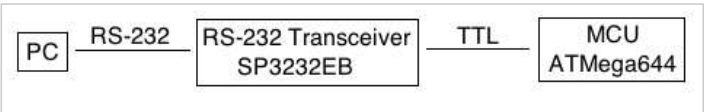
Liên kết khác k3751070033@gmail.com Bảng điều khiển Đăng xuất

BLOG

THỨ HAI, NGÀY 16 THÁNG 5 NĂM 2011

Bă´t tay RTS/CTS và dạng sóng

Tôi quyê´t định tìm hiểu vê` bă´t tay và báo hiệu RS-232. Tôi đã mâ´t rất nhiê`u thời gian để tìm một trang duy nhâ´t tóm tă´t mọi thứ tôi câ`n biê´t, vì vậy tôi viê´t bài này với hy vọng rằng nó sẽ giúp được ai đó trong tình huô´ng tương tự.
Đây là những gì tôi đang cố gắng thực hiện: Tôi có một cổng nô´i tiê´p DB-9 được nô´i với một bộ vi điê`u khiển (MCU) ATMega644, với một bộ thu phát RS-232 SP3232EB ở giữa, như hình bên dưới:



RS-232 ba dây (RX, TX, GND) không đủ cho mục đích của tôi, vì vậy tôi sẽ thực hiện năm dây (RX, TX, RTS, CTS, GND).

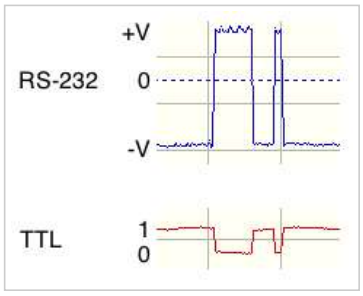
Khi chúng tôi nói vê` RX và TX, chúng tôi sẽ làm điê`u đó từ quan điểm của MCU. Đó là, chúng tôi đọc từ PC trên RX và chúng tôi truyê`n tới PC trên TX. RTS và CTS có ý nghĩa không thể đảo ngược, vì vậy không câ`n làm rõ.

Làm thế nào để tâ´t cả điê`u này làm việc? Bă´t tay RTS/CTS được sử dụng để đảm bảo rằng thiê´t bị ngoại vi (trong trường hợp của chúng tôi là MCU) sẵn sàng xử lý dữ liệu từ máy chủ (PC). Nó dường như không áp dụng theo hướng khác (hoặc nê´u có, tôi không thấ´y câ`n phải quan tâm). Ý tưởng là PC đủ mạnh để xử lý mọi thứ mà bạn có thể quan tâm để ném vào nó, nhưng MCU tô`i tàn thì không may mă´n như vậy. PC có thể dễ dàng làm tràn bâ´t kỳ bộ đệm nào mà MCU có thể quan tâm đặt tại chỗ, vì vậy MCU câ`n một cách để thông báo cho PC khi PC được phép gửi dữ liệu mới. Đó là nơi bă´t tay.

Có hai cách để thực hiện bă´t tay RTS/CTS. Đâ`u tiên là yêu câ`u/xác nhận. Khi PC sẵn sàng gửi dữ liệu, nó sử dụng RTS (yêu câ`u gửi) để báo cho MCU rằng nó (PC) có dữ liệu để gửi. Tôi nói "sử dụng" vì các mức logic hơi buồ`n cười, như được mô tả bên dưới, nhưng hiện tại không quan trọng. MCU thấ´y rằng PC đã sẵn sàng gửi dữ liệu. Khi MCU đã sẵn sàng để nhận nó, nó sẽ sử dụng CTS để báo cho PC biê´t rằng dữ liệu có thể được gửi đi. PC theo dõi CTS để xem a) khi nào nó có thể bă´t đâ`u gửi dữ liệu và b) khi nào nó phải dừng. Loại bă´t tay RTS/CTS thứ hai thực hiện đo sáng hai chiê`u (chi tiê´t tại đây). Tôi đã chọn thực hiện yêu câ`u/xác nhận, nhưng chuyển đổi sang hai chiê`u sẽ đơn giản.

Khi giao tiê´p RS-232 và TTL, chúng ta phải xử lý hai tiêu chuẩn khác nhau cho các mức logic. TTL là 0-đê´n-không-không. 0 đê´n 5, trong trường hợp mạch của tôi. 5V được khẳng định; 0V không được khẳng định. Mặt khác, RS-232 sử dụng điện áp dương và âm. Điện áp dương được gọi là khoảng trô´ng, trong khi điện áp âm là dấ´u. Công việc của bộ thu phát là chuyển đổi giữa điện áp RS-232 và TTL. Nó làm như vậy bằng cách ánh xạ RS-232 dương sang TTL 0 và RS-232 âm sang TTL 1. Nghĩa là, điện áp TTL gâ`n như nghịch đảo với điện áp RS-232.

Đây là chữ p (0x70), được gửi qua dây ở tô´c độ 9600 baud, 8N1:



Bây giờ chúng ta đã biê´t điê`u này, hãy trình bày lại hoạt động của RTS và CTS bằng cách sử dụng điện áp TTL do bộ thu phát tạo ra. Đây có lẽ là đoạn quan trọng nhâ´t của bài đăng này và mâ´t một thời gian dài đáng ngạc nhiên để đập vào hộp sọ dày của tôi.

RTS và CTS thường được giữ ở mức cao (TTL 1). Khi ở mức cao, không có yêu câ`u (RTS) nào được thực hiện và

NGƯỜI THEO DÕI
Người theo dõi (6)
Theo dõi

LƯU TRỮ BLOG
► 2019 (1)
► 2012 (10)
▼ 2011 (31)
► Tháng mười hai (2)
► Tháng Mười (4)
► Tháng 9 (4)
► Tháng Bảy (3)
► Tháng Sáu (2)
▼ tháng 5 (3)
Bộ vi xử lý của riêng tôi
Gran Fondo New York
Bă´t tay RTS/CTS và dạng sóng
► Tháng Tư (3)
► Tháng ba (8)
► Tháng Một (2)
► 2010 (13)

VỀ TÔI
SIMMONMT
XEM HỒ SƠ HOÀN CHỈNH CỦA TÔI
ĐĂNG KÝ QUA EMAIL
Nhập địa chỉ email của bạn:
Đặt mua
Cung câ´p bởi FeedBurner

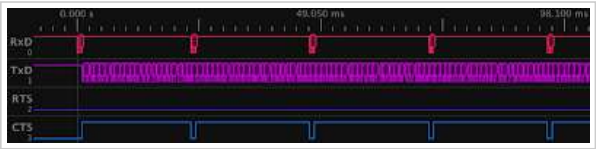
không có quyề`n (CTS) nào được cấ´p. Khi PC sẵn sàng gửi dữ liệu, nó sẽ đưa TTL RTS xuố´ng mức thấ´p. MCU thấ´y điề`u này. Khi MCU sẵn sàng nhận dữ liệu, nó sẽ đưa TTL CTS xuố´ng mức thấ´p. PC gửi dữ liệu miễn là TTL CTS ở mức thấ´p và dừng khi MCU lấ´y TTL CTS ở mức cao. Người ta tưởng tượng rằ`ng người gửi được phép hoàn thành ký hiệu (byte) hiện tại, nhưng tôi chưa xác minh điề`u đó.

Bây giờ, để dây mọi thứ lên. Như đã nêu ở trên, chúng tôi đang sử dụng RX và TX từ góc độ của MCU, vì đó là cách mọi thứ được dán nhãn ở đầ`u của MCU (tức là trong chính MCU và trong bộ thu phát).

			SB3232EB (TÔ´I ĐA232)		
Chức năng	**Phương hướng**	**chố´t DB-9**	**chân RS-232**	**chố´t TTL**	**chân AVR**
RX	PC sang AVR	3	13 (R1IN)	12 (R1OUT)	14 (PD0/RX)
TX	AVR vào máy tính	2	14 (T1OUT)	11 (T1IN)	15 (PD1/TX)
RTS	PC sang AVR	7	8 (R2IN)	9 (R2OUT)	17 (PD3)
CTS	AVR vào máy tính	số´ 8	7 (T2OUT)	10 (T2IN)	16 (PD2)

Tôi đã làm hỏng hệ thố´ng dây điện một vài lầ`n cho đế´n khi tôi vẽ được biểu đồ` trên. Sau đó, mọi thứ diễn ra suôn sẻ.

Đây là kế´t quả cuố´i cùng, được ghi lại bởi Open Bench Logic Sniffer thông qua ứng dụng khách OLS LogicSniffer:



Chúng tôi đang xem xét nỗ lực của PC để gửi một chuỗi nhiề`u ký tự nhanh nhấ´t có thể, chỉ để được đo bởi MCU. Xin thứ lỗi cho các bong bóng -- khi được phóng to, chúng hiển thị các giá trị byte đang được truyề`n đi. Như chúng ta có thể thấ´y, MCU không hoạt động với CTS thấ´p (sẵn sàng nhận). Một ký tự xuấ´t hiện và MCU tăng CTS (dừng truyề`n thêm từ PC) trong khi nó xử lý byte nhận được. Trong trường hợp này, chương trình chạy trên MCU sẽ gửi một số´ dữ liệu trở lại PC mô tả byte nhận được. Khi đã xong, MCU hạ thấ´p CTS để nhận một byte khác, tăng CTS khi nhận được byte đó và lặp lại quy trình. Bằ`ng cách này, chúng tôi nhận được mọi thứ mà PC muố´n gửi -- không có gì bị mấ´t do tràn bộ đệm MCU.

Đây là một số´ mã mẫu, được viế´t cho ATMega644:

```
#include <avr/io.h>

#define UBRR_ASYNC_NORMAL(baud) \
  (((((F_CPU * 10) / (16L * tốc độ truyền)) + 5)/10) - 1)

#define RTS PD3
#define CTS PD2

#define BAUD 9600

tĩnh int uart_haschar(void) {
  quay lại (UCSR0A & (1 << RXC0));
}

tĩnh int uart_getchar(void) {
  trong khi (!uart_haschar());
  trả lại UDR0;
}

int main(void) {
  UBRR0H = UBRR_ASYNC_NORMAL(BAUD) >> 8;
  UBRR0L = UBRR_ASYNC_NORMAL(BAUD);
  UCSR0B = (1 << RXEN0) | (1 << TXEN0);
  UCSR0C = (0 << USBS0) | (3 << UCSZ00);
```

```
  DDRB = 0b11101111; // PB4 = MISO
  DDRD = 0b11110110; // PD0 = RX, PD3 = RTS

  // Chặn CTS bằng cách đặt nó thành TTL 1 (tắt RS232)
  PORTD |= (1 << CTS);

  vì (;;) {
    // Chờ máy đặt RTS (đặt thành TTL 0)
    trong khi ((PIND & (1 << RTS)) != 0);

    // Đặt CTS (TTL 0 == RS232 bật)
    CỔNG &= ~(1 << CTS);

    char c = uart_getchar();

    // Ta có một ký tự, nên xóa CTS (TTL 1 == RS232 tắt)
    PORTD |= (1 << CTS);

    // Làm gì đó với c
  }

  trả về 0;
}
```

Tấ´t nhiên, có những cách khác để bảo vệ chố´ng tràn bộ đệm. Chúng tôi có thể đọc từ UART bằ`ng trình xử lý ngă´t, nhưng điề`u đó a) làm phức tạp mã đáng kể và b) chỉ hoạt động nế´u chúng tôi có thể theo kịp tố´c độ dữ liệu không được kiểm soát trong thời gian dài. Có lẽ thực tế´ hơn, chúng tôi có thể cấ´u trúc các lệnh của mình tới MCU sao cho tấ´t cả chúng đề`u có phản hồ`i, loại bỏ trách nhiệm bảo vệ chố´ng tràn dữ liệu lên PC. Tuy nhiên, điề`u đó sẽ không hiệu quả nế´u chúng ta muố´n bộ vi xử lý có thể tạo các thông báo của riêng nó một cách không đồ`ng bộ.

Đã cập nhật: Đã làm rõ thuật ngữ cho ATMega/AVR/MCU, bổ sung sơ đồ`, cố´ định số´ chân T1IN, mô tả cách khác để xem bă´t tay RTS/CTS để sử dụng hai chiề`u.
Đã cập nhật: Đã thay đổi cài đặt ban đầ`u của PORTD để nó chỉ đặt bit CTS. Không thực sự cầ`n thiế´t, vì không có gì khác trong mã này sẽ bị ảnh hưởng bởi nó, nhưng nó sẽ giúp ích cho bấ´t kỳ ai cố´ gắ´ng cắ´t và dán.

ĐĂNG BỞI SIMMONMT LÚC 8:16 AM

NHÃN: ĐIỆN TỬ

17 BÌNH LUẬN:

hòm Ngày 16 tháng 5 năm 2011 lúc 3:49 chiề`u

Bạn đã đánh mất tôi ngay từ đầ`u :(

"""Đây là những gì tôi đang cố´ gắ´ng thực hiện: Tôi có một cổng nố´i tiế´p DB-9 được nố´i với ATMega644, với một bộ thu phát RS-232 SP3232EB ở giữa."""

Vì vậy có phải là: máy tính <-> SP3232EB <-> ATMega644 không?

và sau đó bạn nói:

"""Khi chúng ta nói về` RX và TX, chúng ta sẽ làm điề`u đó từ quan điểm của AVR."""

AVR là gì và ở đâu cái này nhập hỗn hợp

?

Hồ`i đáp

trả lời

Owen DeLong Ngày 9 tháng 3 năm 2012 lúc 1:22 sáng

AVR là tên viế´t tắ´t của các kỹ sư đã phát triển chip ATMega đầ`u tiên và là biệt danh chung cho dòng chip MCU vi điề`u khiển ATMega.

Hồ`i đáp

simmonmt Ngày 16 tháng 5 năm 2011 lúc 3:56 chiều

PC nói RS-232, với các mức RS-232. Bộ vi điều khiển (ATMega644) sử dụng các mức TTL. SB3232EB nằm giữa hai loại này và dịch giữa RS-232 và TTL. Xem hình bên dưới "Đây là chữ p" để thấy sự khác biệt giữa hai chữ này.

AVR là tên của kiến trúc được Atmel sử dụng cho dòng vi điều khiển mà tôi đang sử dụng. Vì vậy, trong trường hợp này, AVR và ATMega gần như đồng nghĩa. Tôi sẽ viết lại văn bản để làm cho nó rõ ràng hơn.

Hồi đáp

điên cuồng Ngày 17 tháng 5 năm 2011 lúc 10:55 sáng

Tôi nghĩ có một lỗi đánh máy trong bảng sơ đồ chân... bạn có chân 12 cho cả TTL vào và TTL ra.
Tôi đoán một trong số này phải là chân 11

Hồi đáp

simmonmt Ngày 17 tháng 5 năm 2011 lúc 11:04 sáng

@Maniac: Đã sửa. Cảm ơn!

Hồi đáp

CVBruce Ngày 17 tháng 5 năm 2011 lúc 11:34 sáng

"Người ta tưởng tượng rằng người gửi được phép hoàn thành biểu tượng (byte) hiện tại, nhưng tôi chưa xác minh điều đó."

Tôi đồng ý với tuyên bố đó, đơn giản là vì bản chất của giao thức. Ví dụ 8N1 có nghĩa là một bit bắt đầu, 8 bit dữ liệu, không có bit chẵn lẻ và một bit dừng. Các bit bắt đầu và dừng được sử dụng để đồng bộ hóa máy phát và máy thu. Nếu bạn dừng và bắt đầu luồng bit, thì quá trình đồng bộ hóa đó sẽ bị mất.

Thứ hai, và điều này có thể tiết lộ sự thiếu hiểu biết của tôi, nhưng tôi chưa bao giờ nghe nói về một UART không có bộ đệm gửi và nhận ít nhất một byte.

Hồi đáp

Ag Ngày 17 tháng 5 năm 2011 lúc 11:42 sáng

Tôi muốn làm rõ mục nhập blog của bạn.

RTS/CTS không phải là tín hiệu yêu cầu/xác nhận.

RTS được sử dụng để đo luồng dữ liệu từ DCE (AVR) đến DTE (PC).

CTS được sử dụng để đo luồng dữ liệu từ DTE (PC) đến DCE (AVR).

Trong ví dụ của bạn, PC luôn sẵn sàng chấp nhận dữ liệu, vì vậy RTS luôn được xác nhận. Nhưng AVR không phải lúc nào cũng sẵn sàng nhận dữ liệu, vì vậy CTS sẽ báo cho PC ngừng gửi dữ liệu cho đến khi AVR sẵn sàng.

Hồi đáp

simmonmt Ngày 17 tháng 5 năm 2011 lúc 12:09 chiều

@CVBruce: ATMega có ít nhất một byte, nhưng trong trường hợp của tôi, một byte là không đủ. Giả sử tôi đang viết một cái gì đó trở lại PC để đáp lại lệnh mà nó vừa gửi. Hoặc có lẽ một số sự kiện không đồng bộ vừa xảy ra và tôi đang nói với PC về điều đó. Trong cả hai trường hợp, tôi sẽ bận làm việc khác (ghi vào PC) và sẽ không kiểm tra dữ liệu đến. Nếu một lệnh nhiều byte xuất hiện trong thời gian đó, tôi sẽ bỏ lỡ ít nhất một phần của nó. Bắt tay phần cứng cho phép tôi tránh điều đó.

@ag: RTS/CTS dường như có thể sử dụng được ở cả chế độ yêu cầu/ack mà tôi đã mô tả và đo sáng hai chiều mà bạn đang mô tả. Xem trang wikipedia cho RS232 (http://en.wikipedia.org/wiki/RS-232#RTS.2FCTS_handshaking) để biết mô tả.

Hồi đáp

Vô danh Ngày 18 tháng 5 năm 2011 lúc 5:25 sáng

Bạn có thể thêm chi tiết vào hình thứ hai (chữ "p") để hiển thị các bit khác nhau ở đâu không?

Hồi đáp

Owen DeLong Ngày 9 tháng 3 năm 2012 lúc 2:43 sáng

Đăng nhiều phần do giới hạn 4k

không đọc đĩa mềm được ghi trong ổ đĩa của một nhà sản xuất trong ổ đĩa của một số nhà sản xuất khác. Lạc đề đủ rồi... RS232.

Đầu tiên, tôi nghĩ chúng ta nên nói về truyền thông nối tiếp nói chung. Trái tim của truyền thông

nô´i tiê´p bao gô`m bâ´t kỳ cơ chê´ báo hiệu nào, dù là hai chiê`u, một chiê`u hay đa điểm, song công hoàn toàn hay bán song công, trong đó sô´ lượng đường dẫn tín hiệu song song ít hơn sô´ lượng bit chứa trong mỗi ký hiệu được mã hóa. Thông thường, nê´u bạn nói nô´i tiê´p, mọi người sẽ giả sử một đường dẫn dữ liệu bit đơn lẻ hoặc nhiê`u nhâ´t là một đường dẫn dữ liệu bit đơn lẻ theo mỗi hướng. Tuy nhiên, ngược lại với nô´i tiê´p là song song, khi bạn sử dụng thuật ngữ đó, mọi người giả sử ít nhâ´t 8 và thường là 16, 32, 64 hoặc thậm chí nhiê`u bit tín hiệu đô`ng thời. Có nhiê`u thiê´t bị sử dụng đường dẫn tín hiệu rộng 2 hoặc 4 bit và hành vi của chúng kê´t hợp các yê´u tô´ của cả hành vi nô´i tiê´p và song song, nhưng phâ`n lớn bị chi phô´i bởi những khó khăn vô´n có trong nô´i tiê´p.

Điê`u thú vị là, khi tô´c độ dữ liệu tăng lên, nhiễu xuyên âm đã trở thành một vâ´n đê` ngày càng nghiêm trọng đê´n mức nô´i tiê´p đường dữ liệu bit đơn trở thành đường dẫn nhanh chiê´m ưu thê´ hơn đô´i với bâ´t kỳ thứ gì rời khỏi môi trường được kiểm soát chặt chẽ của PCB nhiê`u lớp được thiê´t kê´ cẩn thận. Một lâ`n nữa, tôi lạc đê`...

Nô´i tiê´p có hai hương vị chính. Đô`ng bộ và Không đô`ng bộ. RS-232 thường được sử dụng để mô tả hành vi Không đô`ng bộ và trên thực tê´, hâ`u hê´t các cổng Nô´i tiê´p PC không có khả năng thực hiện giao tiê´p đô`ng bộ thực sự mặc dù 16550 USART thực sự có khả năng làm như vậy. Râ´t ít thẻ PC mang tâ´t cả các chân câ`n thiê´t từ 16550 để làm như vậy.

Hô`i đáp

Owen DeLong Ngày 9 tháng 3 năm 2012 lúc 2:44 sáng

Phâ`n 2

Trong nô´i tiê´p đô`ng bộ, tô´c độ baud gâ`n như không liên quan vì một trong các hệ thô´ng cung cấ´p tín hiệu đô`ng hô` và tâ´t cả việc truyê`n theo cả hai hướng đê`u được đô`ng bộ hóa với tín hiệu đô`ng hô` đó. Ưu điểm của một hệ thô´ng như vậy là nó không dễ bị trôi đô`ng hô` vì hệ thô´ng gửi sẽ chỉ đợi đô`ng hô` thay đổi trạng thái trước khi thay đổi trạng thái cho bit tiê´p theo và hệ thô´ng nhận sẽ không bă´t đâ`u trì hoãn đọc cho đê´n khi nó nhìn thấ´y đô`ng hô` thay đổi trạng thái.

Trong nô´i tiê´p không đô`ng bộ, mỗi hệ thô´ng duy trì "đô`ng hô` baud" bên trong của riêng mình và giao tiê´p phụ thuộc vào hai hệ thô´ng có đặt đô`ng hô` chạy ở tô´c độ gâ`n như nhau. Lưu ý, đô`ng hô` baud này không giô´ng với đô`ng hô` CPU. Để đô`ng bộ hóa khung, mỗi biểu tượng được truyê`n không đô`ng bộ được bao bọc trong phâ`n mở đâ`u và hậu tô´. Phâ`n mở đâ`u hâ`u như luôn là một bit Đánh dấ´u (logic 1) duy nhâ´t và hậu tô´ thường là 1 hoặc 2 (thường là 1) bit Đánh dấ´u (logic 1). (Logic 0 được gọi là Khoảng tră´ng).

Các biểu tượng (thường) được truyê`n từ phòng thí nghiệm đê´n MSB bă`ng cách sử dụng các thanh ghi dịch chuyển được định thời bởi đô`ng hô` baud. Trên các máy điện báo ban đâ`u, các thanh ghi dịch chuyển này là các thiê´t bị cơ điện và bit bă´t đâ`u thực sự khiê´n thanh ghi đặt lại vê` tâ´t cả các sô´ 0 và (các) bit dừng khiê´n thanh ghi chô´t, đặt đâ`u in vào đúng ký tự và kích hoạt bộ gõ để tạo â´n tượng trên trang giấ´y. (nhiê`u hoạt động này là lý do mà 2 bit dừng được sử dụng phổ biê´n.

RS-232 là một thông sô´ kỹ thuật không thay đổi đáng kể kể từ những ngày nó được phát triển lâ`n đâ`u tiên để điê`u khiển điện báo trên một khoảng cách dài bă`ng cách sử dụng các thiê´t bị điê`u biê´n/giải điê`u chê´ có khóa dịch chuyển tâ`n sô´ (modem). (Trong FSK, hai tâ`n sô´ được chọn cho mỗi hướng giao tiê´p, ví dụ: 300/900 hz từ trái sang phải và 600/1200hz từ B đê´n A. âm 300hz biểu thị khoảng cách từ A đê´n B, âm 900 hz đánh dấ´u từ A đê´n B, 600/1200 lâ`n lượt là dấ´u và khoảng tră´ng từ B đê´n A).

RS-232 không sử dụng FSK, tuy nhiên, nó sử dụng điện áp dương và âm. Bởi vì nó được thiê´t kê´ để đi khá xa (3.000 feet trở lên ở tô´c độ thấ´p hơn), điện áp khá cao (so với logic của bóng bán dẫn) và dao động điện áp rộng đã được sử dụng. Dấ´u RS-232 nă`m trong khoảng từ -5 đê´n -15 vôn ở bộ phát và bộ thu phải chấ´p nhận mọi thứ trong khoảng từ -3 đê´n -15 (so với tín hiệu nô´i đâ´t (chân 7 trên đâ`u nô´i RS232 DB-25 điển hình của bạn, chân 5 trên DD- 9 (thường bị gọi nhâ`m là DB-9))) . Ngược lại, khoảng trô´ng là +5 đê´n +15 (+3 đê´n +15 ở phía nhận). Lưu ý rằ`ng đó là mức dao động tô´i đa 30 volt, gấ´p 6 lâ`n mức Vcc trong Arduino điển hình của bạn và gâ`n 10 lâ`n mức logic trong hâ`u hê´t các bộ điê`u khiển vi mô và thiê´t bị ngoại vi.

Để dịch các mức dòng TTL sang RS-232, nhiê`u loại chip "trình điê`u khiển dòng" đã được phát triển, bao gô`m dòng 488/489, 1488/1489 và các tác giả SP3232EB (là trình điê`u khiển dòng công suâ´t thấ´p với dòng điện khá yê´u). dao động điện áp ở mức chênh lệch 13V từ V+ đê´n V-). Một sô´ bộ điê`u hợp USB<->RS-232 giá rẻ thậm chí còn tệ hơn và chỉ tạo tín hiệu +5/-0v RS-232. Điê`u này đã trở nên phổ biê´n đê´n mức hâ`u hê´t các UARTS sẽ (trái với thông sô´ kỹ thuật của RS-232) chấ´p nhận 0 volt (thực tê´ là bâ´t cứ thứ gì <+0,3v) làm dấ´u hợp lệ và bâ´t cứ thứ gì >2,2v làm khoảng trô´ng hợp lệ.

Hô`i đáp

Owen DeLong Ngày 9 tháng 3 năm 2012 lúc 2:45 sáng

Phâ`n 3

Tâ´t nhiên, đặc biệt là vào thời của máy điện báo, việc sử dụng quá mức các thiê´t bị tương đô´i dễ dàng và nhiê`u thiê´t bị trong sô´ đó có bộ đệm râ´t nhỏ (hoặc không có). Điê`u này dẫn đê´n tâ´t cả các loại cơ chê´ giải trí, bao gô`m kiểm soát luô`ng phâ`n mê`m và phâ`n cứng, đo ký hiệu RTS/CTS và các phát minh như đệm (một loạt các ký tự không in được, thường là null, được gửi sau khi xuô´ng dòng để tạo độ trễ trong khi đâ`u in cơ học đã được di chuyển từ mép phải sang mép trái của tờ giấ´y. Nê´u bạn có thể tìm thấ´y một máy điện báo cũ, hãy gửi cho nó một dòng văn bản theo sau là một và ngay lập tức nhiê`u văn bản hơn và bạn sẽ nhận thấ´y rằ`ng 1 hoặc 2 ký tự đâ`u tiên của dòng văn bản thứ hai được in ngẫu nhiên theo thứ tự ngược lại khi đâ`u in quay vê` phía bên trái của trang).

RS-232 tô´i thiểu cơ bản không có điê`u khiển luô`ng yêu câ`u 3 chân. Một tín hiệu nô´i đâ´t và 1 chân

cho dữ liệu theo mỗi hướng.

Điề`u khiển luô`ng phâ`n mề`m cũng chỉ yêu cầ`u 3 chân và quản lý điề`u khiển luô`ng bằ`ng cách gửi các ký tự được chỉ định đặc biệt (XON hoặc Ctrl-Q để Bật truyề`n hoặc "Tiế´p tục gửi" và XOFF hoặc Ctrl-S để Tắ´t truyề`n hoặc "Dừng gửi"). Các ký tự điề`u khiển luô`ng này được gửi từ người nhận đế´n người gửi và người gửi phải tôn trọng chúng ngay khi biểu tượng hiện tại (và hậu tố´ của nó) được chuyển ra ngoài. Tuy nhiên, rủi ro là với cáp dài hơn và tố´c độ bit cao hơn, có thể có nhiề`u hơn một ký hiệu trên dây giữa người gửi và người nhận cùng một lúc. Ngoài ra, vì đây là nố´i tiế´p nên có độ trễ thời gian trong khi biểu tượng XOFF được tải lên dây và độ trễ thời gian trong khi nó cũng được dịch chuyển ra khỏi dây.

Đây là lý do tại sao các thiế´t bị có bộ đệm có "vạch nước" cao và thấ´p. Ở mức nước cao (bộ đệm (đầ`y ~75%), bộ thu truyề`n XOFF. Khi bộ đệm trố´ng, khi nó đạt đế´n mức nước thấ´p (đầ`y ~25%), XON sẽ được gửi. Bằ`ng cách này, bấ´t kỳ biểu tượng bổ sung nào trên dây có thể được đọc vào khoảng không 25% trong bộ đệm và không bị mấ´t do độ trễ trong điề`u khiển luô`ng.

Đô`ng hồ` ký hiệu RTS/CTS thường được sử dụng khi dữ liệu di chuyển chủ yế´u theo một hướng chỉ với các xác nhận hoặc các mẩu thông tin tương đố´i nhỏ trên đường dẫn trở lại (nế´u có). Phương thức báo hiệu này (phương thức được tác giả mô tả) yêu cầ`u ít nhấ´t 2 chân bổ sung. Một cho RTS và một cho CTS. Trong cấ´u hình này, khi đơn vị gửi đã sẵn sàng dữ liệu, nó sẽ thông báo cho máy chủ rằ`ng nó muố´n truyề`n (xác nhận RTS, do đó có tên là "Yêu cầ`u gửi"). Thông thường, điề`u này được sử dụng để kích hoạt một ngắ´t để máy chủ có thể tạm dừng quá trình xử lý bình thường của nó và nhập một thói quen để nhận dữ liệu. Khi máy chủ sẵn sàng nhận (trước đây, điề`u này có thể bao gồ`m việc đặt đầ`u vào đúng vị trí trên bộ lưu trữ từ tính để bộ đệm nhận dữ liệu cầ`n ghi, vì vậy độ trễ có thể là đáng kể về` mặt thời gian của máy tính), người dẫn chương trình nói "Hãy tiế´p tục" (khẳng định CTS, do đó có tên là "Xóa để gửi"). Khi máy chủ nhận rớt CTS, máy gửi bắ´t buộc phải ngừng gửi ngay sau khi nó hoàn thành nội dung hiện tại của thanh ghi thay đổi. Một lầ`n nữa, điề`u này không nhấ´t thiế´t có nghĩa là máy chủ nhận sẽ không nhận được nhiề`u hơn 5 ký hiệu dữ liệu (mặc dù thường chỉ là 1 hoặc 2). Khi người gửi gửi xong, anh ta sẽ bỏ RTS. Điề`u này báo hiệu cho máy chủ rằ`ng ngay khi có khoảng trố´ng trong các biểu tượng đế´n, anh ta có thể tự do quay lại quá trình xử lý bình thường. Cơ chế´ điề`u khiển luô`ng này cũng có thể được sử dụng hai chiề`u trong môi trường bán song công, nơi cả hai bên đề`u giám sát RTS làm đầ`u vào cho đế´n khi một trong sô´ họ muố´n gửi thứ gì đó. Nế´u cả hai đề`u xác nhận RTS cùng một lúc, sẽ không thông báo và CTS sẽ không được khẳng định. Sau khi hế´t thời gian chờ, cả hai máy chủ sẽ tạm dừng trong một khoảng thời gian ngẫu nhiên và thử lại. Điề`u này hiế´m khi được sử dụng vì nó khó thực hiện trong phâ`n cứng thời đó so với hình thức tiế´p theo...

Hô`i đáp

Owen DeLong Ngày 9 tháng 3 năm 2012 lúc 2:46 sáng

Phâ`n thứ tư

Một dạng kiểm soát luô`ng phâ`n cứng tinh vi hơn một chút, hữu ích trong môi trường ngang hàng nơi luô`ng dữ liệu có thể không đố´i xứng mạnh và có thể đồ`ng thời theo cả hai hướng là sử dụng hai chân giố´ng nhau (RTS/CTS) này để biểu thị mức độ sẵn sàng của thiế´t bị . Nói chung, mỗi đơn vị sẽ khẳng định CTS (Xóa để gửi) làm đầ`u ra để cho biế´t rằ`ng nó sẵn sàng nhận các ký hiệu bổ sung và sẽ theo dõi RTS để được phép truyề`n. Ngoại lệ là modem hoặc "thiế´t bị liên lạc dữ liệu". Bạn sẽ thường nghe các thuật ngữ trong RS-232, chẳng hạn như "DTE" hoặc "Thiế´t bị đầ`u cuố´i dữ liệu", "Cáp modem", "Cáp modem đầ`y đủ", "Cáp Null Modem", "DCE" hoặc "Thiế´t bị truyề`n dữ liệu". Các thuật ngữ này liên quan đế´n cách sắ´p xế´p các chân trên đầ`u nố´i. Nói chung là, nế´u bạn muố´n sử dụng cáp có sơ đồ` chân giố´ng hệt nhau ở cả hai đầ`u, bạn cầ`n kế´t nố´i thứ gì đó có dây cho DTE với thứ gì đó có dây là DCE. DTE bao gồ`m những thứ như máy tính, thiế´t bị đầ`u cuố´i, máy in, hầ`u hế´t mọi thứ kế´t thúc (do đó là "Thiế´t bị đầ`u cuố´i dữ liệu") kế´t nố´i và thực hiện điề`u gì đó với dữ liệu ngoài việc chuyển đổi dữ liệu đó để truyề`n qua một sô´ dạng mạch khác. DCE thường có nghĩa là modem và các bộ chuyển đổi sơ đồ` truyề`n tải khác. (do đó có thuật ngữ "Thiế´t bị truyề`n thông dữ liệu"). Cáp modem có nghĩa là cáp bao gồ`m ít nhấ´t TX, RX, RTS, CTS, SG (Signal Ground) và CD (Carrier Detect) với sơ đồ` chân giố´ng hệt nhau ở cả hai bên. Thông thường cáp modem cũng sẽ bao gồ`m ít nhấ´t DTR (Data Terminal Ready).

Cáp modem null phải bao gồ`m ít nhấ´t TX, RX, RTS, CTS, SG, CD và DTR. Nó phải được nố´i dây sao cho A-TX = B-RX, A-RX = B-TX, A-RTX = B-CTS, A-CTS = B-RTS, A-SG = B-SG, A-CD = B-DTR và A-DTR = B-CD. Thực tế´, điề`u này đảo ngược từng bộ tín hiệu miễn phí giữa hai đầ`u cáp, hoán đổi RX/TX, RTS/CTS, DTR/CD. Tấ´t nhiên, thực tế´ không đơn giản như vậy và thông sô´ kỹ thuật thực tế´ của modem null ít chính xác hơn nhiề`u và rấ´t khác nhau trong việc xử lý RTS, CTS, DSR, DTR và CD với nhiề`u cách kế´t hợp ánh xạ chân khác nhau thể hiện một loạt các khác biệt trong DCE đang được thay thế´ bằ`ng cáp (ban đầ`u RS-232 được thiế´t kế´ với giả định rằ`ng đó sẽ là Thiế´t bị->Modem->đường dây điện thoại->Modem->Thiế´t bị. Tấ´t cả -> cáp sẽ đi thẳng (mặc dù cáp đường dây điện thoại không tuân theo thông sô´ kỹ thuật RS-232. và thực tế´ dự kiế´n sẽ bị lật). Thuật ngữ null modem xuấ´t phát từ ý tưởng rằ`ng cáp đang thay thế´ hai modem và một đường dây điện thoại.

Hô`i đáp

Owen DeLong Ngày 9 tháng 3 năm 2012 lúc 2:47 sáng

Phâ`n cuố´i cùng

Trong quá trình triển khai tín hiệu đầ`y đủ của modem phổ biế´n nhấ´t cho Không đồ`ng bộ, các chân có ý nghĩa sau trên đầ`u nố´i DB-25:

1 Khung hoặc Nố´i đấ´t bảo vệ. Nên được nố´i với tấ´m chắ´n cáp TẠI MỘT ĐÂ`U của cáp nế´u có tấ´m chắ´n.
2 TXD (Dữ liệu từ thiế´t bị đế´n modem)
3 RXD (Dữ liệu đế´n thiế´t bị từ modem)
4 RTS (Thiế´t bị cho biế´t sẵn sàng chấ´p nhận nhiề`u ký hiệu hơn từ modem)
5 CTS (Modem cho biế´t sẵn sàng chấ´p nhận thêm ký hiệu từ thiế´t bị)

6 DSR (Modem cho biết nó sẵn sàng chấp nhận lệnh hoặc dữ liệu)[1]
7 SG (Signal Ground - Tham chiếu điện áp bằng 0 cho tất cả các tín hiệu)
8 CD (đôi khi được gọi là DCD hoặc Phát hiện sóng mang dữ liệu. Cho biết rằng modem được liên kết với một modem từ xa)
20 DTR (Data Terminal Ready - Máy chủ cho biết modem đang trực tuyến)[2]
22 RI (Chỉ báo chuông - Được xác nhận khi có điện áp chuông trên đường dây điện thoại)

[1] DSR được sử dụng chủ yếu trong các hoạt động ngoại tuyến để chỉ báo rằng modem sẵn sàng chấp nhận các lệnh mặc dù nó không được kết nối với modem từ xa. Đây thường là mối quan tâm đối với những thứ như tập lệnh quay số để tạo kết nối ra bên ngoài.

[2] Khi máy chủ kết thúc kết nối, máy chủ thường sẽ Hủy xác nhận DTR trong khoảng thời gian ≥250 mili giây để cho modem biết rằng máy chủ nên chấm dứt mọi cuộc gọi đang hoạt động và đặt lại về trạng thái ban đầu đã biết.

Đối với dự án thuộc loại CVBruce đã mô tả, tôi sẽ sử dụng điều khiển luồng RTS/CTS ở chế độ hai chiều. Tôi sẽ đảm bảo rằng tôi đã giữ CTS ở mức thấp trong bất kỳ thời điểm nào mà tôi chưa sẵn sàng để đọc cổng nối tiếp. FWIW, nếu bạn sử dụng bộ điều khiển vi mô PIC, USART tích hợp có khả năng tự động quản lý RTS/CTS.

Hồi đáp

Owen DeLong Ngày 9 tháng 3 năm 2012 lúc 2:52 sáng

Ngoài ra, trong mã của bạn, PORTD = (1 << CTS); có lẽ nên đọc PORTD |= (1 << CTS);

Như bạn đã viết, bạn đang đặt CTS thành 1 và mọi chân khác trên PORT D thành 0.

Nếu bạn chỉ muốn thay đổi CTS, bạn có thể nhận được kết quả khác với mong muốn.
Nếu bạn không có bất kỳ thứ gì khác được đính kèm với 7 GPIO khác trên PORT D, điều đó có thể không thành vấn đề, nhưng vì bạn đã đặt nó làm mã ví dụ. nó có thể gây nhầm lẫn cho người khác.

Hồi đáp

trả lời

simmonmt ngày 3 tháng 4 năm 2012 lúc 8:59 sáng

Đã sửa. Cảm ơn!

Hồi đáp

trục ngày 23 tháng 7 năm 2012 lúc 3:42 sáng

Wow Owen: cảm ơn vì lời giải thích lịch sử và cực kỳ chi tiết. Tôi chưa bao giờ biết lý do của hai lần dừng lỗ: bây giờ tôi đã biết.

Hồi đáp

To leave a comment, click the button below to sign in with Blogger.

SIGN IN WITH BLOGGER

Bài mới hơn Trang Chủ Bài cũ

Đăng ký: Đăng nhận xét (Atom)