LABORATORIO DIDATTICO 5 Pilotaggio di un LED remoto

Pilotare un LED remoto via radio, mediante moduli XBee, in modo che premendo il pulsante ON il LED si accenda e premendo il pulsante OFF il LED si spenga.

Soluzione

Le due stazioni, trasmittente e ricevente, sono composte da (Figura 1):

- un Arduino UNO, caricato con lo sketch opportuno;
- un modulo XBee Serie 1, che è già configurato di default per operare in modalità *peer-to-peer* trasparente, realizzando un collegamento radio punto-punto senza bisogno di effettuare configurazioni;
- una scheda *SparkFun XBee Explorer Regulated*, per interfacciare XBee con Arduino.

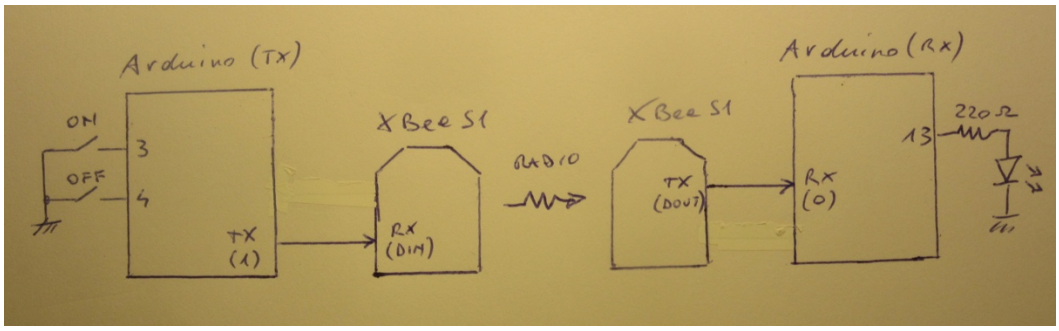

Figura 1 – Schema circuitale.

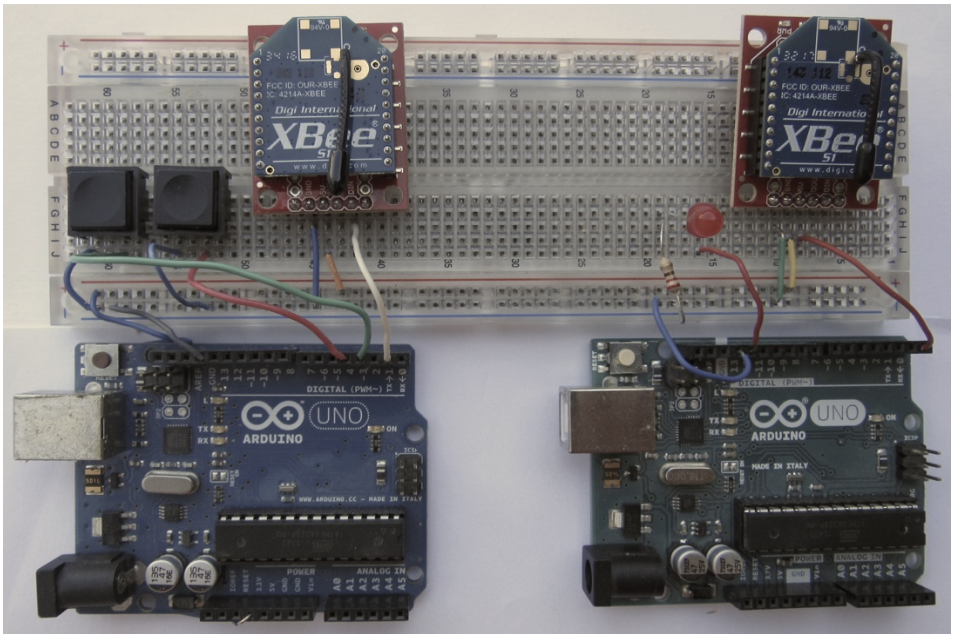

Figura 2 – Schema di montaggio.

Lo sketch del trasmettitore

Individua il pulsante premuto e invia sulla seriale il carattere corrispondente (ON ® “a”, OFF ® “1”).

```
const int pinPulsOn = 3;
const int pinPulsOff = 4;

void setup (){
  Serial.begin (9600);          // impostazione velocità di comunicazione seriale
  pinMode (pinPulsOn, INPUT_PULLUP);
  pinMode (pinPulsOff, INPUT_PULLUP);
}
void loop (){

  if (digitalRead(pinPulsOff) == LOW){       // se è stato premuto il pulsante OFF
    Serial.write ("1");                      // invia il carattere "1"
  }
  else{
    if (digitalRead(pinPulsOn) == LOW){      // se è stato premuto il pulsante ON
      Serial.write ("a");                    // invia il carattere "a"
    }
  }
}
```

Lo sketch del ricevitore

Individua il carattere ricevuto dalla seriale e accende o spegne il LED (“1” ® ON, “a” ® OFF).

```
char carattere;               // variabile che memorizza il carattere ricevuto via XBee (seriale)
const int pinLed = 13;

void setup (){
  Serial.begin (9600);      // impostazione velocità di comunicazione seriale
  pinMode (pinLed, OUTPUT);
}
void loop (){
  carattere = Serial.read();            // acquisizione del carattere inviato via radio (XBee)
  if (carattere == '1'){                // se il carattere inviato è "1"
    digitalWrite (pinLed, HIGH);   // accendi il LED
  }
  if (carattere == 'a'){                // se il carattere inviato è "a"
    digitalWrite (pinLed, LOW);    // spegni il LED
  }
}
```

Si verifichi che il funzionamento del circuito è identico a quello che si ottiene collegando con un cavo il pin 1 (Tx) dell'Arduino trasmettitore con il pin 0 (Rx) del ricevitore; nella trasmissione *point-to-point* l'impiego dei moduli XBee ha la funzione di sostituire il cavo seriale con una trasmissione seriale via radio.

Effettuando semplici modifiche si provi a rendere simmetrico il circuito e il software caricato nelle due schede Arduino, in modo che la trasmissione sia bidirezionale: premendo i pulsanti ON/OFF di un Arduino si pilota il LED dell'altro.

Configurazione dei moduli XBee

Per configurare un modulo XBee è necessario procurarsi:

- un adattatore USB (*XBee–USB adapter*) per collegare XBee al computer (Figura 3a);
- un software per inserire le impostazioni: X-CTU (Figura 3b) è il software che si può scaricare gratuitamente dal sito della Digi, la casa produttrice dei moduli XBee, ma esiste anche XBeeConfigTool della Funnel.

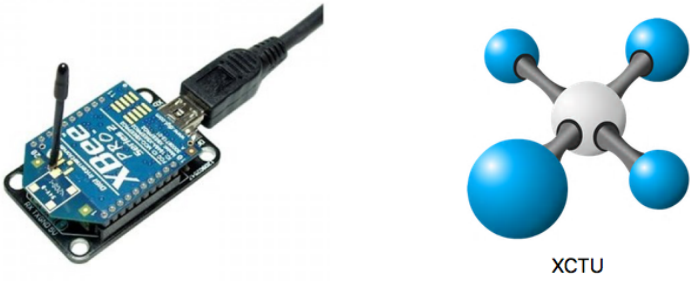

Figura 3 – a) XBee–USB adapter; b) icona del software X-CTU.

I moduli XBee Serie 1 sono configurati di default in modalità *peer-to-peer* trasparente, consentendo di realizzare un collegamento punto-punto senza bisogno di configurazioni.

Mediante il software X-CTU è possibile:

- leggere il numero seriale di un modulo XBee, (*Serial Number High – SH e Serial Number Low – SL*);
- modificare il *baudrate*, che di default vale 9600 bps;
- aggiornare il firmware;
- effettuare test di comunicazione;
- impostare il "ruolo" del modulo: *coordinator*, *router* o *end device*.

Con il software X-CTU si può anche scegliere la modalità di funzionamento:

- trasparente (AT): il dato ricevuto sul pin DIN viene trasmesso via radio, mentre quello ricevuto via radio viene passato sul pin DOUT;
- API (*Application Programming Interface*): consente di creare reti più complesse, organizzando i dati in frame costituiti da quattro campi: 1) un byte delimitatore; 2) due byte che contengono la lunghezza del frame; 3) i byte di dati da trasmettere; 4) un byte di *checksum* per la rivelazione degli errori di trasmissione.