
Interface serial

Pedroso

21 de fevereiro de 2007

1 Introdução

A Transmissão serial é uma das mais importantes formas de transmissão de dados utilizada. O IBM-PC possui uma interface serial assíncrona RS-232c, que suporta a transmissão de dados em taxas de até 115,2K bps. As conexões WAN entre redes normalmente são realizadas utilizando-se interfaces seriais síncronas.

2 Transmissão Assíncrona

Na transmissão não existe nenhuma referência de tempo comum entre transmissor e receptor. Desta forma, o transmissor deve "avisar" o receptor quando for enviar um dado válido para que o ele conheça o instante quando começar a recuperar o sinal. START BIT, que nada mais é do que um aviso que diz ao receptor para coordenar o momento de amostragem para recuperação do sinal.

Quando o receptor percebe a presença do START BIT, ele calcula quais serão os instantes em que ele irá realizar a amostragem do sinal. Como o receptor conhece previamente qual o tamanho do caracter ele poderá calcular quando recuperar os dados. Ao final do caracter, é inserido um STOP BIT, com nível invertido ao START BIT. Desta forma, outro sinal de START poderá iniciar o próximo caracter.

Existem alguns problemas com esta técnica. O primeiro é o fato de que, como transmissor e receptor não tem uma base de tempo em comum, se o número de bits por caracter for muito grande, existe uma tendência de ocorrer uma sobreposição na recuperação dos dados. Como principais desvantagens podemos citar:

1. Os bits de START e STOP foram inseridos para controle, porém não tem valor como dado. Por exemplo, se o tamanho do caracter for 8 bits, e estivermos trabalhando com 1 START BIT e 1 STOP BIT, 20% das informações transmitidas serão informações de controle da própria transmissão.

2. Quanto menor o tempo de permanência do sinal no meio, maior será a quantidade de dados que conseguiremos enviar em um intervalo de tempo. A desvantagem é que quanto menor o tempo de permanência do sinal, mais grave será o problema causado pela diferença de relógio de referência entre transmissor e receptor, conforme foi citado anteriormente.

O padrão RS-232 para interfaces seriais assíncronas foi estabelecido pela EIA (Electronics Industry Association), e é implementado no IBM-PC através de uma interface chamada UART.

3 UART (Universal Asynchronous Receiver Transmitter)

A UART é implementada através do controlador 8250 do IBM PC. Antes de ser utilizado, o 8250 deve ser programado com as características de transmissão: velocidade, número de stop bits, tamanho da palavra e paridade. O 8250 se comunica com a CPU através do processo de interrupção.

A interface utilizada é a RS232C, que utiliza um conector padrão de 25 pinos (DB-25), uma das mais utilizadas para transmissão de dados. Conectores de 9 pinos também são muito utilizados, ambos os conectores são do tipo macho no IBM-PC. Significa que o cabo deve possuir um conector fêmea.

A porta serial transmite um '1' como um nível de -3 a -25 volts e um '0' como um nível de +3 a +25 volts. Portanto, a diferença máxima de tensão é 50 volts, o que permite que os cabos seriais possuam comprimentos relativamente longos sem se preocupar com a perda. A corrente nunca deve exceder 500mA, caso contrário irão ocorrer danos no circuito.

3.1 Elementos da comunicação

DTE - Data Terminal Equipment O DTE é o equipamento que possui dados a serem transmitidos (fornecendo dados sempre em formato digital). Como exemplo de DTE podemos citar um computador ou um terminal.

DCE - Data Communication Equipment O DCE é o equipamento que irá receber os dados do DCE, realizar a transformação do sinal para adequá-lo ao meio de transmissão (seja ele qual for) e realizar a transmissão. Um exemplo simples de DCE é um modem.

3.2 Pinagem dos conectores

Nem todos os pinos disponíveis no conector DB-25 são utilizados. A Tabela 1 mostra o número do pino nos conectores DB-25 e DB-9 e seu significado.

Significado de cada pino:

TD Transmissor, transmissão assíncrona conforme configuração

RD Receptor, recepção assíncrona conforme configuração

RTS A UART informa que está pronta para trocar dados

DTR Oposto ao DSR, a UART informa que está pronta para estabelecer o link

CTS O modem informa que está pronto para trocar dados

DSR O modem informa que está pronto para estabelecer o link

CD Portadorea Detectada

Cabos

4 Controle de fluxo

Se a taxa de transmissão do DTE ultrapassa a capacidade do DCE, cedo ou tarde irá ocorrer um buffer overflow. Para prevenir a perda de dados devido este erro, existem duas táticas: controle de fluxo por software e controle de fluxo por hardware.

O controle de fluxo por software (tambem chamado de Xon/Xoff) utiliza dois caracteres (respectivamente, o ASCII 17 e o ASCII 19 para Xon e Xoff). O modem envia o caracter Xoff quando desejar que o computador pare a transmissão de dados, e o caracter Xon para requisitar que o computador continue transmitindo.

Este tipo de controle de fluxo tem a vantagem de não necessitar de nenhuma linha adicional entre os elementos. No entanto, a principal desvantagem é o acréscimo de informações a serem transmitidas. O controle de fluxo por hardware utiliza os pinos RTS e CTS. Quando o computador deseja enviar dados, ele ativa a linha RTS. Quando o modem estiver pronto para receber, ele irá ativar a linha CTS. Esta técnica é mais eficiente do que o controle de fluxo por software. Contudo, necessita de uma via de extra.

5 Registradores da UART

Cada registrador possui uma finalidade específica. Tudo que for enviado para BASE+0 será imediatamente transmitido. Da mesma maneira, todas as informações recebidas podem ser recuperadas acessando o endereço BASE+0. O registrador LSR informa o status interface. Uma importante informação é se existe dado pronto para ser lido. O significado de cada bit do registrador LSR é apresentado na Tabela 4.

```
#include <stdio.h> #define BASE 0x3F8

void main(void) {
    int c, ch;
    outportb(BASE + 1 , 0);    /* Turn off interrupts */

    /*          PORT 1 - Communication Settings          */
}
```

```
outportb(BASE + 3 , 0x80); /* SET DLAB ON */
outportb(BASE + 0 , 0x03); /* Set Baud rate - Divisor Latch Low Byte */
/* Default 0x03 = 38,400 BPS */
/*      0x01 = 115,200 BPS */
/*      0x02 = 57,600 BPS */
/*      0x06 = 19,200 BPS */
/*      0x0C = 9,600 BPS */
/*      0x18 = 4,800 BPS */
/*      0x30 = 2,400 BPS */
outportb(BASE + 1 , 0x00); /* Set Baud rate - Divisor Latch High Byte */
outportb(BASE + 3 , 0x03); /* 8 Bits, No Parity, 1 Stop Bit */
outportb(BASE + 2 , 0xC7); /* FIFO Control Register */
outportb(BASE + 4 , 0x0B); /* Turn on DTR, RTS, and OUT2 */

printf("\nSample Comm's Program. Press ESC to quit \n");

do {
    c = inportb(BASE + 5); /* LINE STATUS REGISTER (LSR) */
    if (c & 1) {
ch = inportb(BASE); /* If data ready */
        printf("%c",ch);
    }
    if (kbhit()){
        ch = getch(); /* If key pressed, get Char */
        outportb(BASE, ch);} /* Send Char to Serial Port */

    } while (ch !=27); /* Quit when ESC (ASC 27) is pressed */
}
```