RS485 Transmit Data

```
int EN = 2; //RS485 has a enable/disable pin to transmit or
receive data. Arduino Digital Pin 2 = Rx/Tx 'Enable'; High to
Transmit, Low to Receive
††
void setup()†
{
††pinMode(EN, OUTPUT);
††Serial.begin(19200);
}†
††
void loop()
{
// send data†
††digitalWrite(EN, HIGH);//Enable data transmit
††Serial.print('A');
††delay(1000);
}
```

RS485 Receiving Data

```
int ledPin = 13;
int EN = 2;
int val;
††
void setup()†
{
††pinMode(ledPin, OUTPUT);
††pinMode(EN, OUTPUT);
††Serial.begin(19200);
}
††
void loop()
{
††// receive data
†digitalWrite(EN, LOW);//Enable Receiving Data
†val = Serial.read();
††if (-1†!= val) {
††††if ('A' == val) {
††††††digitalWrite(ledPin, HIGH);
††††††delay(500);
††††††digitalWrite(ledPin, LOW);
††††††delay(500);
††††}
```

```
†}
}
```

Sketch per la trasmissione

```
//RS485 has a enable/disable pin to transmit or receive data.
Arduino Digital Pin 2 = Rx/Tx 'Enable'; High to Transmit, Low
to Receive

int EN = 2;
void setup()
{
  pinMode(EN, OUTPUT);
  Serial.begin(19200);
}
void loop()
{
// send data
digitalWrite(EN, HIGH);//Enable data transmit
Serial.print('A');
delay(1000);
}
```

Sketch per la ricezione

```
int ledPin = 13;
int EN = 2;
int val;
void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(EN, OUTPUT);
  Serial.begin(19200);
}
void loop()
{
  // receive data
digitalWrite(EN, LOW);//Enable Receiving Data
val = Serial.read();
if (-1 != val)
{
  if ('A' == val)
  {
    digitalWrite(ledPin, HIGH);
```

```
    delay(500);
    digitalWrite(ledPin, LOW);
    delay(500);
    }
  }
}
```

Programma per la ricezione di informazioni a pacchetti.
Il led collegato al pin digitale 9,si accende solo se lí informazione Ë corretta.

Librerie necessarie per lí utilizzo dellí interrupt del timer.

```
#include <avr/io.h>
#include <avr/interrupt.h>
#define USART_BAUDRATE 9600
#define BAUD_PRESCALE (((F_CPU / (USART_BAUDRATE * 16UL))) -
1)

int a;
int stato=0;
int c=0;
int Byte_Ricevuti=0;
int LunghPack;
boolean flag_RX=false;
byte DatiRX[20];
int tcnt2;
int toggle=0;
int b=0;
int TIMEOUT;
boolean Ab_TIMEOUT;

void setup()

{
  pinMode(9,OUTPUT);
  UCSR0B |= (1 << RXEN0) | (1 << TXEN0);   // Accendere il
circuito di trasmissione e ricezione
  UCSR0C |=(1 << UCSZ00) | (1 << UCSZ01); // Utilizzare
caratteri a dimensioni di8 bit
  UBRR0L = BAUD_PRESCALE; // Caricare gli 8 bit inferiori del
baud rate nei byte inferiori del registro UBRR
  UBRR0H = (BAUD_PRESCALE >> 8); // Caricare gli 8 bit
superiori del baud rate nei byte superiori del registro UBRR
  UCSR0B |= (1 << RXCIE0); // Attivare la USART alla ricezione
completa dell' interrupt (USART_RXC)
  TIMSK2 &=~(1<<TOIE2);
  TCCR2A &=~((1<<WGM21)|(1<<WGM20));
```

```
  TCCR2B &=~(1<<WGM22);
  ASSR &=~(1<<AS2);
  TIMSK2 &=~(1<<OCIE2A);
  TCCR2B |=(1<<CS22) | (1<<CS20);
  TCCR2B &=~(1<<CS21);

  /* Abbiamo bisogno di calcolare un valore corretto per
impostare il cronometro.
   * Impostare il valore 131 nel registro 2 timer counter
   * La matematica Ë alla base di questo:
   * (frequenza CPU) / (valore di prescaler ) = 125000 Hz =
8us.
   * (periodo desiderato) / 8us = 125.
   * MAX(uint8) + 1 - 125 = 131;
   */

  tcnt2=131;
  TCNT2=tcnt2;
  TIMSK2 |=(1<<TOIE2);
  sei();   // Attivare l'opzione Abilita Global Interrupt flag
in modo che gli interrupt possono essere trattati

}
ISR(USART0_RX_vect)   /* intestazione routine d'interrupt
dell'interfaccia seriale     */
{

  if (Byte_Ricevuti==0)
  {
    LunghPack=(unsigned int)UDR0;  /* rappresenta la lunghezza
del pacchetto         */
    Byte_Ricevuti=1;              /* ricevuto il primo dato
*/
    TIMEOUT=0; Ab_TIMEOUT=true;
  }
  else         /* se invece la condizione non Ë vera il
carattere ricevuto  */
  {
    DatiRX[Byte_Ricevuti]=UDR0;     /* Ë un dato del pacchetto
*/

    if (Byte_Ricevuti==LunghPack) /* controlla se il pacchetto
Ë completo  */
    {
      flag_RX=true; /* segnala al main il pacchetto completo
*/
       Ab_TIMEOUT=false;
    }
   ++Byte_Ricevuti;
  }
```

```
}

ISR(TIMER2_OVF_vect)
{
  TCNT2=tcnt2;
  if(Ab_TIMEOUT==true)
  {
    ++TIMEOUT;
    if(TIMEOUT==15)
    {
      Byte_Ricevuti=0;
      TIMEOUT=0;
    }
  }

}


void loop()
{
  if (flag_RX==true)
  {
    if(c==0)
    {
      c=1;
      digitalWrite(9,HIGH);
    }
    else
    {
      c=0;
      digitalWrite(9,LOW);
    }
    flag_RX=false;
    Byte_Ricevuti=0;
  }

}
```