um cabo *crossover* para interligação da placa PME-10 com um computador pessoal numa ligação ponto-a-ponto ou um cabo direto para ligações em redes com distribuição. Essa é a interface para acessar páginas Web e outras aplicações que possam estar gravadas na memória *flash* do microcontrolador PIC18F8720 presente na placa.

A placa PME-10 vem configurada de fábrica com endereço IP 10.10.5.15 e máscara de rede 255.255.255.0. Nesta configuração o protocolo DHCP não está habilitado.

A Figura 3.1 abaixo apresenta o sistema montado:

De posse dessas informações, os seguintes passos devem ser seguidos para se conectar a placa PME-10 para a realização de um primeiro teste com segurança:

1. Energizar a placa PME-10 conectando-a a placa de expansão, FD-01, responsável pela alimentação da PME-10 e por agregar algumas funções demonstrativas, como os LEDs e potenciômetros descritos mais à frente, alimentada por um transformador de 9V DC, conforme mostrado na Figura 3.2 abaixo:

2. Conectar o cabo de rede entre a PME-10 e a placa de rede do computador que contém o navegador Web.

3. Verificar se o LED D3 (vermelho) na PME-10 pisca a cada intervalo de 1 segundo. Esse LED indica o perfeito funcionamento da pilha TCP/IP escrita no memória do microcontrolador. Caso não pisque, é sinal de que a pilha não está sendo acessada por algum motivo.

[2]Praticamente todos os sistemas operacionais atuais suportam TCP/IP.

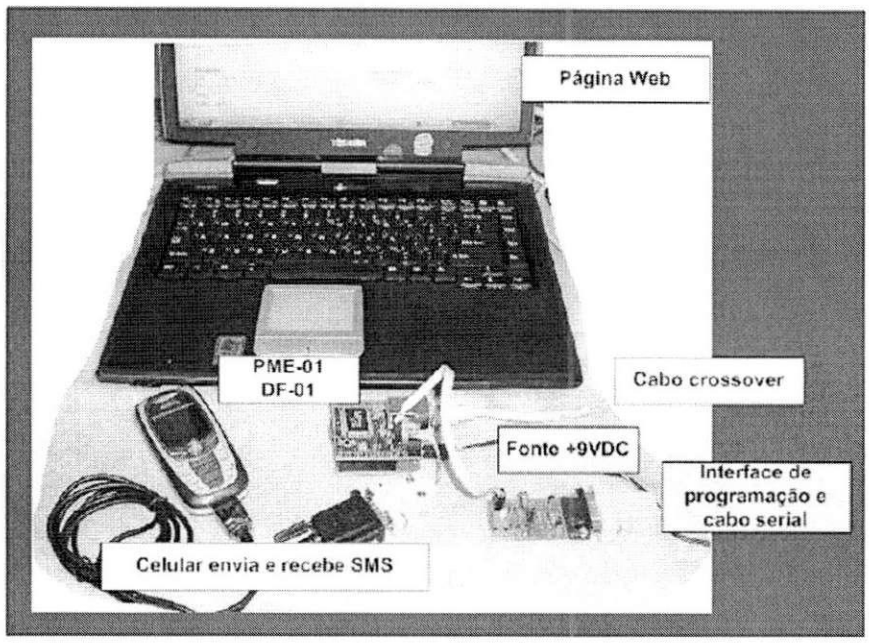

Figura 3.1: Conexão da placa PME-10 com um computador

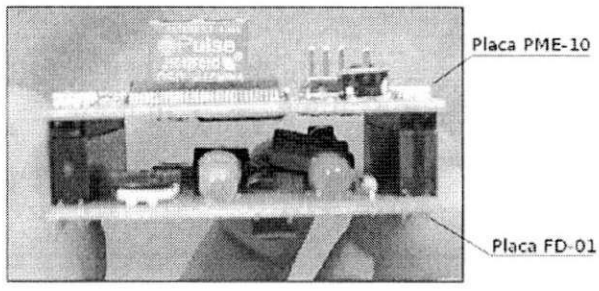

Figura 3.2: Conexão da placa PME-10 com a placa de alimentação FD-01

4. Configurar o computador do seguinte modo:

   (a) Endereço IP: 10.10.5.X (aonde X deve ser diferente de 15)

   (b) Máscara de rede: 255.255.255.0

5. No computador, usando o navegador Web, digitar o seguinte endereço http://10.10.5.15

6. A seguinte página Web (Figura 3.3) deverá ser carregada na tela de computador. Este é um teste padrão, mas pode funcionar como um primeiro exercício.

   O último item pode ser utilizado como um exercício demonstrativo para o laboratório,

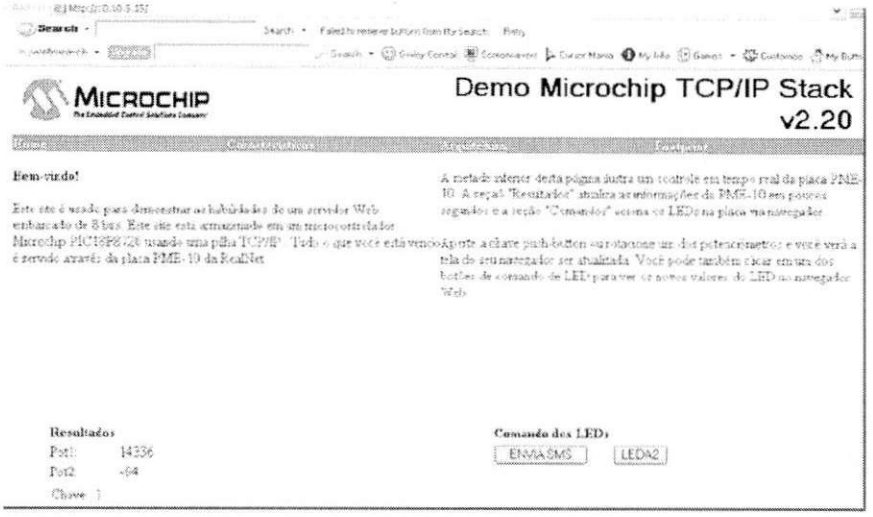

Figura 3.3: Página WEB da PME-10

tendo que o mesmo mostra de forma satisfatória a forma como um computador se conecta a uma página Web na Internet.

No caso desse exercício demonstrativo, os seguintes recursos apresentados pelas placas PME-10 e FD-01 podem ser verificados:

- Variando-se o potenciômetro P1 pode-se verificar que o valor do Pot1 também varia na página do navegador Web.
- No navegador Web, ao se clicar no botão LED1 verifica-se que o estado (ligado e desligado) do LED1 (azul) varia simultâneamente na placa FD-01.
- Apertando a tecla `push-button` na FD-01, verifica-se que o estado da Chave (ligada ou desligada) varia na página do navegador Web.

Após essa apresentação de como conectar a placa de uma forma segura e utilizá-la num rápida introdução ao laboratório de redes, com a pequena demonstração descrita acima, pode-se agora descrever como alterar as configurações da placa. O procedimento de alteração dos arquivos e escrita no microcontrolador mostrado a seguir será utilizado em todos os experimentos utilizando a placa.

3.2 Escrevendo na Placa PME-10

Para se realizar qualquer alteração em qualquer arquivo de projeto, ou mesmo criar novos projetos com a placa PME-10 deve-se utilizar o cabo de programação, ligado a interface serial do computador, que acompanha o kit de desenvolvimento. Os seguintes procedimentos devem ser executados para a realização dessas alterações.

3.2.1 Configurando e Utilizando o MPLAB IDE

A seqüência de programação do microcontrolador é a de se escrever um arquivo hexadecimal e depois inserí-lo na memória do mesmo. Para a alteração dos arquivos fonte em C e posterior criação dos arquivos hexadecimais faz-se uso do compilador C MCC18 (versão estudante) e do ambiente de desenvolvimento MPLAB IDE, ambos propriedades da Microchip®. Eles podem ser encontrados no site www.microchip.com e são distribuídos gratuitamente.

A seguinte seqüência de passos configuram ambos o compilador C MCC18 (versão estudante) e o ambiente de desenvolvimento MPLAB IDE e devem ser seguidas.

1. O primeiro passo é instalar o compilador MCC18 e o ambiente de desenvolvimento MPLAB IDE. Uma sugestão é a de manter os nomes dos diretórios sugeridos durante o processo de instalação, caso contrário problemas com os caminhos dos arquivos poderão ser encontrados.

2. O passo seguinte é criar um projeto e uma área de trabalho no sistema de ambiente integrado MPLAB IDE. Um projeto contém os arquivos necessários para construir uma aplicação (códigos fontes, etc.). Uma área de trabalho contém um ou mais projetos e informações dos dispositivos selecionados, ferramentas de depuração, localização das janelas abertas e outras configurações da IDE. O MPLAB IDE contém um assitente de projetos (`Project Wizard`) para ajudar a criar novos projetos. Neste trabalho, serão utilizados projetos já prontos, que acompanham o kit de desenvolvimento.

3. Na seqüência, fazer uma verificação da seleção de linguagens no aplicativo MPLAB IDE. Na barra de menu selecionar `Project\Set Language Tool Locations`. Na janela `Set Language Tool Locations` selecionar opção `Microchip C18 Toolsuite` (clique em +). Clicar no + de `Executables`. Clicar no + de `Default Search Paths & Directories`. Verificar então os seguintes caminhos[3]:

   ► Executables:

   - MPASM Assembler (mpasmwin.exe)
     `c:\Arquivos de programas\MPLAB IDE\MCHIP_Tools\mpasmwin.exe`
   - MPLAB C18 C Compiler (mcc18.exe)
     `c:\mcc18\bin\c18demo.exe`
   - MPLINK Object Linker (mplink.exe)
     `c:\mcc18\bin\mplink.exe`

[3]Logicamente, esses caminhos variarão a depender do diretório onde o MCC 18 e o MPLAB foram instalados.

► Default Search Path & Directories:

- Output Directory, $(BINDIR)
- Intermediate s Directory, $(TMPDIR)
- Include Search Path, $(INCDIR)
- Library Search path, $(LIBDIR)
  `c:\mcc18\lib`
- Linker_Script Search Path,$(LKDIR)
  `c:\mcc18\lib`

4. No MPLAB IDE, selecionar na barra de menu `Project\Open`. Selecionar o projeto mpnicpg.pjt da pasta `PILHAS\PME10\v1.10\Source` que acompanha o kit[4]. Essa pasta é a localidade dos arquivos e projetos que estão originalmente carregados na PME-10 e que serão utilizados nesses experimentos.

5. No MPLAB IDE, selecionar na barra de menu `View\Project`. Será mostrada uma árvore com todos os arquivos diretamente ligados ao projeto aberto, com os arquivos fonte (`Source Files`), arquivos de cabeçalho (`Header Files`), arquivos objeto (`Object Files`), arquivos de bibliotecas (`Library Files`), *scripts* de ligação (`Linker Scrips`) entre outros. Em `Linker Script` verificar a opção 18f8720.lkr. Esta é óbviamente a opção referente ao PIC 18f8720. Caso outra opção ou nenhuma esteja visível, selecionar esta clicando com o botão direito sobre a pasta e escolhendo o arquivo contido na pasta `\mcc18\lkr` do compilador MCC 18.

6. No menu `Project` selecionar `Build All`. O projeto será compilado e *linkado*. Um novo arquivo mpnicpg.hex é então gerado e salvo na mesma pasta onde está localizado o projeto, estando pronto para ser transferido para a PME-10.

3.2.2 Instalação e configuração do Programa IC-Prog 1.05D

Como dito anteriormente, o procedimento acima descrito serve para criar um arquivo *.hex que traz todas as modificações a serem implementadas no microcontrolador. A inserção desse arquivo na PME-10 se faz por intermédio de um outro programa, o IC-Prog.

O programa IC-Prog é um programador de protótipos desenvolvido por Bonny Gizen (www.ic-prog.com). Para instalar e configurar o IC-Prog os seguintes passos devem ser seguidos:

1. Acessar a pasta ICPROG que acompanha o kit de desenvolvimento.

[4]O kit é acompanhado de um cd com todos os programas e tutoriais utilizados, bem como os projetos descritos aqui.

2. Na barra de menu selecionar `Settings\Options`. Ir para a aba `Language` e selecionar a linguagem `Portuguese`.

3. Executar o programa IC-Prog a partir do disco rígido do computador. Selecionar a serial na qual será colocado o cabo de programação na janela `Configurações/Hardware`. O programador selecionado deve ser `JDM Programmer`. Caso esteja sendo usado o Windows XP ou NT selecionar `Windows API` ou `Direct IO`. A segunda evita mais possíveis erros de gravação. A opção `Retardar I/O` pode estar com o valor mínimo. Caso a programação da placa não tenha sido efetuada corretamente selecionar um valor progressivamente maior da opção `Retardar I/O` até que a programação ocorra sem erros.

4. Na barra de menu selecionar `Configuração\Opções`. Clicar na aba `Programando`. A opção "Verifique depois da programação"não deve estar selecionada.

5. Na barra de menu selecionar `Configuração\Opções`. Clicar na aba `Diversos`. Em "Processo Prioritário"selecionar "Alto".

6. Na barra de menus, selecionar `Configuração\Dispositivo \Microchip PIC\PIC18F8720`.

7. Na barra de menus, selecionar `Arquivo\Abrir` e carregar o arquivo mpniccpg.hex que se encontra na pasta `PILHAS\PME10\v1.10\Source`, a mesma onde o projeto utilizado se encontra.

8. No programa IC-Prog, os registradores de configuração devem estar carregados com os seguintes valores:

   Config1 - `2200`

   Config2 - `0004`

   Config3 - `0183`

   Config4 - `0081`

   Config5 - `C0FF`

   Config6 - `E0FF`

   Config7 - `40FF`

   Estes valores são obtidos da leitura do arquivo *.hex e não inseridos pelo usuário. Uma das formas de verificar se o arquivo *.hex foi corretamente executado ou está sendo corretamente carregado é verificar a alteração nesses valores.

Feitas essas configurações do IC-Prog, o procedimento para a inserção do arquivo *.hex é simples e feito da seguinte forma:

1. Após alimentar devidamente a placa PME-10, conectar o cabo de programação na saída serial do computador e na placa de programação da PME-10.

2. Na interface de programação pressionar ambas as chaves `RESET` e `CK/ADD` na placa de programação de modo a habilitar a programação do microcontrolador da PME-10. (Com relação a posição das chaves, a posição "para baixo"indica a opção de programação enquanto que a posição "para cima"indica a posição de operação, ou seja, de não programação). Durante a gravação, observar que a pilha TCP/IP gravada na placa PME-10 é desconectada, o que é mostrado pelo fato do LED D3 parar de piscar. Após a gravação, o LED volta a piscar indicando que a conexão com a pilha está novamente ativa.

3. Na barra de menu selecione `Comando` e depois `Programar Tudo`. O LED da interface de programação deverá ficar ligado. Um observação cabe nesse ponto: a interface de programação é de custo muito baixo e por conta disso não é capaz de verificar claramente se a programação foi feita de forma correta. Uma maneira de se verificar isso é verificar o led D3. Caso o led D3 não fique piscando imediatamente após voltar a chave de programação para a posição de operação, indica que o programa não foi gravado corretamente.

4. A finalização da programação deve ser aguardada. Após terminar, retornar as chaves para a posição de operação.

Nesse ponto, mais uma observação. Algumas placas podem vir com o microcontrolador PIC18F8720 ou o PIC18F8621. A descrição acima refere-se ao primeiro. No caso de ser usado o segundo microcontrolador, as principais diferenças na programação seriam:

- No MPLAB IDE:

  1. `Header Files` no menu `View\Project`: p18f8621.h
  2. `Linker Scripts` no menu `View\Project`: 18f8621.lkr

- No IC-Prog:

  1. O dispositivo selecionado passa a ser o PIC18F8620.

3.3 Experimento 1: Camada de Aplicação

Esse experimento tem como objetivo principal estudar como opera a camada de aplicação através do exemplo de um servidor Web, ou servidor HTTP. Para isso será proposto a realização de duas atividades.

3.3.1 Atividades Propostas

A primeira atividade proposta para o experimento 1 é a verificação da página existente no microcontrolador da placa PME-10. Nela o aluno pode visualizar o funcionamento de um servidor Web, simplesmente operando as funções descritas na seção 3.1.1.

Uma segunda atividade seria a manipulação do código HTML dessa página Web. O procedimento seria o de modificar o código em um computador, compilá-lo em um arquivo *.hex, usando o MPLAB IDE e finalmente escrever o arquivo no microcontrolador utilizando o IC-Prog, segundo descrito na seção anterior.

A pasta `PILHAS\PME10\v1.10\Source\WebPages` que acompanha o kit contém os arquivos *.html usados na página mostrada na execução da primeira atividade sugerida para esse experimento. O arquivo `Index.html` é o arquivo principal da página. Estão ligados à ele todos os outros que a compõe, como `Header.html`, `Home.html`, `Status.cgi` e `Commands.cgi`.

A seqüência a ser feita para a segunda atividade seria:

1. Usando um editor de textos HTML (no caso de não se possuir um editor específico, pode ser utilizado o próprio bloco de notas), realizar alterações no código HTML das páginas, de forma a alterar o *layout* da página. Para isso, uma pequena introdução em HTML pode ser dada ao aluno, para que o mesmo tenha condições de modificar estas páginas ou criar novas páginas Web. Aplicativos como Microsoft® FrontPage podem ser sugeridos como interface para modificação destas páginas, caso não seja possível ou desejável a introdução à HTML.

2. Na pasta `PILHAS\PME10\v1.10\Source` executar o arquivo `mpfs` através do prompt do DOS. Ao executar esse arquivo, um menu como o mostrado na Figura 3.4 aparecerá. Ao executar o comando mostrado no exemplo 1 do menu da citada figura (no caso da localização dessa instalação, o exemplo é `mpfs c:\webpages mpfsimg.c /c`; conforme mude a localização desse arquivo, esse comando mudará) um novo arquivo mpfsimg.c contendo as imagens das novas páginas geradas será criado.

3. No programa MPLAB IDE, selecionar a opção `Project\Build All` para gerar um novo arquivo mpniccpg.hex usando o novo mpfsimg.c.

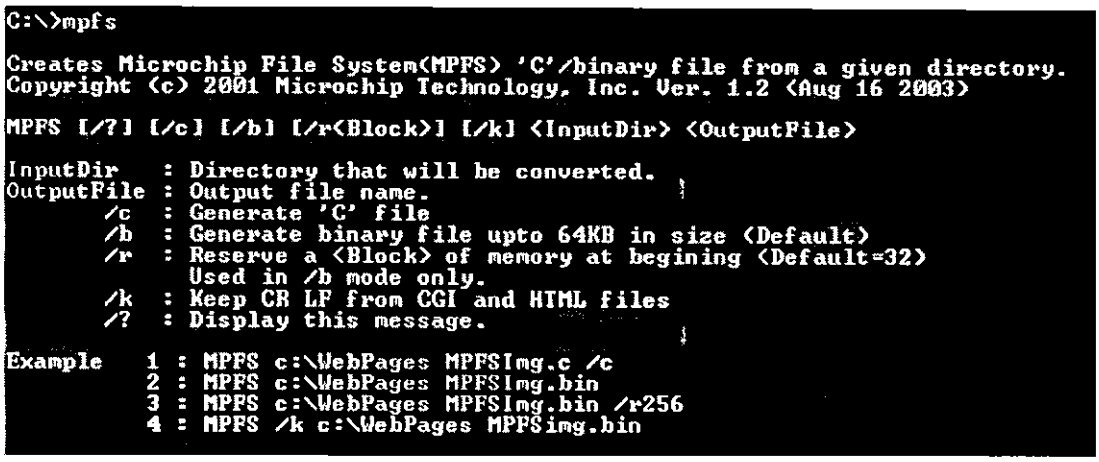

Figura 3.4: Tela inicial do programa mpfs

4. Utilizar o programa IC-Prog e o cabo de programação para gravar o novo arquivo mpniccpg.hex que contém as novas páginas Web, fazendo uso dos procedimentos descritos na seção 3.2.2.

3.3.2 O Servidor HTTP Microchip: uma visão geral

O Servidor HTTP da Microchip utilizado nesse experimento é implementado como uma tarefa cooperativa que co-existe com a pilha Microchip TCP/IP e a aplicação principal do usuário. O Servidor é implementado pelo arquivo "HTTP.C", com a aplicação do usuário implementando duas chamadas de funções. O arquivo de demonstração "Websrvr.c"é usado como modelo de aplicação para criação das interfaces apropriadas.

O Servidor HTTP da Microchip não implementa todas as funcionalidades dos documentos RFC, mas possui características mínimas para implementação em sistemas embarcados. O usuário pode facilmente adicionar novas funcionalidades como desejar.

Qualquer página Web a ser armazenada na PME-10 dever ser convertida para o formato MPFS.

O Servidor HTTP usa o arquivo "index.htm"como a página Web inicial a ser chamada pelo navegador Web, através do comando http://X.X.X.X (aonde X.X.X.X é o endereço IP da placa PME-10, seja estático ou dinâmico). Então todas as aplicações devem incluir um arquivo "index.htm". Se necessário, o nome desta página inicial pode ser trocada modificando a definição do compilador `HTTP_DEFAULT_FILE_STRING` no arquivo "http.c". É muito importante que nenhuma página Web contenha algum dos seguintes caracteres:

- ' ou "

- < ou >

- #
- %
- {, },[ou]
- |
- \
- ~
- ^

Se uma das páginas Web contiver um destes caracteres, a correspondente página Web tornar-se-á inacessível. Nenhum aviso de atenção será dado.

O Servidor HTTP mantém uma lista de tipos de arquivos que ele suporta. Esta informação é usada para avisar um navegador como interpretar um arquivo em particular, baseado na extensão do arquivo. Por padrão, o Servidor HTTP Microchip suporta arquivos ".txt", ". htm", ".gif", ".cgi", ".jpg", ".cla"e ".wav". Se uma aplicação usa tipos de arquivos que não estão incluídos nesta lista, o usuário pode modificar a tabela "httpFiles"através da correspondente enumeração "httpContents"no arquivo "http.c".

O servidor HTTP pode alterar páginas dinamicamente e substituir informações em tempo real, como *status* de entradas e saídas do microcontrolador. Para incorporar estas informações em tempo real, o correspondente arquivo CGI (*.cgi) deve conter uma string '%xx', aonde o caracter '%' serve como código de controle e 'xx' representa o identificador da variável com dois dígitos.

```
<html>
<meta http-equiv="refresh" content="3">
<body>
<table>
    <tr>
        <td><b>Resultados</b></td>
    </tr>
    <tr>
        <td>Pot1:</td>
        <td>%02</td>
    </tr>
    <tr>
        <td>Pot2:</td>
        <td>%03</td>
    </tr>
```

O identificador das variáveis deve estar entre 00-99. Quando o Servidor HTTP encontra %xx, ele remove o caracter % e chama a função `HTTPGetVar`. Se uma página requer o caracter '%' ele deve ser precedido por outro caracter '%'. Por exemplo, para mostrar "23%"em uma página, deve ser escrito "23%%".

A função `HTTPGetVar` tem a seguinte sintaxe:

```
HTTPGetVar(Byte var, Word ref, Byte *val)
```

onde:

`var`: ENTRADA . Identificador da variável cujo valor é para ser retornado.

`ref`: ENTRADA. Usado para uma única transferência ou transferência de vetores.

`val`: SAÍDA. Byte a ser transferido.

Valor de retorno. Se o valor retornado for diferente de `HTTP_END_OF_VAR`, o servidor HTTP chamará a função outra vez. Se o valor retornado for `HTTP_END_OF_VAR` o servidor HTTP não mais chamará a função e assume que a variável foi totalmente transferida.

No exemplo a seguir, o parâmetro `val` representa o estado da porta RB5. deve ser notado também que o retorno `HTTP_END_OF_VAR` indica que a variável tem comprimento de apenas um byte.

```
WORD HTTPGetVar(BYTE var, WORD ref, BYTE *val)
{
    // identificação da variável
    // é referente a RB5
    if ( var == 4 )
    {
//retorna '1' se RB5 estiver em HIGH,
// ou '0' se estiver em LOW
 if ( PORTBbits.RB5 )
                    *val = '1';
            else
                    *val = 0;
            // Reporta ao HTTP que este foi o último
     // byte do valor da variável
            return HTTP_END_OF_VAR;
    }
    // Checa outras variáveis
```

No segundo exemplo a seguir, o número serial do Servidor Web é mostrado.

```
WORD HTTPGetVar(BYTE var, WORD ref, BYTE *val)
{
    // identificação da variável
    // é referente a RB5
    // Se sim, segue o mesmo do Exemplo 1
...
...
    // é referente a variável do número serial
    if ( var == 5)
    {
            // o número serial acaba com um caracter NULL
            // sendo a primeira chamada
            if (ref == HTTP_START_OF_VAR)
            {
                    // Esta é a primeira chamada.
// Inicializa índice para n. série
                    ref = (BYTE)0;
            }
            // Agora acessamos byte no índice corrente
//e salvamos no buffer
            *val = SerailNumberStr[(BYTE)ref];
            // É final da string
            if (*val=='\0')
            {

//A string está sendo transferida.
//Retornamos com HTTP_END_OF_VAR para notificar o
//servidor HTTP que nós estamos finalizando a transferência
//o valor

             return HTTP_END_OF_VAR;
            }

// Ou caso contrário, incrementa o índice do vetor
//e retorna para o
// servidor HTTP
            (BYTE)ref++;
```

```
            // desde que não é o fim da string retorna ref
            return ref;
    else
    // checa por outras variáveis
```

A página "status.cgi"sendo servida pelo servidor HTTP contém a seguinte linha HTML:

```
....
<td>Serial Number=%05</td>
....
```

O servidor HTTP ao processar este arquivo e encontrando a string '%05' faz uma chamada `HTTPGetVar(4, HTTP_START_OF_VAR, &value)`. A aplicação principal implementa `HTTPGetVar` como segue no segundo exemplo.

Os Exemplos 1 e 2 mostram em um navegador Web variáveis que se encontram na memória do microcontrolador. O Exemplo 3 a seguir mostra como alterar valores em um navegador Web no cliente e transferi-los para a memória do microcontrolador.

No navegador Web a seguinte página, mostrada na figura 3.5 permite a alteração da variável "Nível de Potência", "Limite de Potência Inferior"e "Limite de Potência Superior".

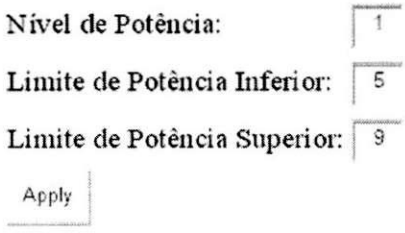

Figura 3.5: Página Web permitindo alteração de variáveis

E o código da página Web que permite alteração de variáveis (`Commands.cgi`) é o seguinte:

```
<body><center>
<FORM METHOD=Get action=Power.cgi>
<table>
<tr><td>Nível de Potência:</td>
<td><input type=text size=2 maxlength=1 name=P value=%07></td></tr>
<tr><td>Limite de Potência Inferior:</td>
<td><input type=text size=2 maxlength=1 name=L value=%08></td></tr>
```

```
<tr><td>Limite de Potência Superior:</td>
<td><input type=text size=2 maxlength=1 name=H value=%09></td></tr>
<tr><td><input type=submit name=B value=Apply></td></tr>
</table>
</body>
```

Esta página mostra uma tabela com o nome das variáveis na primeira coluna e caixa de textos para entrada de valores na segunda coluna. A primeira linha, primeira coluna contém o nome da variável "Nível de Potência"; a segunda coluna é uma caixa de texto para mostrar e modificar o nível de potência. A última linha contém um botão "Apply". Com esta página o usuário tem a habilidade de modificar o nível de potência na caixa de texto e enviar este valor ao microcontrolador na PME-10.

Assumindo que o usuário entre com os valores '5','1' e '9' respectivamente nas caixas de texto do nível de potência, limite inferior de potência e limite superior de potência,e em seguida clicando no botão "Apply", o navegador criará uma requisição HTTP com uma string "`Power.cgi?P=5&L=1&H=9`"e enviará ao servidor HTTP. O servidor chama a função `HTTPExecCmd` com os seguintes parâmetros:

```
argv[0]="Power.cgi",argv[1]="P",argv[2]="5",argv[3]="L",
argv[4]="1",argv[5]="H", argv[6]="9" argc=7
```

A aplicação principal deverá implementar a função`HTTPExecCmd` como segue:

```
void HTTPExecCmd(BYTE *argv, BYTE argc)
{
      BYTE i;
      // Varre todos os parâmetros
      for (i=1; i < argc;i++)
      {
            // Identifica parâmetros
            if ( argv[i][0] == 'P')
// É nível de potência?
            {
                  PowerLevel = atoi(argv[++i]);
            }
            else if ( argv[i][0] == 'L' )
// É limite inferior de potência?
                  LowPowerSetting = atoi (argv[++i]);
            else if (argv[i][0] == 'H')
 // É limite superior de potência?
```

```
                    HighPowerSeting = atoi (argv[++i]);
        }
        // se outra página é para ser mostrada
        //como resultado deste comando
        // copie o nome em maiúsculo para argv[0]
        // strcpy(argv[0], "RESULTS.CGI");
}
```

Neste exemplo, o número total de argumentos excede o padrão de 5. Então o valor de `MAX_HTTP_ARGS` (localizado em "http.c") pode ser modificado para pelo menos 7.

3.4 Experimento 2: Camada de Rede e de Enlace

O segundo experimento tem como objetivo ilustrar a teoria acerca das camadas de rede e de enlace, em especial a questão da atribuição de endereços IP e o endereço MAC para as placas. Cada placa possui um endereço MAC próprio adquirido junto ao IEEE e seu endereço IP pode ser facilmente modificado.

3.4.1 Alterando o Endereço IP e Máscara de Rede

O endereço IP e a máscara de rede podem ser alterados pela modificação do arquivo de `StackTsk.h`, que pode ser aberto no MPLAB IDE usando o comando da barra de menu `File\Open`. As definições abaixo contêm o endereço IP e a máscara de rede:

```
#define MY_DEFAULT_IP_ADDR_BYTE1 (10)
#define MY_DEFAULT_IP_ADDR_BYTE2 (10)
#define MY_DEFAULT_IP_ADDR_BYTE3 (5)
#define MY_DEFAULT_IP_ADDR_BYTE4 (15)
Obs: significa endereço IP 10.10.5.15

#define MY_DEFAULT_MASK_BYTE (0xff)
#define MY_DEFAULT_MASK_BYTE (0xff)
#define MY_DEFAULT_MASK_BYTE (0xff)
#define MY_DEFAULT_MASK_BYTE (0x00)
Obs: significa máscara 255.255.255.0

#define MY_DEFAULT_MAC_BYTE1 (0x00)
#define MY_DEFAULT_MAC_BYTE2 (0x16)
```

```
#define MY_DEFAULT_MAC_BYTE3 (0xa9)
#define MY_DEFAULT_MAC_BYTE4 (0x00)
#define MY_DEFAULT_MAC_BYTE5 (0x00)
#define MY_DEFAULT_MAC_BYTE6 (0x00)
Significa endereço MAC: 00.16.a9.00.00.00
```

Feitas as modificações necessárias, deve-se recompilar o projeto. O novo arquivo `mpnicpg.hex` gerado conterá as modificações que efetuadas. Esse arquivo deve então ser inserido no microcontrolador fazendo uso dos procedimentos descritos anteriormente com o programa ICProg.

3.4.2 Atividades Propostas

Para esse segundo experimento, também duas atividades podem ser propostas. A primeira é visualizar a placa numa pequena rede, com uma ou duas máquinas, sem conexão com a Internet, de forma a se observar a questão de inserção de endereços IP, choque de endereços, bem como a tabela ARP, que faz a tradução entre endereços IP e MAC.

Para isso pode-se fazer uso de uma série de comandos de redes que facilitam a visualização de endereços IP, MAC, tabelas ARP, rotas, entre outras coisas. Dentre os principais comandos, merecem destaque:

- `winipcfg` - No geral, mostra várias informações sobre uma rede, englobando placas de rede, configurações de IP's, servidores DNS, nome de hospedeiros, endereços MAC, etc. É basicamente uma versão gráfica do `ipconfig`.

- `ipconfig` - Fornece informações completas sobre os números IP's fornecidos a(s) placas de rede, por *DialUp* e por placa de comunicação. Mostra também configurações do protocolo pppoa. É a versão em modo texto do `winipcfg`. A sintaxe geralmente usada é `ipconfig/all`, onde todas as informações de todos os hospedeiros ligados à rede são mostrados.

- `netstat` - Mostra conexões de rede, tabela de roteamento, estatísticas de interfaces, conexões mascaradas, e mensagens. A sintaxe básica é:

  ```
  netstat [opções]
  ```

  onde as opções podem ser:

  `-i [interface]`: mostra estatísticas da interface [interface].

  `-M, --masquerade`: se especificado, também lista conexões mascaradas.

`-n, --numeric`: usa endereços numéricos ao invés de tentar resolver nomes de hospedeiros, usuários e portas.

`-c, --continuos`: mostra a listagem a cada segundo até que a seqüência CTRL+C seja pressionada.

Se não for especificada nenhuma opção, os detalhes das conexões atuais serão mostrados.

- `ping` - Interroga um dispositivo de rede numa rede TCP/IP. Funciona basicamente com o propósito de descobrir se um dado dispositivo se encontra na rede, através do envio e recebimento de um pacote de teste. A sintaxe básica é:

```
ping <endereço IP ou nome do hospedeiro> [opções]
```

onde a opção principal é:

`t` interroga uma máquina por um período indefinido ou, até que se pressione a tecla "control + c".

Exemplos:

```
ping 192.168.0.1 -t
ping www.terra.com.br
ping phr34k3r
```

- `tracert/traceroute` - Mostra o caminho percorrido por um pacote desde sua origem até chegar ao seu destino. Este comando mostra na tela o caminho percorrido entre os gateways da rede e o tempo gasto de retransmissão. Este comando é útil para encontrar computadores defeituosos na rede caso o pacote não esteja chegando ao seu destino. Sua sintaxe pode ser diferente para sistemas operacionais linux ou windows.

No linux, a sintaxe é:

```
traceroute [opções] [host/IP de destino]
```

onde:

`host/IP destino`: é o endereço para onde o pacote será enviado (por exemplo, http://forum.plugmasters.com.br). Caso o tamanho do pacote não seja especificado, é enviado um pacote de 38 bytes.

e as opções são:

`-l`: mostra o tempo de vida do pacote (ttl *time to live*)

`-m [num]`: ajusta a quantidade máximas de ttl dos pacotes. O padrão é 30.

`-n`: mostra os endereços numericamente ao invés de usar resolução DNS.

`-p [porta]`: ajusta a porta que será usada para o teste. A porta padrão é 33434.

`-r`: pula as tabelas de roteamento e envia o pacote diretamente ao computador conectado a rede.

`-s [end]`: usa o endereço IP/DNS [end] como endereço de origem para computadores com múltiplos endereços IPs ou nomes.

`-v`: mostra mais detalhes sobre o resultado do traceroute.

`-w [num]`: configura o tempo máximo que aguardará por uma resposta. O padrão é 3 segundos.

Já no windows, a sintaxe é:

```
tracert [opções] [host/IP de destino]
```

onde `host/IP destino` é o mesmo descrito anteriormente e as opções são:

`-d`: não resolver endereços para nomes hosts.

`-h nmax_saltos`: número máximo de saltos para a procura do destino.

`-j lst_hosts`: rota ampliada de origens usada com a lista lst_hosts.

`-w tempo_limite`: tempo limite de espera em milissegundos para cada resposta

- `nbtstat` - Mostra estatísticas de protocolos e conexões de TCP/IP correntes usando NBT (NetBIOS) sobre TCP/IP. A sintaxe é:

```
nebstat [opções]
```

onde as opções são:

`-a`: para listar as máquinas por nome.

`-A`: para listar as máquinas por IP.

`-c`: para listar o nome do cache remoto incluindo os endereços IP.

`-n`: para listar os nomes de NETBIOS Local.

`-r`: para listar nomes resolvidos por Broadcast e por WINS.

`-R`: para recarregar a tabela de cache remoto.

`-S`: para listar a tabela de sessões com os IPs de destino.

`-s`: para listar tabela de sessões convertendo IP de destino para nomes de Hosts pelo arquivo de Hosts.

A segunda atividade seria refazer toda análise feita com esses comandos numa rede maior, com mais máquinas, de forma a permitir que o aluno visualizasse e comparasse o resultado nas duas situações, como tempo de resposta de requisições por exemplo, criando ele próprio uma percepção de como funciona a teoria vista em sala de aula na prática.

Referências Bibliográficas

[1] KUROSE, James F., ROSS, Keith W. *Redes de computadores e a Internet: Uma abordagem top-down*, 3ª edição, Editora Addison Wesley, 2006.

[2] TORRES, Gabriel, *Redes de computadores: Curso completo*, 1ª edição, Editora Axcel Books, 2001.

[3] *www.2ei.com.br*

[4] *www.microchip.com*

Apêndice

Este apêndice descreve o kit de desenvolvimento utilizado nos experimentos, dando ênfase especial à placa PME-10 que o acompanha.

O kit de desenvolvimento

O kit de desenvolvimento, vendido pela 2EI – Eletrônica Embarcada para Internet, mostrado na Figura 3.6 é composto pelos seguintes itens:

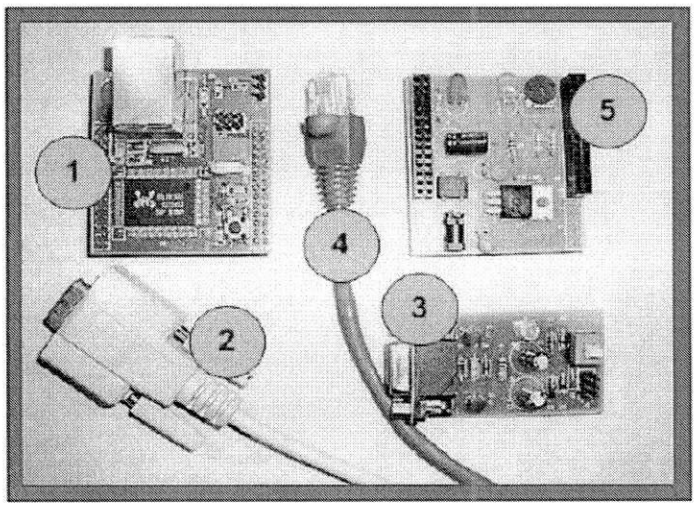

Figura 3.6: Kit de desenvolvimento PME-10

1. Placa PME-10 com conexão para Ethernet/Internet;
2. Um cabo serial para a interface de programação da PME-10;
3. Uma interface de programação serial RS-232 para ICSP (*In Circuit Serial Programmer*) para gravação da memória *Flash* do microcontrolador contido na PME- 10. Somente deve ser utilizada uma interface serial RS-232 verdadeira no computador. Conversores USB para serial provavelmente não funcionarão;

4. Um cabo de rede crossover (10 Base T) para conectar a placa PME-10 com um computador;

5. Uma placa filha contendo fonte de +5VDC, LED, chave push- button e potenciômetro para demonstração de monitoração remota;

6. Um CD-ROM contendo: a pilha TCP/IP Microchip modificada, programa IC-PROG para gravação da memória Flash do microcontrolador PIC18F8720 presente na placa PME-10 e guia do usuário em formato PDF;

7. Guia do usuário.

A placa PME-10

A PME-10 (Figura 3.7) é uma placa microcontrolada (UCP PIC 18F8720) com interfaces Ethernet 10 Mbps e serial RS-232. Ela pode ser adicionada a qualquer rede Ethernet 10/100 Mbps. Ela é fornecida com a pilha TCP/IP da Microchip® modificada, escrita na linguagem C, que pode ser compilada tanto pelo compilador Microchip PIC18 C (MCC 18) quanto pelo compilador Hi Tech. Aplicações típicas incluem servidor Web HTTP, conversor de interface RS 232 para Ethernet, conversor de interface RS485 para Ethernet, controle remoto via Web, etc. O usuário pode armazenar suas páginas Web na memória FLASH do microcontrolador PIC18F8720 usando o cabo serial e interface de programação que acompanha o kit de desenvolvimento. Além da pilha TCP/IP da Microchip, a 2EI fornece funções de enviar e receber mensagens SMS via celular através da interface serial.

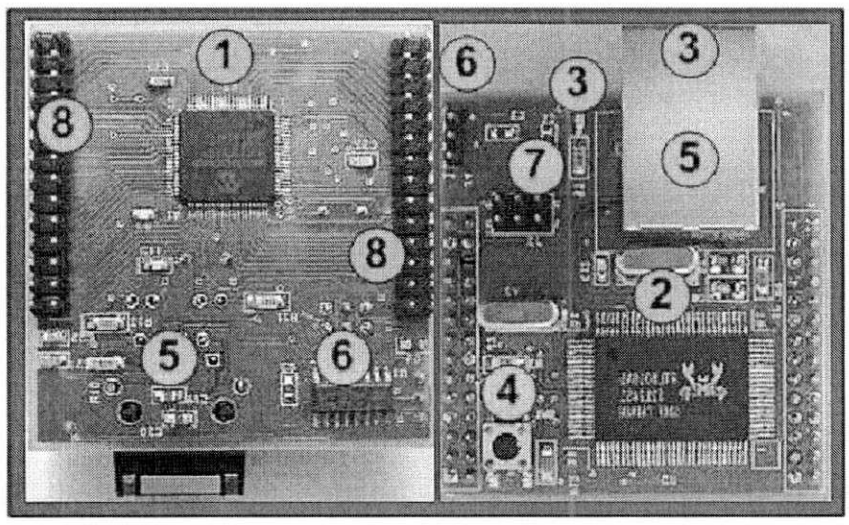

Figura 3.7: Placa PME-10

Os principais componentes da PME-10 são:

1. **Microcontrolador**: PIC18F8720 (clock de 20MHz, 128K de memória Flash, 1024 bytes de memória EEPROM, 3840 bytes de memória RAM e 68 portas de I/O programáveis das quais 16 podem ser configuradas como entradas analógicas de 10 bits) programado com a pilha TCP/IP Microchip modificada. A pilha TCP/IP usa no máximo 11Kbytes de memória Flash e 145 bytes de memória RAM (sem incluir os buffers de transmissão e recepção), assim há bastante área de memória para aplicações do usuário.

2. **Controlador Ethernet**: A PME-10 usa o controlador de Ethernet Realtek RTL8019AS (10Mbps) .

3. **LEDs de indicação**:

   (a) Vermelho: indica que a pilha TCP/IP está sendo executada. É programável pelo usuário. No software fornecido o mesmo pisca a cada 1 segundo.

   (b) Amarelo: indica que a conexão Ethernet está ativa (RJ-45).

   (c) Verde: indica que a placa está transmitindo ou recebendo um pacote de dados (RJ-45).

4. **Chave *push button***: serve para re-inicialização (RESET) do programa pois está conectada ao pino MCLR do microcontrolador.

5. **ConectorRJ-45 (10 Base T)**: Provê conectividade Ethernet padrão.

6. **Conector Tipo Molex Serial RS-232**: Provê os sinais Tx, Rx e GND para transmissão serial assíncrona proveniente do integrado ST232ABD.

7. **Conector ICSP (*In Circuit Serial Programming*)**: Permite à placa PME-10 ser conectada a interface de programação para gravação da memória f*flash* do microcontrolador.

8. **Conectores de Expansão**: Provê acesso a maior parte dos pinos do microcontrolador PIC18F8720. A PME-10 pode ser alimentada diretamente por este conector (+ 5V nos pinos 13 e 14 de J2, GND no pino 16 de J2).

A placa PME-10 é projetada para executar a pilha TCP/IP fornercida pela Microchip com pequenas modificações. São características da pilha TCP/IP fornecida pela Microchip:

- Inclui MAC, IP, ARP, ICMP, TCP, UDP, HTTP, FTP, DHCP e MPFS;
- Suporte a sockets para protocolos TCP e UDP;
- Suporte para compiladores Microchip C18 e Hi Tech PICC18;
- RTOS Independente;
- Software modular.