Corso di Laurea Magistrale in Informatica

Esiot – A.A. 2017/2018

Progetto

Agrumino Configuration by AT Commands

Gruppo 2 - Antonello Meloni & Hicham Lafhouli

I comandi AT

La prima implementazione di un set di comandi AT (AT sta per “Attention”) risale al 1981, quando Dennis Hayes and Dale Heatherington misero in commercio il loro Hayes Smartmodem 300, il quale era in grado di operare in due modi distinti, “data mode” e “command mode” e comunicava attraverso una porta seriale RS-232.

Per passare in “command mode” era necessario inviare al modem una sequenza di escape “+++”, dopodiché i dati inviati attraverso la seriale venivano interpretati dal modem come comandi per operazioni quali effettuazione di una chiamata, chiusura di una comunicazione, impostazione dei parametri della connessione e così via.

Il set dei comandi Hayes, in seguito denominati “AT” per motivi di copiright, consiste in una serie di brevi stringhe di testo, che vengono combinate per produrre le operazioni desiderate, e che possono essere suddivise a formare quattro gruppi di comandi:

a) comandi di base
b) set esteso dei comandi
c) set dei comandi proprietario
d) comandi di registro

La velocità con cui si passò dallo Smartmodem modello 300 al 1200 e poi al 2400 causò la decisione di Hayes di non modificare il sistema di programmazione basato sul set di comandi AT, aprendo di fatto la strada verso la standardizzazione. Infatti, dato il successo dei modem Hayes, anche la concorrenza dotò i propri modem di set di comandi “compatibili”.

Per la semplicità di utilizzo e per le ridotte risorse di calcolo richieste, set di comandi AT, appositamente realizzati, sono l’ideale per la configurazione dei dispositivi IOT.

La libreria AgruminoAt

Le specifiche del linguaggio AT prevedono che sia possibile inserire comandi singolarmente, uno per linea, oppure uno di seguito all'altro, separati da un punto e virgola. Il primo comando della linea deve iniziare con la stringa "AT" seguita da "+CMD".

I successivi debbono essere nella forma "+CMD". Eventuali parametri vanno forniti facendo seguire un segno di "=" al comando. Se il parametro è una stringa deve essere racchiuso tra virgolette.

Per la realizzazione di un interprete di comandi AT con queste caratteristiche è stata implementata la libreria AgruminoAt, basata sulla libreria Agrumino.

A titolo di esempio sono infatti stati implementati i comandi AT per l'accesso a molte delle funzioni da essa offerte.

Allo stato attuale sono stati implementati i seguenti comandi:

Comando	Descrizione
AT	AT ready test
AT+Q	Quit AT interpreter
AT+SLPTIME	With no parameter: returns sleep time in sec, with a numeric parameter within 1 and 4294 sets it
AT+BTLEV	Return battery level
AT+TMP	Return temperature (C°)
AT+SAVECFG	Save configuration to flash memory
AT+READCFG	Read configuration from flash memory
AT+HELP	Print Help
AT+NOP	No operation
AT+LUX	Return illuminance
AT+BTVOLT	Return battery voltage
AT+SOIL	Return soil moisture
AT+BRDON	Turn board on
AT+BRDOFF	Turn board off

Comando	Descrizione
AT+BTCHGR	Return if battery is charging
AT+LEDON	Turn LED on
AT+LEDOFF	Turn LED off
AT+MSQSRV	Print/Set Mosquitto Server address
AT+SSID	Print/Set wifi access point SSID
AT+PSWD	Print/Set wifi access point password
AT+PUBCH	Print/Set Mosquitto Server pub channel
AT+SUBCH	Print/Set Mosquitto Server sub channel
AT+WIFION	Set WIFI AND Mosquitto client ON
AT+WIFIOFF	Set WIFI AND Mosquitto client OFF
AT+DPSON	Set deep_sleep ON
AT+DPSOFF	Set deep_sleep OFF

L’input dei dati

L’input dei dati avviene attraverso una funzione temporizzata che si occupa di ricevere l’input dalla porta USB attraverso la seriale del chip ESP8266.

La temporizzazione si rende necessaria per evitare che la funzione resti bloccata nel ciclo di input indefinitamente e restituisca il controllo al programma principale, nel caso in cui venga meno il collegamento al pc. Il tempo di default è di trenta secondi.

L’analizzatore

L’analizzatore è basato su un automa a stati finiti con quattro stati più uno d’uscita. Allo stato zero, ingresso al sistema, corrisponde l’analisi dell’input relativa all’individuazione del comando da eseguire.

L’analisi viene effettuata un carattere alla volta, sull’input proveniente dalla porta seriale. Quando il sistema allo stato “0” incontra un carattere “;” (char 59) termina la memorizzazione del comando da eseguire e passa allo stato “2” che si occupa di restituirlo, insieme al resto dei caratteri ancora da analizzare. Se si incontra un carattere “=” (char 61), il sistema passa allo stato 3, che si occupa dell’analisi del

parametro relativo al comando appena inserito.

Da qui, se viene individuato un char 34 (virgolette) si passa allo stato “1” , che si occupa dell’analisi di un parametro in formato stringa di testo, altrimenti i caratteri utilizzati saranno considerati facenti parte di un parametro numerico (saranno considerati validi solo i caratteri [0-9] ed il punto decimale).

A partire da qualsiasi stato, se il sistema incontra un char 0, l’analizzatore raggiunge lo stato di uscita.

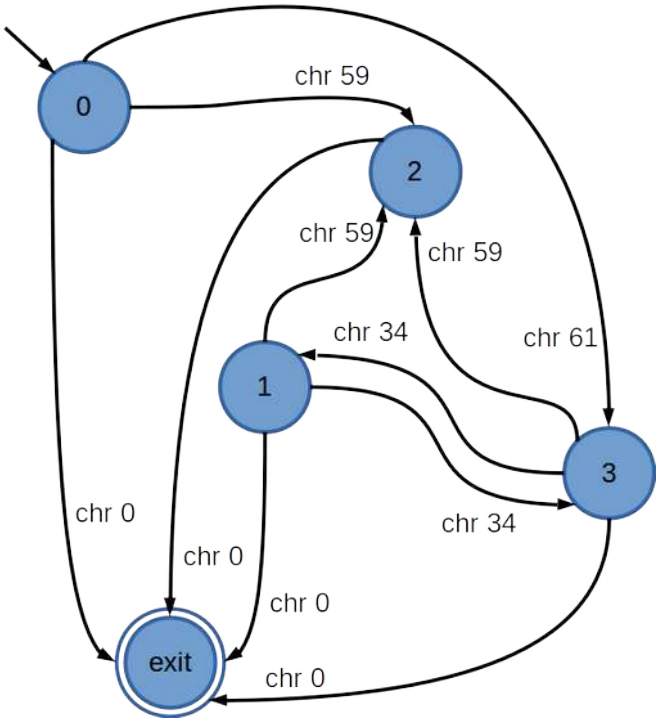

Fig. 1 - Schema automa a stati finiti usato nell’analizzatore.

L’interprete

L’interprete, esegue il suo ciclo principale fino all’immissione del comando di uscita “`AT+Q`”.

Ad ogni esecuzione verifica se il comando immesso corrisponde ad uno dei comandi memorizzati nella tabella dei comandi ed esegue la funzione corrispondente, il cui puntatore a funzione è memorizzato nell’array delle funzioni nella posizione corrispondente al relativo comando.

Tutte le funzioni hanno una dichiarazione simile: restituiscono tutte un void ed hanno per parametri un puntatore a char per il parametro ed un puntatore all’istanza AgruminoAt che contiene tutti i possibili dati relativi alla board `{static void`

`AT_funz0(char* param, AgruminoAt* agruminoAt);}.`

Con tale sistema, è possibile aggiungere, togliere, modificare qualsiasi comando agendo esclusivamente sulla tabella dei comandi, quella degli help associati e quella delle funzioni, nonché sulle funzioni associate a ciascun comando.

Il numero dei comandi implementati è indicato dalla `#define COMMANDS_NUM`

La lunghezza di ogni riga di comandi da `#define ROW_LENGHT` e così via, dando la possibilità di personalizzare l'ambiente per le necessità dell'applicazione principale.

Persistenza dei dati

La scheda Agrumino, per risparmiare energia, trascorre la maggior parte del tempo in uno stato di deep sleep, in cui la board è spenta ed il consumo del processore è ridotto al minimo.

Al risveglio, il programma riprende dalla fase di setup, per poi eseguire la fase di loop: questo ha portato alla nostra attenzione il fatto che qualsiasi variazione ai valori di default ottenuta attraverso comandi AT sarebbe andata persa senza un metodo per assicurarne la persistenza.

Per ottenere quanto necessario, si è fatto ricorso alla definizione di una *structure* a cui si è dato il nome di config.

Allo stato attuale in tale struttura sono presenti i dati relativi al valore in secondi per il deep sleep, all'indirizzo del server mqtt, al SSID e password dell'access point wifi, ai canali sottoscritti e pubblicati sul server mqtt, all'attivazione del wifi e al modo deep sleep, ma può essere aggiunta qualsiasi altra informazione necessaria per il funzionamento delle applicazioni sviluppate sulla libreria.

Tale struttura viene memorizzata sulla memoria flash attraverso l'uso della libreria EEPROM.h che consente di memorizzare nell'ESP8266 fino a 4096 byte.

La configurazione viene salvata e letta attraverso i comandi `AT+SAVECFG` e `AT+READCFG` mediante le funzioni n. 5 e n. 6.

Gli esempi

AgruminoAT e AgruminoAT_wifi sono due esempi di come integrare l'interprete dei comandi AT in una applicazione per Agrumino. Essi sono contenuti nella cartella example all'interno della libreria AgruminoAt e sono accessibili su Arduino IDE attraverso il comando del menu file [Esempi] sotto la voce AgruminoAt nella sezione Esempi da librerie personalizzate, una volta aggiunta la libreria suddetta all'IDE attraverso il comando [Aggiungi libreria da file .ZIP] nel menu Sketch alla voce #include libreria.

L'interprete AT viene avviato attraverso la chiamata del metodo atInterpreter() della classe AgruminoAt.

AgruminoAT

L'applicazione è basata sull'esempio predefinito fornito dalla libreria Agrumino. Essa, ad ogni ciclo, fa lampeggiare il led verde sulla scheda e prevede il deep_sleep per un periodo prefissato (in alternativa può essere usato un delay).

L'avvio dell'interprete è subordinato alla presenza del collegamento USB ed all'immissione di un qualsiasi carattere entro un intervallo di tempo di due secondi ad ogni ciclo.

Una volta avviato l'interprete, si può richiedere l'elenco dei comandi disponibili con `AT+HELP`.

Se si desidera modificare l'intervallo di deep_sleep in secondi, è possibile farlo con `AT+SLPTIME=nuovo valore`. Per poter essere operativo, il cambiamento deve essere salvato nella configurazione, attraverso l'uso del comando `AT+SAVECFG`.

Se, invece, si vuole tornare alla configurazione salvata, eliminando tutte le modifiche apportate dal momento dell'ultimo salvataggio, si può immettere il comando `AT+READCFG`.

Durante la sessione dell'interprete è possibile accedere a tutti i sensori della scheda attraverso comandi AT ed è possibile accendere e spegnere il LED verde della stessa (vedere tabella comandi).

Per uscire dalla sessione dell'interprete occorre inserire il comando `AT+Q`, che interrompe l'interprete e riavvia la scheda. Ogni modifica alla configurazione non

salvata sarà persa.

L’interprete viene arrestato e la scheda riavviata anche in caso di disconnessione della porta USB, dopo un intervallo di tempo di 30 secondi.

AgruminoAT_wifi

Anche questo sketch è basato sull’esempio predefinito fornito dalla libreria Agrumino, a cui è stato aggiunto il supporto per il collegamento ad un server MQTT, attraverso il wifi integrato sul chip ESP8266.

I valori interessati sono:

- mqtt_server indirizzo del server MQTT
- ssid SSID dell’access point WIFI
- password password wifi
- channel_pub canale MQTT in cui la board pubblicherà i messaggi
- channel_sub canale MQTT a cui la board effettuerà la sottoscrizione
- sleep_mode modalità deep_sleep / delay
- ok_start controlla l’avvio del client MQTT e del WIFI

tali valori possono essere impostati tramite altrettanti comandi AT (vedere la tabella comandi e la sequenza d’esempio alla fine del paragrafo “agruminoAT_serial_cmd**”**) e salvati nella configurazione tramite il comando AT+SAVECFG.

Sul canale impostato, la scheda pubblicherà, ad ogni ciclo, i valori della temperatura, dell’umidità del terreno e della carica della batteria.

Il collegamento con il PC

Le comunicazioni per la configurazione di agrumino avvengono attraverso l’interfaccia seriale collegata con una porta USB.

I sistemi operativi basati su Linux assegnano a tali porte un nome di device tipo ttyUSBn, con n numero tra 0 e 9.

Per il collegamento possono essere usati, tra gli altri, i software Arduino-IDE Monitor Seriale, Putty, cu e minicom, attraverso i quali possono essere inseriti manualmente i comandi diretti ad agrumino.

Per rendere possibile l'automazione delle operazioni di configurazione di agrumino in ambiente Linux, è stato realizzato il programma "agruminoAT_serial_cmd", con interfaccia a riga di comando.

agruminoAT_serial_cmd

Il programma, realizzato in C per Linux e basato su un esempio reperito sul portale github [1], utilizza la libreria "*termios.h*" per la gestione delle comunicazione attraverso la porta seriale. Al programma vanno forniti, come parametri, l'identificativo della porta ed il nome del file di testo, contenente i comandi (es. `./Agrumino_AT_serial_cmd /dev/ttyUSB0 cmd.txt`).

Per la sincronizzazione con agrumino, che ad ogni ciclo tra un periodo di deepSleep ed il successivo, attende input per 2 secondi dalla seriale, si è provveduto ad analizzare l'output del medesimo alla ricerca della parola "interpreter", che sancisce l'inizio dell'intervallo utile per la trasmissione dei comandi.

Effettuato l'invio, il programma analizza nuovamente l'output al fine di idividuare la stringa iniziale del logo di agrumino, che sancisce il ritorno al funzionamento normale, visualizzando al contempo il risultato dell'elaborazione dei comandi AT inviati.

L'unica prescrizione da seguire per il corretto funzionamento del programma è la necessità di porre come ultimo comando, nel file dei comandi da eseguire, "`AT+Q`" che termina l'interprete e consente il ritorno al ciclo normale di funzionamento. Naturalmente il penultimo dovrà essere "`AT+SAVECFG`" per il salvataggio della configurazione, se la stessa è stata modificata.

Un esempio di sequenza di comandi per la configurazione di agrumino (per l'utilizzo con l'applicazione "AgruminoAT_wifi" con invio dei dati ogni 60 secondi) è il seguente:

```
AT+SSID="ESIOT";+PSWD="ESIOT1234";+PUBCH="AgruminoAt";+SUBCH="AgruminoAt/cmd";
+MSQSRV="server_address";+SLPTIME=60 ;+DPSON;+WIFION;+SAVECFG;+Q
```

1 https://github.com/xanthium-enterprises/Serial-Port-Programming-on-Linux/blob/master/USB2SERIAL_Read/Reciever%20(PC%20Side)/SerialPort_read.c