# UTILIZANDO A PORTA PARALELA PARA CONTROLE REMOTO DE UM DISPOSITIVO

Ricardo A. M. Valentim [1]
Ivanildo A. Nogueira [2]
Aluízio F. Rocha Neto [3]

***Resumo***

*Este artigo descreve como as portas de entrada e saída de dados podem ser manipuladas de forma a estabelecer uma interatividade e usabilidade ainda maior para o computador, tornando estas portas mecanismos de interação com outros objetos externos ao computador. Através de um dispositivo emissor de ondas de rádio, gerenciado pelo computador via porta paralela, aciona-se e controla-se o movimento de um pequeno carro.*

***Palavras-chave:*** *Porta paralela; sistema operacional; interação; wireless.*

## 1 Introdução

Na atualidade, muito se discute a questão da comunicação e interação homem-máquina. Claro que esta interatividade se dá porque os computadores[4] estão se aperfeiçoando a cada dia, na intenção de se tornar um equipamento cada vez mais amigável. Muito da interatividade homem-máquina iniciou-se no desenvolvimento de sistemas operacionais que fossem capazes de gerenciar dispositivos de entrada e saída de dados, além de promover um ambiente gráfico mais amigável e intuitivo.

A comunicação através de portas de entrada e saída de dados (E/S) é tratada neste trabalho como uma forma alternativa de oferecer mais aplicabilidade ao uso do computador. Fundamentalmente, a implementação de um projeto de comunicação com portas de entrada e saída necessita da construção de um software que possa estabelecer a comunicação com o sistema operacional de forma correta e precisa, pois esta comunicação irá ocorrer através das chamadas de sistema de baixo nível.

[1] Discente do Curso de Bacharelado em Sistemas de Informação da FARN. E-mail: ricardo@farn.br
[2] Discente do Curso de Bacharelado em Sistemas de Informação da FARN.
[3] Docente do Departamento de Informática da FARN. E-mail: aluizio@farn.br
[4] Termo que inclui, além do hardware, o sistema operacional que é o maior responsável pela comunicação com os dispositivos de hardware (TANENBAUM, 2000).

A comunicação do software com o sistema operacional tornará possível o envio de quaisquer comandos ao dispositivo que se deseja controlar. Ao se programar as portas de entrada e saída, não se estará acessando diretamente o dispositivo controlado, mas sim o Sistema Operacional, que por sua vez, através de sua manipulação com o hardware promoverá o acesso ao dispositivo desejado.

O sistema operacional deve proporcionar uma interface entre a E/S em nível físico e em nível de usuário ou lógica, pois a E/S é bastante complexa em muitos casos. Ela não envolve apenas a movimentação de dados de um lugar para outro; ela requer a comunicação entre dois dispositivos eletrônicos completamente diferentes (SHAY, 1996).

Observa-se então que através desse tipo de comunicação é possível manipular quase todos os equipamentos. Tais dispositivos podem ser eletrônicos ou até mesmo mecânicos. Para tanto, é importante o desenvolvimento de softwares que possibilitem o gerenciamento de objetos externos ao PC (Computadores Pessoais), o qual é o objetivo deste trabalho.

O software desenvolvido neste trabalho é o responsável direto pela comunicação fim-a-fim com os objetos externos. Todo o processo de interação entre o software e o dispositivo exterior ocorre em tempo real via porta paralela. Sendo assim, o sistema desenvolvido, que se denomina Sistema de Gerenciamento de Objetos (SGO), consolida de forma consistente a integração entre hardware e software, o que nem sempre é trivial.

Segundo o professor de Engenharia da Computação Dr. Roger S. Pressman, a montagem de um sistema de tempo real faz com que o engenheiro de sistemas se defronte com difíceis decisões referentes a hardware e software (PRESSMAN, 1995).

O foco desta pesquisa é enfatizar a implementação de um sistema totalmente dedicado, no qual o controle do aplicativo desenvolvido com o objeto gerenciado ocorre remotamente.

## 2 A Porta Paralela

Um microcomputador não teria muita importância se não pudesse entrar com dados externos para o processamento, nem como exteriorizar os dados resultantes de algum processamento (TORRES, 2001).

Para que o micro possa estabelecer uma comunicação com periféricos deve

existir uma espécie de porta, a qual permite a entrada e saída de dados. Estas portas podem ser: a porta paralela, a serial e a USB (*Universal Serial Bus*) que estão integradas à placa-mãe do microcomputador. Neste projeto é descrito sucintamente o funcionamento da porta paralela, pois essa interface foi objeto de estudo deste trabalho.

Atualmente, já há uma grande variedade de periféricos que utilizam a porta paralela, popularmente conhecida como *porta de impressora*. Tais periféricos incluem *zip drives*, gravadores de cds externos, *scanners*, entre outros. Tudo isso só é possível porque a interface paralela tornou-se bidirecional, devido à evolução do PC. Isto é, pode enviar e receber dados 'simultaneamente'. Neste sentido, a proposta deste trabalho é mostrar que, além desses periféricos já conhecidos, pode-se dar mais usabilidade à porta paralela estabelecendo uma comunicação desta interface com qualquer objeto que possa ser controlado. Aplicando as técnicas corretas, qualquer interface de E/S pode ser usada para manipular objetos externos. A interface paralela é uma boa alternativa pela baixa complexidade.

A bidirecionalidade da interface paralela mostrou-se bastante relevante, pois, através desse recurso, pode-se obter um alto desempenho no quesito taxa de transmissão. Os modos de operação da interface paralela estão basicamente restritos a SSP[5], EPP[6] e ECP[7], cada uma com sua taxa de transferência distinta. O modo que tem uma melhor taxa de transferência é o ECP. Esse modo de operação é um avanço sobre o modo EPP, aumentando significativamente a taxa de transferência da porta paralela, (TORRES, 2001). O modo utilizado para implementação do projeto foi o EPP.

A identificação das portas paralelas é feita com a sigla LPT (*Line Printer* – Impressora de linha), sendo que, para cada porta é acrescentado um número que segue uma ordenação, ou seja, o primeiro endereço de conexão com a interface paralela ganha um rótulo de LPT1, o segundo de LPT2 e assim por diante. Na prática, esses rótulos têm apenas uma simbologia figurativa para facilitar a identificação, pois o que ocorre realmente é que o sistema operacional atribui a cada interface endereços de memória. Assim, o software desenvolvido realiza a transmissão e recepção de dados através de escrita e leitura nesses endereços, respectivamente.

Deve-se ter atenção para o fato de que a mesma interface, no caso LPT1,

[5] SSP (*Standard Parallel Port*) é o modo de operação padrão, unidirecional e com taxa de transferência máxima de 150KB/s (TORRES, 2001).

[6] EPP (*Enhanced Parallel Port*) este modo de operação existe em todos os micros atuais. Traz duas novas características à porta paralela padrão: modo bidirecional e aumento da taxa de transferência (TORRES, 2001).

[7] ECP (*Extended Capabilites Port*) esse modo de operação é um avanço sobre o modo EPP, aumentando significativamente a taxa de transmissão da porta paralela (TORRES, 2001).

tem dois endereços distintos, um para escrever e outro para ler dados. Podemos observar o endereço do dispositivo de entrada e saída configurado através da BIOS (*Basic Input/Output System*) do PC ou no gerenciador de dispositivos do Windows.

Por padrão, o endereço em hexadecimal 378h é o utilizado para a LPT1. Isso não significa que esse endereço seja estático. Ele pode ser mudado a qualquer momento, desde que se tenha noção do que se esteja fazendo, pois qualquer procedimento errado poderá gerar conflitos e o periférico ficar inoperante. Desta forma, na implementação do projeto observou-se cuidadosamente o tratamento deste endereço, ou seja, o software realiza todas as suas atividades sem causar danos às configurações já propostas pelo sistema operacional. Nos Quadros 1 e 2, é mostrado o endereçamento das LPTs.

Quadro 1: Endereços de Escrita da Porta Paralela

| Nome da Porta | Endereço de memória | Endereço da Porta | | Descrição |
|---|---|---|---|---|
| LPT1 | 0000:0408 | 378 hexadecimal | 888 decimal | Endereço base |
| LPT2 | 0000:040A | 278 hexadecimal | 632 decimal | Endereço base |

Quadro 2: Endereços de Leitura da Porta Paralela

| Nome da Porta | Endereço de memória | Endereço da Porta | | Descrição |
|---|---|---|---|---|
| LPT1 | 0000:0408 | 378 hexadecimal | 888 decimal | Endereço base |
| LPT2 | 0000:040A | 278 hexadecimal | 632 decimal | Endereço base |

## 2 O Projeto

O projeto foi desenvolvido no ambiente Windows. Foi utilizada, como ambiente de programação, a plataforma de desenvolvimento *Borland Delphi 5.0*. Essa ferramenta foi escolhida porque com ela é possível realizar rapidamente a construção de uma interface gráfica bastante amigável para o SGO.

O SGO tem por finalidade gerenciar os movimentos de um protótipo (pequeno carro), sem a utilização de fio, isto é, utilizando-se do sistema de comunicação *wireless*. Em linhas gerais, o sistema promove uma conexão entre dois objetos, transmissor e receptor, utilizando a emissão de ondas eletromagnéticas em freqüências de rádio e sinais luminosos na faixa de infravermelho, utilizando como meio de transmissão o ar.

Assim, o SGO é um sistema computacional que estabelece a comunicação entre dispositivos com arquiteturas bastante diversificadas. Para isso, o sistema desenvolvido utilizou vários recursos de programação implementados em cima da interface paralela, como por exemplo, a implementação de multitarefa (*multithreading*), sem a qual não seria possível obter as informações em tempo real. Foi através dessas técnicas implementadas que a interface paralela serviu como meio para que o SGO pudesse enviar sinais digitais para a conversão em sinais de rádio freqüência.

É por meio dessa manipulação de sinais que o SGO consegue obter do objeto algumas informações, tais como: distância percorrida pelo carrinho, a velocidade que o carrinho está atualmente, velocidade média e percepção da existência de algum obstáculo que poderá ocasionar uma colisão. Para este último caso, o sistema recebe um estímulo enviado por um sensor que está acoplado na frente do experimento. Após receber este estímulo, o SGO envia um comando que leva à aceleração do carrinho a 0 (zero), fazendo com que o carro pare, evitando a colisão.

Observando todo o conjunto de hardware e software utilizados na implementação deste projeto, podem-se destacar três entidades:

**Centro de controle** é a parte do sistema que dá funcionalidade ao projeto, ou melhor, é o conjunto de códigos de programação que é acionado toda vez que se tem a necessidade de enviar ou receber um dado. É implementado pelo SGO. O mesmo pode gerar ações que serão enviadas na forma de comandos ao objeto gerenciado e, a partir de informações que chegam, estas são interpretadas de acordo com a situação que está ocorrendo naquele momento com o objeto gerenciado, gerando novas ações. Sendo assim, o centro de controle é considerado como sendo o "cérebro" do sistema.

**Objeto passivo** é a parte do sistema que aguarda comandos acionados pelo centro de controle. Ao entrar em ação, este gera informações que são passadas ao centro de controle a fim de que o mesmo as gerencie da forma que lhe convier. Neste projeto, é implementado pelo carrinho.

**Sistema de comunicação** é o elo entre o centro de controle e o objeto passivo, ou seja, engloba todos os dispositivos de comunicação entre estas duas entidades. É implementada pelo Sistema Operacional, a interface paralela e o sistema de transmissão por rádio freqüência.

Para observar o funcionamento deste sistema, construiu-se um fluxo de informações que pode ser observado na Figura 1.

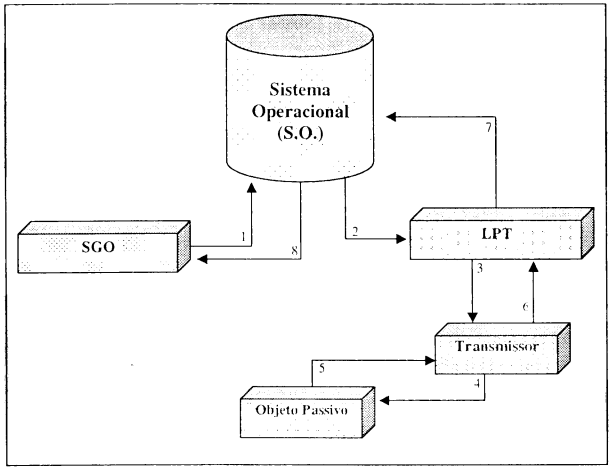



Figura 1: Arquitetura da Solução

No Quadro 3 há uma identificação dos fluxos de informações. É através destes fluxos de sinais que o SGO obtém uma percepção do mundo exterior.

Quadro 3: Fluxo de Informação entre os Componentes

| Fluxo | N°Origem | Destino | Objetivo |
|---|---|---|---|
| 01 | SGO | S.O. | Enviar comando para o objeto gerenciado |
| 02 | S.O. | LPT | Escrever a seqüência de bits na porta paralela |
| 03 | LPT | Transmissor | Os bits são transformados em pulsos elétricos |
| 04 | Transmissor | Objeto | Os pulsos elétricos são transformados em sinais de rádio freqüência (RF), que são captados pela antena do objeto |
| 05 | Objeto | Transmissor | O objeto envia informação através de difusão de rádio freqüência, que são captados pela antena no Transmissor |
| 06 | Transmissor | LPT | Os sinais de RF viram pulsos elétricos na interface paralela |
| 07 | LPT | S.O. | O S.O. lê o dado que acaba de chegar na LPT |
| 08 | S.O. | SGO | O SGO recebe o dado que o S.O. acaba de receber, verifica a informação e imediatamente envia um novo comando se for necessário |

## 3 O Protótipo

O carrinho que faz parte da implementação deste trabalho foi adquirido totalmente pronto, mostrando que realmente é possível manipular muitos objetos externos ao PC, sem se preocupar como eles funcionam, e sim com a maneira como se deve comunicar com eles. Este carrinho era totalmente alheio ao PC e agora faz parte de um sistema que depende deste mesmo PC para que se possa entrar em funcionamento. Além do carrinho faz parte do protótipo um circuito eletrônico responsável por receber os sinais elétricos enviados pelo SGO e convertê-los em sinais de rádio freqüência. O SGO, que é o 'cérebro' do sistema, é detalhado a seguir. Um ponto relevante no carrinho é um sensor infravermelho[8] posto na caixa de redução, este sensor envia um sinal cada vez que a roda completa um giro de 360° graus, com isso podemos calcular a distância percorrida, com esta grandeza é possível descobrir outras, como velocidade, velocidade média e aceleração.

### 3.1 O Algorítmo

O SGO é composto basicamente de duas funções. Uma responsável por enviar dados e outra responsável por checar se há algum dado na porta paralela, tudo isso ocorrendo simultaneamente e em tempo real. Este paralelismo, como já foi dito, é tratado pelo SGO através do recurso de multitarefa oferecido pelo sistema operacional, ou seja, enquanto o sistema está recebendo e executando comandos a partir de sua interface com o usuário, o mesmo, está realizando um serviço de checagem de dados na LPT1(379h).

Quadro 4 – O Algoritmo do SGO

```
Programa SGO;

{Módulos responsáveis por ler e enviar dados na interface paralela.}
Constantes

  LPT1 = 378h; // Endereço da porta paralela para escrita;
  LPTR = 379h; // Endereço da porta paralela para leitura;

// Módulo de Escrita de dados.
Procedimento enviabyte(bit: inteiro); // Acessar a porta paralela;
Início
```

[8] O sensor usado foi o mesmo encontrado em impressoras para indicar que acabou o papel.

{Código escrito em *assembly* para enviar dados para porta paralela:}

```
asm // diretiva usada para escrever códigos em assembly.

 mov dx, LPT1  //move o valor 378h para o registrador de dados dx.
 mov al, bit   //move o valor de bit para registrador al(8 bits).
 out dx, al    // Escreve no endereço 378h o valor que tem em          bit.
Fim asm;
```

Final Procedimento;

{————————————————————————————————————————}

// Módulo Leitura de dados
Função recebebyte(end: inteiro):inteiro; // Escrever na LPT;
 Var
  ret : byte;

Início

{Código escrito em *assembly* para ler dados na porta paralela:}

```
asm

  mov dx, LPTR  // move o valor $379 para registrador de dados dx.
  in al, dx     // lê o valor que está em dx e armazena o valor encontrado no
registrador al.
  mov ret, al   // move o valor lido na paralela para variável ret.

Final asm;
```

 recebyte ? ret; {retorna o dado que está na porta paralela}

Final função;

{————————————————————————————————————————}
// implementação do sistema ou inicialização do programa.

Início
 Var
  ValorPorta, ValorEntrada, Conta360 : Inteiro;
  Sair : Caractere;

```
{Este bloco de comandos é responsável por enviar dados}

IniciaMultitarefa;
ValorEntrada ? 0; {Garante que a variável será iniciada com zero}

Repita
  Mensagem ('Menu');
  Mensagem ('128 – Para Frente, 64 – Para Traz, 32 – Direita,
        16 – Esquerda' );
  Leia (ValorPorta);
  Enviabyte (ValorPorta);
  Mensagem ('Deseja encerrar o programa (s/n)?');
  Leia (sair);
Até que (sair = 's' ou sair = 'S');

Multitarefa
 Enquanto verdade Faça
 Início
  Tempo(10); // O processo entra em espera por dez milésimos de
        // segundo evitando o loop infinito

  Se recebebyte(0x379) = 120 Então  // Sinal proveniente do sensor
                    // de presença de obstáculos
    enviabyte(0);

  Se recebebyte(0x379) = 8 Então    // Sinal proveniente do
                    // sensor de giro da roda.
        Conta360 ? Conta360 + 1;      // Incrementa o nº de giros
 Fim enquanto;

Fim.
```

No *Delphi* não existe função para acessar diretamente um dispositivo de hardware. Para isso, foi utilizado o recurso de embutir código *Assembly* para instruções de baixo nível, ou melhor, recorremos a diretiva "ASM" que no *Delphi* significa que daquele ponto até o próximo *end* as linhas de comandos serão escritas em *Assembly*. Além disso, o *Delphi* oferece uma classe *TThread* que facilita bastante a questão da multitarefa, pois é através desta classe que o SGO realiza a verificação dos dados na interface paralela. Claro que todo o código do sistema não se resume a este algoritmo; colocou-se este para dar uma noção da lógica empregada no sistema. A velocidade e a distância aproximada percorrida pelo carrinho são obtidas através de cálculos a partir da variável Conta360. Os cálculos seriam os seguintes: Para calcular a distância, fazemos:

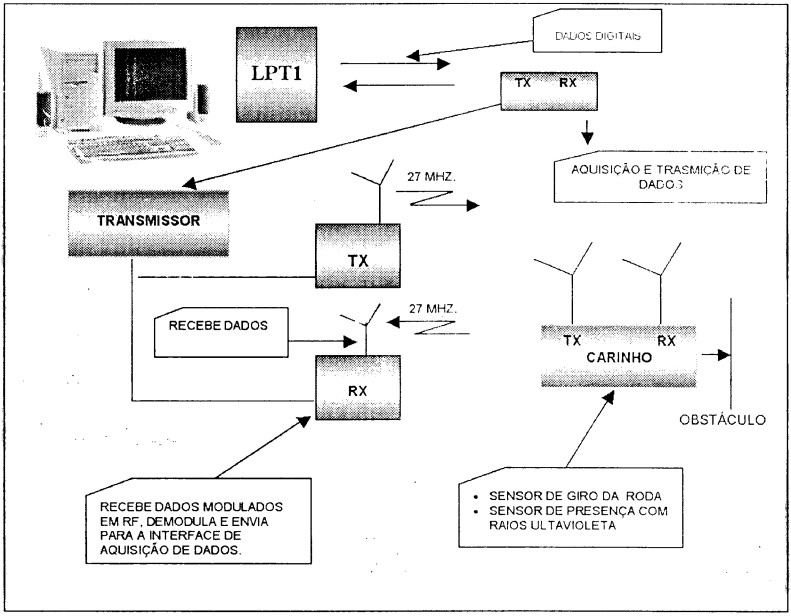
Distância = (Diâmetro_da_Roda * Conta360);

Para calcular a velocidade, temos:

Distância / Tempo, em que Tempo é o instante em que o carrinho parou subtraído do instante em que ele partiu. Estas grandezas medidas pelo SGO são apenas valores aproximados, uma vez que o equipamento utilizado não provê informações mais precisas.

### 3.2 O Circuito Elétrico do Transmissor

Descrevemos o transmissor de forma ilustrativa, mostrado na Figura 2, no intuito de fornecer uma idéia básica de como se processa a transmissão dos dados sem fio.




## 4 Metodologia Aplicada

Através de pesquisas na Internet sobre como utilizar a porta paralela para acessar e/ou controlar objetos do mundo real, foi encontrada uma infinidade de

novas tecnologias que já fazem uso destes recursos de hardware. Partindo daí, observou-se a estrutura empregada e descobriu-se o processo de comunicação entre a interface paralela e os objetos gerenciados. Assim, foram desenvolvidas técnicas que nos ajudaram a tornar possível o envio e recebimento de dados com objetos externos ao PC.

Em um segundo momento, procurou-se fundamentar teoricamente a nossa pesquisa a fim de que ela obtivesse um caráter mais científico. Recorreu-se a autores como Gabriel Torres que, através de suas obras, nos permitiu esmiuçar a interface paralela, que foi o principal objeto de estudo. Pesquisou-se também autores que descrevessem o sistema operacional como recurso de comunicabilidade software-hardware. Desta maneira, conseguiu-se ter um entendimento mais aprofundado de implementação do projeto. Foi fundamental nesta pesquisa conhçcer bem a funcionalidade dos sistemas operacionais, no que tange a comunicação com dispositivos de entrada e saída de dados.

## 5 Resultados

O sistema desenvolvido tem um grau de complexidade relativamente médio, uma vez que teve que se definir regras de manipulação do dispositivo controlado. Segundo Willian Shay, professor do Departamento de Ciências da Computação da Universidade de Wisconsin-Gree Bay, escrever um software para controlar um dispositivo requer muita atenção a detalhes. O programador deve conhecer as especificações técnicas do hardware, o que em geral requer um cuidadoso estudo dos manuais técnicos (SHAY, 1996).

O projeto concede conhecimento teórico e empírico no que diz respeito a todo o processo envolvido no seu desenvolvimento, propiciando segurança no desenvolvimento de sistemas que manipulem objetos externos ao computador, através das interfaces de entrada e saída. O sistema proposto alcançou os resultados desejados, pois além de conseguir enviar dados através da porta paralela, também foi possível receber dados do dispositivo controlado, de modo que o software desenvolvido interagiu com o meio exterior sem a intervenção humana.

Com isso, os resultados obtidos com este projeto são bastante satisfatórios, uma vez que o mesmo serviu como ponto de partida para projetos ainda maiores e mais ousados.

## 6 Considerações Finais

A informática tem andado a passos largos. Contudo, ainda existe uma infinidade de possibilidades a serem implementadas. Este projeto de pesquisa permi-

tiu buscar ainda mais informações no intuito de poder ampliá-lo para um tipo de sistema inteligente. Sendo assim, percebe-se o quanto é importante desenvolver sistemas que possam prover a automação de objetos usados nas empresas, nas residências, nos veículos, enfim, em tudo que faça parte do cotidiano das pessoas e das corporações.

Dominar este tipo de técnica ou ao menos ter conhecimento de suas possibilidades é de suma importância, uma vez que a sociedade moderna exige sistemas cada vez mais completos a fim de resolver problemas cada vez mais complexos. Em princípio, a grande vantagem destes sistemas que resolvem problemas complexos é que os mesmos agregam muito mais valor às soluções.

**Referências**

PRESSMAN, R. S. **Engenharia de software**. São Paulo: Makron Books, 1995.

SHAY, W. A. **Sistemas operacionais**. São Paulo: Makron Books, 1996.

TANENBAUM, A. S.; WOODHULL, A. S. **Sistemas operacionais**. 2. ed. Porto Alegre: Bookman, 2000.

TORRES, G. **Hardware completo**. 4. ed. Rio de Janeiro: Axcel Books, 2001.

***Abstract***

*This paper shows how the input and output ports can be handled to even greater computer inter-operability and usability when they become interactive mechanisms with other computer external objects. By means of a radio wave transmission device managed by a parallel port, a small car engine is started and its movements controlled controlled.*

***Key words**: parallel port; operational system; interaction; wireless.*