Moduli Bluetooth 02 - programmazione AT

Author: Mirco Piccin aka pitusso

La programmazione AT permette di cambiare alcuni parametri dei moduli BT.

Differenze tra moduli

Esistono varie versioni di questi moduli, anche se in alcuni casi sono identici per aspetto
La differenza risiede nel firmware e quindi nelle loro funzionalità e modalità di programmazione.

- I moduli **HC05** hanno un ampio set di comandi AT a disposizione, compersi quelli per cambiare modalità (Master/Slave); è necessario **inviare LF e CR (\r\n)** alla fine di ogni comando AT.
La programmazione è possibile solo dopo aver messo in HIGH il pin relativo alla programmazione PIO11 (34)

- i moduli **HC06** hanno set di comandi AT limitato; con tale fw non è necessario inviare LF o CR (\r\n) alla fine del comando AT.
La programmazione può essere fatta a patto che non ci sia una connessione in atto.
I comandi vanno scritti su editor, copiati e incollati sulla console: questo fw effettua un polling continuo, pertanto è importante impartire un comando al secondo ca. [1].
Vengono distribuiti nelle versioni Master o Slave ma non è possibile passare da una modalità all'altra.
Il firmware è noto come "linvor", e questo è il nome di default assegnato.

Per tutti i moduli (compreso HC07), la configurazione di base è:
baud rate 9600 (HC06 e HC07) / 38400 (HC05), data 8 bit, stop bit 1, no parity,no flow control.
Pin per il pairing: 1234

Per gli esempi Linux viene utilizzata una macchina Debian Squeeze, con Gnome 2.3 come desktop manager.
Per gli esempi Windows viene utilizzata una macchina Windows 7.
Per gli esempi Arduino, viene utilizzato l'IDE 1.0 e una Arduino 2009 con bootloader originale.

con Arduino

Il modo più semplice per passare comandi AT con Arduino, è utilizzare uno sketch che inoltri al modulo i comandi impartiti da Serial monitor.
Per poter utilizzare il Serial monitor (e quindi la seriale hw), e contemporaneamente instaurare una connessione seriale con il modulo BT, si ricorre ad una Software Serial, ovvero una seriale sw.
Se si utilizza un IDE antecedente al 1.0, consiglio di utilizzate la NewSoftSerial [8].
Dalla versione dell'IDE 1.0 in poi, la libreria SoftwareSerial [9] è di fatto la NewSoftSerial, che ha sostituito la libreria precedente.

Per approfondimenti, rimando alla pagina di reference della libreria [9].

Come codice si può utilizzare questo:

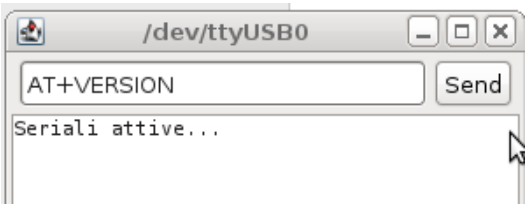
```
#include <SoftwareSerial.h>     //IDE >= 1.0
//#include <NewSoftSerial.h>    //IDE <= 0023

const int rxPin = 2;
const int txPin = 3;
const int atPin = 4;

SoftwareSerial bluetooth(rxPin, txPin);  //IDE >= 1.0
//NewSoftSerial bluetooth(rxPin, txPin);   //IDE <= 0023

void setup() {

  Serial.begin(9600);
  bluetooth.begin(9600);
  pinMode(atPin, OUTPUT);

  Serial.println("Seriali attive...");
  digitalWrite(atPin, HIGH);
}

void loop() {
  if (bluetooth.available()) {
    Serial.write(bluetooth.read());
  }

  if (Serial.available()) {
    bluetooth.write(Serial.read());
  }
}
```

che altro non è che l'esempio fornito a corredo della libreria SoftwareSerial, con poche modifiche.
Connettiamo il modulo BF ad Arduino (con Arduino non alimenata), secondo quanto descritto nella sezione HW.

In particolare, utilizzando una seriale sw, utilizzeremo pin di Arduino diversi da RX (D0) e TX (D1).
In questa riga di codice

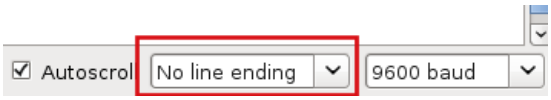
```
SoftwareSerial bluetooth(rxPin, txPin);  //IDE >= 1.0
```

dichiariamo come pin RX e TX rispettivamente il pin D2 e D3.
Una volta caricato lo sketch, e aperto il Serial Monitor, potremo passare i nostri comandi AT al modulo, e leggerne la risposta.



Aperto il serial monitor, invio il comando
AT+VERSION
al modulo BT

Leggo quindi la risposta:
OKlinvorV1.5
(in questo esempio sto utilizzando il modulo HC06 con fw linvor)



Per l'utilizzo con moduli HC06, selezionare "No line ending" nella finestra del Serial Monitor

Nel caso di utilizzo con modulo HC05 (o altro, che richieda l'invio di \r\n a seguito dei comandi AT), selezionare "Both NL & CR" nella finestra del Serial Monitor.
In questo modo, verranno aggiunti \r\n come fine stringa, a tutto quel che viene inviato.

In questo caso sono stati invitati i comandi
AT+VERSION?
e
AT+NAME?
(notare anche il tipo di risposta, molto diversa rispetto al modulo con fw linvor)

con adattatore usb2serial / Arduino (as serial adapter [6])

- hardware:
convertitore usb2serial; in questo esempio utilizzo un convertitore basato su IC CP2102 (USB to UART Bridge Controller);
- software:
putty (widows e linux) / gtkterm (linux) / CLI (linux)

Si inizia connettendo l'adattatore usb2serial così come indicato nella sezione connessione HW. Quindi si connette l'adattatore al pc.

Linux

Per ottenere il device associato al convertitore, utilizziamo il comando dmesg:

```
$ dmesg
[236353.684185] usb 1-5.4: new full speed USB device using ehci_hcd and address
29
[236353.782142] usb 1-5.4: New USB device found, idVendor=10c4, idProduct=ea60
[236353.782151] usb 1-5.4: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[236353.782158] usb 1-5.4: Product: CP2102 USB to UART Bridge Controller
[236353.782163] usb 1-5.4: Manufacturer: Silicon Labs
[236353.782168] usb 1-5.4: SerialNumber: 0001
[236353.782427] usb 1-5.4: configuration #1 chosen from 1 choice
[236354.267075] USB Serial support registered for cp210x
[236354.267166] cp210x 1-5.4:1.0: cp210x converter detected
[236354.352178] usb 1-5.4: reset full speed USB device using ehci_hcd and
address 29
[236354.445262] usb 1-5.4: cp210x converter now attached to ttyUSB1
[236354.445317] usbcore: registered new interface driver cp210x
[236354.445322] cp210x: v0.09:Silicon Labs CP210x RS232 serial adaptor driver
```

in questo caso il device creato è /dev/ttyUSB1.

Nel caso utilizzassimo una Arduino come adattatore seriale, si otterrebbe un output del genere (utilizzata qui una 2009, con una UNO l'output sarebbe diverso, e il device /dev/ttyACMx)

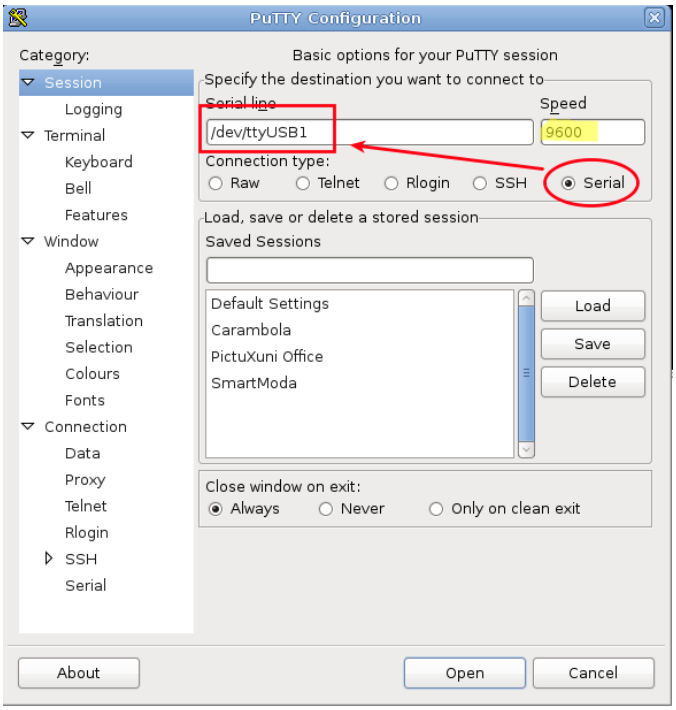
```
[356920.416269] usb 1-5.3: new full speed USB device using ehci_hcd and address
87
[356920.514745] usb 1-5.3: New USB device found, idVendor=0403, idProduct=6001
[356920.514754] usb 1-5.3: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[356920.514761] usb 1-5.3: Product: FT232R USB UART
[356920.514766] usb 1-5.3: Manufacturer: FTDI
[356920.514770] usb 1-5.3: SerialNumber: A600aeNC
[356920.515023] usb 1-5.3: configuration #1 chosen from 1 choice
[356920.518446] ftdi_sio 1-5.3:1.0: FTDI USB Serial Device converter detected
[356920.518516] usb 1-5.3: Detected FT232RL
[356920.518521] usb 1-5.3: Number of endpoints 2
[356920.518526] usb 1-5.3: Endpoint 1 MaxPacketSize 64
[356920.518531] usb 1-5.3: Endpoint 2 MaxPacketSize 64
[356920.518536] usb 1-5.3: Setting MaxPacketSize 64
[356920.518822] usb 1-5.3: FTDI USB Serial Device converter now attached to
ttyUSB0
```

in questo caso il device creato è /dev/ttyUSB0.
Questo è il device a cui dobbiamo connetterci per poter proseguire con la programmazione AT.

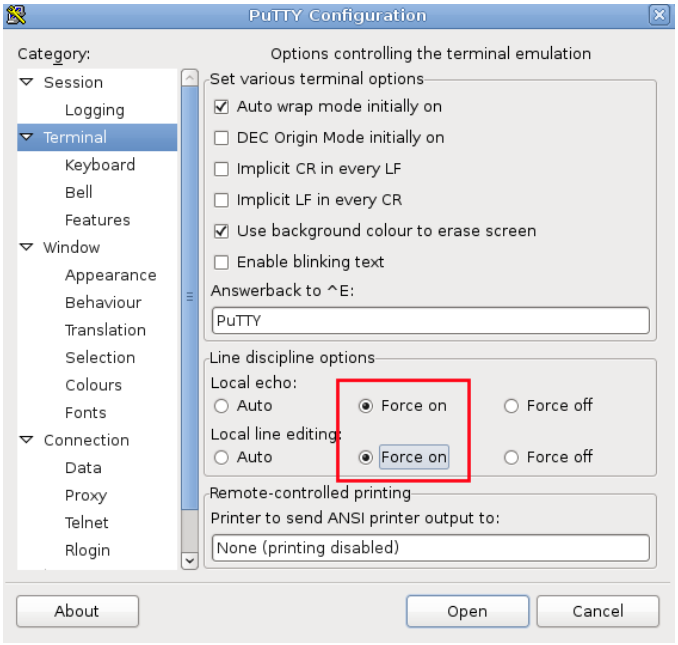
Vediamo ora 2 programmi grafici per gestire la connessione al modulo.

Putty

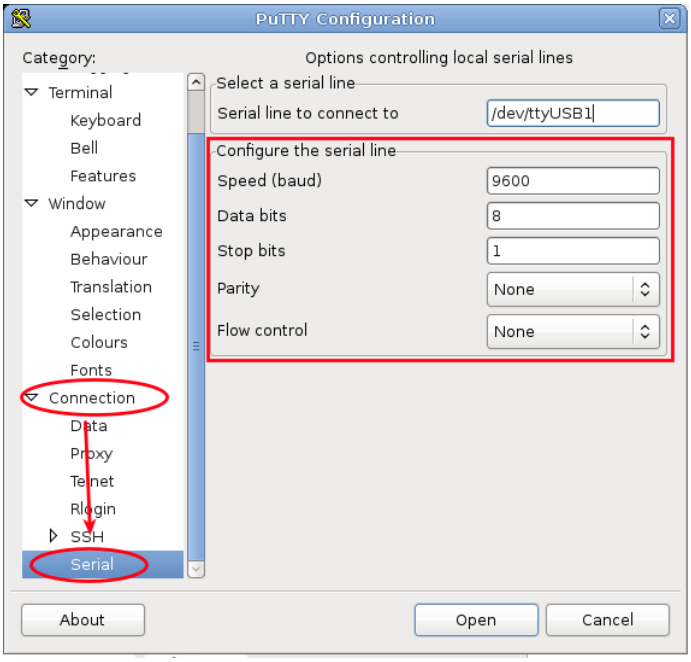
Putty [4] è un programma disponibile sia per Linux che per Windows, pertanto qui ne parliamo per entrambi gli OS.
Una volta avviato, selezionare il tipo di connessione (Serial), e la Serial Line (nel nostro caso /dev/ttyUSB1). Impostare la velocità a quella dichiarata di default (nel ns. caso 9600).



forzare il *local echo* e *local inline editing*, nella sezione Terminal



Nella sezione Connection -> Serial, definire infine i parametri di connessione:

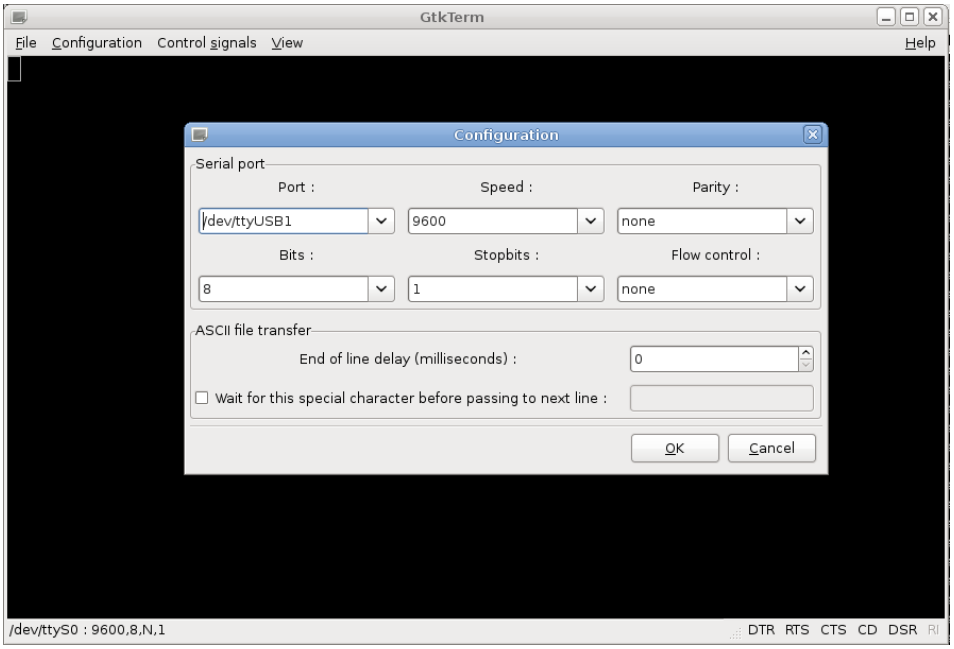


una volta terminati questi passaggi, è possibile aprire la connessione usando il tasto “Open”. In seguito, vedremo come impartire i comandi AT.

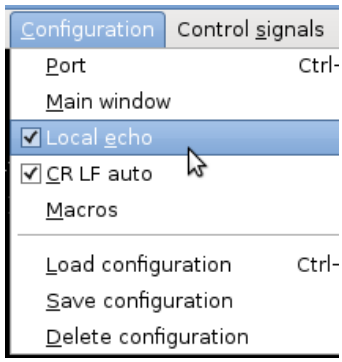
GTKterm

GTKterm [5] è un programma disponibile solo per Linux. E’ disponibile come pacchetto o direttamente nei repository per quasi tutte le distribuzioni Linux.

Una volta avviato, selezionare il menu Configuration -> Port



Anche qui, specificare i valori di default di connessione del modulo, e il device relativo.
Abilitare il local echo, in modo da visualizzare i comandi che digitiamo.
E' possibile anche abilitare il CR LF automatico; torna molto utile nella programmazione dei moduli HC05, evitando così di dover inserire "\r\n" al termine di ogni comando.



La connessione è così attiva.
In seguito, vedremo come impartire i comandi AT.

CLI (Command Line Interface)

Tutti i comandi, se non specificato diversamente, si possono eseguire come user normale (no root). Per approfondire il funzionamento dei singoli comandi, è sufficiente utilizzare al pagina di manuale :

```
$ man <comando>
```

Una volta noto il device (ottenuto con dmesg), è possibile utilizzare la semplice console per

interfacciarsi con il device seriale.

Per visualizzare l'output seriale, è possibile usare il comando

```
cat    oppure    tail -f    oppure    tailf
```

quindi se il device seriale è /dev/ttyUSB1, il comando da lanciare sarà:

```
$ cat /dev/ttyUSB1
```

Per lanciare comandi, invece, si userà il comando

```
echo
```

Quindi se vogliamo interrogare il device seriale, per ottenerne la versione di fw (comando: AT+NAME?\r\n - nota che \r\n non è richiesto per moduli HC06)

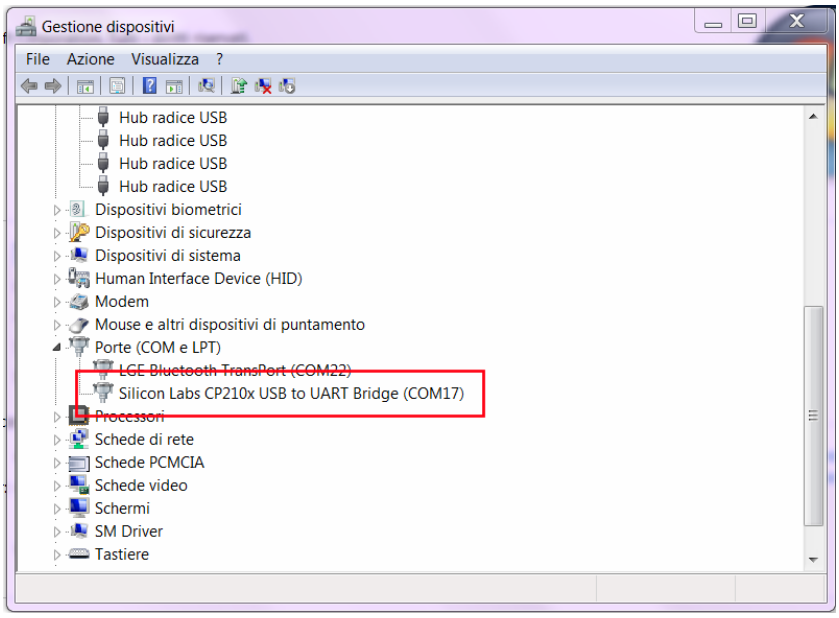
```
$ echo -en  'AT+VERSION?\r\n' > /dev/ttyUSB1
```

L'output del comando lanciato, apparirà nella finestra in cui abbiamo lanciato il *cat.*

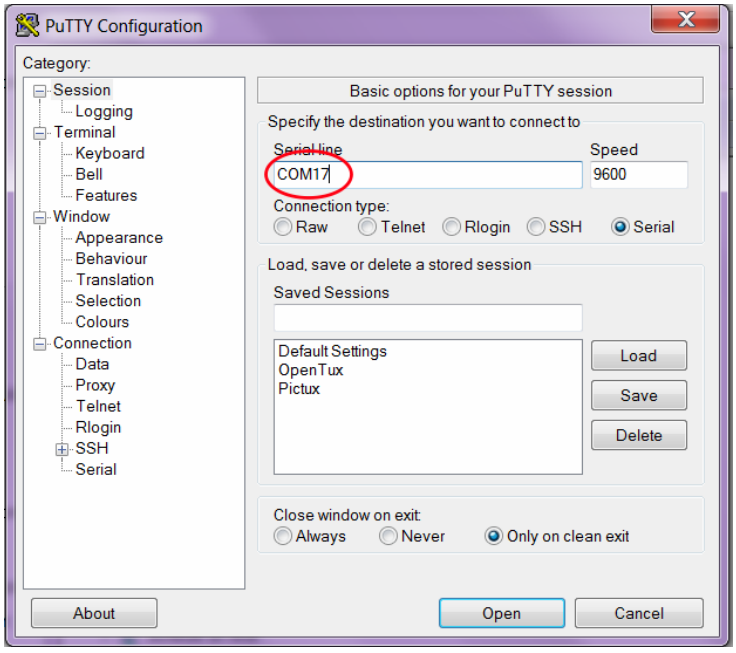
Per approfondire l'utilizzo della seriale da CLI e alternative a quanto scritto, è possibile consultare la pagina LinuxTTY su Arduino Playground [7]

WINDOWS

Sotto Windows si può utilizzare Putty, rimando pertanto alla sezione ad esso dedicata nella parte Linux.
L'unica variazione sarà nel nome del device.
Una volta connesso il nostro adattatore al pc, da risorse del pc, identifichiamo la COM appena creata:



questa COM sarà quella utilizzata in Putty.

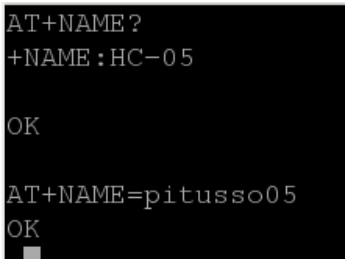


Comandi AT disponibili

Ogni modulo ha il suo set di comandi, vasto nel caso di HC05, limitato nel caso di HC06.
[TODO: postare link a documentazione, o copia incolla tabelle AT]

Per i moduli **HC05**, rifarsi a questo documento: [TODO: link documento]
Negli esempi sotto riportati, non si vedono \r\n in quanto gestiti in automatico da GTKTerm (vedi configurazione nei punti precedenti).

Per esempio, per leggere e cambiare nome del modulo:



AT+NAME?
mostra il nome del modulo (default: HC-05)

AT+NAME=pitusso05
setta in nome del modulo (nei prossimi esempi il modulo avrà tale nome :-D)

AT+VERSION?
mostra il livello del firmware

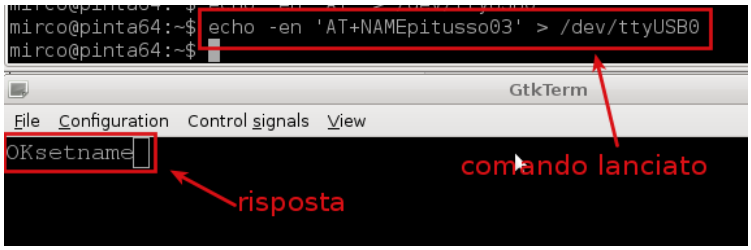
[TODO : inserire più esempi]

Per i moduli **HC06**, i comandi disponibili sono [1]:

Command	Response	Note
AT	OK	Usefull to check connection and baudrate
AT+VERSION	Linvor1.5	Get the version of the module
AT+BAUDx	OKyyyy	Set the baudrate : x can take the following values : ● 1 for 1200 bps ● 2 2400 bps ● 3 4800 bps ● 4 9600 bps ● 5 19200 bps ● 6 38400 bps ● 7 57600 bps ● 8 115200 bps ● 9 230400 bps ● A 460800 bps ● B 921600 bps ● C 1382400 bps
AT+NAMEString	OKsetname	Change bluetooth device name : String can be any string you want ! be creative !! (20 characters limited)

AT+PINxxxx	OKsetpin	Set the bluetooth pincode : 1234 by default

Non inserire per questi moduli “\r\n” al termine del comando AT
Per esempio, per cambiare nome del modulo in “pitusso03”:



l’invio è avvenuto da console in quanto, a causa del fw linvor, è praticamente impossibile riuscire a digitare un comando AT prima che il polling del modulo non lo interpreti incompleto, dando errore.

Webography

[1] http://byron76.blogspot.it/2011/09/one-board-several-firmwares.html
[2] http://blueman-project.org/
[3] http://txapuzas.blogspot.it/2009/12/paperbluetooth-bluetooth-shield-para.html
[4] http://www.putty.org/
[5] http://gtkterm.feige.net/
[6] http://www.zoobab.com/use-the-arduino-as-a-serial-adaptor
[7] http://arduino.cc/playground/Interfacing/LinuxTTY
[8] http://arduiniana.org/libraries/newsoftserial/
[9] http://arduino.cc/hu/Reference/SoftwareSerial

Mirco Piccin aka pitusso, pictux@gmail.com